Evaluating Cybersecurity Architecture in ICS/OT Environments Using Graphs and Hypergraphs

Zach Bovaird

July 3, 2024

Abstract

In Industrial Control Systems (ICS) and Operational Technology (OT) environments, ensuring robust cybersecurity architecture is critical. This paper explores the application of graph and hypergraph theory in evaluating cybersecurity within these environments. Critical metrics such as adjacency matrices, Laplacian matrices, spectral radius, and Fiedler vectors are derived using tools like Gephi and next-generation firewalls with built-in network topology capabilities. Additionally, the role of hypergraphs, using metadata from GrassMarlin to analyze network traffic and identify anomalies such as botnets, is examined. The focus includes predicting bottlenecks and detecting the spread of botnets through changes in degree centrality and traffic patterns. The implementation includes exporting GraphML files from GrassMarlin and using Gephi-generated CSV files for analysis in a Jupyter Notebook environment. A conceptual understanding of linear algebra and matrices is assumed, and anything further is explained in the paper.

1 Introduction

The threat landscape to ICS/OT environments is rapidly changing due to the increased number and complexity of cyberattacks. In the first half of 2024 alone, the number of exploits in Metasploit for OT/ICS/SCADA systems has surged from less than 100 to over 200. These new exploits focus on banner grabbing, remote code execution, and installing Meterpreter payloads on HMIs and SCADA servers. Traditional security measures often fall short in the face of these attacks, and so cybersecurity teams must develop new tools in order to adapt to this evolving environment. This paper presents a mathematical approach using graph and hypergraph theory to enhance cybersecurity architecture evaluation. This approach is cost-effective and is meant to supplement, not replace, existing cybersecurity solutions. By leveraging these mathematical techniques, organizations can add an additional layer of defense to their cybersecurity infrastructure, enhancing the detection and mitigation of potential

threats without significant additional costs. Additionally, this approach helps to analyze the network segmentation of zones and conduits as specified in the IEC 62443 standard.

2 Basics of Graph Theory

2.1 Nodes and Edges

In graph theory, a graph G is composed of a set of nodes V and a set of edges E. Nodes (or vertices) represent entities such as devices or servers, while edges represent the connections between these entities, such as communication links or data flows.

2.2 Adjacency Matrix

The adjacency matrix A of a graph is a square matrix used to represent a finite graph. The elements of A indicate whether pairs of vertices are adjacent or not in the graph. Mathematically,

$$A_{ij} = \begin{cases} 1 & \text{if there is an edge between node } i \text{ and node } j, \\ 0 & \text{otherwise.} \end{cases}$$

2.3 Laplacian Matrix

The Laplacian matrix L is defined as $L = D - A$, where D is the degree matrix (a diagonal matrix where D_{ii} is the degree of vertex i). The Laplacian matrix is used to find important properties of the graph, including connectivity and bottleneck prediction.

$$L = D - A$$

where

$$D_{ii} = \text{degree of node } i.$$

2.4 Spectral Radius and Fiedler Vectors

2.4.1 Spectral Radius

The spectral radius ρ of a graph is the largest absolute eigenvalue of its adjacency matrix. The spectral radius provides a measure of the graph's connectivity, resiliency, and redundancy. In the context of ICS/OT cybersecurity, a node with a high spectral radius might become a critical point of failure if compromised. Analyzing the spectral radius helps identify such nodes, enabling preemptive security measures to be implemented.

Mathematically, the spectral radius ρ is given by:

$$\rho = \max(\lambda_i)$$

where λ_i are the eigenvalues of the adjacency matrix A.

2.4.2 Fiedler Vectors

The Fiedler vector v (sometimes referred to as the algebraic connectivity) is the eigenvector corresponding to the second smallest eigenvalue (λ_2) of the Laplacian matrix L. The Fiedler vector is crucial in spectral graph theory and has significant implications for network analysis. In ICS/OT cybersecurity, the Fiedler vector helps in identifying clusters and community structures within the network. By examining the Fiedler vector, it is possible to determine which nodes are more likely to form a bottleneck or critical junction in the network. The most common approach is to segment the network by placing the nodes corresponding to the positive Fielder vector components into one sub-network, and the nodes corresponding to the negative Fielder vector components in another.

