5-2023

A Costing Framework for the Dynamic Computational Efficiency of the Network Security Detection Function

Afdam Howarth

A Costing Framework for the Dynamic Computational Efficiency of the Network Security Detection Function

By

Adam Howarth

BA University of Southern Maine 2020

A Thesis

Presented to the Affiliated Faculty of

The College of Science, Technology, and Health

at the University of Southern Maine

Submitted in Partial Fulfillment of Requirements

For the degree of Master of Cybersecurity

Portland & Gorham, Maine

May 2023

Adam Howarth

5/5/2023

Cybersecurity

ABSTRACT

This study developed a comprehensive framework to systematically evaluate the economic implications of security policy implementation in IT-centric business processes. Focusing on the detection aspect of the NIST cybersecurity framework, the research explored the interrelation between business operations, computational efficiency, and security protocols. The framework comprises nine components, addressing the gap between cost projections and security policy enforcement. The insights provided valuable perspectives on managing security expenses and resource allocation in information security, ensuring alignment with revenue and expenditure outcomes while emphasizing the need for a comprehensive approach to cost management in information security management.

University of Southern Maine

Master of Cybersecurity

This thesis was presented

by

Adam Howarth

It was presented on

4/27/2023

and pending approval of:

Lori L. Sussman, Ed.D, Faculty Advisor

University of Southern Maine

ACKNOWLEDGEMENTS

Primarily, I express my deepest gratitude to my thesis advisor, Dr. Lori Sussman, for the unwavering support, guidance, and mentorship from my earliest inquiries into enrollment for the master's program through the completion of this project. Your expertise and encouragement were invaluable in helping me navigate the complex landscape of cybersecurity, and I am genuinely thankful for the opportunity to learn from such a distinguished professional.

I would also like to express my appreciation to the entire faculty of the cybersecurity master’s program. Each of you has played a crucial role in shaping my understanding of cyber security, helping me identify my shortcomings, and fostering the growth of a well-rounded information security skill set.

I am grateful for the support and camaraderie of my fellow students in the program, who have enriched my experience through our discussions, collaborations, and friendships. Your diverse perspectives and insights have been essential to my learning and personal growth.

I would like to acknowledge the contributions of my colleagues from my professional work experience. Your expertise and mentorship have allowed me to apply the lessons learned in the classroom to real-world scenarios, strengthening my skills and deepening my understanding of the field.

This thesis represents not only the culmination of my academic journey in the master's program but also the invaluable experiences and lessons gained from my personal work experience. It is a testament to the incredible support and guidance I have received from my mentors, colleagues, friends, and family. I am grateful to each of you for participating in this journey and helping me become the information security professional I am today.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Integration of information technology into business processes is a critical component of business transformation. With the growing prevalence of cybersecurity threats, organizations must consider business objectives and security policies when incorporating technology into their more automated operations (Senate Committee on Homeland Security and Governmental Affairs, 2018). "Achieving digital transformation goals is impossible without taking into account information security considerations" (Sandhu, 2021, p. xiv). As information technology assets become more embedded in business processes, security policies, roles, and systems may impose constraints that counterbalance the potential benefits. Information security is often perceived as a cost center within organizations, leading to challenges in allocating costs and resources (Gordon & Loeb, 2002). This study developed a methodology for information technology management that considers not only the cost dynamics of security policy detection requirements but also the perception of information security as a cost center when making decisions about business process augmentation.

Focusing on the Detection Function of the NIST Cybersecurity Framework

The National Institute of Standards and Technology (NIST) cybersecurity framework comprises five functions, and increasingly the implementation of security policies focuses on addressing these functions (Greenwald, 2013). These high-level functional abstractions include identification, protection, detection, response, and recovery. Together this framework defines the "…five key pillars of a successful and wholistic cyber security program" (NIST, 2018). The complexity and functionality of the algorithms required for these policy objectives differ,

encompassing availability attacks, data flow, and topographic anomaly detection. This research concentrated on the detection function of the NIST framework, which is the most computationally sensitive, as it processed real-time event data (Stonebraker et al., 2014).

Analytical Frameworks

There is no formal methodology for assessing the impact of a proposed business process augmentation on the detection function of an organization's security operations. A mapping function that translates the domain of detection analytic capacity requirements into a range of monetary costs would enable a direct comparison with the expected revenue changes associated with technology-based business process augmentation.

This study aimed to develop a formal method for determining proposed business process augmentations. The importance of this research was due to the increased number of monitored information technology assets that affected the costs of analytic detection requirements. This method considered security operation costs when projecting the revenue changes of proposed business process augmentations.

The central research question for this project was: How can security operations management assess the costs of technology-driven enhancements to business processes in the context of the security detection function? Addressing this question involves examining security policy-driven detection outcomes, relationships between business processes and network complexity, and the deltas observed in resource requirements.

Conceptual Framework

The theoretical concepts of digital business process transformation, analytic detection outcomes, and security monitoring (Von Solms & Niekerk, 2013) are foundational conceptual

frameworks for this project. The comprehensive literature review explained the importance of each. The method proposed by this project operates within the intersection of these concepts, mapping business process revenue opportunities to security operation costs while considering principles from each area (Li et al., 2010, p. 19). The expected outcome of this project was a framework that enables security cost allocation among operating departments which require security.

This project made several assumptions about the proposed business process augmentation, the detection algorithms employed, and the security monitoring coverage strategy defined by local security policies. These included the identification of process cost centers, the approximation of core algorithmic logic isolation, and non-uniform departmental compliance requirements.

The proposed method's significance lies in its ability to enable security operations to connect the computational efficiency of the security detection function with other functional areas of the organization. This method facilitates resource allocation within the organization, reducing information security costs (Horngren et al., 2015, p. 565). By providing security operations input on business process augmentation, organizations can ensure the intended revenue/cost impact.

Definition of Terms

This project employs several unique terms that warrant definition.

Network. A system implemented with a collection of interconnected components. Such components may include routers, hubs, cabling, telecommunications controllers, key distribution centers, and technical control devices (Ross et al., 2020).

Security. A condition that from the establishment and maintenance of protective measures that enable an enterprise to perform its mission or critical functions despite risks posed by threats to its use of information systems. Protective measures may involve a combination of deterrence,

avoidance, prevention, detection, recovery, and correction that should form part of the enterprise's risk management approach (Committee on National Security Systems, 2015, 176).

Security Function. The hardware, software, or firmware of the system is responsible for enforcing the system security policy and supporting the isolation of code and data (Ross et al., 2020, 58).

Computational Efficiency. The efficiency of an algorithm can be captured by a function T from the set of natural numbers N to itself such that T(n) is equal to the maximum number of basic operations that the algorithm performs on inputs of length n (Arora & Barak, 2007, p. 13).

This project proposed a method for calculating the cost of security policy enforcement concerning proposed technology-based business process augmentations. We utilized Arora and Barak's (2007) definition of computational efficiency as the foundational building block of this research represented in (1). Composition with resource cost quantification methods allowed for a mapping between computational efficiency and cost, as shown in (2) (Sikeridis et al., 2018, p. 2). Expressed as a composite (3) is a definition for cost as a function of computational efficiency and, therefore, input cardinality.

$$CE = T(n) \tag{1}$$

$$Cost = C(x) \tag{2}$$

$$Cost = C(T(n)) \tag{3}$$

Gordon and Loeb (2002) asserted that "there are no fixed costs of information security" (p. 443). From this concept, this project focuses on the cost dynamics as seen in (4), given dynamics in computational efficiency as shown in (5).

$$\Delta Cost = \Delta C(x) \tag{4}$$

$$\Delta CE = \Delta T(n) \tag{5}$$

This project identified two sources of dynamics in the computational efficiency of security detection. The first was dynamics in detection function input. By deploying information technology to support business solutions, (6) defines the change in input cardinality caused by business process augmentation (Gordon & Loeb, 2002, p. 438). Diestel (2017) defines a graph as (7) a set of nodes and edges (p. 2). Business process defines the information technology requirements of an organization. The management of business processes "...sits at the intersection of computer science, information systems engineering, management science, and industrial engineering" (Reijers, 2021, p. 4). Dynamics in business processes that result in a topological change to the network are shown by (8) to impact computational efficiency directly, given that nodes and edge cardinality are the input for detection algorithms (Hamilton et al. 2017, p. 2).

$$\Delta CE = T(\Delta n) \tag{6}$$

$$G = (V, E) \tag{7}$$

$$\Delta CE = T(\Delta n) \ni \Delta n = f(\Delta V, \Delta E) \tag{8}$$

Next, the project identified dynamics in security policy detection requirements as a source of computational efficiency dynamics (Whitman & Mattord, 2018, p. 161). This resulted in the formulation of (9), where the dynamics of computational efficiency are the composite of applicable detection algorithms.

$$\Delta CE = \Delta T(n) \ni \Delta T(n) = \Delta(t_1(n_1) + \ldots + t_x(n_x)) \tag{9}$$

We leveraged (8) and (9) for the project's research objective of determining how security operations management should assess the costs of technology-driven enhancements to business processes in the context of the security detection function. With this functional definition, our costing framework can map organizational plans directly to security related costs, as seen in (10) and (11).

$$\Delta Cost = T(\Delta n) \quad (10)$$

$$\Delta Cost = \Delta T(n) \quad (11)$$

By conducting experimental simulations that evaluated the impact of augmentations on various detection analytics' computational efficiency, this study demonstrated a repeatable method for approximating local analytic requirements and their application to the proposed costing framework.

