Journal of Cybersecurity and Privacy

MDPI

Article

An Approach for Anomaly Detection in Network Communications Using *k-Path* Analysis

Mamadou Kasse [1,2,3,*], Rodolphe Charrier [1], Alexandre Berred [2], Cyrille Bertelle [1] and Christophe Delpierre [3]

[1] Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes, UFR Sciences and Technology, University of Le Havre, 25 rue Philippe Lebon, 76058 Le Havre Cedex, France; rodolphe.charrier@univ-lehavre.fr (R.C.); cyrille.bertelle@univ-lehavre.fr (C.B.)
[2] Laboratoire de Mathématiques Appliquées du Havre, UFR Sciences and Technology, University of Le Havre, 25 rue Philippe Lebon, 76063 Le Havre Cedex, France; alexandre.berred@univ-lehavre.fr
[3] Risk n Tic, 93360 Neuilly Plaisance, France; christophe.delpierre@risknticc.com
* Correspondence: mamadou.kasse@etu.univ-lehavre.fr

Abstract: In this paper, we present an innovative approach inspired by the *Path-scan* model to detect paths with *k* adjacent edges (*k-path*) exhibiting unusual behavior (synonymous with anomaly) within network communications. This work is motivated by the challenge of identifying malicious activities carried out in vulnerable *k-path* in a small to medium-sized computer network. Each observed edge (time series of the number of events or the number of packets exchanged between two computers in the network) is modeled using the three-state observed Markov model, as opposed to the *Path-scan* model which uses a two-state model (active state and inactive state), to establish baselines of behavior in order to detect anomalies. This model captures the typical behavior of network communications, as well as patterns of suspicious activity, such as those associated with brute force attacks. We take a perspective by analyzing each vulnerable *k-path*, enabling the accurate detection of anomalies on the *k-path*. Using this approach, our method aims to enhance the detection of suspicious activities in computer networks, thus providing a more robust and accurate solution to ensure the security of computer systems.

Keywords: cybersecurity; cyberattacks; Markovian model; generalized maximum likelihood ratio; computer networks; network traffic

Citation: Kasse, M.; Charrier, R.; Berred, A.; Bertelle, C.; Delpierre, C. An Approach for Anomaly Detection in Network Communications Using *k-Path* Analysis. *J. Cybersecur. Priv.* **2024**, *4*, 449–467. https://doi.org/10.3390/jcp4030022

Academic Editor: Feng Wang

Received: 6 June 2024
Revised: 4 July 2024
Accepted: 15 July 2024
Published: 19 July 2024

1. Introduction

In the context of information systems, a widely accepted definition by Hawkins (Hawkins, 1980) describes an anomaly as an observation that deviates so significantly from other observations that it raises suspicions that it was generated by a different mechanism [1]. Anomalies are of paramount importance in computer networks because they can signal unusual but significant activities. For instance, a sudden increase in network traffic to an unknown destination might indicate an attempt at hacking or a security breach, necessitating immediate response to protect data and systems from potential threats [2]. Therefore, anomaly detection is a crucial issue in many fields, especially cybersecurity. It involves detecting unexpected or rare events in data streams, commonly referred to as abnormal events. Anomaly detection in networks encompasses various methods and approaches, each with its own strengths and applications. It aims to identify unusual behaviors that might indicate intrusions or other cyberattacks. An overview of anomaly detection in networks using network flows is provided in [3]. A promising method in this domain is the *k-path* approach [4], which leverages network flow events to identify paths exhibiting anomalies using a time series of flow events between pairs of computers.

In 2007, Estonia was the target of a series of large-scale cyberattacks that specifically targeted several of its major banks [5]. These attacks were significant enough to force the

country's leading bank to shut down its online services for several hours, disrupting financial transactions and causing widespread concern among the populace. Moving forward to May 2017, the world witnessed the emergence of WannaCry, a ransomware attack that quickly became one of the most notorious cyberattacks in history. WannaCry spread rapidly across the globe, infecting over 200,000 computers in more than 150 countries within a matter of days. The attack crippled critical systems in various sectors, including healthcare, finance, and government, causing billions of dollars in damages [6]. These incidents brutally remind us of the growing threat posed by cyberattacks to our interconnected digital systems. However, most attackers operate using brute-force attacks, resulting in abnormal spikes in network traffic. Other attackers prefer a more subtle approach by sending low-rate streams, thus blending in with normal flows generated by trusted sources on the network [7]. These low-rate tactics highlight the different strategies used by cybercriminals to infiltrate and disrupt digital systems, with intrusion being the predominant objective for most attackers.

By definition, when an individual undertakes an action for which they are not legally authorized, it is termed intrusion. This intrusion can originate from external sources, through attacks such as unauthorized access attempts or exploitation of security vulnerabilities, or it can result from an intruder exceeding their authorized limits [8]. The actions taken by an intruder, whether from an external or internal source, may follow a well-defined pattern, often represented by the intrusion path. This path can be illustrated by several stages, as shown in Figure 1:

- Initial infection and local research: The intruder often starts by infecting a system or network via various vectors, such as malware or social engineering techniques. Once infiltrated, the intruder conducts local research to identify other potential resources to compromise.
- First Traversal and Further Exploration: After the initial infection, the first traversal of the network occurs. The intruder explores accessible systems to gather information and identify their potential target.
- Complete Traversal and Malicious Actions: Once the intruder has identified their target and obtained the necessary privileges, they can undertake malicious actions.

Let us consider an example where an attacker launches a phishing attack targeting employees of a company:

Step 1:

- An employee opens a malicious email containing a link to a compromised site.
- The compromised site exploits a vulnerability in the employee's browser to install malware on their workstation.
- The malware begins scanning the local network for other vulnerable systems. It identifies an internal server used for storing sensitive data.

Step 2:

- Once on the internal server, the malware begins collecting information about the system, such as credentials stored in plain text in configuration files.
- The intruder uses the retrieved credentials to log in to the server and explore the stored data. They discover a database server containing customer information.

Step 3:

- Using the access privileges obtained on the internal server, the intruder accesses the database server and extracts sensitive customer data, including credentials and financial information.
- The intruder uses this information to conduct fraudulent transactions and access sensitive user accounts.

Each of these steps represents a critical phase of the attack, and detecting these behaviors at an early stage can be crucial in preventing damage and limiting the impact of the intrusion.

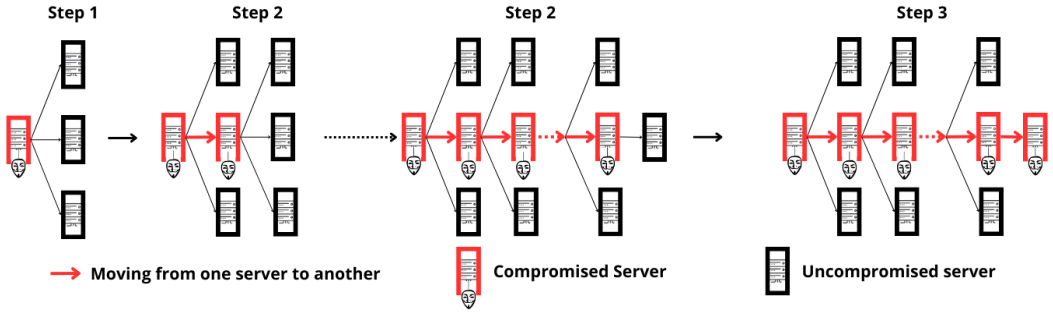


Figure 1. Representation of an intrusion.

Typically, an intrusion is rarely direct. It often employs the tactic of "lateral movement" [9] to traverse multiple servers within the computer network and form a path leading to its target. Despite significant advancements in cybersecurity, the persistent demand for solutions to identify these paths, which are often vulnerable, indicates that challenges remain. Researchers are, therefore, working on developing methods capable of detecting and characterizing unusual behaviors within these paths, aiming to enhance network security against intrusions.

In this article, we address the problem of anomaly detection within these paths through the analysis of network flow data. We begin our discussion by presenting flow data, its processing, and modeling. Then, we review previous work on anomaly detection, such as the work by Joshua Neil and his colleagues [4]. Their work forms the basis of our contribution, which involves considering the number of packets exchanged between two connected servers and applying a three-state observed Markov model to account for the three types of behavior in exchanges between two connected servers: no exchange, exchanges corresponding to a flux peak, or exchanges are moderate. We propose replacing the *Active* state of the *Path Scan* model [4], which represents an exchange of flow between two servers, with two new states. We introduce the *Peak* state, indicating a massive increase in flows, indicating periods where the system is at maximum load, and the *Moderate* state, which characterizes less intense but still significant flow levels. This modification provides a more nuanced representation of the system conditions, facilitating better adaptation. By integrating these new states into the *Path Scan* model, our contribution aims to improve its ability to detect anomalies in the system.

