ASSESSING SECURITY IN THE MULTI-STAKEHOLDER PREMISE OF 5G: A SURVEY AND AN ADAPTED SECURITY METRICS APPROACH

by
Muhammad Shafayat Oshman

A thesis submitted to
the Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of
the requirements for the degree of

MASTER OF COMPUTER SCIENCE

School of Computer Science

at

CARLETON UNIVERSITY

Ottawa, Ontario
January, 2022

Abstract

The fifth-generation (5G) mobile telecom network has been a focal point of research amongst the community in recent times. Features and characteristics like dynamicity (which enables the dynamic allocation of resources to stakeholders on a need basis), low latency, higher network capabilities etc., makes 5G attractive to telecom operators. But as is the case with any new technology, new threats have emerged and made their way into the ecosystem. Therefore, we have a need to evaluate these threats and evaluate the state of security of 5G deployments. Security metrics provide a pathway to quantitatively measure the security posture of an environment by taking these new threats into account. Existing literature contains a plethora of security metrics proposals designed for different system setups. We need to analyze if and how existing security metrics can be applied to a 5G environment. As such, this thesis aims to do a state-of-the-art survey to explore the diverse set of security metrics in literature, understand the process of deriving security metrics. We then employ this knowledge to investigate the factors that hinder the usage of existing security metrics in a 5G environment, identify solutions from existing academic proposals on how to address those factors, propose an adapted security metrics approach to address the multi-ownership nature of 5G and discuss the design and analysis of Argus-5, a tool designed to verify the feasibility of the adapted security metrics approach and simulate various scenarios.

Acknowledgements

All praise to the Almighty for all the blessings He bestowed me with.

First and foremost, I would like to express my eternal gratitude to my supervisor, Dr. Lianying Zhao for his constant support, guidance and words of encouragement. I would have never made it this far without the help that I have received from him over the course of my studies. His insights have helped me tremendously in shaping my research. I can confidently say that whatever growth I have achieved as a researcher has been solely due to the guidance of Dr. Zhao.

I would like to also thank my defence examiners, Dr. Guy-Vincent Jourdan and Dr. AbdelRahman Abdou for their feedback and suggestions during the oral examination, which helped me to further strengthen the quality of my thesis and Dr. Alan Tsang for chairing my defence. I want to send over my thanks and regards to my industrial supervisors from Ericsson, Dr. Makan Pourzandi, Dr. Mengyuan Zhang, Dr. Fereydoun Farrahi Moghaddam and Dr. Taous Madi, for their input to my research as well as the discussions during the time of our collaboration.

Lastly, I would like to thank my parents, friends and family members for their support during stressful times and for providing me with the inspiration to work harder.

Prior Publication

A publication has arisen as a direct result of the research in this thesis. While these works represent joint contributions of all authors, any sections reproduced in this thesis are the sole work of the thesis author, with suggestions and positioning contributions by co-authors. The work is listed below.

Chapter 4 contains text and ideas from our paper "Towards 5G-ready Security Metrics" [151], co-authored with Lianying Zhao, Mengyuan Zhang, Fereydoun Farrahi Moghaddam, Shubham Chander and Makan Pourzandi and published at the IEEE International Conference on Communications (ICC) 2021.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

$5GC$ 5G Core Network

AMF Access and Mobility Management Function

$AUSF$ Authentication Server Function

CPE Common Platform Enumerationl

CSP Cloud Service Provider

$CUPS$ Control and User Plane Separation

CVE Common Vulnerabilities and Exposure

$CVSS$ Common Vulnerability Scoring System

$HARM$ Hierarchical Attack Representation Model

$HTTP$ HypterText Transfer Protocol

$ICMP$ Internet Control Message Protocol

KPI Key Performance Indicator

LTE Long-Term Evolution

LVI Load Value Injection

$MANET$ Mobile Ad Hoc Network

MEC Multi-edge Access Computing

MPL Mean of Path Lengths

$MTFF$ Mean Time to First Failure

$MTTB$ Mean Time to Breach

$MTTR$ Mean Time to Recovery

$MVNO$ Mobile Virtual Network Operators

NDN Named Data Networking

NFV Network Function Virtualization

NRF Network Repository Functions

NSM Network Security Metrics

NVD National Vulnerability Database

$OSPF$ Open Shortest Path First

PDN Packet Data Network

PGW Packet Gateway

RAN Radio Access Network

$RBAC$ Role-based Access Control

RTU Remote Terminal Unit

$SCAP$ Security Content Automation Protocol

SDN Software-defined Networks

$SFTP$ Secure File Transfer Protocol

SGW Serving-Gateway

SGX Software Guard Extension

SMF Session Management Function

$SNMP$ Simple Network Management Protocol

SOA Service Oriented Architecture

UDM Unified Data Management

UE User Equipment

UPF User Plane Function

VNF Virtual Network Function

VO Virtual Operator

Chapter 1

Introduction

The fifth-generation (5G) of telecom networks has created lots of discussions in industry and academia by offering flexibility and higher performance, unseen in the previous generations. According to the 2021 Ericsson mobility report [28], the number of 5G subscriptions worldwide will cross 500 million this year, more than doubling from 2020 and in 2022, number of 5G subscriptions will be more than 1 billion, 2 years less than what it took 4G to reach that number. This statistic is a testament to the popularity of 5G. Some of the unique features offered by 5G include, but are not limited to, dynamicity [106], which is achieved through network slicing [48] and multi-edge access computing [133]. 5G also attributes a higher data rate [79], about 10-100 times more capability than existing 4G networks and low latency, which can accommodate high-performance use cases like industrial applications and massive IoT applications [52]. We also have multiple stakeholders sharing the same platform, as opposed to the traditional single-ownership model of the previous telecom networks.

1.1 Problem Statement and Motivation

It is evident that 5G brings in a whole new template, concerning performance, ownership model and network landscape. However, with the new benefits, new threats have also emerged. Therefore, we have a need to identify these threats and evaluate the state of security of 5G environments. Hence, we dive into the world of security metrics and see if we can come up with an adapted security metrics for 5G. However, security metrics has been a long debated topic, with opposing views such as security metrics is a weak hypothesis [137]. On the other hand, we also come across supporting arguments that while security metrics may not be comprehensive, we still need some metrics to decide which components to go for, some basis for comparison. We can consider an example: if a system administrator has two deployment scenarios, A and

B and if he has to choose which deployment to proceed with, some data is needed for comparison. Is A more secure than B?