CHAPTER 2

LITERATURE REVIEW

Business process management, security computational efficiency, and information security policy are the thematic pillars (Gordon & Loeb, 2002, p. 439). Literature aggregated under these themes establishes evidence in answering the primary research question and our method of experimentation.

Information Technology-based Business Process Augmentation Business Solution

Rajarathinam, Chellappa, and Nagarajan (2015) discussed the augmentation of business processes with information technology in their research. Their framework partitioned business processes into management, operational, and supportive classes. This partitioning allowed for the consideration of variable algorithmic requirements and input data cardinality.

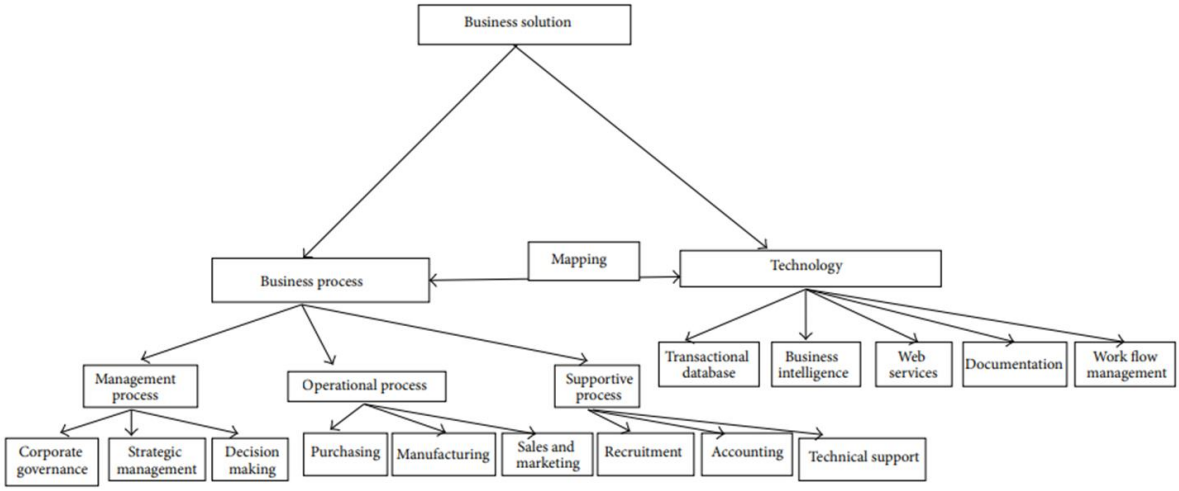
Similarly, Rajarathinam et al. (2015) introduce the term "business solution" to represent the pairing of business process and information technology solutions (p. 2). The concept of a business solution binds the technology requirement domain with business function. This combination determines cost allocation along with security policy constraints. Information

security management utilizes this pairing to determine the relevant detection computational efficiency dynamics.

Figure 1.

Business Solution Mapping

Reprinted from Rajarathinam, Chellappa, & Nagarajan, 2015, Figure 2



The Rajarathinam et al. (2015) framework outlined the fundamental principles to consider while assessing business processes; consistency, consonance, competitive advantage, and feasibility (p. 5). This project aims to balance their principle of competitive advantage, "the strategy must provide for the creation and maintenance of a competitive advantage in the selected area of the activity" and feasibility, "the strategy must neither overtax available resources nor create unsolvable subproblems" (Rajarathinam et al., 2015, p. 5).

Detection Algorithm Computational Efficiency

Bastian and Weir (2020) discuss the challenges of efficient algorithm selection. These authors form a mapping between tasks and analytical approaches, partitioning detection tasks into

prescriptive, predictive, and descriptive classes. The detection algorithm complexity domain at focus in this project is discretized into these classes and then mapped to analytical approaches, creating an interface for evaluating the delta in computational efficiency between business states. Bastian and Weir conclude that algorithm selection automation did not significantly improve over an expert system approach, supporting this project's intent to inform experts on business process dependencies in algorithm selection.

Distributed Denial of Service Detection

Bouyeddou, Kadri, Harrou & Sun, Y. (2020) present an innovative method for network intrusion detection using a nonparametric Kullback-Leibler distance-based approach in their study titled "Nonparametric Kullback-Leibler distance-based method for networks intrusion detection." The paper demonstrates the potential of this approach in detecting Distributed Denial of Service (DDoS) attacks and other types of network intrusions, which is crucial in examining the impact of business process augmentation on network security. Furthermore, the authors propose a novel method to identify network intrusions using the Kullback-Leibler (KL) distance, a measure of the divergence between two probability distributions.

The nonparametric KL distance-based method aims to enhance the detection of intrusions in real-time by comparing the statistical distribution of network traffic data against a reference distribution representing normal traffic. An intrusion detected is positive if the KL distance between the two distributions exceeds a predefined threshold. In analyzing the relationship between business process augmentation and security policy, Bouyeddou et al.'s (2020) research are relevant as it offers an advanced technique for network intrusion detection, including DDoS attacks. The KL distance-based method may provide valuable insights into how changes in business processes and network configurations can affect the detection of network intrusions.

Penukonda and Paramasivam (2021) present a behavior-based DDoS detection algorithm for data centers in the cloud. Their proposed method focuses on detecting distributed denial-of-service (DDoS) attacks in a cloud environment by analyzing network traffic patterns and classifying them as legitimate or malicious. This approach relies on threshold values to determine whether traffic patterns indicate an ongoing DDoS attack. However, one potential limitation of Penukonda and Paramasivam's (2021) approach is a reliance on predetermined threshold values, which may not be optimal for all network traffic scenarios.

While Penukonda and Paramasivam's (2021) behavior-based DDoS detection algorithm provides a valuable approach for detecting DDoS attacks in cloud environments, the reliance on fixed threshold values could limit its adaptability. The KL divergence-based method by Bouyeddou et al. (2020) may offer a more appealing alternative due to its nonparametric approach and adaptability to evolving network traffic patterns. The KL divergence method's adaptability to various traffic patterns and its focus on comparing probability distributions offer notable advantages over methods that rely on fixed thresholds for specific traffic features. This adaptability and flexibility make the KL divergence-based method a more appealing choice for DDoS detection in the context of this project with a computational efficiency of (12), a composition of the product of node count and histogram width, and the edges processed in histogram realization.

$$T_{KLDDOS}(n) = f(|V|)(histogram\ width) + g(|E|) \quad (12)$$

Data Exfiltration Detection

Data exfiltration refers to the unauthorized transfer of sensitive data from an organization's network to an external destination. Therefore, detecting and preventing data exfiltration is a critical aspect of cybersecurity. Proposals of numerous methods exist to address this challenge. In

this literature review, we focus on the producer-consumer ratio (PCR) method, a technique for detecting data exfiltration by monitoring data flow between different network entities.