The rest of the document is organized as follows: Section 2 discusses the processing and modeling of flow data. Then, Section 3 presents related work. Section 4 describes the methodology used in our research. Section 5 presents the results obtained using the generated datasets. Finally, Section 6 concludes the paper.

2. Data Processing and Modeling

In computer science, an event is an action or occurrence recognized by a software system. This occurrence can come from various sources such as operating systems, networks, servers, firewalls, antivirus software, database queries, hardware infrastructures, etc. Events are typically recorded in special files called event logs [10]. They come from different sources, such as firewalls, routers, and switches, thus providing an external perspective on activities that may be internal to the network or occur between an internal server and an external server. To collect event streams, organizations usually use automated collection methods such as NetFlow [11].

NetFlow, the most widely used standard for flow data statistics enables the monitoring and recording of all traffic entering or exiting an interface (A physical or logical connection point on a network device such as a router, switch, or a computer's network card), as illustrated in Figure 2.

Figure 2 shows the NetFlow data collection process on a network. Computers and printers send data through routers. The router, enabled with NetFlow, analyzes and

aggregates this traffic into distinct flows. These flows are then grouped into flow export datagrams and exported using the User Datagram Protocol (UDP).

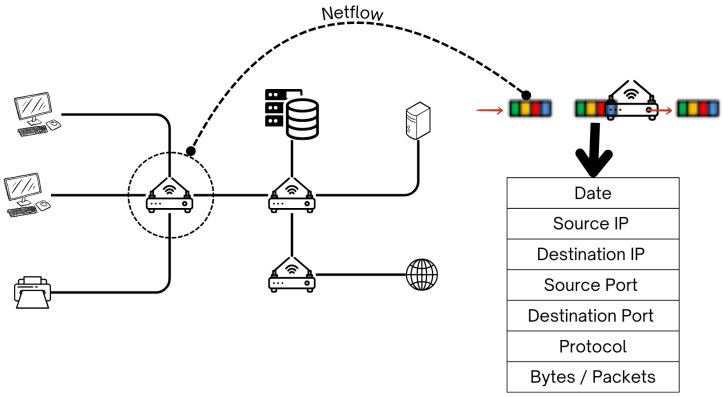


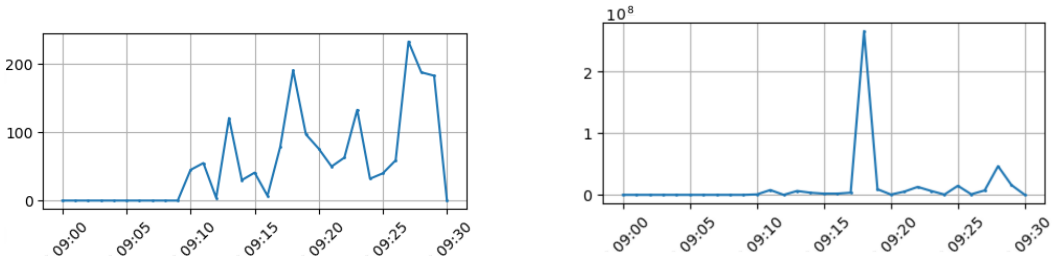
Figure 2. Flow event collection process.

These datagrams are received and processed by a flow collector. The format of each NetFlow record contains at least the following information:

- The date the flow was recorded.
- The duration for which the flow was active was often measured in seconds.
- The source and destination IP addresses (data can be replaced by the source name).
- The type of IP protocol (e.g., TCP = 6; UDP = 17; ICMP = 1).
- The source and destination TCP/UDP port numbers.
- The number of packets in the flow.
- The number of bytes in the flow's packets.

However, to effectively utilize this data for detecting suspicious behaviors, careful preparation is essential. Firstly, it is crucial to select relevant features for analysis. Among these, commonly included are the number of exchanged packets, the total volume of transferred bytes, the source and destination ports used, the duration of the flow record, the date and time of the event, and the involved communication protocol.

Furthermore, we recorded flow events per minute and packets exchanged per minute between 9:00 and 9:30 on a pair of connected addresses in the LNLA network [12]. Figure 3a,b illustrates the number of events and the number of packets recorded per minute.



(**a**) (**b**)

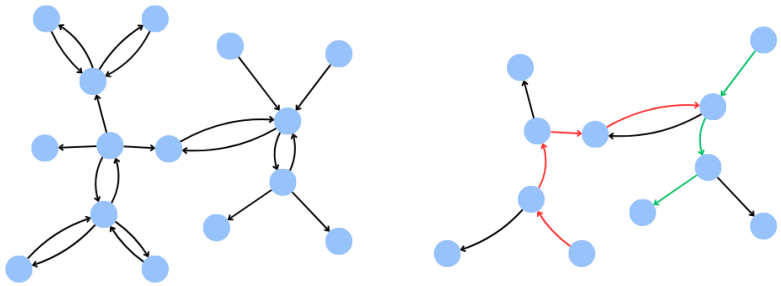
Figure 3. (**a**) Event count per minute between 9:00 and 9:30; (**b**) Packet count per minute between 9:00 and 9:30.

Examining Figure 3a we observe a dynamic showing the number of events per minute against the date we notice a pronounced variability with several event peaks exceeding 100 events per minute. In contrast to Figure 3a, Figure 3b illustrates a notable packet peak occurring at 9:18 a.m., indicating a sudden increase in traffic at that precise moment. These two behaviors exhibit irregular and potentially intense network activity at certain times, suggesting transitions between periods of no activity, moderate activity, and activity spikes.

However, to effectively use this data to detect suspicious behavior, careful preparation is essential. First and foremost, it is crucial to select relevant features for analysis. Among these, commonly included are the number of exchanged packets, the total volume of transferred bytes, the source and destination ports used, the duration of the flow record, the date and time of the event, and the involved communication protocol. These features provide a basis for detecting suspicious activities in network traffic. Additionally, it is important to model the mapping of the studied network from these event logs in order to identify active paths (each pair of computers, which compose this path, communicates through network flows) and analyze them. This simplifies the detection work. This graphical modeling involves grouping events associated with each pair of connected network equipment. These aggregated data also allow the construction of actionable time series, describing activity between well-defined time periods for each pair of network equipment.

A graph [13], denoted G, is defined as a pair (V, E), where V is a set of vertices and E is a set of edges, disjoint from V. This pair is accompanied by an incidence function ψ that associates with each edge of G a pair of vertices (which are not necessarily distinct) from V. If e is an edge and u and v are the vertices such that $\psi(e) = \{u, v\}$, then we say that e connects u and v, and these vertices u and v are called the endpoints of e. The numbers of vertices and edges of G are denoted $v(G)$ and $e(G)$; these two parameters are called the order and size of G, respectively. In the context of a computer network, each vertex of the graph can represent an IP address. The edges can represent exchanges between these IP addresses. From this graph, we can now identify the set of active paths. These will be paths of multiple adjacent edges. In Joshua Neil and colleagues' article, these paths are called *k-paths* of G.

Although Figure 4a represents the network structure for a small to medium-sized enterprise, it shows the absence of communication in the network, although the links (edges) may represent potential communication flow opportunities between IP addresses in the future. Whereas Figure 4b depicts the network using flow data over a 30-min period, using this flow data we are able to illustrate the active side of the network with the aim of identifying only paths with flow activities. For example, the *four-path* in red and the *three-path* in green.



(**a**) (**b**)

Figure 4. (**a**) represents the graphical depiction of a computer network of a small to medium-sized enterprise, and (**b**) illustrates the graphical representation of the network using active paths during a 30-min period. In (**b**), we have depicted an example of a *four-path* in red, and an example of a *three-path* in green.

The vulnerable *k-path* captures the essence of many network attacks, which traverse a path through the network. In Section 4, we will address the methodology for detecting these attacks using modeling of the number of events or the number of packets per minute on each edge of the active *k-path*, based on the work of [4].

As a reminder, the *Path-scan* model, associated with the observed Markov model, is a detection technique developed to identify abnormal subgraphs, such as paths (*k-path*). This

model relies on a probabilistic representation of flow data, where two states are observed: the active state and the inactive state.

