Towards attack detection in traffic data based on spectral graph analysis

Anonymous submission

No Institute Given

Abstract. Nowadays, cyberattacks have become a significant concern for individuals, organizations, and governments. These attacks can take many forms, and the consequences can be severe. In order to protect ourselves from these threats, it is essential to employ a range of different strategies and techniques like detection of patterns, classification of system behaviors against previously known attacks, and anomaly detection techniques. This way, we can identify unknown forms of attacks. Few of these existing techniques seem to fully utilize the potential of mathematical approaches such as spectral graph analysis. This domain is made of tools able to extract important topological features of a graph by computing its Laplacian matrix and its corresponding spectrum. This framework can provide valuable insights into the underlying structure of a network, which can be used to detect cyberthreats. Indeed, significant changes in the topology of the graph result in significant changes in the spectrum of the Laplacian matrix. For this reason, we propose here to address this issue by considering the network as a dynamic graph composed of nodes (devices) and edges (requests between devices), to study the evolution of the Laplacian spectrum, and to compute metrics on this evolving spectrum. This way, we should be able to detect suspicious behaviors which may indicate that an attack is occurring.

Keywords: cybersecurity · cyberattacks · anomaly detection · graph analysis · Laplacian Matrix · graph spectrum · graph topology.

1 Introduction

Cybersecurity is a critical concern for computer networks in the current century [10], where these networks are sensitive to countless types of attacks that can undermine their security and integrity. Several forms of attacks exist on computer networks [37], two common forms of attacks are the *Distributed Denial of Service* (DDoS) and *Denial of Service* (DoS). DoS and DDoS attacks can have significant impacts on the targeted systems and networks, causing downtime, service disruptions, and other adverse effects that can undermine their functionality and performance.

Cybersecurity community tries to detect them using different types of algorithms, often referenced as *anomaly detection* algorithms. They are mainly based on statistics [27,1,2], machine learning [33], deep learning [19,35,41,25], or even on graph neural networks [36,8,31,9](GNN). These algorithms are detailed and categorized in the next section.

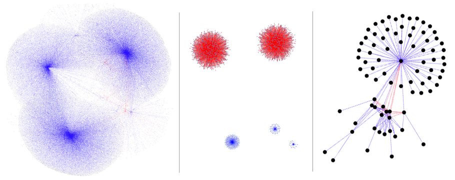

Fig. 1: IP-IP graph representation for Ton-IoT, IoT Healthcare Security, and Bot-IoT data sets

Traffic data can be represented with a graph [7], where nodes are computers and edges are the connections, we can thus model the traffic on a network as a *dynamic graph*. Based on various observations and analyses of datasets [7], it can be concluded that in star graphs mainly, the central node is typically regarded as a *server/hub*, while the leaves are considered as *devices/endpoints*. The edges connecting the *hub* to the *clients* are weighted based on the number of requests originating from each leaf to the central node at a given time t. Our proposition is then to use *spectral graph analysis* to be able to exhibit topological properties of the traffic data using its *Laplacian spectrum*. The core concept is to design metric functions that can assess the spectrum of traffic data, tracking changes in patterns over time. These functions quantify the various characteristics of the traffic, evaluating its impact on the overall spectrum and identifying any notable patterns or anomalies that emerge over time. However, these spectral metrics do not exist yet in the context of cybersecurity. That is why we propose in this paper a study of a pattern which is recurrent in many cybersecurity data sets (see Figure 1): the *star graphs*. So, we will empirically study the spectrum of a star graph depending on its parameters; we will then observe the spectrum of a graph made of several star graphs when they connect to each other; we will propose several spectral metrics that we estimate relevant in this kind of context; and we will compute these metrics in different scenarios (inspired of our data sets) to demonstrate their relevance to detect attacks.

There are the highlights: (1) we investigated the spectrum behaviour of star graphs and their derivatives, and (2) we developed four metrics that make systems administrators able to detect suspicious behaviors in network traffic.

2 Cybersecurity Background

Cyberspace has been experiencing an ongoing expansion as the amount of data being introduced into it continuously grows. Over time, cyberspace has increasingly become integrated into nearly every aspect of human life, including banking, hospitals, education, emergency services, and military operations. Consequently, the complexity of this digital domain has also increased. This heightened level of complexity has led to the emergence of cyber attacks as a significant threat. The typical categorization of cyber attacks can be classified according to their Purpose, Legal classification, Severity of involvement, Scope, and Network types as mentioned in [37]. Let's go deeper into the Purpose-based classification of cyber attacks, which includes DoS (Denial of Service) and DDoS (Distributed Denial of Service).

Denial of service (DoS) A DoS attack is one of the most common attack in the literature is Denial of Service (DoS) attacks, which pose a threat to the availability of a given service. These attacks may be targeted at individual clients or may seek to exhaust the network's resources, such as the Access Point (AP), thereby rendering the service inaccessible to all clients utilizing it [29,21]. As mentioned in [26] one type of DoS attack is characterized as *Data Flooding*, where the attacker endeavors to fully utilize the available bandwidth of a network, host, or device by sending a flood of massive data. This results in an overwhelming volume of data to be processed, causing the system to become incapacitated.

Distributed denial of service (DDoS) A DDoS attack employs numerous computers to execute a coordinated DoS attack on one or more targets. The perpetrator leverages client/server technology to amplify the impact of the DoS attack by utilizing the resources of multiple unsuspecting accomplice machines, which serve as the attack platforms. [26,22]. DDoS attacks do not aim to penetrate the victim's system, rendering traditional security defenses ineffective. The primary objective of a DDoS attack is to inflict harm on a target for personal motives, financial gain, or to increase notoriety [34,15].