Bullard and Gerth (2014) presented (13) the producer-consumer ratio metric, a statistic for detecting data exfiltration based on monitoring data flow between network entities.

$$PCR = \frac{srcBytes - dstBytes}{srcBytes + dstBytes} \quad (13)$$

From this definition, we can infer (14) the computational efficiency of PCR calculation. This equation represents the total computational efficiency of this method as the composition of the per node statistic calculation and per edge consideration made in its calculation.

$$T_{PCR}(n) = f(|V|) + g(|E|) \quad (14)$$

The core idea behind this method is to analyze the ratio of data produced (sent) by a network entity to the data consumed (received) by that entity. This ratio can indicate unusual data flows, such as those indicative of data exfiltration. For this reason, this project utilized PCR calculation as the basis for evaluating computational efficiency dynamics for assets subject to data exfiltration monitoring.

Graph Neighborhood Traversal

Inductive representation learning on large graphs, like computer networks, has become an essential technique for capturing complex patterns and relationships in various real-world datasets (Hamilton et al. 2017, p. 1). In this literature review, we focus on the computational efficiency of the breadth-first search (BFS) algorithm, a common graph traversal technique, in the context of inductive representation learning on large graphs.

Breadth-First Search (BFS) is a widely used graph traversal technique that explores the nodes of a graph in layers, starting from a source node and moving outwards, with a time complexity of (15) (Cormen et al., 2009, p. 595).

$$T_{BFS}(n) = O(|V| + |E|) \quad (15)$$

The utilization of BFS in GNNs to efficiently sample neighborhoods of nodes enable the models to scale to input large graphs (Hamilton et al., 2017). Moreover, BFS forms the basis for all neighborhood aware detection algorithms due to this property. For this reason, this project considered the computational dynamics of BFS.

Security Policy

Security policies play a crucial role in guiding organizations through their cybersecurity efforts, with the focus being on infrastructure, compliance, and legal aspects. Alok et al. (2022) highlighted the importance of developing comprehensive security policies, identifying various policy classes such as privacy, web, cloud, information, physical, data retention, access control, data protection, network, and email communications. By partitioning the policy space, they provide a structured approach to addressing business process augmentation and the necessary detection capacity in the context of security policy. The Payment Card Industry Data Security Standard (PCI DSS) is a prominent example of a security policy framework that offers guidance on securing payment card data. It emphasizes the need for monitoring network activity to detect and respond to security incidents effectively (PCI Security Standards Council, 2021).

Apart from the PCI DSS requirements, organizations should also consider other policy classes outlined by Alok et al. (2022) to create a well-rounded security strategy. Algorithm selection for security detection monitoring should be based on industry-accepted best practices, considering factors such as the type of data monitored, network architecture, and potential threat

landscape (PCI Security Standards Council, 2021). In cases with network reconfiguration, organizations must assess the implications for security detection monitoring algorithms. Changes in network components or data flow might necessitate adjustments to existing algorithms to maintain their effectiveness. Developing a comprehensive security policy incorporating PCI DSS requirements and other relevant policy classes is essential for organizations exposed to commerce and payment-related infrastructure. Adhering to industry-accepted best practices and monitoring network activity for security incidents will help organizations safeguard against data breaches and other security threats.

Under the theme of information technology-based business process augmentation, we considered the business solution mapping for dynamics in network topology (Rajarathinam et al., 2015, p.2). Regarding algorithm selection and complexity, we choose to employe the methods Bouyeddou et al.'s (2020) Kullback-Leibler based DDOS detection, Bullard and Gerth's (2014) producer-consumer ratio calculation of data exfiltration detection, and general breadth-first search. Finally, addressing the policy theme, Alok et al. (2022) provide a class-based method for policy structure. This model describes business augmentation as a composition of sub-policy.

CHAPTER 3

METHODOLOGY

How can security operations management assess the costs of technology-driven enhancements to business processes in the context of the security detection function? Addressing this research question required the study of the two identified sources of dynamics, input (10) and detection composition (11). We conducted an experimental process to identify the impact of various input on each of the detection algorithms in focus.

Setting

This research utilizes summarized statistics from actual business topology and networked entity usage profiles to inform a simulation allowing for the study across the dimensions of business structure and observation point configuration. These topologies inform the simulation, describing how the business process-related entities connect and the network infrastructure (Li et al., 2020). As the research questions acknowledge, the initial state of the organization's structure constrains business processes and monitoring requirements.

Participants/Sample

This project involves the participation of two businesses with network topologies representing opposite sides of the business structure dependence spectrum. The first organization, which this project calls Network A, is a medium/large retail operation. This business's network topology is highly dependent on the business structure. "A typical retail network at the downstream end of the supply chain generally comprises a set of retail stores, distribution centres[sic] (DCs), customers and the transport network that connects them together as a spatial system within which they interact" (Chhetri et al., 2017).

Network A: Retail Business Network Analysis

Network A is a computer network consisting of eighteen distinct locations, operating within a single internal /16 network address space. The network serves a business with seventeen retail stores and a central office, where each location has a unique third octet. This configuration connected the retail stores and central office through a well-structured network design that offers security, isolation, and access control, ensuring the protection of sensitive data while maintaining efficient communication between the locations.

Retail Stores Configuration. Each of the seventeen retail stores within Network A contains between 5 and 12 point-of-sale (POS) systems, an inventory computer, and an office computer. A local subnet connected the devices of each store, enabling communication and resource sharing among them. To ensure the security of sensitive store data and maintain isolation from potential threats, restrictions on devices within the retail stores allow access only to resources within their local subnet and the central office subnet. This design helps protect each retail store from unauthorized access and potential security breaches.

Central Office Configuration. The central office in Network A houses a server room for inventory management and sixty user desktops. The server room is crucial to the overall business operation, as it facilitates effective inventory management across the retail stores. Unlike retail stores, devices in the central office have internet access, allowing users to access external resources and perform necessary online tasks. However, to maintain a secure environment for critical business systems, restrictions prevent communication between user desktops and devices within the server room.

Security and Access Control in Network A. Network A's design strikes a balance between security, access control, and resource sharing among the retail stores and central office. By limiting access to resources within their respective subnets and the central office subnet, the retail stores can effectively share information with the central office without compromising security. Similarly, the central office can maintain a secure environment for critical systems while providing users with necessary internet access by restricting communication with the server room.

In summary, Network A's network configuration effectively addresses the unique requirements of a retail business with multiple store locations and a central office. In addition, the

design prioritizes security and access control, ensuring the protection of sensitive data while facilitating efficient communication and resource sharing among the business locations.

Network B: Corporate Headquarters Network Analysis

The second organization, which this project calls network B, is a software company with centralized compute resources, cloud service reliance, and a 100% work-from-home employee base. This end of the business structure spectrum shows a low level of dependency between topology and business structure. For this reason, the spectrum of results serves as an analog for the application of direct experimental observation evaluation. This business structure is standard after the covid pandemic forced organizations to reduce physical interactions between employees (O'Reardon & Rendar, 2020). In this configuration, the network serves multiple departments and facilitates communication between various teams, including finance, marketing, human resources, and IT. The network's primary objective is to ensure efficient interdepartmental communication and resource sharing while maintaining a secure environment for sensitive data.

Departmental Configuration. Each department within Network B has its subnet, allowing devices within the department to communicate and share resources efficiently. The number of devices within each subnet varies depending on the size and nature of the department. To maintain a structured network environment, restrictions limit devices with access to resources only within their local subnet, while communicating with other departments through a central resource server.

Central Resource Server. The central resource server in Network B plays a crucial role in interdepartmental communication and resource sharing. This server houses shared resources such as documents, applications, and databases required by various departments. By routing

communication between departments through the central resource server, Network B maintains a secure environment for sensitive data and prevents unauthorized access.

Security and Access Control in Network B. Network B's design ensures a balance between security, access control, and interdepartmental communication. By isolating departmental subnets and facilitating communication through the central resource server, the network can effectively protect sensitive data from unauthorized access. Additionally, the use of strict access controls and authentication mechanisms helps maintain a secure environment for critical business systems.

Data

Network monitoring in an ideal world would consist of full packet capture, with every detail of traffic flows archived. "NetFlow [IPFIX] can fill up some of the gaps and challenges regarding the collection of packet captures everywhere on the network. It is easier to store large amounts of NetFlow [IPFIX] data because it is only a transactional record" (Santos, 2016, p. 49). The use of network summary statistics available through the aggregation of IPFIX network observational logs constrained this research. The experiment requires manipulation of business network topology, the development, and execution of network event-generating simulation code produced observational records consistent with what is available from a production IPFIX data stream.