The active state indicates the presence of positive records on an edge, meaning that there have been events or packets recorded on this edge during the considered time period. In contrast, the inactive state corresponds to the absence of activity on this edge during the studied period. Under the assumption that the attacker's behavior leads to an increase in the transition probability from the inactive state to the active state, a hypothesis test is conducted for each edge.

3. Related Works

Anomaly detection using continuous data has become a crucial area of research due to the proliferation of data in recent decades and the need to detect events indicating that a process has changed significantly from its previous behavior. This is particularly true in the fields of audiology [14], imaging [15,16], engineering [17], and computer science [18]...

In the field of cybersecurity, various anomaly detection systems have been proposed in the literature, notably in the work of Neil el al., 2013 [4]; Kimura et al., 2014 [19]; Céline Lévy-Leduc, 2014 [20]; Evangelou et al., 2016 [21]; Evangelou et al. 2020 [22] and Corentin Larroche, 2021 [23]. Some of these studies have focused on developing models using data flows to detect anomalies in network communications. For example, in the 2018 study by Bo Zong et al. [24], the authors introduced a model called Deep Autoencoding Gaussian Mixture Model (DAGMM) for unsupervised anomaly detection. This model combines a deep autoencoder with a Gaussian Mixture Model (GMM) to detect anomalies in data streams. The anomaly detection process involves using the deep autoencoder to generate a low-dimensional representation and reconstruction error for each input data point. This information is then fed into the Gaussian mixture model to estimate the probability of each data point being generated by the model. In the 2020 study by Sajjad Hosseinzadeh et al. the Gamma-Normal Mixture Model (GNM) is presented as a crucial component for statistical anomaly detection in network traffic. The authors propose a fast statistical anomaly detector at the aggregate level for two types of anomalies: DDoS and Flood. They then study the compatibility between this model and network traffic using various tests.

Although these anomaly detection methods are capable of identifying irregularities, they are based on specific assumptions about the distribution of packets exchanged or events. The difficulty lies in the lack of precise knowledge of these distributions, which leads us to consider the observed Markov model. This model evaluates the probability of transition from one state to another as a function of network flow events, as shown in the 2013 study by Joshua Neil and colleagues. While this model is effective at detecting anomalies on specific paths through network flow data, it has limitations. In particular, it can struggle to spot anomalies when there is a rapid succession of exchanges. In other words, an unusual sequence of events over a very short period may escape the detection system associated with the observed Markov two-state model. Furthermore, this model does not take into account the quantity of packets exchanged and the spikes in flux that occur in these exchanges but rather focuses on the number of exchanges. These limitations can be problematic in scenarios where the quantity of exchanged packets is crucial for anomaly detection, such as in a brute force attack or a DoS attack, where the volume of exchanged packets can be a key indicator.

In this paper, time series of communications are analyzed using the three-state Markov model and the generalized likelihood ratio test (GLRT) to detect anomalies, in particular, those resulting from brute force behavior that is often human-induced.

4. Three-State Markovian Approach

This section presents a framework for anomaly detection in an active *k-path*, based on the observed Markov model and the generalized likelihood ratio.

4.1. Definition of the States

Most anomaly detection methods based on the observed Markov model and the generalized likelihood ratio detect network anomalies using information extracted from flows, even though some flows may persist for a considerable duration. These methods are, therefore, more suitable for offline detection [4]. In order to detect malicious traffic at an early stage, Joshua Neil et al. [4] aim to identify an anomaly based on the specific number of exchanges between two IP addresses. In this paper, aimed at furthering the method described in [4], we present an approach to identify such anomalies, based on the specific number of packets exchanged between two IP addresses.

From an active graph, let us consider a *k-path* of adjacent edges, denoted as

$$\gamma = \{e_1, e_2, \cdots, e_{k-1}, e_k\} \subset E.$$

By collecting flow data along each edge of this path, we gather information about the number of packets exchanged per minute between the involved IP addresses.

Let $e \in \gamma$; at discrete time points $t \in \{t_0, t_1, \cdots, t_n\}$ with $t_i = t_0 + i$ where t_0 is the initial date and n is the total number of minutes in the sequence, we designate a time window (e, I), where $I = \{[t_{i-1}, t_i[, i = 1, \cdots, n\}$.

Let $X_t(e)$ be the number of packets recorded on e during the period $[t-1, t]$. Let $X(e, I)$ represent the data in window (e, I) we have:

$$X(e, I) = (X_{t_i}(e))_{i=1,\cdots,n}.$$

For each value of $X_{t_i}(e)$, we can deduce a state for the edge e. Figure 5 associated with Figure 3b illustrates the three zones where the variable X_t can be located. Each zone represents a state of the edge during the given period.

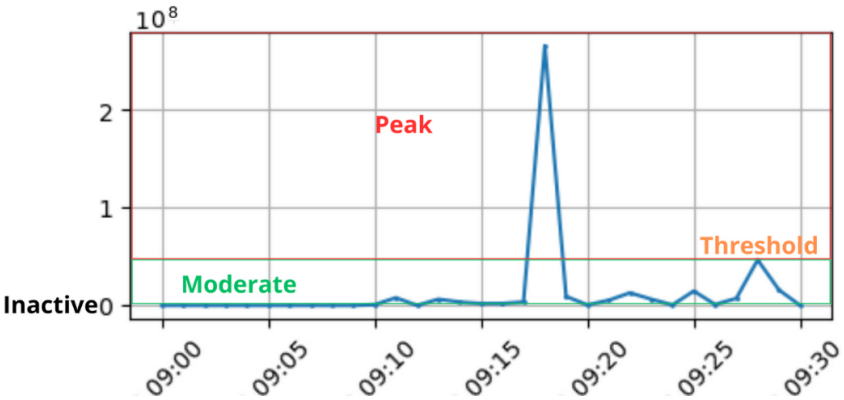


Figure 5. Zones of different states.

The first zone, *State 1*, corresponds to the absence of packets; it represents periods of one minute where the counters are zero. In this configuration, we qualify this state as *inactive*.

The second zone, *State 2*, is characterized by the presence of activity resulting in an exchange of packets, but at a moderate level. This means that the number of packet exchanges is controlled and does not exceed a defined threshold. In this regard, we describe this state as *moderate*.

Finally, the third zone corresponds to cases where exchanges explode, far exceeding the defined threshold. This scenario, *State 3*, represents periods of intense activity where the number of packet exchanges is well beyond the established threshold. We qualify this state as *Peak*.

In summary, these three zones describe the different possible states of the edge, ranging from inactivity to moderate activity, up to intense activity. This classification helps to better understand and analyze the behavior of the edge in different contexts and levels of activity.

We have a data process $\mathbf{S}_{t_i}(e)$ representing the state of the edge during the period $[t_{i-1}, t_i[$; it takes values in $\mathbb{S} = \{1, 2, 3\}$ with

$$\mathbf{S}_{t_i}(e) = \begin{cases} 1 & \text{if } X_{t_i}(e) = 0 \\ 2 & \text{if } 0 < X_{t_i}(e) \leq X_{\text{Peak}}(e) \\ 3 & \text{if } X_{t_i}(e) \geq X_{\text{Peak}}(e). \end{cases}$$

In this formulation, $X_{\text{Peak}}(e)$ represents the threshold of $X_{t_i}(e)$, determining the transition from the moderate state to the peak state.

A network data flux peak occurs when there is a sudden and significant increase in the volume of data traversing through a computer network over a defined period. However, determining an appropriate threshold to conclude that a flux peak has occurred can be challenging. One commonly used approach is to employ statistical analysis techniques to model the distribution of network data flux. These models help determine threshold values based on features such as the mean, standard deviation, quantiles, or other statistics (see the works of Standardization Working Group No. 5 [25]). The threshold values thus defined can then be used to detect periods where the data volume significantly exceeds expected levels, indicating the presence of a flux peak.

In the future, we plan to propose a study on selecting the threshold $X_{\text{Peak}}(e)$ for detecting data traffic peaks. In our simulation, we will use the mean (μ) and standard deviation (σ) to define this threshold. The mean represents the typical value of data flux, while the standard deviation measures the dispersion of data around the mean. By setting the threshold at a certain number of standard deviations above the mean, we can identify periods where the data volume exceeds what is considered moderate. We define:

$$X_{\text{Peak}}(e) = \mu(e) + h \times \sigma(e),$$

where $\mu(e)$ is the mean of the data $X_{t_i}(e)$, $\sigma(e)$ is the standard deviation of the data $X_{t_i}(e)$, h is a coefficient that determines the number of standard deviations above the mean to define the threshold. The larger h is, the higher the threshold will be, and thus more stringent in terms of peak detection.