3 State-of-the-Art

For *anomaly detection* [2], main approaches are *statistical ones* [3] and ML-based ones [5]. Among the most recent statistical approaches, we can refer to a real-time network anomaly-detector called *ReTiNA* [28] and an unsupervised three-stage framework for detecting anomalous network behaviour on-the-fly. We can also cite [18] presenting a scalable, principled, probability-based technique for detecting outlying connectivity behavior. Traditional systems use elementary statistics techniques and are often inaccurate [17]. For this reason, a ML-based approach called CAMLPAD has been proposed [17]. This system streamlined, holistic approach begins with retrieving a multitude of different species of cybersecurity data in real time using ElasticSearch, then runs several ML algorithms. The calculated anomalies are assigned an outlier score which serves as an indicator for whether an alert should be sent to the system administrator that there are potential anomalies in the network. The ML-based techniques are supervised algorithms using Logistic Regression, Decision Trees, Naive Bayes techniques, Support Vector Machines, k-Means, k-Nearest Neighbors, and Random Forests.

In network security, there are not much labeled data to train efficient classifiers [7]. Furthermore, existing labeled data are often specific to the context and thus may not be useful for other applications. Hopefully, graph-based machine learning techniques have the potential to make significant impact in the next-generation cybersecurity systems. Walk-based sampling is a technique whereby graph structured data is sampled via walks through the graph (see [32,16]). In the simplest case, random walks are performed, which converts the unstructured graph data into structured sequences of nodes and edges which can then be processed by traditional ML techniques. In [8], they detect lateral intrusions thanks to unsupervised learning of graphs. The possible tracks [7] to handle dynamicity of networks are time series graph learning [31]. In [9], they propose StrGNN, an end-to-end structural temporal GNN model for detecting anomalous edges in dynamic graphs.

Deep learning [41,25] has received widespread concerns in the graph data field. Kipf *et al.* [20] proposed a generalisation method of *Graph Convolutional Networks* (GCN's), currently the best choice for graph data learning tasks. HyperGCN [40] generalized the simple graph convolution operation of GCN to the hypergraph domain by using hypergraph Laplacian to capture more complex or beyond pairwise relationships between nodes. Graph attention networks [38] encode the hidden representations of each node in the graph by introducing the self-attention mechanism to attend over its neighbors. Liu *et al.* [24] propose a weakly supervised multi-label image classification framework based on GCN with learning the semantic label co-occurrence in an image.

When a network is represented by a graph, we can extract topological properties thanks to spectral graph analysis [14] using their Laplacian spectrum. Several features follow: the number of eigenvalues equal to zero (the number of connected components of the graph), the greatest eigenvalue (bipartiteness [4]), etc. Robustness [23] to edge rewiring of this spectrum has been shown in the sense that *a small modification of the graph weights implies a small change in the graph spectrum*. Several approaches utilizing spectral analysis have been developed for cybersecurity applications. In [13], they detect changes in the periodicity of Transmission Control Protocol (TCP) packet transport associated with round-trip times: using power spectral density (PSD), they can defend against denial of service (DoS) attacks. Another approach [39] employs graph theory, diffusion, and spectral methods to analyze forensic evidence. This method demonstrates the potential of graph-spectral and kernel-based methods in extracting structural characteristics of the evidence graph. By modeling the propagation of suspicion in the attack scene using heat diffusion, this approach extracts attack scenarios and creates attack case profiles. A method [11] called MC-GPCA has been introduced to reduce the complexity of a single graph by breaking it down into several centrality features and reducing their correlation. In the same paper, another method called MC-GDL performs dictionary learning on a group of graphs using multiple centrality features. Both methods utilize spectral decompositions using singular value decomposition (SVD). In [42], a fraud detection framework is developed to identify various cybersecurity attacks by analyzing the eigenvalues and eigenvectors of the adjacency matrix. This approach studies how to identify attackers by characterizing their distributions in the spectral space. In [6], they discover static graphs patterns, model dynamic graphs, cluster evolving graphs, and detect anomalies in dynamic graphs. Last, in [12], they merge discrete Fourier transform (DFT) with a hypothesis testing framework to address shrew attacks.

4 Mathematical background

Let us recall the background necessary to handle star graphs. A (finite) *graph* G is a pair (V, E) with $V = \{v_i\}_{i \in [1,n]}$ the set of *vertices*, and $E \subseteq V \times V$ the set of *edges*. We can represent G by its *adjacency matrix* $A = (a_{i,j})_{i,j}$, with $a_{i,j}$ the value of A at row i and column j. In the binary case, each $a_{i,j}$ belongs to $\{0, 1\}$ and $a_{i,j} = 1$ iff $(v_i, v_j) \in E$. In the present paper, $a_{i,j}$ belongs to $\mathbb{R}^+$; we can then call this term the *weight* of the edge (i, j) and it represents "how much" v_i is adjacent to v_j. Here, we will treat only *undirected graphs*, that is, $(v_i, v_j) \in E$ is equivalent to $(v_j, v_i) \in E$. The direct consequence is that $a_{i,j} = a_{j,i}$ for any $i, j \in [1, n]$. The *degree matrix* $D = (d_{i,j})_{i,j}$ is a diagonal matrix where $d_{i,i}$ is equal to $\sum_{j \in [1,n]} a_{i,j}$. The *Laplacian matrix* [14] is then $L = D - A$. Since L is a squared matrix of rank n, we can compute its set of *eigenvalues* $\{\lambda_1, \ldots, \lambda_n\}$. The *spectrum* Λ of L is the set of eigenvalues of L sorted in the increasing order. When G is made of several connected components (CC's), the spectrum of G is the sorted concatenation of the spectra of the CC's of G.

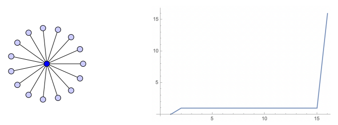

Fig. 2: G_*^{15} and its spectrum.

A *star graph* is a special type of tree where a "central" node is connected to nodes of degree 1 (see Figure 2); a node which is not central is thus a *leaf* of this graph. We will denote a star graph using the following notations: $G_*^{\mathfrak{D}} = (V_*^{\mathfrak{D}}, E_*^{\mathfrak{D}})$ with $V_*^{\mathfrak{D}} = \{c, \ell_1, \ldots, \ell_{\mathfrak{D}}\}$, $E_*^{\mathfrak{D}} = \bigcup_{i=1}^{\mathfrak{D}} \{(c, \ell_i)\}$. Note that $\mathfrak{D} \in \mathbb{N}$ is the degree of the central node c.