This dilemma can be solved through some metrics. In this case, it would have been difficult to come up with an answer without taking some metrics into account. Although there might be some skepticism regarding security metrics, we still need it to make decisions regarding deployments, security patches etc. To better understand why we need metrics, we can draw an analogy from the insurance sector: in the case of insurances, the insurance companies need to collect data like age, weight, health history, and combine all these to come up with a metric value to adjust premium for the insurance policyholder.

Some curiosity might also arise regarding the need for an adapted security metric approach because we could simply take existing metrics and apply it to 5G. There are several security metrics designed for simple to complex IT environments but the issue is: there might be gaps which may hinder the applicability of existing security metrics. We attempt to identify the gaps between 5G and existing security metrics and what are the challenges that show up when we try to apply existing security metrics to a 5G/NFV environment. For example, existing security metrics are designed to evaluate individual systems. However, 5G is a collection of several systems belonging to different stakeholders. Also, 5G/NFV demonstrate a layered structure [37], with the physical infrastructure at the bare bottom, and the network functions hosted on top of the physical layer and the relations between these layers are not as simple or straightforward as the technicalities between them prevent data sharing, which complicates the process of deriving the metric. We can set a local component as a target but then it needs to consider the effect of surrounding stakeholders, which is difficult due to the trust boundaries.

Figure 1.1 shows a simplified overview of the 5G environment, which is one possible deployment scenario of 5G.

An observation from Figure 1.1 is that 5G shares a similar layered nature like public cloud infrastructures. An argument could be made that since 5G shares lots of similarities with cloud, 5G security could share similar issues faced by cloud security, hence cloud security metrics could also be a possible candidate to be applied to 5G

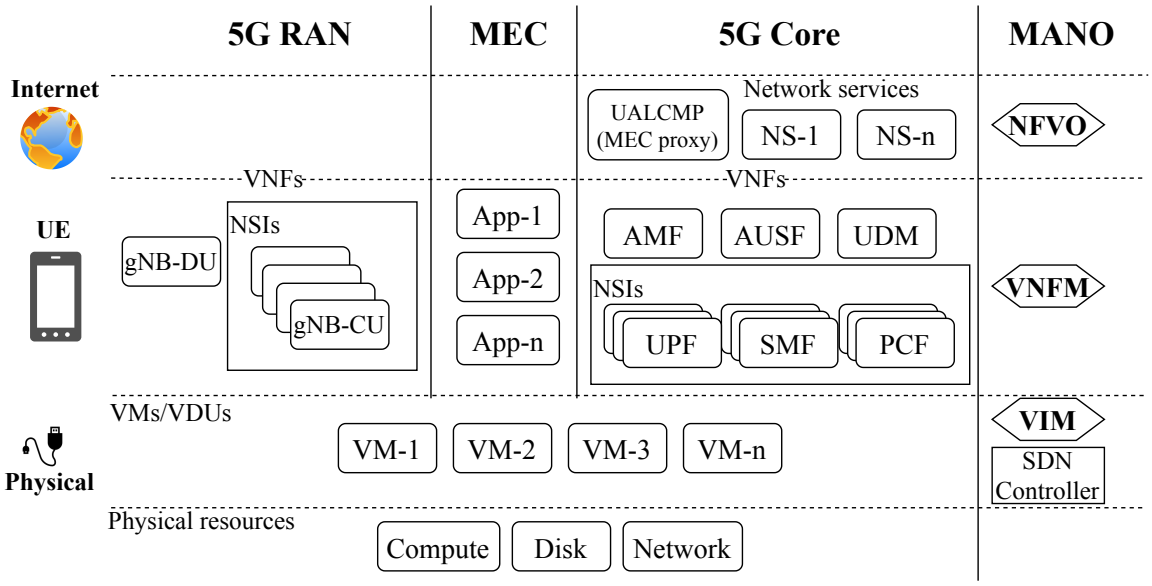

Figure 1.1: An Overview of a possible 5G environment

environments. While that statement is partially true, there are two factors that also need to be addressed when considering applicability of cloud security metrics in 5G environments:

- Telecommunication networks brings in attack vectors [60], [72], [120] such as attacks originating from the base stations, which are not seen in cloud (more discussion in Chapter 4 Section 4.3)

- Cloud security partially applies to 5G as there exists a choice to deploy 5G infrastructure on top of public cloud infrastructure i.e., cloud provider hosting the physical infrastructure at the bottom, while communication service providers and virtual operators rent the cloud infrastructure to deploy network functions as well as provide services to end users. Although they share similar characteristics, aspects such as multi-ownership are more widely formulated in 5G due to the established business implications. However, the multi-stakeholder dilemma has not been addressed in cloud as well. While cloud security issues mostly focuses on co-residency problems, such as how VM placements could be manipulated to gain side-channel advantages (from the adversary perspective, more discussed in subsection 3.4.7), multi-stakeholder dilemma has not been

visible in academic proposals centering around cloud security metrics, therefore allowing no such security metrics derivation in a cloud environment.

Hence, the adapted security metrics approach to be discussed in our thesis, could also be used for cloud, as a use case.

1.2 Contributions

This thesis makes 3 contributions: a security metrics survey, a proposal to adapt existing security metrics for 5G and a tool to validate the proposed adaptation. Details as follows:

Contribution 1: We conduct a survey into the state-of-the-art of security metrics. This survey has two components:

- We investigate the process of metric derivation: identifying the risks that arise from vulnerabilities and other weaknesses in the system, quantifying the risks, model the attacks that may occur from exploiting the vulnerabilities, aggregate the quantified values and derive the metric.

- We investigate the gaps between the existing security metrics and the 5G telecommunication network

Contribution 2: We propose some potential solutions to the identified gaps between the existing security metrics and 5G, including the details of our novel proposal of Modular Security Metrics, an adapted security metric approach to address the multi-ownership nature of 5G.

Contribution 3: We discuss the design and purpose of Argus-5, which facilitates the generation of graphs, containing representation of resources present in a 5G network with modular support and simulation pertaining to calculation of security scores.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- **Chapter 2:** Provides a brief discussion of 5G/NFV technologies and terminologies.
- **Chapter 3:** (Contribution 1) Conducts a survey of the state-of-the-art of security metrics and identifies the gap between existing security metrics and new-generation telecom networks.
- **Chapter 4:** (Contribution 2) Proposes potential solutions from the existing academic proposals for the identified gaps between existing security metrics and 5G, as well as propose Modular Security Metrics.
- **Chapter 5:** (Contribution 3) Demonstration of Argus-5, an automated resource graph and attack graph generation tool for 5G, that can accommodate various simulation parameters and scenarios.
- **Chapter 6:** Presents the simulation results done by the proposed tool, Argus-5.
- **Chapter 7:** Concludes the thesis and discusses future directions of this research.