4.2. Modeling, Estimation, and Hypothesis Testing

We thus define the observed Markov model over e, characterized by nine parameters,

$$p_{\alpha,\beta}(e) = \mathbb{P}(\mathbf{S}_t(e) = \beta | \mathbf{S}_{t-1}(e) = \alpha) \text{ for } (\alpha, \beta) \in \mathbb{S}^2,$$

where $p_{\alpha,\beta}(e)$ represents the transition probability from state α to state β.

We assume that the initial state of the model is fixed and known. Thus, the likelihood of the model is formulated as follows:

$$\mathcal{L}\Big((p_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2} | \mathbf{s}(e, I)\Big) = \prod_{(\alpha,\beta)\in\mathbb{S}^2} \big(p_{\alpha,\beta}(e)\big)^{n_{\alpha,\beta}(e)},$$

where $\mathbf{s}(e, I) = (\mathbf{s}_{t_1}(e), \mathbf{s}_{t_2}(e), \cdots, \mathbf{s}_{t_n}(e))$ is the sequence of observation states on (e, I) and $n_{\alpha,\beta}(e)$ is the number of times the consecutive pair $\{\alpha, \beta\}$ has been observed in $\mathbf{s}(e, I)$ with

$$\sum_{(\alpha,\beta)\in\mathbb{S}^2} n_{\alpha,\beta}(e) = n - 1$$

Estimating the maximum likelihood parameters of the model is a crucial step in fitting the model to the observed data. The most commonly used method for this task is the Maximum Likelihood Estimation (MLE).

The log-likelihood (ℓ) is then obtained by taking the natural logarithm of the likelihood function, thus transforming the product into a sum:

$$\ell\Big((p_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2}|\mathbf{s}(e,I)\Big) = \sum_{(\alpha,\beta)\in\mathbb{S}^2} n_{\alpha,\beta}(e)\log(p_{\alpha,\beta}(e))$$

The objective is to maximize this log-likelihood function with respect to the parameters $\theta(e,I) = (p_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2}$. This can be achieved using numerical optimization techniques. Sometimes it is necessary to consider constraints on the parameters; one common constraint is that for a given state $\alpha \in \mathbb{S}$, the total transition probability from that state must be equal to 1:

$$\sum_{l\in\mathbb{S}} p_{\alpha,l}(e) = 1.$$

We can introduce a Lagrange multiplier, denoted c_α, and modify the likelihood function to include the penalty term $c_\alpha(\sum_{l\in\mathbb{S}} p_{\alpha,\beta}(e) - 1)$. The parameter estimation problem then becomes that of maximizing this penalized likelihood function:

$$\mathfrak{L}(p_{\alpha,\beta}(e), c_\alpha) = \sum_{(\epsilon_1,\epsilon_2)\in\mathbb{S}^2} n_{\epsilon_1,\epsilon_2}(e)\log(p_{\epsilon_1,\epsilon_2}(e)) - c_\alpha(\sum_{l\in\mathbb{S}} p_{\alpha,l}(e) - 1).$$

Using the method of Lagrange multipliers, we seek the value of the parameter $p_{\alpha,\beta}(e)$ and the Lagrange multipliers c_α that nullify the partial derivatives of the Lagrange function $\mathfrak{L}(p_{\alpha,\beta}(e), c_\alpha)$ with respect to the parameters $p_{\alpha,\beta}(e)$ and the Lagrange multipliers c_α. Differentiating $\mathfrak{L}(p_{\alpha,\beta}(e), c_\alpha)$ with respect to $p_{\alpha,\beta}(e)$ and setting the derivative to zero, we obtain:

$$\frac{\partial\mathfrak{L}(p_{\alpha,\beta}(e), c_\alpha)}{\partial p_{\alpha,\beta}(e)} = \frac{n_{\alpha,\beta}(e)}{p_{\alpha,\beta}(e)} - c_\alpha = 0$$

Furthermore, since $\sum_{\delta\in\mathbb{S}} p_{\alpha,\delta}(e) = 1$, we then have:

$$c_\alpha = \sum_{\delta\in\mathbb{S}} n_{\alpha,\delta}(e),$$

$$\hat{p}_{\alpha,\beta}(e) = \frac{n_{\alpha,\beta}(e)}{\sum_{\delta\in\mathbb{S}} n_{\alpha,\delta}(e)}.$$

Now, we are discussing hypothesis testing within a statistical framework, specifically regarding the determination of whether new observations in a particular path (γ) are likely to have been generated by a known function of the parameters of the model associated with that path ($\theta_0(\gamma) = (\theta_0(e_1), \theta_0(e_2), \cdots, \theta_0(e_k))$), as opposed to hypotheses indicating that these parameters have changed. In other words, given new observations $\mathbf{S}(\gamma, I_1) = \mathbf{s}(\gamma, I_1)$, the goal is to test the null hypothesis $H_0 : \theta(\gamma, I_1) = \theta_0(\gamma)$ against the alternative hypothesis that can be formed by restricting the global parameter space Θ to a subset $\Theta_A \subset \Theta$. The statistical test of the generalized likelihood ratio is a relevant measure to use in this context. It is defined as follows:

$$\lambda(\gamma, I_1) = -2\log\left(\frac{\mathcal{L}(\theta_0(\gamma)|\mathbf{s}(\gamma, I_1))}{\sup_{\theta\in\Theta_A}\mathcal{L}(\theta(\gamma, I_1)|\mathbf{s}(\gamma, I_1))}\right)$$

with

$$\gamma = \{e_1, e_2, \cdots, e_{k-1}, e_k\} \subset E.$$

In other words, this statistic measures the divergence between the likelihood of the model under the null hypothesis and the maximum likelihood under all alternative hypotheses.

In Joshua Neil et al. [4], the authors introduce the assumption of independence between the edges of a *k-path*. This assumption translates to:

$$\lambda(\gamma, I_1) = \sum_{e \in \gamma} -2 \log \left(\frac{\mathcal{L}(\theta_0(e)|\mathbf{s}(e, I_1))}{\sup_{\theta \in \Theta_A} \mathcal{L}(\theta(e, I_1)|\mathbf{s}(e, I_1))} \right).$$

So, this assumption gives us the ability to model the behavior of each edge of the *k-path* to deduce a modeling of the *k-path*, γ.

To design a generalized likelihood ratio test, it is imperative to constrain our global parameter space to include alternatives reflecting the behaviors of attackers we seek to detect. These alternatives must be formulated generically to encompass a variety of behaviors. Our underlying hypothesis is that the attacker's behavior leads to an increase in the maximum likelihood estimators of the model parameters. Specifically, we posit that the attacker's actions result in an increase in certain transition probabilities between specific pairs of states such as $(\alpha, \beta) \in \mathbb{S}^2$ where $(\alpha \leq \beta) \in \mathbb{S}^2$. Additionally, the attacker's behavior will also manifest as a decrease in the frequency of observing state 1 in the sequences of observations.

For each $e \in \gamma$, this concerns the transition probabilities from state α to state β with $\beta \geq \alpha$ where $\beta > 1$.

$$\begin{cases} H_0 : \forall \alpha, \beta, \ \ \hat{p}_{\alpha,\beta}(e) \leq \tilde{p}_{\alpha,\beta}(e) \ \text{ with } \ \beta \geq \alpha \ \text{ or } \ \hat{f}_1(e) \geq \tilde{f}_1(e) \\ H_1 : \exists \alpha, \beta, \ \ \hat{p}_{\alpha,\beta}(e) > \tilde{p}_{\alpha,\beta}(e) \ \text{ with } \ \beta \geq \alpha \ \text{ and } \ \ \hat{f}_1(e) < \tilde{f}_1(e), \end{cases}$$

where $(\tilde{p}_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2}$ are the historical parameter values on e, $\tilde{f}_1(e)$ is the frequency of observing the state 1 in the historical observations on e, $(\hat{p}_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2}$ are the parameter values that maximize the likelihood of the model for a new series of observations on e, and $\hat{f}_1(e)$ is the frequency of observing the state 1 for the new series of observations on e.