5 Studying star graphs

Let us now introduce our results about star graphs and derivatives.

5.1 Spectral properties of an isolated star graph

Let $G_*^{\mathfrak{D}}$ be some graph star whose central node has a degree $\mathfrak{D} \geq 0$ and whose each edge has the same weight $\mathcal{W}_{SG} > 0$. We define λ_1, λ_2, and λ_{max} as the first, second, and last eigenvalues in $\mathit{\Lambda}$ sorted from the lowest to the greatest eigenvalue. The second lowest value in $\mathit{\Lambda}$ is also known as the *algebraic connectivity* (AC). Since the exact formula of the Laplacian spectrum (see Figure 2) of a star graph is $(0, \mathcal{W}_{SG}, \ldots, \mathcal{W}_{SG}, (\mathfrak{D}+1) * \mathcal{W}_{SG})$, $\lambda_1 = 0$, $\lambda_2 = \mathcal{W}_{SG}$ and $\lambda_{max} = (\mathfrak{D}+1) * \mathcal{W}_{SG}$ as soon as the size of L is 3 or more.

5.2 Spectral properties of combinations of star graphs

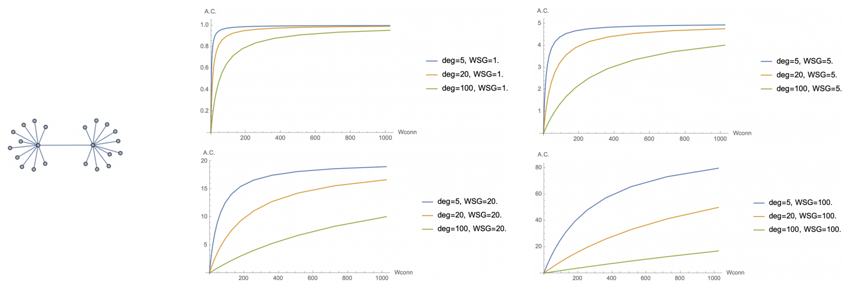

Fig. 3: AC as a function of the "connecting weight" $\mathcal{W}_{conn}$ in $G_{*,2}^{10}$. When $\mathcal{W}_{conn}$ increases, λ_2 tends to $\mathcal{W}_{SG}$, and when $\mathfrak{D}$ or when $\mathcal{W}_{SG}$ increases, this convergence is slower.

Let us merge two disjoint star graphs isomorphic to $G_*^{\mathfrak{D}}$ into $(V_{*,2}^{\mathfrak{D}}, E_{*,2}^{\mathfrak{D}})$ to obtain $G_{*,2}^{\mathfrak{D}}$ (see Figure 3) with $V_{*,2}^{\mathfrak{D}} = \{c^1, \ell_1^1, \ldots, \ell_{\mathfrak{D}}^1\} \cup \{c^2, \ell_1^2, \ldots, \ell_{\mathfrak{D}}^2\}$, and $E_{*,2}^{\mathfrak{D}} = \bigcup_{i=1}^{\mathfrak{D}} \{(c^1, \ell_i^1)\} \cup \bigcup_{i=1}^{\mathfrak{D}} \{(c^2, \ell_i^2)\} \cup \{(c^1, c^2)\}$. The weight of the edge (c_1, c_2) is set at $\mathcal{W}_{conn} \in \mathbb{R}^+$ and the ones of the remaining edges are set at $\mathcal{W}_{SG} > 0$. The goal in this section is to observe the behaviour of the spectrum of $G_{*,2}^{\mathfrak{D}}$ when $\mathfrak{D}$, $\mathcal{W}_{SG}$, and $\mathcal{W}_{conn}$ vary. The observations following from Figure 3 are that the bigger the ratio $\frac{\mathcal{W}_{conn}}{\mathcal{W}_{SG}}$ is, the more the AC λ_2 tends rapidly to $\mathcal{W}_{SG}$. Also, the bigger $\mathfrak{D}$ is, the more λ_2 tends more slowly to $\mathcal{W}_{SG}$.

6 From simulations of threats to metrics

To be able to define metrics, we need to simulate threats and observe how the spectrum behaves. So, let us generalise the star graph formula to any $\mathcal{N} \in \mathbb{N}^*$ as the graph made of a branch made of $\mathcal{N}$ nodes and where $\mathfrak{D}$ additional edges start from each of its nodes.

Scenario 1 (see Figure 4) is the following: we propose here to start from a graph made of $\mathcal{N} = 8$ separated star graphs, all isomorphic to G_*^{10}. During this

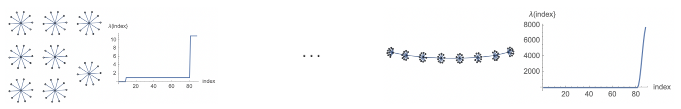

Fig. 4: Scenario 1: at each step, we connect two components with an edge of weight $\gg 1$ (200 in our case, when $\mathcal{W}_{SG} = 1$).

experiment, we are going to connect the different central nodes (the "servers") step by step, as if an intrusion were occurring. The main observation is that the information about the traffic is mainly stored in the $\mathcal{N}$ first and the $\mathcal{N}$ last values of the spectrum, the remaining eigenvalues staying equal to the star graph weight $\mathcal{W}_{SG} = 1$ during all the experiment.

Observations relative to the $\mathcal{N}$ lowest values of the spectrum: During this scenario, we observed that, since we start from $\mathcal{N}$ CC's, we have then $\mathcal{N}$ zeros in the spectrum of the total graph. Since we connect two CC's at each step, we loose one zero in the spectrum. The lowest eigenvalue being not equal to 0 corresponds to the smallest non-zero CC's AC's. This value is placed at position $(\mathcal{N} - M(t) + 1)$ where $M(t)$ is the number of connections we have made at time t. Last, only one zero remains at the end of the scenario since the graph is connected.

Observations relative to the $\mathcal{N}$ highest values of the spectrum: at the beginning, we have a plateau at the end of the spectrum because the graph is the concatenation of $\mathcal{N}$ separated subgraphs maximal eigenvalues. While connections are occurring, the number of plateaus decreases until it vanishes. During this same process, the maximal eigenvalue increases until it reaches a very high value of 8000 compared to the connection value of 200.