Using packet capture data from these environments, the network usage of each network entity informed behavioral profiles. We used these profiles to create network traffic flows representative of the actual topology. The composition of each network flow record results in a complete set of records for the observed period, which we analyzed for high-level summary statistics relevant to this experiment. The composition of observation points allows for the

distributed and passive collection of data at scale, this experiment considers this factor as a linear scaling factor (Claise et al., 2013).

We selected the algorithms for distributed denial of service detection, data exfiltration detection, and generalized breadth-first search detection from techniques identified in the literature review. Kullback-Leibler divergence is employed as a convolutional per the findings of Bouyeddou et al. (2020). The performance of these algorithms is the data focus of this study. With the per entity time of execution, captured memory and compute usage statistics demonstrated algorithmic requirements. The delta between the control and augmented business process instances reveals the augmentation's analytic impact. Collecting algorithmic requirements and analytic impacts for each network with total observation point coverage and core-only observation strategy produces four result sets.

Analysis

Data collected during network simulation is the required detection analytic capacity, measured as the proportion of overall detection resource usage. Therefore, the product of the observed proportion and the overall cost is directly comparable to the proposed business process augmentation's revenue/cost implications.

By virtualizing the execution environment, log data pulled from the hypervisor gives direct insight into the percentage of overall resource cost used by detection algorithms. Solving this product turns the analytic impact of a business process augmentation into a cost figure directly comparable to the proposed revenue increase enabled by the change. The decision-making framework created by this analysis process allows organizations to consider the security detection cost when considering business process augmentation across the dimensions of business structure, detection algorithm objective, and observation point strategy.

Participants' Rights

"Publishers of network data are interested in protecting the privacy of a number of entities: the network users, the network's security procedures, and the hosts that operate on the network" (Coull et al., 2009, p. 5). Network topology information procured from real-world businesses requires anonymization, providing accurate network design without exposing details of the source network (Coull et al., 2009, p. 1). A prefix-preserving IP address pseudonymization strategy provides the required anonymization (Fan et al., 2004). This method maintains the implicit structure of IP addresses as node labels without exposing the actual business address structure.

Potential Limitations of the Study

TraceWrangler, a popular PCAP anonymization tool in the Wireshark community, utilizes the methods presented by Coull et al. (2019). The deployment of TraceWrangler anonymized network b without significant information loss. It was impossible to preserve a useful Network A model, as it is highly structurally dependent. This limits the study to high-level network statistics derived from network traffic flow metadata without leaking host topology-specific information. Using network observational data as input generates an induced representation of the underlying network. Analytic requirements and impact data from the induced network represent network topology (Hamilton et al., 2017). Information technology entities in this study are host network interfaces, leaving the complexity of user space for future research, which would expand the resulting framework for user-oriented observational data. This allows for a description of networks by total node count and relations count.

Pilot Study

The initial simulation run revealed an inconsistency between multi-hop routes and their existence in the generated IPFIX data. The resolution of this issue required submitting generated conversations to all active observation points and checking the local routes and connected networks at each point to check observation inclusion. While generalized pilot network topologies occurred, direct consultation with the network staff at each company utilized in the final experiment enables high-detail simulation entity inclusion and behavioral profile modeling. This result informed the full-scale study to instead rely only on a high-level node and edge statistic summarization, leaving the intricacies of collection strategy for further study under the protection function (NIST, 2018).

CHAPTER 4

DATA COLLECTION, ANALYSIS, AND PRELIMINARY FINDINGS

Demonstrating computational efficiency requires running an analytic service and collecting resource utilization. Through experimental data collection, this project shows the use of high-level network statistics to approximate resource usage in network security detection. The results of the algorithms are outside this project's scope, with the resource required to operate being the metric in focus.

We chose to use Docker as the execution environment for the experiment. Executing programs in a Docker container provides several benefits that enable the collection of the program's resource requirements. First, Docker containers provide an isolated, lightweight environment deployable on any host system. The execution of programs within a standardized environment, independent of the underlying hardware or operating system, makes it easier to collect and compare resource usage data across different configurations.

Regarding data collection, Docker provides several built-in monitoring and resource management features that allow the collection of data on resource utilization and managing resource allocation. For example, the Docker stats command provides real-time statistics on CPU, memory, disk I/O, and network I/O usage for each running container. We gain insights into the resource requirements of the program collection and analysis of docker statistics (Potdar, Kengond, & Mulla, 2020). The collection of data for this study, using the Docker stats command, provided real-time statistics on the resource utilization of running Docker containers.

The execution of the stats command happened periodically during the experiment, and the storage of resulting data in a log file allowed later analysis. These metrics will provide valuable insights into the resource requirements of the containers under various conditions and in the

evaluation of the effectiveness of the proposed optimization techniques (Casalicchio & Perciballi, 2017). The Docker containers independently executed various security detection algorithms, and the input network data was varied to simulate different states of network reconfiguration. The data collection happened multiple times under different experimental conditions to evaluate the resource requirements of the proposed security detection techniques.

The data collection for each configuration and network scenario and the resulting metrics allowed the assessment of the required resources. Utilizing the average resource requirements from multiple runs provides a result set that reduces the impact of random variation. Analysis log files containing the data collected during each experiment store the metrics. Investigating anomalous log files determines their accuracy and qualification for inclusion in the study. This approach enabled a comprehensive evaluation of the resources required under various conditions, providing valuable insights into their effectiveness and limitations.

Description of the Sample

Docker stat streaming produces a series of records that include the following instantaneous quantitative fields; CPU percent utilized, memory consumption in base two measure (KiB, MiB, GiB, etc...), memory percent utilized, network I/O, storage I/O, and process count. This approach allows for the collection of continuous data throughout the execution of the program rather than at discrete intervals. Collecting data this way can capture changes in resource utilization over time and identify patterns or trends not represented in aggregate data. Streaming instantaneous resource utilization statistics can be particularly useful for understanding how a program's resource requirements change under different experimental conditions or scenarios. For example, it may be possible to observe how the CPU utilization of a program changes over time as the input data changes or to monitor memory usage as the program executes

a specific task. By analyzing this data, it is possible to gain insights into how the program consumes resources.

Data Collection Method

Data collection required the setup of a docker environment. Each container consumes IPFIX input data and runs a specific detection algorithm. This pattern allowed for the controlled consumption of input data and ensured that the same input data was in the same order in each experimental run. When run simultaneously, the command that starts the container and the docker stats command start both the experiment and the observation. The stats command ran for a duration of 1200 seconds at one sample per second.

Data Analysis Method

Analysis of the stat collection output, making it possible to gain insights into the computational complexity of the program running in the container. “Different algorithms devised to solve the same problem often differ dramatically in their efficiency” (Cormen et al., 2009, p. 12). A vital component of the analysis stage is the identification of algorithmic computational efficiency due to input complexity caused by augmentation.

The Docker stats command, used to collect real-time CPU utilization data, allowed for plotting the data over time. Analyzing the resulting graph makes it possible to identify CPU usage patterns and estimate the program's algorithmic computational efficiency. In the case where CPU usage remains constant over time, this may suggest that the program has a constant time efficiency, indicating that it performs the same number of operations regardless of the input size.

If the CPU usage increases linearly with the input size, this may suggest that the program has a linear response to input. The number of operations increases proportionally with the input

size. Finally, suppose the CPU usage increases exponentially with the input size suggesting the program has exponential computational efficiency where logical operations grow exponentially with the input size. Applying these principles to the memory usage statistics collected determines the memory response of computational efficiency (Maidana, Parhizkar, Gomola, Utne, & Mosleh, 2023).

CHAPTER 5

RESULTS

The results section of this study addresses the research question via response information in both the dynamics in input (10) and algorithmic composition (11). The experimental process identifies resource requirements of different algorithmic classes when subjected to perturbations in the simulated network. The following is an analysis of the data collected during the experiments.

The basis for evaluation of resource requirements for the different algorithmic classes was the metrics collected from the Docker stats command. The analysis of these metrics follows the methodology presented by Casalicchio and Perciballi (2017), which provides a comprehensive framework for measuring Docker performance. By adapting their approach to the specific context of our research, we were able to assess the impact of network reconfigurations on the resource consumption of the selected detection algorithms.

To analyze CPU usage effectively, we adapted the methodology Meng et al. (2019) proposed to analyze smartphone usage to detect security attacks. By employing a similar approach to examine the CPU utilization patterns of our detection algorithms, we were able to

identify trends and correlations that helped us assess the impact of network reconfigurations on the resource consumption of the selected detection algorithms.