Thus, for any edge $e \in \gamma$, we can determine the statistic of the generalized likelihood ratio $\lambda(e, I_1)$ given by:

$$\lambda(e, I_1) = -2 \log \left(\frac{\mathcal{L}\left((\tilde{p}_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2} | \mathbf{S}(e, I_1) = \mathbf{s}(e, I_1) \right)}{\mathcal{L}\left((\hat{p}_{\alpha,\beta}(e))_{(\alpha,\beta)\in\mathbb{S}^2} | \mathbf{S}(e, I_1) = \mathbf{s}(e, I_1) \right)} \right).$$

We are seeking a p-value for the observed generalized likelihood ratio test, $\lambda(\gamma, I_1)$. The assumption of independence among network communications [4] implies:

$$\lambda(\gamma, I_1) = \sum_{e \in \gamma} \lambda(e, I_1).$$

Under conditions of moderate regularity, the generalized likelihood ratio test $\lambda(e, .)$ is asymptotically distributed according to a χ^2 distribution with degrees of freedom equal to the number of free parameters. However, this only applies under certain conditions (see Casella and Berger, 2001, p. 516 [26]). Let $\Lambda(e, \cdot)$ denote the distribution of $\lambda(e, \cdot)$. Let $\Lambda(e, \cdot)$ be defined as follows:

$$\Lambda(e, \cdot) = B(e, \cdot) \times G(e, \cdot) \ \text{ where } \ B(e, \cdot) \sim \text{Bernoulli}(p_e)$$

and

$$G(e, \cdot) \sim \text{Gamma}(\xi_e, \eta)$$

where p_e and ξ_e are specific parameters exclusively for $\Lambda(e, \cdot)$, and η is specific exclusively for $\Lambda(\gamma, \cdot) = \sum_{e \in \gamma} \Lambda(e, \cdot)$.

Given that $\lambda(e, \cdot)$ could be zero, $\Lambda(e, \cdot)$ must have a point mass at zero, captured by $B(e, \cdot)$. To model the positive part of the distribution of $\Lambda(e, \cdot)$, the Gamma distribution is appealing as it equals a χ^2 distribution with degrees of freedom ν_e when $\xi_e = \frac{\nu_e}{2}$ and $\eta = 2$.

The asymptotic distribution of $\lambda(e,\cdot)$ is then the sum of independent random variables distributed according to a χ^2 distribution with zero inflation.

Let $\lambda_r(e) = \lambda(e, I_r)$ be the statistic of the generalized likelihood ratio test on e for a sequence of observations $\mathbf{s}(e, I_r)$; through maximum likelihood estimation, for a sample of size m, $(\lambda_1(e), \lambda_2(e), \cdots, \lambda_m(e))$, we estimate the parameters p_e and ξ_e. The likelihood of the sample is given by:

$$\mathcal{L}(p_e, \xi_e, \eta | \lambda_1(e), \lambda_2(e), \cdots, \lambda_m(e)) = \prod_{q=1}^{m} (1-p_e)^{\mathbb{I}(\lambda_q(e)=0)} \times \left(\frac{\lambda_q(e)^{\xi_e - 1} \exp^{-\frac{\lambda_q(e)}{\eta}}}{\Gamma(\xi_e)\eta^{\xi_e}} \right)^{\mathbb{I}(\lambda_q(e)>0)}.$$

The log-likelihood is given by:

$$\begin{aligned} \ell(p_e, \xi_e, \eta | \lambda_1(e), \lambda_2(e), \cdots, \lambda_m(e)) = \sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)=0)\log(1-p_e) + \mathbb{I}(\lambda_q(e)>0) \\ \times \left((\xi_e - 1)\log(\lambda_q(e)) - \frac{\lambda_q(e)}{\eta} - \xi_e \log(\eta) - \log(\Gamma(\xi_e)) \right) \end{aligned} \tag{1}$$

- **Estimation of** p_e: To achieve this, we advocate for the use of maximum likelihood estimation. Upon differentiation, we obtain:

$$\frac{\partial \ell(p_e, \xi_e, \eta | \lambda_1(e), \lambda_2(e), \cdots, \lambda_m(e))}{\partial p_e} = -\sum_{q=1}^{m} \frac{\mathbb{I}(\lambda_q(e)=0)}{1-p_e} + \sum_{q=1}^{m} \frac{\mathbb{I}(\lambda_q(e)>0)}{p_e}.$$

Setting

$$-\sum_{q=1}^{m} \frac{\mathbb{I}(\lambda_q(e)=0)}{1-p_e} + \sum_{q=1}^{m} \frac{\mathbb{I}(\lambda_q(e)>0)}{p_e} = 0,$$

we obtain:

$$\frac{\sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)=0) + \sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)}{\sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)} = \frac{1}{p_e},$$

$$\frac{m}{\sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)} = \frac{1}{p_e}.$$

Hence

$$\hat{p}_e = \frac{\sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)}{m}.$$

- **Estimation of** ξ_e: To estimate ξ_e, we will directly use numerical optimization of

$$\ell(p_e, \xi_e, \eta | \lambda_1(e), \lambda_2(e), \cdots, \lambda_m(e)).$$

- **Estimation of** η: For this, we recommend using the derivative of

$$\ell(p_l, \xi_l, \eta | \lambda_1(e), \lambda_2(e), \cdots, \lambda_m(e))$$

with respect to η and setting it to zero, we obtain:

$$\sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)\left(\frac{\lambda_q(e)}{\eta^2} - \frac{\xi_e}{\eta} \right) = 0 \quad \text{with } \eta > 0$$

$$\frac{\sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)\lambda_q(e)}{\eta} = \sum_{q=1}^{m} \mathbb{I}(\lambda_q(e)>0)\xi_e \quad \text{with } \eta > 0$$

$$\frac{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\lambda_q(e)}{\eta}=\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\xi_e \quad \text{with } \eta>0$$

$$\frac{\eta}{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\lambda_q(e)}=\frac{1}{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\xi_e} \quad \text{with } \eta>0$$

$$\eta=\frac{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\lambda_q(e)}{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\xi_e} \quad \text{with } \sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\xi_e\neq 0$$

$$\hat{\eta}=\frac{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\lambda_q(e)}{\sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\hat{\xi}_e} \quad \text{with } \sum_{e\in\gamma}\sum_{q=1}^{m}\mathbb{I}(\lambda_q(e)>0)\hat{\xi}_e\neq 0$$

We denote the maximum likelihood parameters on e of a path by $(\hat{p}_e, \hat{\xi}_e, \hat{\eta})$.

Let I_r be defined by $I_r = \{[t_1+i, t_1+i+1[, i=0,\cdots,30\}$ where t_1 is the initial date.

The p-value is computed by:

$$\begin{aligned}
p\text{-}value(\gamma, I_r) =& \mathbb{P}(\Lambda(\gamma, I_r) > \lambda(\gamma, I_r)) \\
=& \sum_{b_{e_1}=0}^{1}\sum_{b_{e_2}=0}^{1}\cdots\sum_{b_{e_k}=0}^{1}\mathbb{P}(\Lambda(\gamma, I_r) > \lambda(\gamma, I_r)|b_1, b_2, \cdots, b_k) \\
& \times\prod_{e\in\gamma}\mathbb{P}(B(e, I_r) = b_e) \\
=& \sum_{b_{e_1}=0}^{1}\sum_{b_{e_2}=0}^{1}\cdots\sum_{b_{e_k}=0}^{1}\left(\prod_{e\in\gamma}(1-p_e)^{1-\mathbb{I}(B(e,I_r)=b_e)}(p_e)^{\mathbb{I}(B(e,I_r)=b_e)}\right) \\
& \times\left(1-F_\Gamma\left(\lambda(\gamma)|\sum_{e\in\gamma}\mathbb{I}(B(e, I_r)=b_e)\hat{\xi}_e, \hat{\eta}\right)\right)
\end{aligned}$$

where F_Γ is the cumulative distribution function of the Gamma distribution.

The threshold for p-values is a pre-selected threshold, denoted as r_0, representing the significance level of the statistical test. It indicates the maximum probability of incorrectly rejecting the null hypothesis when it is actually true. In other words, the p-value threshold is used to determine if a p-value obtained from a statistical test is sufficiently low to reject the null hypothesis. The threshold, r_0, in these cases, could be defined as follows:

- If the p-value $(\gamma, I_r) > r_0$, then the observation on (γ, I_r) is considered normal.
- If the p-value $(\gamma, I_r) \leq r_0$, then the observation on (γ, I_r) is considered abnormal.

5. Results

In this section, we describe the simulation process used to evaluate the performance of the three-state model compared to the *Path-scan* model.