Metric 1 For the first metric, we propose to introduce a function which increases when interconnections occur in the network. We call it *connectedness* and we define it as $\mu_1(t) = \frac{\exp \frac{1}{\mathcal{Z}(t)}}{\exp(1)}$ where $\mathcal{Z}(t)$ is the number of zeros in the spectrum. Therefore, the more μ_1 tends to 1, the more an intrusion may be possible. This metric tends to $\exp^{-1} \approx 0.367$ when $\mathcal{Z}(t)$ tends to $+\infty$, and it tends to 1 when $\mathcal{Z}(t)$ tends to 1.

Metric 2 To be able to take into consideration at the same time interconnections and connecting edge weights (see Figure 3), we propose to introduce our second metric: $\mu_2(t) = \sum_{p=2}^{p=\mathcal{N}} (\exp^{\lambda_p(t)} - 1)$. This metric is influenced by the occurrence of connections as well as the weight of those connections. When there are many connections with high connecting weights, the *flood value* increases accordingly.

Metric 3 Another possible way to take into consideration interconnections and edge weights is to consider the maximal eigenvalues. For this reason, we introduce a third metric called the *wiringness*: $\mu_3(t) = \sum_{p=n-\mathcal{N}+1}^{p=n} \lambda_p(t)$. It always increases when connections occur and its slope across time depends on the packets sizes (the weights) and on the number of connections. It generally increases when the transmitted packets sizes are big and when the requests are numerous.

Metric 4 The fourth and last introduced metric $\mu_4(t)$ is called *asymmetry*, and is equal to $\mu_4(t) = \#\{k \in [2, n] \ ; \ \Lambda(t)[k] - \Lambda(t)[k-1] > \varepsilon\}$. It corresponds to the number of variations of $\Lambda(t)$ when the index k goes from 2 to n. When we have several identical patterns in the network, it tends to be low. At the contrary, when connections occur, the smoothness of $\Lambda(t)$ increases and thus the asymmetry increases too, which is a sign of possible threat. The value $\varepsilon = 10^{-12}$ is used to tackle the approximation errors.

Computational complexity Let us assume that n is the number of nodes in G, and that the adjacency matrix A of G is given. Computing D needs at most n operation n times, and computing L needs n^2 operations. The spectrum of L can be computed in $O(n^3 + (n \log^2(n)) \log(b))$ according to [30] with 2^{-b} the relative error bound. Our metrics being in $O(n)$ when the spectrum is given, the total complexity of our metrics is then in $O(n^3 + (n \log^2(n)) \log(b))$.

7 Detecting attacks

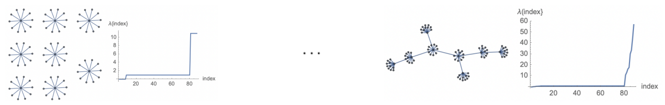

Fig. 5: Scenario 2 made of suspicious connections across all the servers.

In the sequel, we propose two new scenarios, still with a fixed number of nodes, which will allow us to estimate the threat from traffic data. Scenario 2 is the following: we start as usual from $\mathcal{N} = 8$ disconnected star graphs. At each time, we connect two servers with a weight equal to 10, which is much weaker than in the previous experiment. We say that this scenario is *suspicious* in the sense that we consider that it is not usual that many servers are interconnecting in a short time, but it can happen without being an attack. Scenario 3 is the following: we start as usual from $\mathcal{N} = 8$ disconnected star graphs. During each step of the experiment, we will establish a connection between a client node, which is a non-central node, and a server node, which is central in one of the initial star graphs. The corresponding edge in the adjacency matrix will be assigned

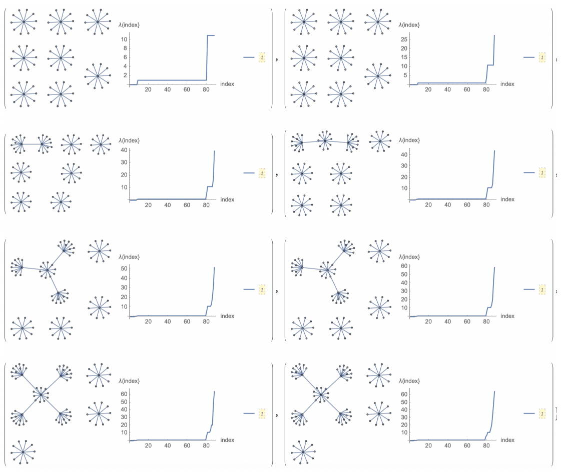

Fig. 6: Scenario 3 made of normal connections across all the servers.

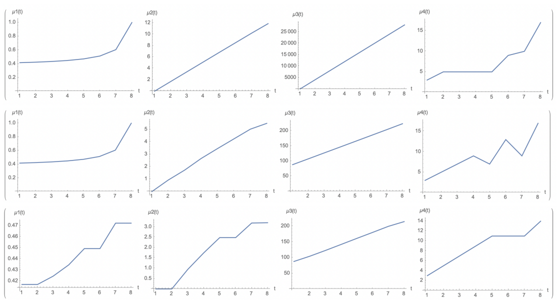

Fig. 7: From left to right, the four dynamic metrics μ_1 to μ_4 in Scenarios 1 to 3.

a weight of 10 to represent this connection. Alternatively, we may establish a connection between two server nodes, also with a weight of 10 assigned to the corresponding edge. We say that this scenario is *normal.*

Let us recall that the bigger the metric, the bigger the risk of a threat. If we observe the dynamic metrics, we understand that, compared to a normal case, the threatening case is easily detected thanks to μ_2 which goes from 3 (normal case) to 6 (suspicious case) and 12 (threatening case). Moreover, we see that this impression is supported by μ_1 showing that we progressively interconnect the different servers in the network (we have 0.475 in the normal case vs. 1.0 in suspicious and threatening cases). Furthermore, μ_3 is dramatically increasing (from 200 to 25000) in the threatening case, which means that the highest eigenvalues increase rapidly when threat is present. Last, μ_4 goes from 14 in the normal case to 18 in the suspicious and threatening cases. It is important to remark that the more we have big values among the four dynamic metrics, the most we have to consider that the situation is critical (one high metric is rarely sufficient to decide).