We ran the Golang program in a Docker container and collected the Docker stats for CPU utilization, memory usage, network I/O, and disk I/O using the Docker stats command. Analyzing the collected data, we assess the baseline resource consumption for each metric under different computational workloads. This baseline served as a reference point for comparing the resource requirements of the security detection algorithms under investigation.

To establish a reliable reference point for comparing the resource consumption of security detection algorithms, we created a custom Golang program that simulates computational tasks with a specified time complexity. By creating a baseline that reflects various computational workloads, we can answer ordinal research questions about the performance of the security detection algorithms under different network conditions.

To establish a baseline for Docker stats, we developed a custom Golang program that simulates computational tasks with a specified input complexity (Whitman & Mattord, 2018, p. 161). The program takes three input arguments: node count, time complexity factor, and frequency. It performs N * F atomic computations representing the simplest case with time complexity proportional to N * F. By varying the values of N and F, the generation of different computational workloads simulates various ideal resource consumption levels within a Docker container, given the observed performance of the security detection algorithms. The collected data provided a baseline for resource consumption under different computational workloads as a reference point for comparing the resource requirements of the security detection algorithms under investigation.

Discussion of Results

The experimental results demonstrated a clear ranking of computational efficiency between the different network reconfigurations. In this section, we discuss the implications of these findings and how they can inform security operations management in assessing the costs of technology-driven enhancements to business processes in the context of the security detection function.

Ordinal Ranking of Computational Efficiency

The experimental results showed that logarithmic growth between network reconfigurations had a more significant impact on the resource consumption of some security detection algorithms than others. This finding is crucial for security operations management because it highlights the importance of understanding the relationship between network reconfigurations and the performance of security detection algorithms. By recognizing these relationships, security operations management can make informed decisions about adopting specific technology-driven enhancements to business processes, considering the potential impact on the security detection function. The ranking of computational efficiency observed in this study can serve as a guideline for security operations management to prioritize network reconfigurations that minimize the resource consumption of security detection algorithms. When reviewing plotted results evaluation of Equation (10) is along the x-axis of Figures 2-9. The composition of the results from multiple algorithm results at the same input complexity allows evaluation of (11).

Baseline

Beginning with the baseline, Figure 2 shows a summarization of the CPU utilization. Here we observed an approximate doubling in resource requirements between logarithmic changes in

input scale. The observation of a generalized minimum usage in small datasets shows similar usage. Figure 3 shows memory consumption per input configuration. Observation shows fixed memory allocation in the baseline case and proportional to the cardinality of input.

BFS

Figures 4 and 5 show the CPU and memory usage for each input configuration applied to the breadth-first-search algorithm. In this case, we can see CPU usage proportional to the logarithmically varied input sizes. Observation shows memory usage approximately double with logarithmic input variation.

KLDDOS

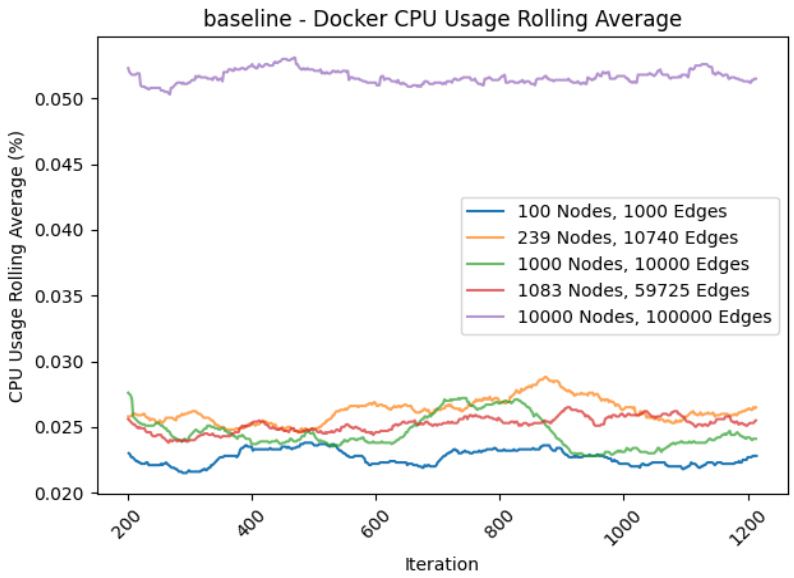
Figures 6 and 7 show the CPU and memory usage for each input configuration as applied to the Kullback-Leibler Distributed Denial of Service detection algorithm. CPU usage is significantly more sensitive to the increase in edge cardinality. Memory usage shows a base level of memory required independent of input, with slow usage growth as input cardinality increases.

PCR

Figures 8 and 9 show the CPU and memory usage for each input configuration as applied to the producer-consumer ratio-based data exfiltration detection algorithm. There is an initialization impact of pcr that optimizes overtime. This study has attributed this behavior to Golang memory allocation calls and the allocation itself. CPU shows minimal relation between input configuration, like the baseline results. Memory is proportional to the node cardinality as the PCR metric storage per node interacted with observed edges.

Figure 2.

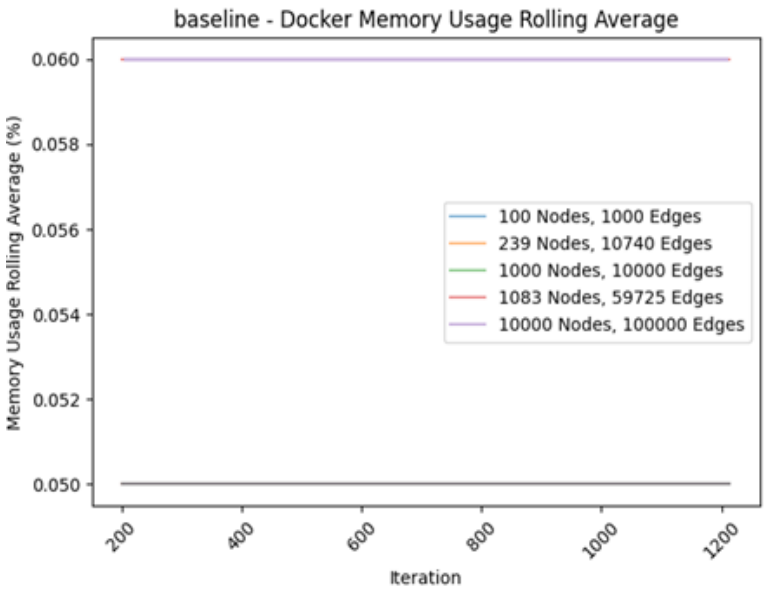
Baseline – Docker CPU Usage Rolling Average



Note. The collected CPU usage metrics from execution of the baseline container smoothed with an averaging window of width 100.

Figure 3.

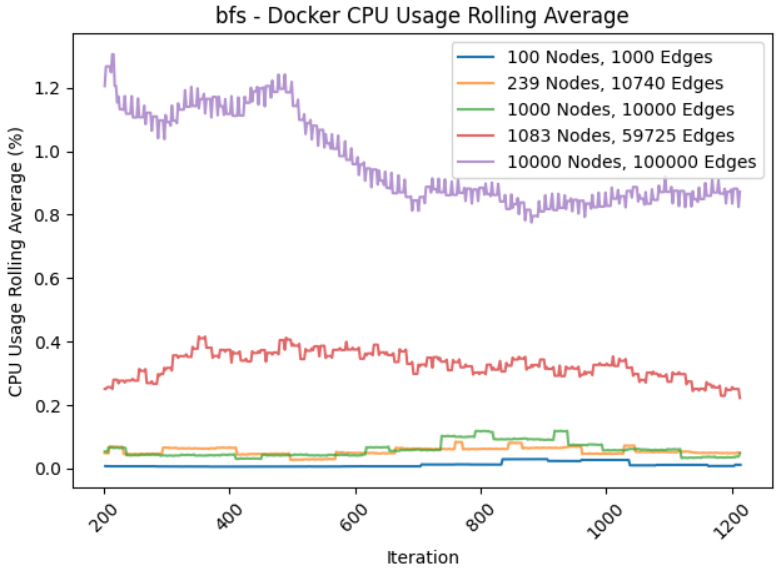
Baseline – Docker CPU Usage Rolling Average



Note. The collected memory usage metrics from execution of the baseline container smoothed with an averaging window of width 100.