Thus, as part of our evaluation of the performance of the three-state model, we rely on this model as a reference to compare the effectiveness of our anomaly detection approaches in flow data.

5.1. Generation of Flow Data

We generated simulated flow data to represent network communications over a *three-path*, γ, defined by:

$$\gamma = \{(A, B), (B, C), (C, D)\}$$

and represented by the Figure 6:

Figure 6. Graphical representation of γ.

Each data record includes information such as the date and time of communication, communication duration, source and destination addresses, the protocol used, the number of packets exchanged, etc. These data were randomly generated using appropriate distributions for each attribute.

5.2. Training the Models

We generated training and adjustment data for three edges (A, B), (B, C), and (C, D) over a period from January 1st to January 31st. The total number of events for each edge is set to 13,780, 14,210, and 14,880, respectively, with specific means and standard deviations for the number of packets: 325 (standard deviation 50) for edge (A, B), 357 (standard deviation 47) for edge (B, C), and 332 (standard deviation 62) for edge (C, D). For each edge, packets were generated following a normal distribution, with randomly added peaks of 451 for edge (A, B), 672 for edge (B, C), and 405 for edge (C, D), simulating extreme variations. Timestamps were randomly assigned to these packets.

Furthermore, we identified a traffic peak when the number of packets exceeded X_{Peak}, determined using training data under normal conditions, with

$$X_{\text{Peak}} = \mu + 4\sigma,$$

where μ is the mean of X_t during the training phase and σ is its standard deviation. This approach ensures that the threshold is positioned significantly above the mean, making it less sensitive to minor fluctuations and more robust for detecting significant variations.

The Tables 1 and 2 summarize the values of the training parameters for each edge according to the model:

Table 1. Observed two-state Markov model, active and inactive states (*Path-scan* model).

Index	Src	Dest	$p_{\text{inactive,inactive}}$	$p_{\text{inactive,active}}$	$p_{\text{active,inactive}}$	$p_{\text{active,active}}$	p	τ	η
0	A	B	0.681	0.318	0.679	0.320	0.479	0.330	6.045
1	B	C	0.670	0.329	0.672	0.327	0.533	0.367	6.045
2	C	D	0.654	0.345	0.657	0.342	0.520	0.330	6.045

Table 2. Observed three-state Markov model using the number of packets exchanged per minute.

Index	Src.	Dest.	X_{Pic}	f_1	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}	p_{23}	p_{31}	p_{32}	p_{33}
0	A	B	505	0.680	0.681	0.310	0.007	0.680	0.311	0.007	0.632	0.361	0.005
1	B	C	545	0.671	0.670	0.317	0.012	0.672	0.315	0.011	0.682	0.302	0.015
2	C	D	505	0.655	0.654	0.336	0.008	0.657	0.332	0.010	0.651	0.333	0.014
Index	**Src.**	**Dest.**	p	τ	η								
0	A	B	0.610	0.557	6.068								
1	B	C	0.610	0.557	6.068								
2	C	D	0.564	0.515	6.068								

These models have identical transition probabilities from the inactive state to the inactive state, which is reflected as

$$p_{\text{inactive,inactive}}(e) = p_{1,1}(e), \ \ \forall \ e \in \gamma.$$

The transition probability from the active state to the inactive state in the *Path-scan* model is equivalent to the average of the transition probabilities from the moderate state to the inactive state and the transition probability from the peak flow state to the inactive state. This combination is represented by

$$p_{\text{active,inactive}}(e) = \frac{p_{3,1}(e) + p_{2,1}(e)}{2}, \ \ \forall \ e \in \gamma.$$

The transition from the inactive state to the active state ($p_{\text{inactive to active}}$) in the adjusted model is equivalent to the transition from the inactive state to the moderate state or the peak flow state. This increased responsiveness is represented by

$$p_{\text{inactive,active}}(e) = p_{1,3}(e) + p_{1,2}(e), \ \ \forall \ e \in \gamma.$$

The transition from the active state to the active state in the adjusted model is equivalent to the transition from the moderate state or the peak flow state to the moderate state or the peak flow state. This integration of different activity levels is represented by

$$p_{\text{active,active}}(e) = \frac{p_{2,3}(e) + p_{2,2}(e) + p_{3,3}(e) + p_{3,2}(e)}{2}, \ \ \forall \ e \in \gamma.$$

5.3. Determination of the p-Value Threshold (r_0)

To establish the threshold (r_0) for each model, we implemented a process to generate a scenario reflecting normal behavior. For the period from February 1 to February 28, we generated data packets for three edges with specific means and standard deviations: 200 (standard deviation of 45) for the edge (*A, B*), 205 (standard deviation of 48) for the edge (*B, C*), and 151 (standard deviation of 41) for the edge (*C, D*). Peaks were introduced randomly within intervals according to a specific distribution. Each peak was assigned based on predefined values for each edge, using a set of defined weights. For example, the number of peaks was selected from the values 0 and 1 with respective weights of 0.9 and 0.1, thus favoring the value 0 more frequently. This process aims to simulate extreme variations that can occur under real network conditions. Additionally, we ensured that no edge could be involved in more than 14 different exchanges within 30 min. Timestamps were randomly assigned to the generated data packets for each edge. To achieve a false alarm rate of one alarm per day, we selected the tenth smallest *p*-value from the resulting list of *p*-values, as chosen in the two-state scan path model [4]. The Table 3 illustrates the *p*-value thresholds for each model.

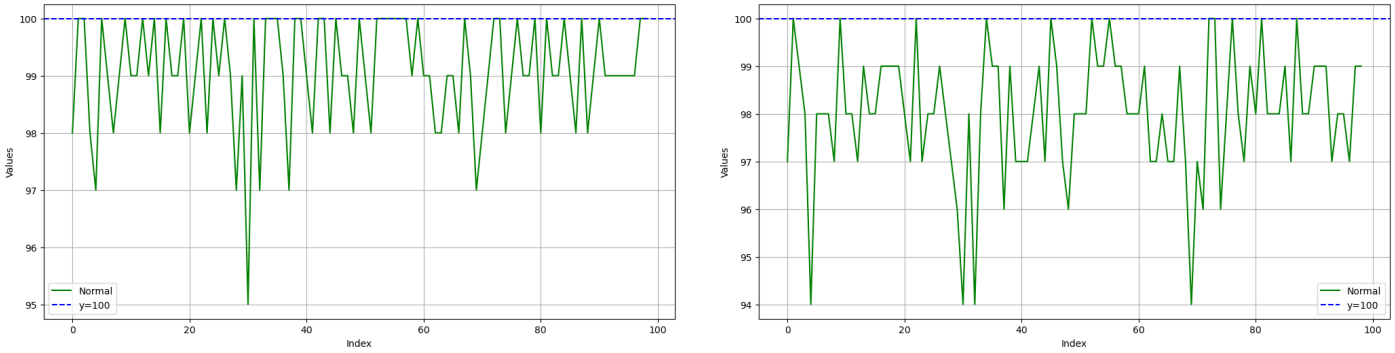
Table 3. Thresholds (r_0) for the models.

Models	*Path-scan* model associated with the observed Markov model	Three-state model using the number of packets exchanged per minute
Threshold (r_0)	0.13846332431809666	0.3322934604988478

5.4. Comparison of Models

We evaluated the performance of different models to detect unusual behaviors on the studied path. Specifically, we compared the *Path-scan* model associated with the Observed Markov Model (OMM) with two states, as presented in Joshua Neil et al.'s article [4], against a three-state model using the number of exchanged packets. The latter model considers the occurrences of the inactive state observed in the observations. These models underwent several rounds of simulations. To reduce false positives, we adjusted the hypothesis test by adding 0.2 to the historical parameters.

In the first scenario, we generated data reflecting the normal behavior of the studied path. To conduct this, we followed the data generation process used during the threshold determination phases for *p*-value. A total of 100×100 tests, each lasting 30 min, were conducted, all reflecting normal behavior. Subsequently, we randomly divided the tests into 100 series of 100 tests to determine the prediction accuracy percentages with a confidence interval. The results obtained for the different series for each model are presented in Figure 7. Figures 8 and 9, respectively, illustrate the box plots for the predictions of the *Path-scan* model associated with the observed Markov model and the three-state model using the number of packets exchanged per minute.



(**a**) (**b**)

Figure 7. Scenario No. 1: In green, the evolution of the number of times a model predicts "normal" behavior across the series of 100 tests is observed. In blue, the expected values for "normal" behaviors are depicted. (**a**) *Path-scan* model associated with the observed Markov model; (**b**) Three-state model using the number of packets exchanged per minute.