8 Limitations

We were not able to handle the attack called *Man in the Middle* (MitM) where the attacker intercepts and listen to any communication and exchange of data manipulated between two endpoints. This is due to the fact that spectrum variations are too low. Moreover, we assumed in this paper that the number of nodes is a constant, which is not always true in practice.

9 Conclusion

In this paper, we show how much using the Laplacian spectrum of a graph is promising to detect attacks: Laplacian spectrum contains a lot of information about the network topology and makes us able to detect several threats in a dynamic and interpretable way thanks to the proposed metrics. As future works, we propose to develop metrics specific to the other recurrent patterns we can find in cybersecurity traffic datasets, so that our study will be more complete. We will also investigate directed graphs, since the direction of data traffic is important to identify sources of requests.

References

1. Adams, N.M., Heard, N.A.: Data analysis for network cyber-security. World Scientific (2014)
2. Adams, N.M., Heard, N.A.: Dynamic networks and cyber-security, vol. 1. World Scientific (2016)
3. Ahmed, M., Mahmood, A.N., Islam, M.R.: A survey of anomaly detection techniques in financial domain. Future Generation Computer Systems **55**, 278–288 (2016)
4. Bauer, F., Jost, J.: Bipartite and neighborhood graphs and the spectrum of the normalized graph laplacian. arXiv preprint arXiv:0910.3118 (2009)
5. Bhattacharyya, D.K., Kalita, J.K.: Network anomaly detection: A machine learning perspective. Chapman and Hall/CRC (2019)
6. Bilgin, C.C., Yener, B.: Dynamic network evolution: Models, clustering, anomaly detection. IEEE Networks **1** (2006)
7. Bowman, B., Huang, H.H.: Towards next-generation cybersecurity with graph ai. ACM SIGOPS Operating Systems Review **55**(1), 61–67 (2021)
8. Bowman, B., Laprade, C., Ji, Y., Huang, H.H.: Detecting lateral movement in enterprise computer networks with unsupervised graph {AI}. In: 23rd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2020). pp. 257–268 (2020)
9. Cai, L., Chen, Z., Luo, C., Gui, J., Ni, J., Li, D., Chen, H.: Structural temporal graph neural networks for anomaly detection in dynamic graphs. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 3747–3756 (2021)

10. Cavelty, M.D.: Cyber-security. In: The Routledge handbook of new security studies, pp. 154–162. Routledge (2010)
11. Chen, P.Y., Choudhury, S., Hero, A.O.: Multi-centrality graph spectral decompositions and their application to cyber intrusion detection. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 4553–4557. IEEE (2016)
12. Chen, Y., Hwang, K.: Collaborative detection and filtering of shrew ddos attacks using spectral analysis. Journal of Parallel and Distributed Computing **66**(9), 1137–1151 (2006)
13. Cheng, C.M., Kung, H., Tan, K.S.: Use of spectral analysis in defense against dos attacks. In: Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE. vol. 3, pp. 2143–2148. IEEE (2002)
14. Chung, F.R.: Spectral graph theory, vol. 92. American Mathematical Soc. (1997)
15. Douligeris, C., Mitrokotsa, A.: Ddos attacks and defense mechanisms: classification and state-of-the-art. Computer networks **44**(5), 643–666 (2004)
16. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
17. Hariharan, A., Gupta, A., Pal, T.: Camlpad: Cybersecurity autonomous machine learning platform for anomaly detection. In: Future of Information and Communication Conference. pp. 705–720. Springer (2020)
18. Heard, N., Rubin-Delanchy, P.: Network-wide anomaly detection via the dirichlet process. In: 2016 IEEE Conference on Intelligence and Security Informatics (ISI). pp. 220–224. IEEE (2016)
19. Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. Eai Endorsed Transactions on Security and Safety **3**(9), e2 (2016)
20. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
21. Kolias, C., Kambourakis, G., Stavrou, A., Gritzalis, S.: Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. IEEE Communications Surveys & Tutorials **18**(1), 184–208 (2015)
22. Kumar, G.: Denial of service attacks–an updated perspective. Systems science & control engineering **4**(1), 285–294 (2016)
23. de Lange, S., de Reus, M., Van Den Heuvel, M.: The laplacian spectrum of neural networks. Frontiers in computational neuroscience **7**, 189 (2014)
24. Liu, Y., Chen, W., Qu, H., Mahmud, S.H., Miao, K.: Weakly supervised image classification and pointwise localization with graph convolutional networks. Pattern Recognition **109**, 107596 (2021)
25. Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A.: Kitsune: an ensemble of autoencoders for online network intrusion detection. arXiv preprint arXiv:1802.09089 (2018)
26. Mitrokotsa, A., Douligeris, C.: Ddos attacks and defense mechanisms: a classification. In: 3rd IEEE International Symposium on Signal Processing & Information Technology (ISSPIT 2003). pp. 190–193 (2003)
27. Neil, J., Hash, C., Brugh, A., Fisk, M., Storlie, C.B.: Scan statistics for the online detection of locally anomalous subgraphs. Technometrics **55**(4), 403–414 (2013)
28. Noble, J., Adams, N.: Real-time dynamic network anomaly detection. IEEE Intelligent Systems **33**(2), 5–18 (2018)