Chapter 2

5G Preliminaries

In this section, we provide brief descriptions of technologies and terminologies related to 5G. We take a look at the technologies like network slicing, network function virtualization, software-defined networking etc. The technologies and terminologies defined here will be used in the following chapters.

2.1 5G Architecture

Compared to the traditional single owner model of previous generation of telecom networks, 5G utilizes a multi-ownership model. Each stakeholder is hosted in a shared platform, separated from each other for privacy and security. While the previous telecommunication generations provided static allocation of resources to the service seeking entities, 5G employs a more dynamic nature [83], which allows 5G operators to allocate resources to users dynamically according to the need, thanks to network slicing [31] and multi-access edge computing [133]. This flexibility and performance boosting capabilities and dynamicity effectively turn the 5G space into a multi-party play i.e. we have multiple stakeholders in a common platform, sharing the space. At the bare bottom, we have the Cloud Service Provider (CSP), who own the hardware and the infrastructure. Communication service providers act as the operator and rent the infrastructure from the cloud vendors and deploy their network functions within. The mobile virtual network operators (MVNO) rents the virtual network from the CSPs and provide service to the consumers or end users. The flexibility of 5G environment and platform sharing is achieved by deploying a clear boundary between the application logic and the network infrastructure through Network Function Virtualization (NFV) [95] and Software-defined Networks (SDN) [65].

2.2 Components in a 5G system

The 5G system design is service-based, unlike prior generations [86]. Architectural elements are described as network functions that offer their services via interfaces of a common framework to any network functions that are allowed to access the given services when permitted. Every network function can use Network Repository Functions (NRF) [119] to learn about the services provided by other network functions. This architectural model was designed to allow deployments to make use of the newest virtualization and software technologies, as well as principles like modularity, reusability, and self-containment of network activities.

The 5G System (5GS) will have three main components as defined below:

- 5G Radio Access Network (5G-RAN)
- 5G Core Network (5GC)
- User Equipment (UE)

2.2.1 5G Radio Access Network and 5G Core Network

5G-RAN. A 5G Radio Access Network (RAN) uses 5G radio frequencies to provide connectivity to User Equipment (UE) such as mobile device, such as a cell phone, tablet, or modem, to deliver services and applications. The RAN consists of various types of equipments including small cells, network towers, base stations known as next generation NodeB (gNB) and dedicated in-building and home systems that connect mobile users and wireless devices to the main core network.

5GC. The composition of the 5GC, also known as Network Functions, have been significantly simplified to enable various data services and requirements, with the majority of them being software-based so that they may be altered according to need. The 5GC consists of the following key network functions:

- The Access and Mobility Management Function (AMF) acts as a entry point for the UE connection such as personal devices, applications etc and oversees authentication, connection, mobility management between the serving network and device. It receives connection and session related information from the UE.

- Session Management Function (SMF) is selected by the AMF for managing the user sessions.

- The User Plane Function (UPF) transports the data (user plane) between the User Equipment (UE) and the external networks.

- The Authentication Server Function (AUSF) allows the AMF to authenticate the UE and grant access to the 5GC.

- Unified Data Management (UDM) function is used for accessing subscription information of the UE.

2.2.2 User Plane and Control Plane

User Plane and Control Plane are part of the 5GC and have been separated in 5G Control and User Plane Separation (CUPS) strategy introduced in the 3GPP specification 14 [74], for flexible placement of User Plane Functions (central, closer to the RAN, etc.). CUPS dissociates Packet Gateway (PGW) control and user plane functions. This allows the data forwarding component (PGW-U), an equipment in the 4G LTE evolved packet core which connects the LTE network to other packet data networks, to be decentralized and mapped to the UPF for the 5G Core. The user plane function consists of a single entity User Plane Function (UPF). It combines functionality from previous EPC Serving-Gateway (S-GW) and PDN-Gateway (P-GW) [58].

The Control Plane in 5G consists of two components - AMF and SMF and the User Plane in 5G consists of UPF.

2.3 Network Slicing

Network slicing is one of the defining features of 5G. Network slicing allows the separation of multiple virtual networks, by slicing the physical network into several virtual networks on a per-service basis [32], operating upon the same physical hardware. These slices can each be allocated to different purposes, increasing performance and

efficiency. The slices can be configured separately depending on their purposes, allowing the operators to modify spectrum capacity, coverage in dynamic nature. A network slice is composed of a collection of network functions and settings that are combined together for a specific use case or business model [150].

2.4 Multi-edge Access Computing (MEC)

As a result of the advent of cloud computing, edge computing [57] moves application hosting from centralised data centres to the network edge, bringing customers and the data generated by apps closer together. Edge computing is widely regarded as one of the most important pillars for achieving the rigorous Key Performance Indicators (KPIs) [126] of 5G, particularly in terms of low latency and bandwidth efficiency. MEC is a set of cloud services that run at the network's edge and execute certain tasks in real-time or near-real-time that would normally be handled by centralised core or cloud infrastructures. MEC brings computational capacity closer to the end user, enabling applications and services that require special connectivity, such as ultra-low latency. MEC functions by transferring data from the cloud to the user. Data does not have to travel as far when certain processing capabilities is moved out of the public cloud and closer to the end user and end devices, which means it may be processed faster. It allows for the acceleration of content, services, and applications by boosting their responsiveness. MEC is critical in enabling telecommunication operators to deliver on their 5G promises by significantly reducing latency, increasing connection speed, enhancing network security, and improving quality of service to end users by bringing computing and processing closer to where data is generated and analysed, as well as where real-time decisions are made.

2.5 Network Function Virtualization (NFV)

NFV is a key enabler in 5G. The goal behind NFV [95], [42], [146] is to decouple actual network equipment from the functions that execute on it. This means a network function, such as a firewall, can be routed to a service provider as a software instance. Many different types of network equipment can then be consolidated into servers of

increased capacity, switches, and storage, which can be hosted in data centres. A service can be decomposed in this way into a series of Virtual Network Functions (VNFs), which can then be implemented in software operating on one or more industry standard physical servers. The VNFs can then be moved and instantiated at multiple network locations, for example, to introduce a service targeted at clients in a certain geographic location, without having to buy and install new hardware. NFV enables service providers to have more flexibility to further open up their network capabilities and services to users and other services, and the ability to deploy or support new network services faster and cheaper, which leads to better service quality.