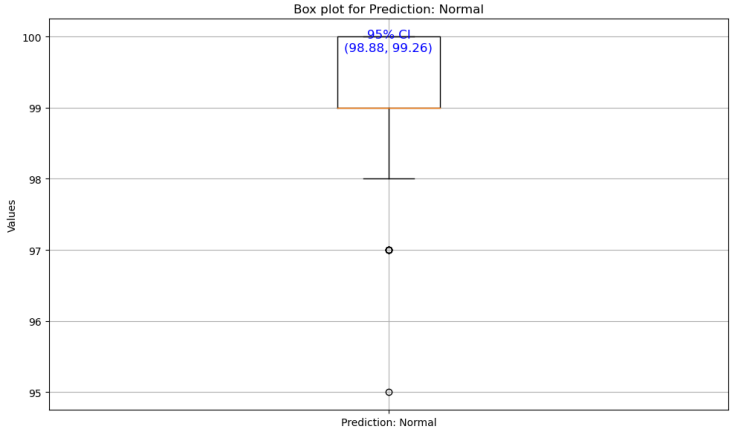


Figure 8. Scenario No. 1: *Path-scan* model associated with the observed Markov model.

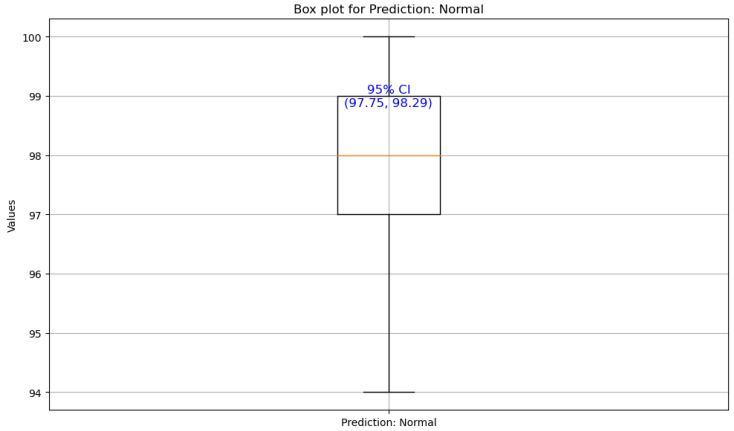


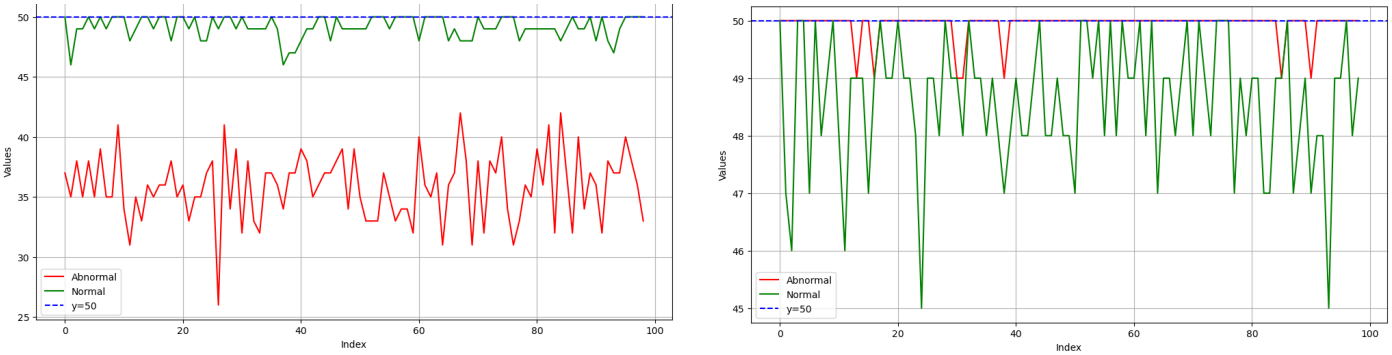
Figure 9. Scenario No. 1: Three-state model using the number of packets exchanged per minute.

The results of the first scenario indicate that both models have comparable sensitivities for detecting normal behaviors. However, the observed two-state Markov model, combined with the *Path-scan* method, slightly outperforms our three-state model for this specific task. Indeed, our three-state model shows a false positive rate ranging from 1.71% to 2.25%, whereas the observed two-state Markov model exhibits a false positive rate between 0.74% and 1.22%. These results highlight the effectiveness of the two-state model in identifying

normal behaviors while minimizing false positives. However, a more in-depth analysis is necessary to evaluate the sensitivity of these models under different data flow conditions, particularly in the presence of brute force attacks. Such an evaluation would help determine if the models maintain their effectiveness in more complex and varied scenarios.

In the second scenario, 100×100 tests were conducted, equally divided between tests reflecting the normal behavior of the studied path and tests reflecting abnormal behavior with one to three peaks per test. Following this data generation phase, the tests were randomly distributed into 100 series of 100 tests each, with each series containing exactly 50 normal tests and 50 abnormal tests. This random distribution creates realistic and varied conditions to evaluate the accuracy of prediction models by testing their ability to effectively distinguish between normal and abnormal behaviors in a given environment.

The results obtained for the different series for each model are presented in Figure 10. Figures 11 and 12, respectively, illustrate the box plots for the predictions of the *Path-scan* model associated with the observed Markov model and the three-state model using the number of packets exchanged per minute.



(**a**) (**b**)

Figure 10. Scenario No. 2: In green, the evolution of the number of times a model predicts "normal" behavior across the series of 50 tests is observed. In red, the evolution of the number of times a model predicts "abnormal" behavior across the series of 50 tests is presented. In blue, the expected values for "abnormal" and "normal" behaviors are depicted. (**a**) *Path-scan* model associated with the observed Markov model; (**b**) Three-state model using the number of packets exchanged per minute.

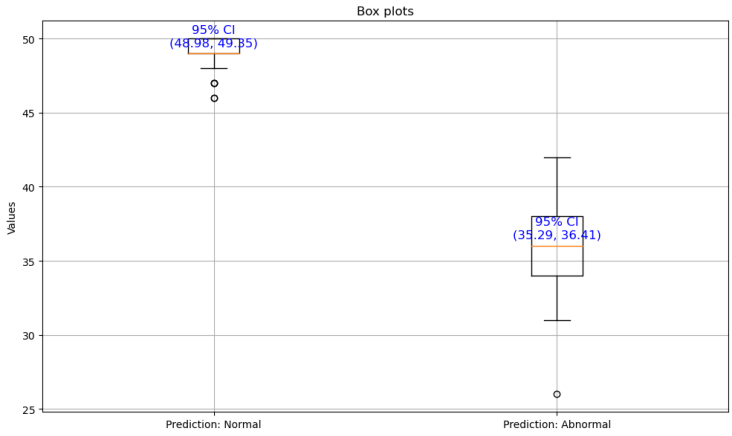


Figure 11. Scenario No. 2: *Path-scan* model associated with the observed Markov model.

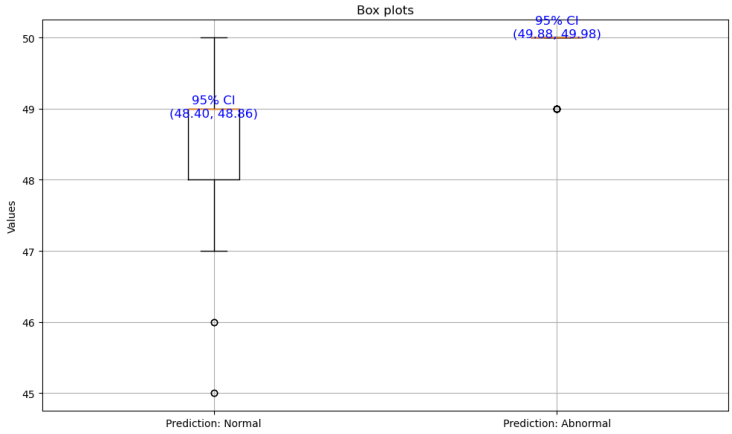


Figure 12. Scenario No. 2: Three-state model using the number of packets exchanged per minute.