The Fiedler vector is derived from the Laplacian matrix L:

$$Lv = \lambda_2 v$$

where λ_2 is the second smallest eigenvalue of L.

2.4.3 Importance in ICS/OT Cybersecurity

1. Identifying Critical Nodes and Edges: Nodes or edges with high values in the Fiedler vector may represent critical points in the network. These points are essential for maintaining network connectivity and, if compromised, could lead to significant disruptions.

2. Community Detection: The Fiedler vector can be used to partition the network into smaller communities or clusters. This is particularly useful in detecting abnormal behavior within subgroups of the network, which may indicate localized attacks or infections.

3. Predicting Bottlenecks: By analyzing the spectral radius and Fiedler vector, cybersecurity teams can predict potential bottlenecks in the network. These bottlenecks are more likely to occur at nodes with high eigenvalues and low associated eigenvectors.

3 Tools for Graph Analysis

Tools such as Gephi provide capabilities to visualize and analyze graph metrics. Moreover, some next-generation firewalls (NGFWs) have integrated network topology features that allow real-time monitoring and analysis.

4 Application in Predicting Bottlenecks

By analyzing the adjacency and Laplacian matrices, and deriving eigenvalues and eigenvectors, it is possible to predict bottlenecks in the network. For instance, a high spectral radius may indicate nodes with high connectivity, potentially becoming bottlenecks under heavy traffic. The Fiedler vector can identify critical edges or nodes whose removal might significantly impact the network's connectivity.

For example, consider a network with an adjacency matrix A:

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

The degree matrix D is:

$$D = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

The Laplacian matrix L is:

$$L = D - A = \begin{pmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 2 \end{pmatrix}$$

The eigenvalues of L are calculated, and the Fiedler vector is used to identify potential bottlenecks.

5 Basics of Hypergraph Theory

5.1 Nodes and Hyperedges

In hypergraph theory, nodes are connected by hyperedges, which can link any number of nodes, unlike edges in traditional graphs. This allows for more complex relationships between entities.

5.2 Tensors in Hypergraph Analysis

Tensors are higher-dimensional generalizations of matrices used to represent hypergraphs. GrassMarlin, a passive network monitoring tool (similar to WireShark) developed by the NSA specifically for ICS/OT environments, captures metadata that can be represented as tensors, providing a more detailed view of network interactions. A tensor T in hypergraph analysis can be represented as:

$$T = \sum_{i,j,k} t_{ijk} e_i \otimes e_j \otimes e_k$$

where t_{ijk} represents the connectivity and interaction strength between nodes i, j, and k.

GrassMarlin can collect various types of metadata, including: - Device and host information (IP addresses, MAC addresses, hostnames) - Network protocols in use (such as Ethernet/IP, Modbus, DNP3, etc.) - Open ports and services running on devices - Network interface details - Network segments and subnets - Communication patterns between devices - Passive fingerprinting of device types and operating systems

5.3 Understanding Tensors in Network Analysis

5.3.1 What is a Tensor?

A tensor is a multi-dimensional array of numbers, extending the concept of scalars (zero-dimensional), vectors (one-dimensional), and matrices (two-dimensional) to higher dimensions. For example, a 3-dimensional tensor can be visualized as a cube of numbers.

5.3.2 Tensor Representation in Network Analysis

In the context of network analysis, especially for detecting cybersecurity threats like botnets, tensors can represent complex interactions among nodes.

- **Tensor Components**: Nodes are represented as indices in the tensor, and interactions are represented as values within the tensor elements.
- **Example of a 3-Tensor**: Consider a small network with nodes A, B, C, and D. A 3-tensor T can be constructed to represent their interactions.

$$T = \begin{cases} t_{ABC}, & \text{interaction among nodes } A, B, \text{ and } C \\ t_{ABD}, & \text{interaction among nodes } A, B, \text{ and } D \\ t_{BCD}, & \text{interaction among nodes } B, C, \text{ and } D \end{cases}$$

If the interaction between nodes B, C, and D doubles due to botnet activity, the tensor element t_{BCD} would increase accordingly.