29. Oseni, A., Moustafa, N., Creech, G., Sohrabi, N., Strelzoff, A., Tari, Z., Linkov, I.: An explainable deep learning framework for resilient intrusion detection in iot-enabled transportation networks. IEEE Transactions on Intelligent Transportation Systems **24**(1), 1000–1014 (2023). https://doi.org/10.1109/TITS.2022.3188671
30. Pan, V.Y., Chen, Z.Q.: The complexity of the matrix eigenproblem. In: Proceedings of the thirty-first annual ACM symposium on Theory of computing. pp. 507–516 (1999)
31. Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T., Leiserson, C.: Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 5363–5370 (2020)
32. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710 (2014)
33. Salloum, S.A., Alshurideh, M., Elnagar, A., Shaalan, K.: Machine learning and deep learning techniques for cybersecurity: A review. In: AICV. pp. 50–57 (2020)
34. Stein, L.D.: Web security. Addison-Wesley, Massachusetts **26**, 1–4 (1998)
35. Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R., Ghogho, M.: Deep learning approach for network intrusion detection in software defined networking. In: 2016 international conference on wireless networks and mobile communications (WINCOM). pp. 258–263. IEEE (2016)
36. Teru, K., Denis, E., Hamilton, W.: Inductive relation prediction by subgraph reasoning. In: International Conference on Machine Learning. pp. 9448–9457. PMLR (2020)
37. Uma, M., Padmavathi, G.: A survey on various cyber attacks and their classification. Int. J. Netw. Secur. **15**(5), 390–396 (2013)
38. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
39. Wang, W., Daniels, T.E.: Diffusion and graph spectral methods for network forensic analysis. In: Proceedings of the 2006 workshop on New security paradigms. pp. 99–106 (2006)
40. Yadati, N., Nimishakavi, M., Yadav, P., Louis, A., Talukdar, P.: Hypergcn: Hypergraph convolutional networks for semi-supervised classification. arXiv preprint arXiv:1809.02589 **22** (2018)
41. Yin, C., Zhu, Y., Fei, J., He, X.: A deep learning approach for intrusion detection using recurrent neural networks. Ieee Access **5**, 21954–21961 (2017)
42. Ying, X., Wu, X., Barbara, D.: Spectrum based fraud detection in social networks. In: Proceedings of the 17th ACM conference on Computer and communications security. pp. 747–749 (2010)

10 Supplementary Materials

In this section, we add several scenarios to show how our metrics make us able to detect that something suspicious is happening on the network. Note that we consider in the first four scenarios that a weight lower than 4 as “normal”, and a weight around 30 or more as really suspicious. Also, before we indicate what is a huge value for a metric, we have to proceed to the normal requests in the following two network configurations.

10.1 Additional Scenario 1

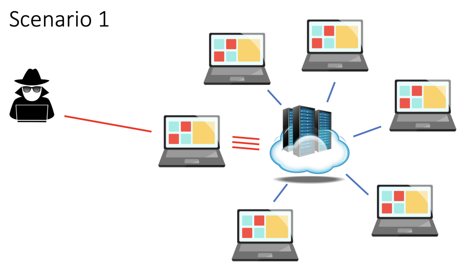

Fig. 8: Additional scenario 1

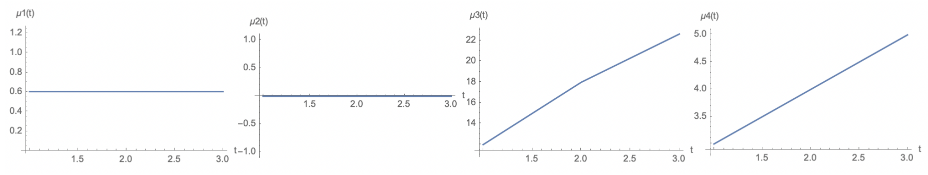

Fig. 9: Behaviour of the metrics during a normal traffic

Calibration of the metrics To calibrate our metrics, we realize the following scenario: one user or a device is already connected to the network, where there is a connection of communication with the server (see 9). Some time later, another user does the same. We observe that μ_1 and μ_2 stay flat. At the same time, μ_3 and μ_4 reach the values 22 and 5 respectively; this values will be considered as our references of a normal traffic in the following scenario.

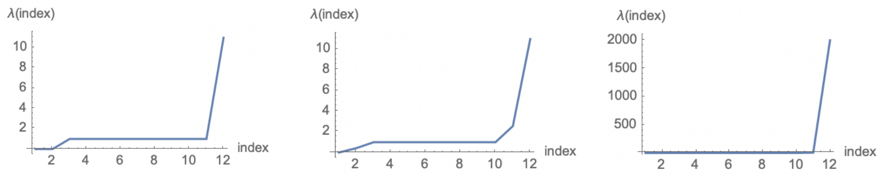

Fig. 10: Evolution of the spectrum during additional Scenario 1

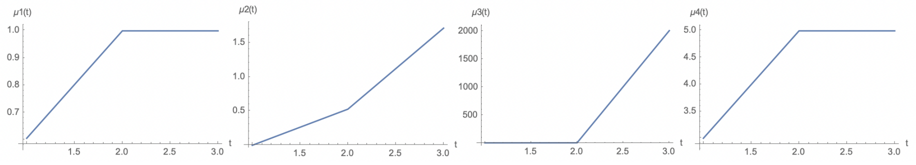

Fig. 11: Dynamic metrics during additional Scenario 1

The attack In the scenario described in Figure 8, we can see that a hacker will be connecting to a device that is already connected to the main server, trying to deny the access of any device on the network to the server. Observing Figure 10, we can see that during the scenario, the spectrum is changing, but we cannot be sure that something suspicious is happening. Looking at Figure 11 describing the dynamic metrics evolving during the experiment, μ_4 shows that we are loosing the symmetry of the network, but it behaves just like the case of calibration, for that it is not sufficient to confirm anything. However, we can observe that μ_1 reaches the value 1 at step 2 (due to the connection of the hacker to one of the devices). This sign shows a potential threat. Then, at step 3, μ_2 increases and μ_3 reach a huge value (compared to calibration behaviour), which confirms that someone is requesting too many packets. There is almost no doubt that there is an attack, and that the system administrator must intervene.