2.6 Software-defined Networks (SDN)

Software-defined networking (SDN) [61] is designed to make networks more flexible and agile. SDN is an approach to separate network control and forwarding planes, which enables network control to be directly programmable and the bottom infrastructure is abstracted for applications and network services. The main idea behind SDN is to move the control structure away from network hardware. This will enable external control of data through software entity, termed as controller. The controller is placed between hardware and applications. Essentially, in 5G networks [117], the administrators will then be able to manage the networks and introduce new services or changes. The enriched ability of 5G SDN is to provide network virtualization, automation, and create new services on top of virtualized resources. The 5G SDN architecture is dynamic and easily manageable through external controls, making it perfect for the dynamic, high-bandwidth nature of 5G use cases.

2.7 Isolation: Technical and Institutional

In the typical cloud scenario, to ensure that none of the hosted parties can access data of another party, the parties are isolated from one another, both technically and institutionally. What isolation refers to is blocking the flow of information from flowing over from one party to another. Technical isolation, referring to modes of isolation through use of technology, either software or hardware, can be achieved

by leveraging the Trusted Computing technologies such as Intel SGX [113], Intel Virtualization Technologies (VT-x) [136], Intel Virtual Machine Extension (VMX) [46], AMD-V [19], role-based access control (RBAC) [29] etc. Institutional isolation is achieved by the parties entering into mutual agreements and rules/regulations defined by various standards, that ensure the providing party is not able to access data of the service seeking party.

2.7.1 Technical isolation technologies

We take the example of Intel SGX to demonstrate how technical isolation leverages such technologies to attain its goals. Intel's Software Guard Extensions (SGX) [113] is the latest addition to the domain of trusted computing technologies, which aim to solve the secure remote computation problem by leveraging trusted hardware in the remote computer. The trusted hardware establishes a secure container, and the remote computation service user uploads the desired computation and data into the secure container. The trusted hardware protects the data's confidentiality and integrity while the computation is being performed on it. An enclave (secure container) in SGX only includes the computation's private data and the code that operates on it. For example, users could upload encrypted photographs to a cloud service that does image processing on private medical images. The encryption keys would be sent by the users to software running within an enclave. The code for decrypting images, the image processing method, and the code for encrypting the results would all be stored in the enclave. Outside the enclave, the code that receives and saves the encrypted images reside.

2.8 Multi-tenancy in Cloud

In cloud computing, multi-tenancy allows customers to share computing resources in a public or private cloud [6]. The data of each tenant is kept separate from those of other tenants. The structure of multi-tenant clouds can be compared to that of an apartment building. Within the agreement of the entire building, each resident has access to their own apartment, and only authorised individuals are allowed to enter the specific units. However, utilities such as water, power, and common rooms

are shared by the entire building. In cloud computing, multi-tenancy makes a larger pool of resources available to a broader group of individuals without compromising privacy or slowing down applications. Cloud computing's virtualization of storage locations offers for greater flexibility and accessibility from practically any device or location. Multi-tenancy ensures efficient usage of system resources, but also gives pathway to potential attack vectors, as discussed in later sections. Multi-tenancy in cloud computing allows for flexibility and ease of access to resources by making a pool of resources available to a group of tenants without fixed and dedicated allocation, without sacrificing privacy and security or slowing down applications.

Chapter 3

Risk Assessment and Security Metrics in IT Environments

Cyber threats are evolving every day. Attackers are constantly looking for new ways to attack individuals and organisations alike. It's becoming easier for amateur hackers to access high-level malicious software, with the seemingly growing popularity of "malware as a service" [24]. Malware as a service refers a paid service that rents out botnets solely for the purpose of propagating malware. Available through the dark web, it is essentially a complete package of malware software and servers that is ready to deploy instantly – and, more troubling fact, MaaS is simple enough for amateurs to use. MaaS is just one example of a growing number of threats that are emerging everyday, threatening the security ecosystem. Hence, cybersecurity must evolve as well. Evolving security must deal with present day threats as well as account for new risks that might show up in the future. Risk management is of utmost importance so as to minimize potential losses. The risk management cycle may involve risk identification, risk mitigation/transfer, and risk monitoring. This is where security metrics come into play. In all phases of the cycle, we need to be able to quantitatively understand the risk [141]. Metrics are needed to ensure that mitigation techniques deployed are also fulfilling their intended purposes. Without quantitative data, it would be difficult to understand the level of risks as well as design mitigation techniques for each risk. In the words of Lord Kelvin, a distinguished British mathematical physicist and engineer,"when you can measure what you are speaking about and express it in numbers, you know something about it" [13], which shows that risk quantification provides us a better insight regarding how to handle the risk. To understand the nature of security metrics and risk quantification, we looked into existing literature to examine several different surveys conducted on different domains of security metrics. Pendleton et al. [104] conducted an extensive survey on System Security Metrics, based on the understanding of attack-defense interactions, and focused on how a system security

state can evolve as an outcome of cyber attack-defense interactions. HU et al. [44] conducted a survey of attack graph based network security metric, showing extensive work on how metric models are introduced and then proceeding to highlight security metric models based on attack graphs. Wu et al. [143] conducted a survey on the security of mobile ad hoc network (MANET), showing the risk assessment process of MANET. Ramos et al. [112] conducted survey on network security metrics (NSMs) based on models, that allow to quantitatively evaluate the overall resilience of networked systems against attacks. After examining several surveys, we observed that they were limited by dimensionality, which is to say that the focus of the surveys were only on one or two specific categories (system, network etc) of security metric. To the best of our knowledge, we did not find any surveys that were comprehensive or covered multiple fields of metrics. Since IT systems consist of several sub-components, such as various softwares, network protocols, rule set, configuration policies etc, each having their own metrics to determine security, a survey is needed that would cover the state-of-the-art of security metrics from multiple angles and not limited by dimensionality, but rather cover the discussion based on multiple risk factors such as vulnerabilities, policy issues etc, which will help in a better understanding of these metrics. It is important to not be limited by dimensionality because risk is relative to the observer and is a subjective thing - it depends upon who is looking, hence it is beneficial to look at it from all angles to get a comprehensive view of perceived security. We use this as a motivation to conduct a new survey of security metrics. The goal of this survey will be to examine security metrics in relevance to risk management: identifying risks, measuring risks, quantifying the risks, modelling the attacks possible through the exploitation of the risks and finally derive the metric.