Figure 4.

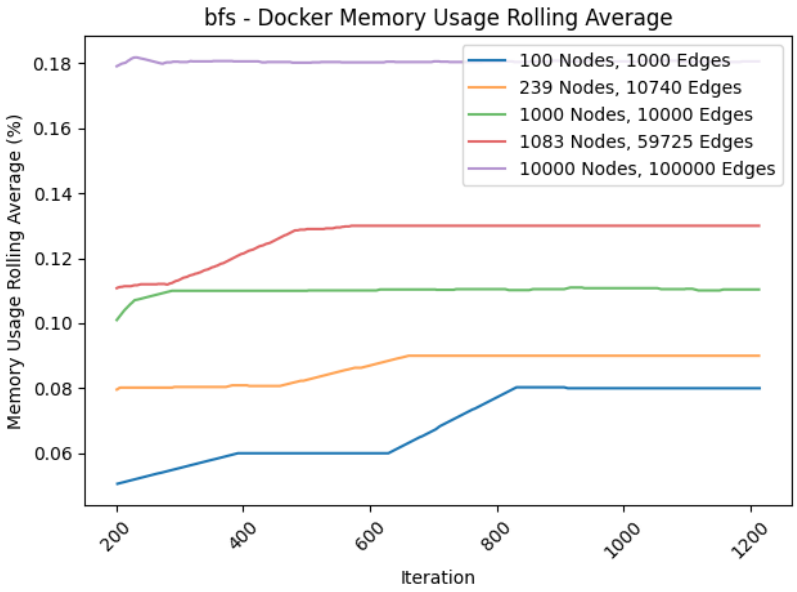
BFS – *Docker CPU Usage Rolling Average*



Note. The collected CPU usage metrics from execution of the breadth-first search container smoothed with an averaging window of width 100.

Figure 5.

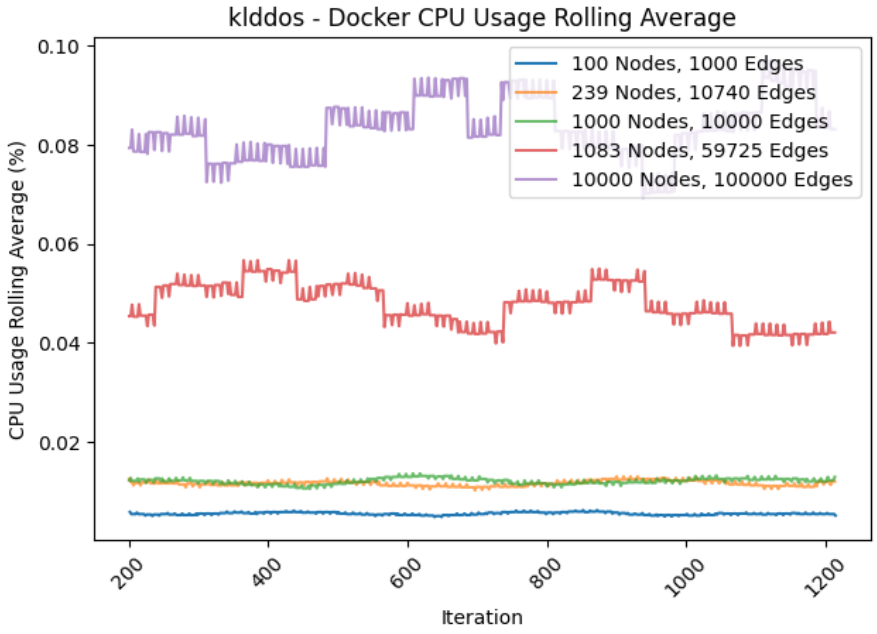
BFS – Docker Memory Usage Rolling Average



Note. The collected memory usage metrics from execution of the breadth-first search container smoothed with an averaging window of width 100.

Figure 6.

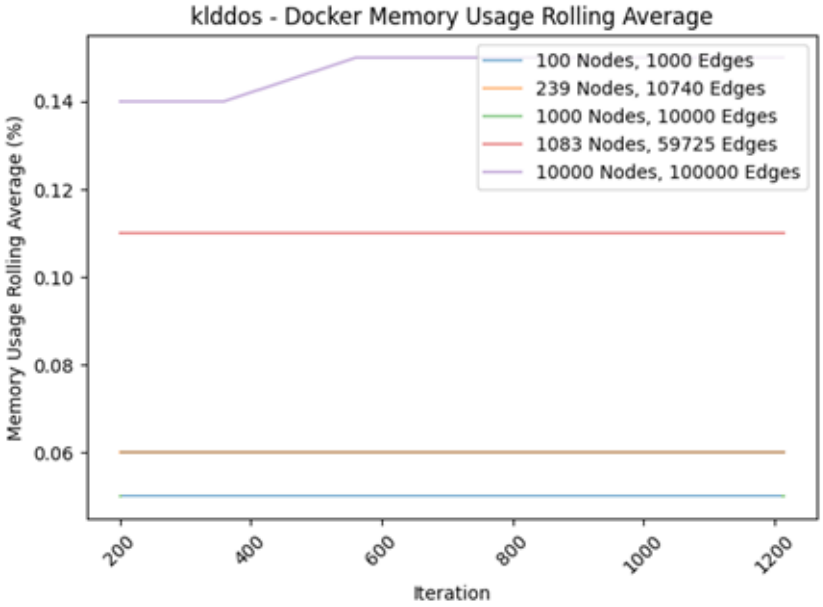
KLDDOS – Docker CPU Usage Rolling Average



Note. The collected CPU usage metrics from execution of the Kullback-Leibler DDOS container smoothed with an averaging window of width 100.

Figure 7.

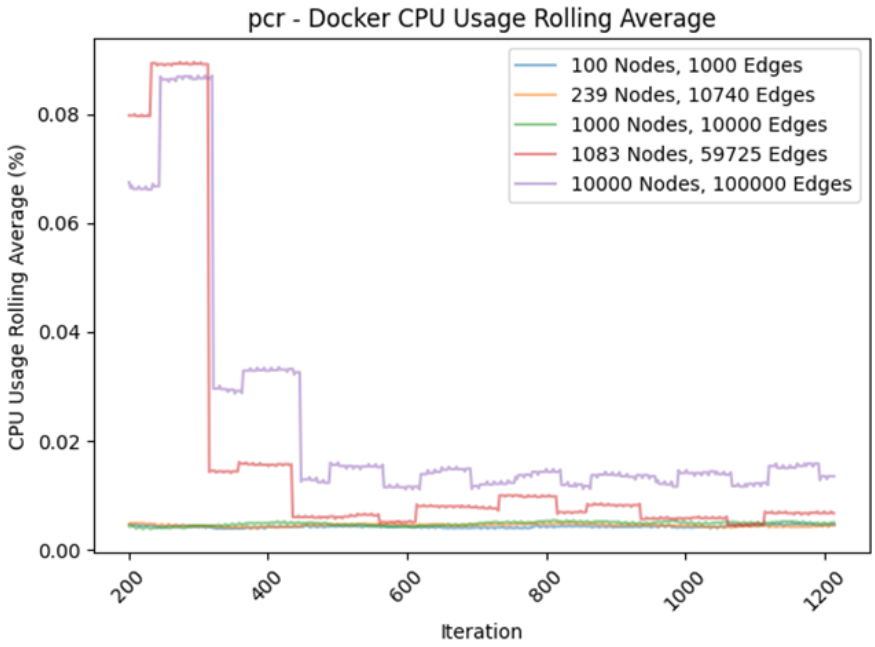
KLDDOS – Docker CPU Usage Rolling Average



Note. The collected Memory usage metrics from execution of the Kullback-Leibler DDOS container smoothed with an averaging window of width 100.

Figure 8.