10.2 Additional Scenario 2

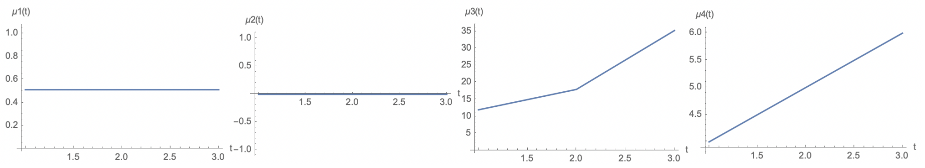

Fig. 12: Behaviour of the metrics during a normal traffic

Calibration of the metrics To calibrate our metrics, we realize the following scenario: some users or devices are already connected to the network, where there is a connection of communication with the server (see 12). On the same network we have also some devices connected to a hub. We can observe that μ_1 and μ_2 stay flat (like in the previous calibration). At the same time, μ_3 and μ_4 reach the values 35 and 6 respectively; this values will be considered as our references of a normal traffic in the coming scenario.

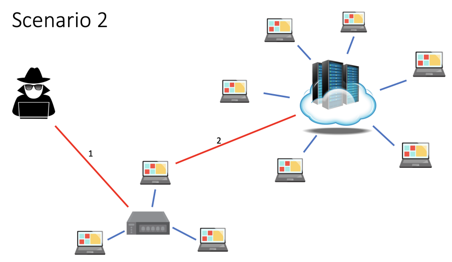

Fig. 13: Additional scenario 2

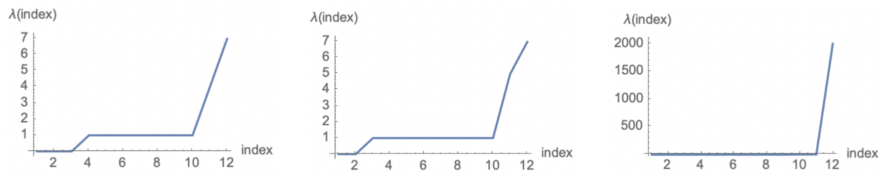

Fig. 14: Evolution of the spectrum during additional Scenario 2

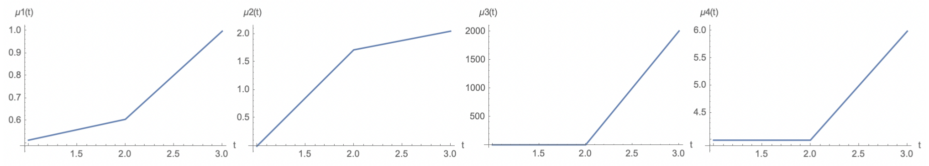

Fig. 15: Dynamic metrics during additional Scenario 2

The attack In this scenario (see Figure 13), a hacker connects directly to the hub, where he is now able to access all devices on the network, then goes through one of the devices to reach the server where he is able to flood many requests aiming to deny access to it (here we set the weight of the connection at 50, which is considered to be very huge for this network). The spectrum apparently progress in the same manner as before (see Figure14). However, the metrics show that the attack is completely different (see Figure 15). It is very clear to observe the evolution of μ_2 which increases simultaneously. Also, metrics μ_1 (reaching the value 1) and μ_4 (which is increasing) show suspicious behaviour. Last, μ_3 is multiplied by a factor equal to 10, which confirms the attack.

10.3 Additional Scenario 3

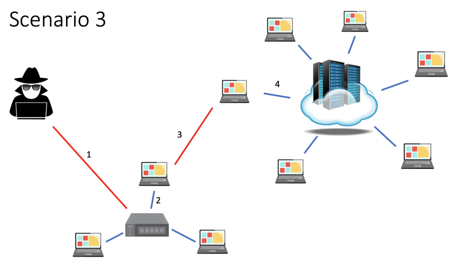

Fig. 16: Additional scenario 3

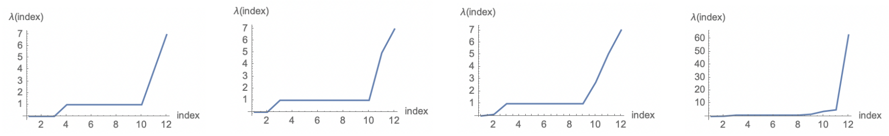

Fig. 17: Evolution of the spectrum during additional Scenario 3

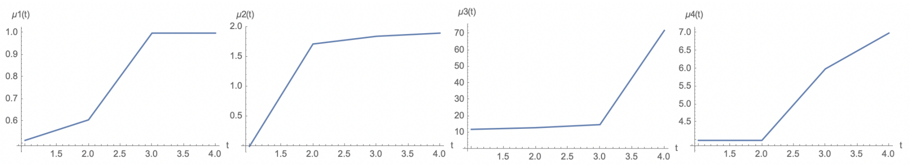

Fig. 18: Dynamic metrics during additional Scenario 3

In this scenario (see Figure 16), a hacker connects directly to a hub, then goes through one device in the hub-network to reach another device on the server-network. Then, he starts flooding the pathway reaching the server to deny any access from any user on the network to it (this time the connection is equal to 29). The global behaviour looks like the previous experiment (see Figure17 and Figure 18). Indeed, the difference is that the connection to the server is not direct. However, the metrics still detect the threat.

10.4 Additional Scenario 4

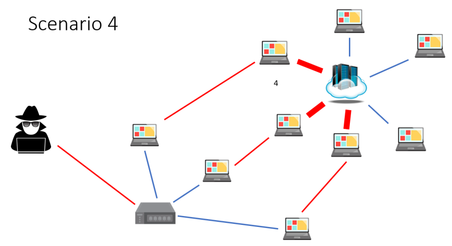

Fig. 19: Additional scenario 4

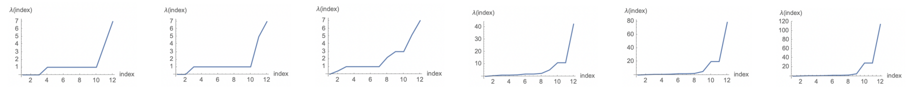

Fig. 20: Evolution of the spectrum during additional Scenario 4

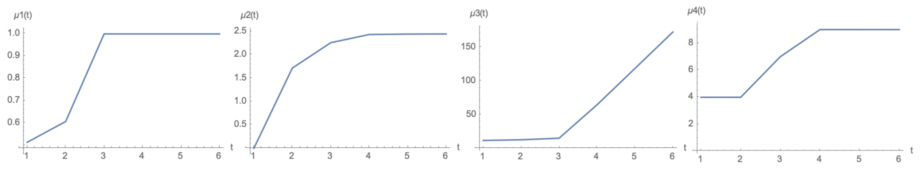

Fig. 21: Dynamic metrics during additional Scenario 4

In this scenario (see Figure 19), a hacker connects directly to a hub, then he controls the three devices of this hub-network, after that he pass through them to access the other devices of the server-network aiming to flood the server and deny any request using different packet-requests of 99, 200, and then 200 at each step. The global behaviour of the dynamic metrics looks like the previous experiment but the values are bigger (see Figure 20 and Figure 21). Our metrics are once more very useful to detect threat.