3.1 Risks in Cybersecurity

Risks refer to the possibility of incurring loss, which can be in terms of data, monetary loss, loss of goodwill etc. In large-scale systems, risk is defined as a triplet [21]: the specifics of what can go wrong, the likelihood that this event will occur, and the consequences when the risk is exploited by outside threat, and this triplet can be expressed as threat $\times$ vulnerability $\times$ consequence. To expand on the triplet, threat

can be defined as any circumstance or event with the potential to adversely impact organizational operations through unauthorized access, data modification etc. [130]. NIST [98] defines vulnerability as weakness in a system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat. Risks can originate from vulnerabilities, architectural weaknesses, misconfigurations etc. Consequences can be defined as the aftermath of exploitation of such risks. Consequence dictates the cost (loss) faced by the victim of such exploits, which occur typically in the form of data loss, loss of financial value etc. Logically, a clear and quantitative way of expressing and measuring risks is needed so that it can be properly judged, along with all other implications, to facilitate proper decision making. This decision making influences how the risks are mitigated and monitored, for example a risk with higher severity would call for a faster response than a risk with low severity. The quantitative requirement can be fulfilled by security metrics, which can act as indicators on how successful the risk management has been. Security metrics present a set of analytics that can potentially help system administrators to tailor the risk management to their own needs.

3.2 Understanding Security Metrics

The quantitative demand of cybersecurity risk management can be fulfilled by using security metrics. As threats and risks emerge, system owners and organization divert funds and resources to manage and mitigate them. However, there remains a dilemma regarding the effectiveness of the risk and threat mitigation techniques.

- Do I need to divert more resources to patch the threats?
- How is the security level after the latest rounds of patches?
- How does a new software or component installation affect the security of the deployment?
- How vulnerable am I to any future attacks?
- How many security controls do I need to place to be safe?

Security metrics helps answer all these questions by helping to quantify the risks. A metric refers to assigning a value to an object while measurement is the process of estimating attributes of an object. According to Yee et al. [145], security metrics are needed to:

- Provide quantative analysis of security measures.
- Support decision making processes such as where to divert more resources etc.
- Assist in the maintenance and compliance of security policies as defined by the security experts (how often do users need to change their passwords).
- Gradually improve the security posture of the environment and increase the capability of facing threats more effectively.

In the later sections, we will attempt to break down the derivation process of security metrics: how to measure risks, factors associated with measuring risks, how to model the risks in a quantified manner and derive the metric.

3.3 Measuring Risks for Security

The attempt to measure the level of security of a computing environment (e.g., a computer) is naturally complicated by the wide scope and is not derived purely based on established scientific methodologies. If security was more scientific in nature, security could be analyzed, just as performance is analyzed, and security metrics could be systematically derived and predicted, just as performance metrics are derived and predicted. This is further aggravated by the fact that security metrics are often based on assumptions but very little comparison done against empirical data [137], as opposed to formal scientific experiments done and tested on several test subjects based on established scientific methodologies, with comparison and verification done with data. Additionally, the security metrics premise is centred around the actions and behaviour of human adversaries, which are unpredictable in nature [23]. According to Stolfo et al. [129] and Yee et al. [145], there are two reasons for the distance between security metrics and science:

- Security metrics must be easy to comprehend, yet when reduced to their simplest form, they fail to represent the essence of what is happening in the actual world. There is a disconnect between mathematical security metric derivations and real-world implementations. RSA implementations, for example, are frequently broken due to side channels (such as timing and power consumption), inadequate random-number generation, insecure key storage, and programming errors, despite mathematical derivations demonstrating the algorithms' resilience.

- It is vital to comprehend attacker ingenuity in devising new attacks, in order to comprehend the security of a computer system. These calls for measurements that presume the attacker is capable of launching any attack, which are difficult to design because we'd have to know about every possible attack. However, we will never know all of the methods an attacker can corrupt the system, therefore it is difficult to fully cover all attack vectors.

To facilitate our security metrics discussion, we consider two directions in our survey:

- Generic security metrics, the security level of the computing environment, as opposed to application-specific goals. We are interested in the likelihood of an attack compromising the owner-expected behavior of the environment. However generic security metrics are not quite straightforward because owner-expected behaviour will differ from one system to another. What might be expected by one owner of a system might be a security threat in another system. To understand how the owner expects their system to behave, a threat model must be defined, which encompasses what the owner considers an ideal behaviour and what behaviour inside the environment is considered a threat or suspicious.

- Relative security, which compares how secure one system is compared to some other system or measurements. As long as we cannot model in all the factors affecting security as in an ideal case, there will be no absolute security. For example, as mentioned by Dambra et al. [23], prediction can be complicated if future attacks are from sophisticated and motivated adversaries (unpredictable).

Still, the purpose and usefulness of current security metrics lie in the fact that they provide a basis for comparison. For example, a metric might be able to measure how much more secure a system f′ is than a system f, where f′ is some transformation of the original system. For security, we could note that a system with m independent defenses is f(m) times is more secure. Conversely, a system that accepts m times as many different inputs will be g(m) less secure [129]. In a different vein, we can scale a module's relative security value by a function that depends on the module's development and testing environment.

3.3.1 Threat Model for Security Metrics

Before identifying risks and develop security metrics, the threat model of the system comes into question as what risks are going to be evaluated is going to depend on what the owner considers as "threats". Security metrics programs are often based on assumptions like there is a secure way to manage any system, and the task of security management is to maintain that state [103], [9]. Security metric values and percentages are difficult to interpret without a defined threat model that answers the question "Secure against what?" [81]. It would be impossible to tell whether specific attacks are stopped, to estimate the number of hosts that may be compromised for a specific threat type, or to assess whether different defensive tactics, such as patching known vulnerabilities more quickly, can significantly lower risk.

A good threat model must establish the following [80]:

- what security conditions must be observed to determine the risk of an attack
- how to compute the risk of an attack on the basis of observed security conditions
- how to prioritize the risk of an attack across network entities, such as persons, devices, and accounts
- how to design the network so that it is easy for network administrators to take actions that mitigate risk and to eliminate security conditions that enable attacks

It is observable that traditionally the threat models exclude platform owners and there exists limited definition of insider threat [12], which leads to widely believed notion that threats and attacks generally originate from outside the network to compromise resources inside the network, underestimating the threats posed by the insiders such as the owners, users [93].