The results of the second scenario indicate that both models exhibit comparable sensitivities for detecting normal behaviors, as observed in the first scenario. However, considering the injection of attacks such as brute force attacks and other types of attacks, our three-state model slightly outperforms the observed two-state Markov model combined with the *Path-scan* method for this specific task. Indeed, our three-state model shows a false negative rate ranging from 0.02% to 0.22%, whereas the observed two-state Markov model exhibits a false negative rate between 13% and 14%. These results suggest that the three-state model is more effective in detecting abnormal behaviors, with a lower likelihood of missing these behaviors (false negatives). This superiority can be attributed to the three-state model's ability to capture greater variability in data patterns, enabling it to better distinguish between normal and abnormal behaviors even in environments with highly active attacks.

These initial results indicate that the three-state model using the number of packets exchanged per minute demonstrates similar or even superior capability in detecting abnormal behaviors compared to the *Path-scan* model associated with the observed Markov model.

6. Conclusions

In this article, we propose an innovative approach for anomaly detection in computer networks, focusing on the analysis of *k-paths*. By utilizing the observed Markov model and generalized likelihood ratio test (GLRT), we have developed a robust methodology to characterize normal behavior and detect suspicious activities. Our results have demonstrated the effectiveness of our approach in identifying anomalies in network data flows, particularly when associated with significant changes. Through well-established hypothesis testing and rigorous parameter estimation of the model, we have provided accurate tools for early detection of malicious behaviors along these paths.

Ultimately, our approach makes a significant contribution to computer network security by providing advanced tools for detecting malicious activities. One research direction involves applying the model using the hidden Markov model [4], while considering the three states of the model and packet flows. Indeed, the model associated with the observed Markov model does not account for the distribution of packet counts exchanged. Additionally, it is crucial to consider work hours, downtime, and tool update times within a company to model a system capable of accounting for a more complex dimension. Another research direction concerns anomaly detection on each host, particularly supporting vulnerable path assets; queries (generating network flows) are indeed sent and received by hosts. Therefore, analyzing the behavior of individual hosts can provide valuable insights for attack detection. New processes and services, as well as file accesses, are potential data sources for identifying suspicious activities. However, collecting these data requires significant software engineering effort.

Author Contributions: Conceptualization: M.K., R.C., A.B. and C.B.; Methodology: M.K., R.C., A.B. and C.B.; Software: M.K. and R.C.; Validation: R.C., A.B. and C.B.; Formal Analysis: M.K.; Investigation: M.K.; Data Curation: M.K.; Writing—Original Draft Preparation: M.K.; Writing—Review and Editing: M.K.; Supervision: R.C., A.B. and C.B.; Funding Acquisition: C.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the LITIS laboratory, the LMAH laboratory, and Risk'n Tic, under a CIFRE agreement with the French National Association for Research and Technology (ANRT). We thank these entities for their financial support to cover the publication fees of this article.

Data Availability Statement: The data supporting the results reported in this study are publicly available at the following address: https://github.com/MamadouKasse1994/Donn-es-flux-g-n-r-es/blob/main/Data_set_zip.zip, accessed on 3 July 2024.

Conflicts of Interest: The authors declare no conflicts of interest. The funding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Ahmed, M.; Mahmood, A.N.; Hu, J. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **2016**, *60*, 19–31. [CrossRef]
2. Anwar, S.; Mohamad Zain, J.; Zolkipli, M.F.; Inayat, Z.; Khan, S.; Anthony, B.; Chang, V. From intrusion detection to an intrusion response system: Fundamentals, requirements, and future directions. *Algorithms* **2017**, *10*, 39. [CrossRef]
3. Ranshous, S.; Shen, S.; Koutra, D.; Harenberg, S.; Faloutsos, C.; Samatova, N.F. Anomaly detection in dynamic networks: A survey. *Wiley Interdiscip. Rev. Comput. Stat.* **2015**, *7*, 223–247. [CrossRef]
4. Neil, J.; Storlie, C.; Hash, C.; Brugh, A. Statistical Detection of Intruders within Computer Networks Using Scan Statistic. In *Data Analysis for Network Cyber-Security*; Imperial College Press: London, UK, 2013; pp. 71–104.
5. Ventre, D. *Cyberattaque et Cyberdéfense*; Lavoisier: Paris, France, 2011.
6. Algarni, S. Cybersecurity attacks: Analysis of "wannacry" attack and proposing methods for reducing or preventing such attacks in future. In *ICT Systems and Sustainability: Proceedings of ICT4SD 2020*; Springer: Singapore, 2021; Volume 1, pp. 763–770.
7. Grana, J.; Neil, J.; Wolpert, D.; Xie, D.; Bhattacharya, T.; Bent, R.W. A likelihood ratio anomaly detector for identifying within-perimeter computer network attacks. *J. Netw. Comput. Appl.* **2016**, *66*, 166–179. [CrossRef]
8. Li, L.; Lu, Y.; Yang, G.; Yan, X. End-to-End Network Intrusion Detection Based on Contrastive Learning. *Sensors* **2024**, *24*, 2122. [CrossRef] [PubMed]
9. MITRE Corporation. MITRE ATT&CK®: Enterprise Matrix. Available online: https://attack.mitre.org/matrices/enterprise/ (accessed on 12 March 2023).
10. Sharif, A. Qu'est-ce Qu'un Event Log? Available online: https://www.crowdstrike.fr/cybersecurity-101/observability/event-log/ (accessed on 17 May 2023).
11. Hofstede, R.; Čeleda, P.; Trammell, B.; Drago, I.; Sadre, R.; Sperotto, A.; Pras, A. Flow monitoring explained: From packet capture to data analysis with netflow and ipfix. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 2037–2064. [CrossRef]
12. Turcotte, M.J.M.; Kent, A.D.; Hash, C. Chapter 1: Unified Host and Network Data Set. In *Data Science for Cyber-Security*; World Scientific Publishing Europe Ltd.: London, UK, 2018; pp. 1–22.
13. Bondy, J.A.; Murty, U.S.R. *Théorie des Graphes*; Traduit de l'anglais par F. Havet; Springer: Berlin/Heidelberg, Germany, 2008.
14. Caplot, A. Analyse de Profils Audiologiques par Apprentissage Statistique. Doctoral Dissertation, Université de Montpellier, Montpellier, France, 2022.
15. Cogranne, R.; Retraint, F. A new tomography model for almost optimal detection of anomalies. In Proceedings of the 2013 IEEE International Conference on Image Processing, Melbourne, Australia, 15–18 September 2013; pp. 1461–1465.
16. Pinon, N.; Trombetta, R.; Lartizien, C. Détection d'anomalies dans l'image ou l'espace latent des auto-encodeurs basés sur des patchs pour l'analyse d'images industrielles. *arXiv* **2023**, arXiv:2307.02495.
17. Chandola, V. Anomaly Detection for Symbolic Sequences and Time Series Data. Doctoral Dissertation, University of Minnesota, Minneapolis, MN, USA, 2009.
18. Forrest, S.; Hofmeyr, S.A.; Somayaji, A.; Longstaff, T.A. A sense of self for unix processes. In Proceedings of the 1996 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 6–8 May 1996.
19. Kimura, T.; Ishibashi, K.; Mori, T.; Sawada, H.; Toyono, T.; Nishimatsu, K.; Watanabe, A.; Shimoda, A.; Shiomoto, K. Spatio-temporal factorization of log data for understanding network events. In Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April–2 May 2014.
20. Lévy-Leduc, C. Several approaches for detecting change-points in high-dimensional network traffic data. In *Data Analysis for CyberSecurity*; Imperial College Press: London, UK, 2013.
21. Evangelou, M.; Adams, N.M. Predictability of netflow data. In Proceedings of the 2016 IEEE Conference on Intelligence and Security Informatics (ISI), Tucson, AZ, USA, 28–30 September 2016; pp. 67–72.

22. Evangelou, M.; Adams, N.M. An anomaly detection framework for cyber-security data. *Comput. Secur.* **2020**, *97*, 101941. [CrossRef]
23. Larroche, C. Network-Wide Intrusion Detection through Statistical Analysis of Event Logs: An Interaction-Centric Approach. Doctoral Dissertation, Institut Polytechnique de Paris, Paris, France, 2021.
24. Zong, B.; Song, Q.; Min, M.R.; Cheng, W.; Lumezanu, C.; Cho, D.; Chen, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In Proceedings of the ICLR 2018 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
25. Anon. Détermination du seuil et de la limite de détection en spectrométrie gamma. 1989. Available online: https://inis.iaea.org/search/search.aspx?orig_q=RN:21054264 (accessed on 11 October 2023)
26. Casella, G.; Berger, R.L. *Statistical lnference*; Duxbury Press: Pacific Grove, CA, USA, 2002.