10.5 Additional Scenario 5

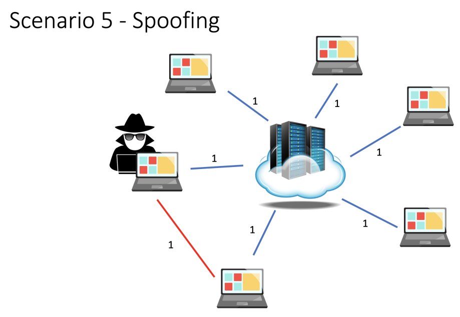

Fig. 22: Additional scenario 5

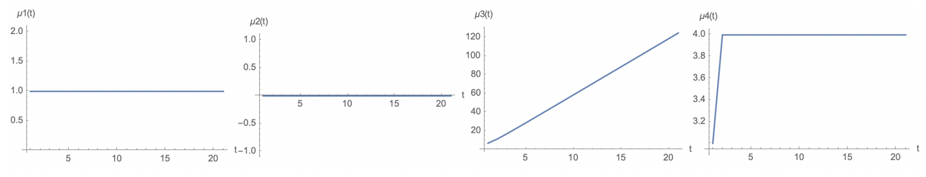

Fig. 23: Dynamic metrics when the networks traffic is normal

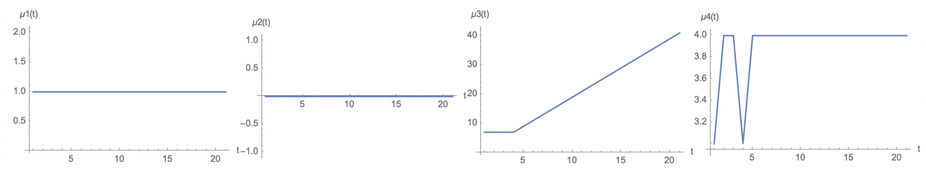

Fig. 24: Dynamic metrics during additional Scenario 5 (Spoofing)

In this scenario (see Figure 22), a hacker spoof one of the server-network devices which is already connected to the network (where the attacker here is acting as if he is the user with the same IP, where he directly connects to another device. The traffic remains "normal" in the sense that packets are of size 1, however this process creates a cycle in the graph (the topology changes). The global behaviour of the dynamic metrics is very similar to that of the calibration, except that $\mu_4(t)$ oscillates at the beginning of the attack, which is clearly an alarm showing that there is changes in the topology (see Figure 24). Note that this case is particularly subtle to detect, since only one metric shows that a suspicious behaviour is occurring; but in all the other presented scenarios, several metrics showed that there is a suspicious behaviour on the network.

10.6 Additional Scenario 6

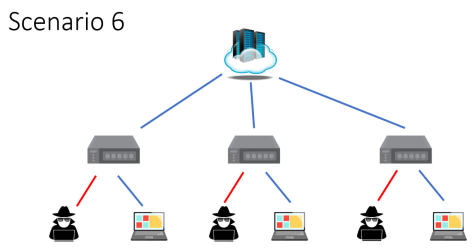

Fig. 25: Additional scenario 6

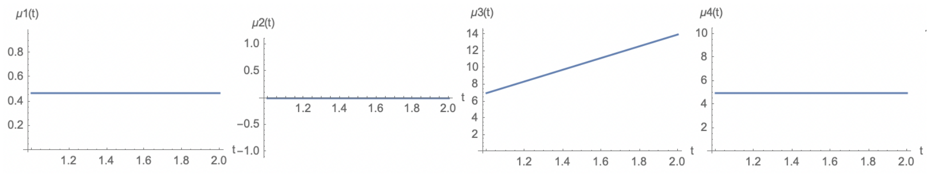

Fig. 26: Dynamic metrics when the networks traffic is normal

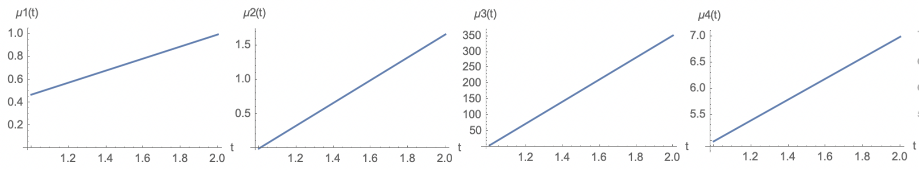

Fig. 27: Dynamic metrics during additional Scenario 6

In this scenario (see Figure 25), there is a network comprising a central server and three hubs, each of which is connected to a host device that is responsible for requesting data from the server. However, the network is targeted by three attackers who infiltrate the network and connect to each of the hubs. These

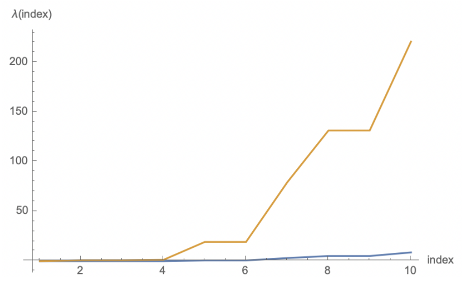

Fig. 28: Spectra of a normal traffic (in blue) and during the attack (in orange)

attackers launch a parallel attack, flooding the network with traffic to deny access to the server for the legitimate hosts on the network. Every metric reaches very high values (see Figure 28 compared to Figure 26), there is no doubt about what is happening. Note that in this scenario, we could have detect the attack even without our metrics (see Figure 28).