3.3.2 Scope of Risk Measurements

Along with the threat model, the scope of risk measurements should also be defined when calculating the metric. The scope of measuring risks depends on the goals of the administrator. While measuring risks, we have to take into account the life cycle of a deployment, to understand at what point the metric becomes relevant for use. Any system undergoes several cycles of design, development, testing and then gets deployed. Each of the stages of this cycle have their own criteria of management. The system life cycle begins with a blueprint, consisting of infrastructure design. The system is then programmed by software and hardware components and deployed in an environment. Once the system has been deployed we are able to collect real data regarding the security. Data can also be collected in run-time. In this life cycle, security metrics are concerned with blueprint and deployment data only. The reason behind this is security metrics requires static data to compute as well as dynamic computation requires constant monitoring of the system, which can call for additional resources and can be expensive for the owner of the environment. Data collected during blueprint or design phase can be used to improve or optimize the deployment scenario and data collected during deployment can be used to optimize security controls and patches. One thing to note here is that security metric is always concerned with the future i.e., we use data available in present to compute metrics, and predict what might happen in future, indicating risks and potential loss factors. We have computer forensics to uncover incidents that happened in the past. We use detection systems such as intrusion detection to detect security events happening in the present or vulnerability detectors to detect vulnerabilities present in the system/network. We use these data to compute the metric for future. When an actual attacker takes advantage of the vulnerabilities to mount the attack, an

exploit happens. Measuring exploits/attacks (through forensics) does not directly indicate risks, as the bad consequence has already happened. Nonetheless, it does provide statistical and analytical information as to what is (not) likely to happen in the future.

3.4 Identifying the Risks

The first step to measuring security is to identify the sources of the risks. This starts by marking the areas in the system that are vulnerable or open to intrusion by the attacker. In the following subsections, we continue to investigate the sources of risks and losses.

3.4.1 Attack Surface

Attack surface can be defined as the exposed interfaces of the system that are visible to the outside world, which can be used by the attacker to enter into the system. It is a widely held notion that "the smaller the attack surface, the more secure the system [89]". The way the system is designed and set up already determines where/how it can be attacked. For instance, a system taking input may be potentially more vulnerable than a system taking no input, as it needs to sanitize input properly.

As a well known security evaluation method, attack surface utilizes the intrinsic properties from a system, i.e., the resources an attacker could use to compromise this system. Manadhata et al. [89] proposes a three-dimensional attack surface metric based on the attack vectors from Windows systems, i.e., channel, method and untrusted data item. Intuitively, channel represents possible ways to connect to the system, method refers to functions that could be exploited by attackers, and untrusted data item means data sources that could be manipulated by attackers. The combination of the three types of resources indicates the risk level in one system. Thus, reducing attack surface would intrinsically enhance the security level of a system. Even if a system does not have inherent vulnerabilities, its attack surface contributes largely to the possibility of exploitation by attackers. If any attacker successfully mounts an attack through the exposed interfaces, system resources can be modified using these interfaces and thus any resource that is accessible through these exposed

interfaces are also part of the attack surface, even if they are not directly visible to the outside network. If a system has a large number of exposed interfaces, it contributes to the factor of attackability i.e., how likely the system is to be attacked through these exposed surfaces. Securing a system against such attacks would be to decrease the attack surface of the system. Decreasing the attack surfaces require changes in configurations of the system. Any changes to the environment must be compared to the previous version to make sure it is more secure than the previous deployment. Manadhata et al. [89] took such attack surfaces into account and proposed an attack surface metric to measure the security level of different versions or deployments. Furthermore, Manadhata et al. [88] made use of their attack surface metric to show how attack surface of two different versions can be judged by comparing their attack surface, using two open source FTP Daemons as use cases. As decreasing attack surfaces is considered one of the major defenses against such approaches, Kurmus et al. [69] demonstrated that by tailoring the kernel, it is possible to reduce the attack surface by 50-85%. Although this might not be directly related to the attack surface reduction in a generic cloud environment, but this demonstration serves as an example that how attack surfaces are a chief contributor in the overall security of a system, that by making. Sun et al. [132] devised a new protection scheme that utilises a clear distinction between Internal Attack Surface (IAS) and External Attack Surface (EAS) and thus causing an expansion in the EAS with fake network and software vulnerabilities to purposely confuse the attacker from identifying the real attack surface.

The attacks making use of interface-induced vulnerabilities typically interact with the software by connecting to a channel (e.g., socket) or invoking a method (e.g., Application Programming Interface) offered by the software or sending/receiving data items to/from the software. The methods, channels, and data items are considered as resources. An entry-point exit-point framework [89] was introduced to identify relevant resources, where an entry point is a method for receiving data from the environment, and an exit point is a method for sending data to the environment. Each resource may or may not contribute to a software's attack surface. The contribution of a resource to the software's attack surface is the likelihood that the resource can be abused to launch attacks, where the likelihood depends on the privilege needed

for using the resource. The potential damage caused by the abused resource is called damage potential. On the other hand, an attacker needs to make efforts to have privileges to use resources. This metric is specific to an environment, implying that one can compare the attack surfaces of two different versions of the same software running on the same environment. This metric is measured by an absolute scale. Attack surface is not defined dependent on software vulnerabilities. Reducing the attack surface (e.g., uninstalling a software) does not necessarily improve security [97]. Putting the interface-induced vulnerabilities into the context, a desirable metric would be the susceptibility of a software system to certain attacks according to a threat model, perhaps through how the exercise of attack surface is dependent on the features of attack surfaces.

Although the attack surface concept initially focuses on a signal system or software application, researchers have expanded the effort to explore network-level attack surface. We believe the attack surface of a cloud/NFV/5G environment could serve as a general indication of the security level of such large/complex systems. Various researches have shown how the novelty and flexibility of 5G has led to the rise of new attack surfaces [68], [66].