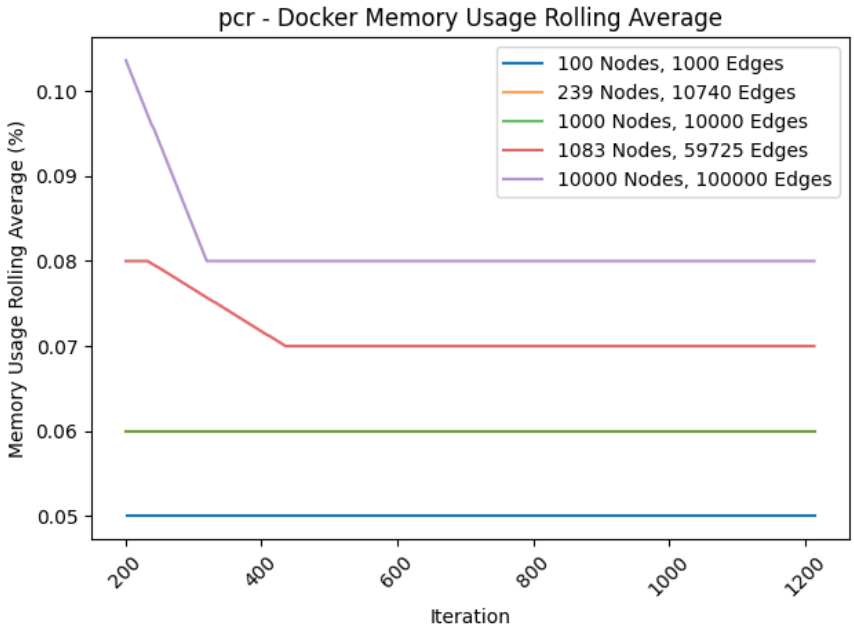
PCR -Docker CPU Usage Rolling Average



Note. The collected CPU usage metrics from execution of the producer-consumer ratio container smoothed with an averaging window of width 100.

Figure 9.

PCR -Docker CPU Usage Rolling Average



Note. The collected Memory usage metrics from execution of the producer-consumer ratio container smoothed with an averaging window of width 100.

Table 1.

Table of Results per input and algorithm variance

Dataset	Container	Avg. CPU Usage (%)	Rank (CPU)	Avg. Mem Usage (%)	Rank (Mem)
100_1000	baseline	0.02	1	0.05	1
239_10740	baseline	0.02	1	0.05	1
1000_10000	baseline	0.03	2	0.05	1
1083_59725	baseline	0.03	2	0.05	1
10000_100000	baseline	0.05	3	0.06	2
100_1000	klddos	0.01	1	0.05	1
239_10740	klddos	0.01	1	0.06	2
1000_10000	klddos	0.01	1	0.06	2
1083_59725	klddos	0.05	2	0.11	3
10000_100000	klddos	0.09	3	0.15	4
100_1000	bfs	0.02	1	0.08	1
239_10740	bfs	0.05	2	0.09	2
1000_10000	bfs	0.07	3	0.11	3
1083_59725	bfs	0.29	4	0.13	4
10000_100000	bfs	0.87	5	0.18	5
100_1000	pcr	0.00	1	0.05	1
239_10740	pcr	0.00	1	0.06	2
1000_10000	pcr	0.01	2	0.06	2
1083_59725	pcr	0.01	2	0.07	3
10000_100000	pcr	0.01	2	0.08	4

CONCLUSION

This study has unified the relevant concepts that cause network reconfiguration and how those reconfigurations induce dynamics in the cost of security detection. With the inclusion of a repeatable experimental process for sourcing metrics used in cost calculation the paper concludes by proposing the following costing framework for the dynamic computational efficiency of the network security detection function. This framework addresses the research question of how security operations management can assess the costs of technology-driven enhancements to business processes in the context of the security detection function by providing a structured approach to estimating and analyzing the fiscal impact of implementing security measures in a business environment.

A Costing Framework for the Dynamic Computational Efficiency of the Network Security Detection Function

The following series of steps comprise this costing framework.

Maintain a Business Solution Map. Ensure there is a strong understanding of the relationship between the business processes and the technical assets supporting them.

Determine the Business Solutions Impacted. Identification of the specific business solutions the technology-driven enhancements affect. This is the component of the framework that determines the dynamics in computational efficiency related to input complexity.

Enumerate Relevant Compliance and Security Policy. Enumerate the various compliance requirements and security policies that need to be adhered to when implementing technology-driven enhancements. This component represents the computational efficiency dynamic induced by recompositing of detection algorithms.

Select Appropriate Detection Algorithms. Based on the security policies, select suitable detection algorithms for network observations of the business solution. This component also

represents the computational efficiency dynamic induced by recompositing of detection algorithms.

Sample the Current Environment. By sampling the existing network environment and generating statistics, simulations approximately reflect the current state of the network.

Simulate with Current and Projected Input. Perform simulations using both the current network statistics and a dataset with the proposed changes, providing a comprehensive view of the potential impact on the system.

Analyze the Usage Dynamics. This step involves collecting and analyzing the simulation results to determine the changes in resource utilization resulting from the technology-driven enhancements.

Calculate the Cost of the Usage Delta. Estimate the cost of the additional resource utilization by considering the historical cost of operation and scaling it according to the changes in resource usage.

Allocate the Cost Delta. Allocate the cost delta to the operational department responsible for proposing the technology-driven enhancements, providing a clear picture of the financial implications of the changes.

By following these steps, the framework offers a systematic solution for security operations management to assess the costs of technology-driven enhancements to business processes in the context of security detection function, enabling organizations to make well-informed decisions and optimize their security investments.

Table 2.

A Costing Framework for the Dynamic Computational Efficiency of the Network Security Detection Function

Action	**Requirements**
Maintain a business solution map	Maintain a mapping between business function and the technical assets that support them (Rajarathinam, Chellappa, & Nagarajan, 2015, p.2).
Determine the business solutions impacted	Reduce business process augmentation to the set of business solutions affected.
Enumerate the relevant compliance requirements and security policy	Using the business solution mapping, identify the business compliance related to the process and the security policy for the technical asset (PCI Security Standards Council, 2018, requirement 10).
Select the appropriate detection algorithms	Using the relevant security policy, select the family of algorithms applicable to the business solution's network observations.
Sample the current environment	Take samples from affected network environment and generate node and edge cardinality statistics for usage in simulation (Yu et al., 2019).
Simulate with current and projected input	Perform simulations with the current network statistics as well as a dataset with the proposed delta n (Yu et al., 2019).
Analyze the usage dynamics	Collect statistics and identify the proportional delta in resource utilization.
Calculate the cost of the usage delta	Consider historical cost of operation and scale by the delta in resource utilization (Horngren et al., 2015 p. 840).
Allocate the cost delta	Allocate the cost delta to the operational department which proposed the business solution augmentation (Horngren et al, 2015 p. 565).

This project successfully developed a comprehensive costing framework for evaluating the impact of implementing security policies and algorithms within a business environment. By considering factors such as business solutions, compliance requirements, technical assets, and security policies, the framework offers a systematic approach to assessing the costs associated with enhancing security measures. The utilization of detection algorithms, network environment sampling, and simulations enables the framework to generate accurate and actionable data for decision-making.

The proposed framework not only quantifies the cost of usage delta but also facilitates the allocation of these costs to the respective operational departments responsible for the business solution augmentation. The framework enables organizations to make informed decisions on resource allocation and to better understand the financial implications of adopting new security measures. The costing framework, therefore, serves as a valuable tool for organizations seeking to optimize their security investments while minimizing the impact on their operational efficiency.

Future research could further refine the framework by incorporating additional variables or exploring alternative detection algorithms. Additionally, case studies and empirical validation of the framework within real-world scenarios would be beneficial in demonstrating its practical applications and effectiveness.

Limitations and Future Research

While this study provides valuable insights into the relationship between network reconfigurations and the resource consumption of security detection algorithms, the consideration of some limitations is important when interpreting the results. First, the use of high-level network summarization statistics and focus on IP networks constrained this study. Second, the execution

of experiments using simulated network traffic may not fully capture the complexity and variability of real-world network environments. Third, experimental input space logarithmically scales both node and edge counts when natural network traffic would compare better against a test grid with an approximate proportion between nodes and edges. Future research could expand the scope of the study by considering additional network configurations, exploring user-oriented observational data, and analyzing the performance of security detection algorithms in more diverse and complex network environments.