5.4 Importance of Monitoring Network Traffic

Monitoring network traffic is crucial for two reasons. First, research by Dragos has shown the ability to introduce malicious OPC and DNP3 traffic alongside regular traffic, which often goes undetected by traditional cybersecurity solutions. Second, many botnets now communicate with the bot-herder via third-party apps like Telegram, making it more difficult for traditional cybersecurity

solutions to detect these communications. Hypergraph analysis has the potential to detect both of these issues by providing a more comprehensive view of network interactions.

5.4.1 Operations on Tensors

To detect anomalies, operations such as summing interactions to calculate centrality or comparing tensors across different time frames can be performed.

6 Example Calculation

6.1 Initial State

Assuming initial balanced interactions among four nodes A, B, C, D:

$$T_{\text{initial}} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

This tensor shows each node interacting equally with the others (interaction strength of 1).

6.2 After Botnet Infection

Suppose the interaction among nodes B, C, and D doubles due to botnet activity:

$$T_{\text{infected}} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 2 & 2 \\ 1 & 2 & 0 & 2 \\ 1 & 2 & 2 & 0 \end{pmatrix}$$

6.3 Calculating Degree Centrality

Summing the interactions for each node to calculate centrality:

$$\text{Centrality}_{\text{initial}} = \sum T_{\text{initial}} = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix}$$

$$\text{Centrality}_{\text{infected}} = \sum T_{\text{infected}} = \begin{pmatrix} 3 \\ 5 \\ 5 \\ 5 \end{pmatrix}$$

6.4 Detecting Anomalies

Comparing the centralities to detect significant changes:

$$\text{Anomalies} = \text{Centrality}_{\text{infected}} - \text{Centrality}_{\text{initial}} = \begin{pmatrix} 0 \\ 2 \\ 2 \\ 2 \end{pmatrix}$$

Nodes B, C, and D show increased centrality, indicating potential botnet activity.

7 Using Hypergraphs to Detect Botnets

Hypergraphs can be particularly useful in detecting sophisticated cyber threats like botnets. By monitoring changes in degree centrality (a measure of node importance) among nodes, unusual patterns that may indicate botnet activity can be identified. In the absence of communication with a command-and-control (C2) server, analyzing the metadata through hypergraphs can reveal subtle changes in network traffic, pointing to compromised nodes.

7.1 Example: Identifying a Spreading Botnet

Consider an ICS network where a botnet is spreading. Traditional graph analysis might show changes in degree centrality, with certain nodes becoming increasingly central as they communicate more frequently. Hypergraph analysis, using metadata from GrassMarlin, can provide additional context, such as changes in the types of communication or shifts in traffic patterns, offering a deeper understanding of the botnet's spread.

For instance, if the network's communication is represented as a tensor T, the changes in its elements over time can be observed. An increase in specific tensor components t_{ijk} might indicate that certain nodes are engaging in more complex interactions, potentially signifying botnet activity.

By continuously monitoring these tensor components, anomalies indicative of a botnet can be detected. For example, a sudden increase in interactions among a specific set of nodes, represented by changes in T, can trigger further investigation.

7.2 Using Exported GraphML Files and Gephi CSV Files for Analysis

To perform this analysis, metadata can be exported from GrassMarlin as GraphML files for the hypergraphs, while Gephi generates CSV files of the graphs. These files can then be analyzed in a Jupyter Notebook environment using libraries such as NetworkX, matplotlib, and NumPy.

8 Conclusion

Graph and hypergraph theories provide powerful tools for evaluating cybersecurity architectures in ICS/OT environments. By leveraging mathematical techniques and modern analysis tools, bottlenecks can be predicted and cyber threats detected more effectively. This approach enhances the robustness of ICS/OT networks, ensuring their continued operation in the face of evolving cyber threats.

9 References

1. Chung, Fan R. K. "Spectral Graph Theory." CBMS Regional Conference Series in Mathematics, Number 92, 1997. 2. Newman, M. E. J. "Networks: An Introduction." Oxford University Press, 2010. 3. GrassMarlin User Guide. National Security Agency (NSA). 4. Gephi: The Open Graph Viz Platform. `https://gephi.org` 5. Fortinet NGFW: `https://www.fortinet.com/products/next-generation-firewall`