3.4.2 Software Vulnerabilities

While attack surface determines how a system can be attacked, a system can already have vulnerabilities, directly contributing to what the attacker can exploit to mount a successful attack, once gaining entry through attack surfaces. Attack surfaces describes how exposed (through entry/data extraction points to the system) someone is to an attack while vulnerabilities are weaknesses that expose the risk (possible exploits by the attacker) such as improper inputs leading to the bug in a software to be activated, giving escalated privilege to the attacker. The relation between attack surface and vulnerabilities is that to gain entry through the attack surface, the attacker must make use of some vulnerability, for example, a door can be an attack surface (entry point to the building), but to break in through the door, the lock must have a vulnerability that will aid the attacker to compromise the lock security and open the door. The software vulnerabilities are an indication of what has already gone wrong

inside the system, for example, a software with bug in its code, which has now allowed the attacker to utilize race conditions to gain advantage and execute their agenda with escalated privilege. Any system's Achilles' heel is the dormant vulnerabilities that lie inside it, waiting to be exploited. We might have vulnerabilities that appear with changes in attack patterns, that might not have been factors of concern before. For example, a system with all its code well protected would not be considered vulnerable, until the advent of the return-oriented programming (ROP [18], targeting the stack) and data-oriented programming (DOP [51], targeting data objects). Therefore over time, an attack vector of the attack surface may become a vulnerability inside the system, due to improvements of attack techniques. The risk of vulnerability exploitability can be assessed based on two software properties - attack surface entry points and how many components inside the software can be reached/accessed through those entry points [147]. If a vulnerability is found in one of the entrance points or in a function that is called directly or indirectly by the entry points, it is reachable. The damage potential-effort ratio in the attack surface metric, as well as the existence of system calls judged risky, can be used to assess the possibility of an entry point being used in an attack. Because of the growth of the open-source software community, enormous amounts of software code are now available, allowing machine learning data mining tools and other adversaries to leverage rich patterns within software code and mount exploits [78]. Some vulnerabilities can be exploited anytime, regardless of the environmental or deployment factors. Some vulnerabilities need suitable configurations to become active and exploitable for example, Load Value Injection (LVI) attacks on Intel CPU, which is a micro-architectural attack that allows an attacker to inject their own values into the victim's code. The data injection can either be instructions or memory addresses, allowing the attacker to obtain data from Intel SGX. A vulnerability's properties can be studied to find out how that vulnerability can be exploited. Vulnerabilities discovered in a system can naturally be an indicator of the system's security level. A naive way to do so could be counting the number of vulnerabilities. However, not all vulnerabilities pose equal threats to the system. Each vulnerability has its own severity level, determined by its technical nature and indicative of how much risk it poses to that particular system

and thus it becomes important to identify a security metric for that environment. Chandra et al. [17] proposed a framework for identifying the suitable security metric for a software system and if unavailable, develop a security metric for the environment. Researchers have also made applications of neural networks to understand vulnerable coding patterns and semantics, which can work as indicators of potential software vulnerabilities [78].

3.4.3 Vulnerability Chaining

While individual vulnerabilities pose security threats in their stand-alone condition, several vulnerabilities, existing in different components, can be chained together to perform an attack, i.e., attack path. Contrary to the popular perception that only vulnerabilities with high severity pose danger to the system, the chaining of vulnerabilities can combine several low-severity vulnerabilities to create greater damages. A low-severity vulnerability can aid the attacker in getting access to important resources, with which then the attacker mounts bigger attacks. Wang et al. [140] demonstrated a probabilistic security metric that joins all the vulnerabilities in the system and gives a score in the form of probability, outlining which path can be taken by an attacker. Jajodia et al. [53] proposed a topological analysis on how several vulnerabilities can be chained together to mount an exploit.

3.4.4 Protocol Problems

For systems that rely on various protocols such as network, security etc, it introduces additional attack vectors due to protocol vulnerabilities. It is important that the target network/system's protocols have weaknesses in order to carry out a successful assault. These are different than software vulnerabilities because they are not application specific bugs or issues with the code but rather flaws with the protocols being utilized. These flaws are primarily due to two factors [144]. One is the protocol design technique, which will have an influence on the nodes that are executing that protocol if it is attacked. The other stems from the nefarious usage of a legitimate protocol.

Types of network protocols includes:

- Communication protocols are basic data communication tools like TCP/IP and HTTP
- Security protocols like HTTPS, SFTP etc.
- Management protocols maintain and govern the network through protocols such as ICMP and SNMP.

There are several other protocols that govern communications such as industrial communication protocols, such as the IEC 61850 and ModBus protocols [118]. We have seen numerous attacks that have been mounted based on issues with various protocols. Wagner et al. [138] showed an attack where a ClientHello message of SSL 3.0 is modified to look like a ClientHello message of SSL 2.0. This would force a server to rollback to the more vulnerable SSL 2.0. Brumley et al. [14] outlined a timing attack on RSA based SSL/TLS. The attack extracts the private key from a target server by observing the timing differences between sending a specially crafted ClientKeyExchange message and receiving an Alert message inducing an invalid formatted PreMasterSecret. Even a relatively small difference in time allows to draw conclusions on the used RSA parameters. Sosnovich et al. [127] revealed a series of attacks using fundamental protocol vulnerabilities of OSPF, a very widely used protocol for Internet routing, which can harm routing in huge networks with complex topology. Several weaknesses were discovered by INL researchers during examinations of the cybersecurity of real-world process control system and among the most important findings of the cybersecurity evaluations are weak authentication and integrity checks in industrial communication protocols [67]. Between control servers and field devices such as remote terminal units (RTUs) and between other control system components, industrial communication protocols have been found to use inadequate authentication and integrity checks, and in some cases no integrity checks at all [118], which can and have led to attacks such as man-in-the-middle attack, dns spoofing, protocol specific memory corruption etc. Various research proposals regarding attack modelling often fail to address protocol issues as one of the weaknesses and was addressed by Stan et al. [128], where the authors extended MulVAL's (a popular attack graph generating tool) attack modelling to include protocol vulnerabilities.

3.4.5 Policy Issues and Misconfigurations

Even a well protected system will sometimes be compromised due to misconfigurations and/or issues in the policies that govern security, access, operations etc. For example, firewalls are deployed to filter traffic between trusted and untrusted parts of a networks, as well as to monitor their incoming and outgoing interaction with other networks. However, any misconfigured policies in the firewall's rule set will result in unauthorized data gaining access to the system and leading to compromise. By taking such misconfigurations into account, Cuppens et al. [22] proposed a set of algorithms, that assist in the management of policy misconfigurations. The proposal relies on the analysis of the interactions between the filtering rule sets and rewrite a new rule set free of any potential misconfigurations. Access-control policy misconfigurations that cause requests to be erroneously denied can result in wasted time, user frustration, and, in the context of particular applications (e.g., health care), very severe consequences [8]. To capture such changes in policies, Abedin et al. [2] introduced Policy Security Score, a method of analyzing and assigning scores to security policies, by analyzing and comparing different versions of policies and examining their vulnerabilities. Another potential issue is that devices from different vendors may require different methods of configuration [92]. Inevitably, when the size of the network increases, it becomes increasingly complex to manage so many different set of policies or rule sets, catered to each of the equipment designed by distinct vendors, which could lead to inconsistent rule matching between different policies, resulting in allowing wrong traffic to enter the network [84], [41]. Haddad et al. [40] demonstrated several approaches to validate the use of a common language for defining access control policies between all network devices in order to avoid conflicts and inconsistency.