References

Arora, S., & Barak, B. (2007). Computational Complexity: A Modern Approach (Draft). Princeton University. Retrieved from https://theory.cs.princeton.edu/complexity/book.pdf

Bastian, N. D., & Weir, J. (2020, July 13). Algorithm selection framework for cyber attack detection. Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning. https://doi.org/10.1145/3395352.3402623

Bongertz, J. (2013). SEC-4 Trace File Sanitization NG [Slide show; Slides]. Sharkfest 2013: Wireshark Developer and User Conference, San Francisco, California, United States of America. sharkfest.wireshark.org. https://web.archive.org/web/20130903100239/https://sharkfest.wireshark.org/sharkfest.13/presentations/SEC-04_Trace-File-Sanitization-NG_Jasper-Bongertz.pdf

Bouyeddou, B., Kadri, B., Harrou, F., & Sun, Y. (2020). Nonparametric Kullback-Leibler distance-based method for networks intrusion detection. 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI). doi:10.1109/ icdabi51230.2020.9325642

Bullard, C., & Gerth, J. (2014). PCR - A new flow metric: Producer Consumer Ratio [PowerPoint slides]. FloCon 2014. https://qosient.com/argus/presentations/Argus.FloCon.2014.PCR.Presentation.pdf

Casalicchio, E., & Perciballi, V. (2017). Measuring Docker Performance. International Conference on Performance Engineering. https://doi.org/10.1145/3053600.3053605

Casalicchio, Emiliano & Perciballi, Vanessa. (2017). Auto-Scaling of Containers: The Impact of Relative and Absolute Metrics. 10.1109/FAS-W.2017.149.

Chhetri, P., Kam, B., Lau, K. H., Corbitt, B., & Cheong, F. (2017). Improving service responsiveness and delivery efficiency of retail networks: A case study of Melbourne. International Journal of Retail & Distribution Management, 45(3), 271-291. https://doi.org/10.1108/IJRDM-07-2016-0117

Claise, B., Cisco Systems, ETH Zurich, Trammell, B., & Aitken, P. (2013, September). RFC 7011 - Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. IETF RFC 7011. https://tools.ietf.org/html/rfc7011

Committee on National Security Systems. (2015). Committee on National Security Systems (CNSS) Glossary (CNSS Instruction 4009). National Security Agency, Fort George G. Meade, MD.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). MIT Press. Retrieved from https://sd.blackball.lv/library/Introduction_to_Algorithms_Third_Edition_(2009).pdf

Coull, S. E., Monrose, F., Reiter, M. K., & Bailey, M. (2009). The Challenges of Effectively Anonymizing Network Data. 2009 Cybersecurity Applications &Amp; Technology Conference for Homeland Security. https://doi.org/10.1109/catch.2009.27

Dempsey, K., Chawla, N. S., Johnson, A., Johnston, R., Jones, A. C., Orebaugh, A., Scholl, M., & Stine, K. (2011). Information security continuous monitoring (ISCM) for federal information systems and organizations (NIST Special Publication 800-137). National Institute of Standards and Technology. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-137.pdf

Diestel, R. (2017). Graph Theory (5th ed.). Springer-Verlag. Retrieved from https://web.xidian.edu.cn/zhangxin/files/20020101_111456.pdf

Fan, J., Xu, J., Ammar, M. H., & Moon, S. B. (2004). Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme. Computer Networks, 46(2), 253–272. https://doi.org/10.1016/j.comnet.2004.03.033

Gordon, L., Loeb, M. (2002). The Economics of Information Security Investment. ACM Transactions on Information System Security. 5. 438-457. 10.1145/581271.581274. Retrieved from https://www.researchgate.net/publication/220593665_The_Economics_of_Information_Security_Investment

Greenwald, J. (2013, November 4). Cyber security framework welcomed; Voluntary plan may spur broader coverage. Business Insurance, 47(22), 0003. https://link.gale.com/apps/doc/A348425719/ITOF?u=maine_usm&sid=summon&xid=fa6a5c0a

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), Advances in Neural Information Processing Systems 30 (pp. 1024-1034). Curran Associates, Inc. https://cs.stanford.edu/people/jure/pubs/graphsage-nips17.pdf

Horngren, C. T., Datar, S. M., & Rajan, M. V. (2015). Cost Accounting: A Managerial Emphasis (14th Global ed.). Retrieved from https://vera.staff.unri.ac.id/files/2015/11/Cost-Accounting-A-Managerial-Emphasis-by-Horngren-Datar-Rajan-14th-Global-Edition.pdf

Li, S., Du, S., Huang, W., Liang, S., Deng, J., Wang, L., Huang, H., Liao, X., & Su, S. (2020). A light-weighted data collection method for DNS simulation on the cyber range. KSII Transactions on Internet and Information Systems, 14(8), 3501-3518. https://doi.org/10.3837/tiis.2020.08.020

Li, A., Yang, X., Kandula, S., & Zhang, M. (2010). CloudCmp: Comparing public cloud providers. In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement (pp. 1-14). https://doi.org/10.1145/1879141.1879143

Maidana, R. G., Parhizkar, T., Gomola, A., Utne, I. B., & Mosleh, A. (2023). Supervised dynamic probabilistic risk assessment: Review and comparison of methods. Reliability Engineering & System Safety, 230, 108889. https://doi.org/10.1016/j.ress.2022.108889

National Institute of Standards and Technology. (2018). The five functions [PowerPoint slides]. Retrieved from https://www.nist.gov/document/thefivefunctionspptx

O'Reardon, M., & Rendar, M. (2020). Managing Security Risk: How COVID-19 Pandemic and Work-from Home Arrangements Pose New Security Considerations. Employee Relations Law Journal, 46(2). https://www.pillsburylaw.com/images/content/1/4/141663/Managing-Security-Risks.pdf

Penukonda, Qubeb & Paramasivam, Ilango. (2021). Design and analysis of behaviour based DDoS detection algorithm for data centres in cloud. Evolutionary Intelligence. 14. 10.1007/s12065-019-00244-3.

Potdar, A., Narayan, D G., Kengond, S., & Mulla, M. M. (2020). Performance Evaluation of Docker Container and Virtual Machine. Procedia Computer Science, 171, 1419–1428. https://doi.org/10.1016/j.procs.2020.04.152

Rajarathinam, V., Chellappa, S., & Nagarajan, A. (2015). Conceptual Framework for the Mapping of Management Process with Information Technology in a Business Process. The Scientific World Journal, 2015, 1–7. https://doi.org/10.1155/2015/983832

Reijers, H. A. (2021). Business Process Management: The evolution of a discipline. Computers in Industry, 126, 103404. https://doi.org/10.1016/j.compind.2021.103404

Ross, R., Pillitteri, V., Dempsey, K., Riddle, M., & Guissanie, G. (2020). Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations (NIST Special Publication 800-171, Revision 2). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-171r2

Sandhu, K. (2021). Advancing Cybersecurity for Digital Transformation. Handbook of Research on Advancing Cybersecurity for Digital Transformation, 1–17. https://doi.org/10.4018/978-1-7998-6975-7.ch001

Santos, O. (2016). Network Security with Netflow and IPFIX: Big Data Analytics for Information Security [Kindle E-book]. Cisco Press. Retrieved January 30th 2021, from https://Amazon.com

Senate Committee on Homeland Security and Governmental Affairs. (2018). Cyber threats facing America: An overview of the cybersecurity threat landscape : Hearing before the committee on homeland security and governmental affairs, united states senate, one hundred fifteenth congress, first session, may 10, 2017. United States Government Publishing Office. https://www.govinfo.gov/content/pkg/CHRG-115shrg27390/pdf/CHRG-115shrg27390.pdf

Sikeridis, D., Papapanagiotou, I., Rimal, B. P., & Devetsikiotis, M. (2018). A Comparative Taxonomy and Survey of Public Cloud Infrastructure Vendors. arXiv preprint arXiv:1710.01476.

Von Solms, R., & van Niekerk, J. (2013). From information security to cyber security. Computers & Security, 38, 97-102. https://doi.org/10.1016/j.cose.2013.04.004 Retrieved via https://profsandhu.com/cs6393_s19/Solms-Niekerk-2013.pdf

Whitman, M. E., & Mattord, H. J. (2018). Principles of Information Security (6th ed.). Cengage Learning. Retrieved from http://almuhammadi.com/sultan/sec_books/Whitman.pdf

Yu, L., Zwetsloot, I. M., Stevens, N. T., Wilson, J. D., & Tsui, K. L. (2019). Monitoring dynamic networks: A simulation-based strategy for comparing monitoring methods and a comparative study. (). Ithaca: Cornell University Library, arXiv.org.