3.4.6 Human Angle of Security Metrics

While vulnerabilities can arise from technical issues, there remains one missing angle: the user of the system itself. Matsumoto et al. [91] presented the malicious administrator problem in Software-Defined Networks, in which one or more network administrators attempt to sabotage routing, forwarding, or network availability by misconfiguring controllers. We discuss two types of user vulnerabilities, as described

by Pendleton et al. [104], in terms of:

- users' reasoning bias
- users' analytical limitation

How prone the user is to phishing attacks or insider threats is a common example of vulnerabilities exposed by the user's reasoning bias, while weak passwords are frequently the result of limited human memory. An attacker can simply crack a weak password, causing the authentication process to fail. Phishing vulnerability is frequently influenced by human reasoning bias or personality factors [122]. The user's online behaviour is also linked to malware vulnerability. Malware is more likely to infect users who frequently install multiple applications. Furthermore, if consumers visit a lot of websites, they are more vulnerable to malware infection [71]. It would be beneficial if we could measure how vulnerable a particular user is to cases of social engineering. It is vital to analyse the key aspects a user relies on to make security decisions, such as the user's disposition, personality, or prejudices, in order to evaluate the overall risk produced by a collection of individual vulnerabilities. Although little research has been done on this topic, it is critical to analyse the relationships between a user's vulnerability and his or her reasoning biases or personality features. Investigating the impact of users' personality traits, reasoning biases, can be utilised to create defence mechanisms to mitigate the user's susceptibility.

3.4.7 Inherent Weaknesses

In addition to software vulnerabilities, a system might have other weaknesses, which make it vulnerable to attacks. One such example could be the co-residency problem in cloud computing, where malicious users utilize side channels, vulnerabilities and extract private information from virtual machines co-located on the same server. When it comes to a virtualized environment, the traditional model of independent computing systems interconnected with communication channels may not apply as is. In cloud, resources are no longer stand-alone, they are more connected and hence have threats emerging from multiple angles. Ristenpart et al. [114] showed that it is possible to map the internal cloud infrastructure and deliberately place a malicious

VM onto the same physical server the target VM is likely to reside on. Having placed the malicious VM co-resident with the victim VM, they showed preliminary results on a variety of cross-VM attacks such as denial of service (DoS), and side channel attacks such as remote keystroke monitoring via timing inference and others.

Hasan et al. [45] conducted and analyzed the co-resident attacks and corresponding defense strategies, with respect to benign and malicious VMs and the VM Monitor (VMM). Han et al. [43] analyzed commonly used VM allocation policies to investigate co-residency attacks and observed that server's configuration, oversubscription and background traffic have a large impact on the ability to prevent attackers from co-locating with the targets. Miao et al. [94] proposed the security-aware VM placement approach (SecVMP) to minimize co-residency and mediate conflicts between tenants for proactively mitigating co- resident attacks in cloud. Caron et al. [16] proposed a security metric designed for cloud environments and using this metric, came up with optimized placements of VMs in line with user security requirements, which include a list of adversaries the user wants to avoid being placed in the same physical machine withm avoiding co-residency attacks. Alhebaishi et al. [4] modeled cross-layer and co-residency attacks in the NFV stack and used optimized VM placement to mitigate such attacks.

3.4.8 Zero Day Vulnerabilities

Zero-day vulnerabilities are another category of vulnerabilities and play an important factor in analyzing the security posture. Zero-day vulnerabilities are weaknesses that have been discovered by the vendor/researcher as well as the adversary, but not yet reported. Reported vulnerabilities have scores associated with them (to be discussed in later sections) which helps in quantifying risks, while zero-day vulnerabilities do not have any severity ratings, making it difficult to quantify. Radhakrishnan et al. [111] conducted a survey of zero-day malware attacks, demonstrating how malware developers make use of such malware to bypass security mechanisms. Bilge et al. [11] performed an empirical study of zero day attacks, where they identify 18 vulnerabilities exploited before disclosure, of which 11 were not previously known to have been employed in zero-day attacks. Wang et al. [142] proposed a security metric to

address this issue, where instead of attempting to rank unknown vulnerabilities, the metric counts how many such vulnerabilities would be required to compromise the target; a larger count implies more security because the likelihood of having more unknown vulnerabilities available, applicable, and exploitable all at the same time will be significantly lower. Li et al. [76] proposed a metric to measure the system's resilience against zero-day attacks, which is the minimum effort required by zero-day exploits to compromise a system and then apply this metric to evaluate different defense countermeasures to decide with defense mechanism can be deployed by the system against zero-day attacks.

3.5 Ecosystem of Security Metrics

Once we understand how to identify the threats and sources of risks, the next step would be to quantify the risks and derive the metric that reveals the security posture of the environment. This process involves:

- Set up the threat model
- Collect risk information from the environment.
- Quantify the risks.
- Model the attacks using the collected data.
- Aggregate the quantified values and classify.

Since state-of-the-art analysis tools and framework only take into account information regarding software vulnerabilities, the risk information input from the system are generally software vulnerability data, obtained by using vulnerability scanners in the system. However, several existing proposals also deal with inputs of other categories. For example, when a company wants to conduct security audit to determine the compliance of the employees with security guidelines, such security guidelines then become an input for the metric calculation, and several such frameworks for security audits [99], [26], [105] have been proposed in existing literature. Some security

proposals also depend on actively taking operator or administrator knowledge as input to build attacker models and adjust attacker capabilities when modelling attacks, terming such inputs as "Attacker Skill Level" [64]. Some security evaluators [63] also take service dependencies as well as malefactor models as input to compute the metrics.

For subsequent discussion, we only consider software vulnerabilities as relevant risk information input for the metric computation, as they are more widely used in existing security proposals.

3.5.1 Input Collection

The first step to determining the security posture of the environment is to scan for any weakness that exists in the environment. Multiple inputs are required from the environment for the metric derivation. Examples of inputs include:

- Vulnerability data using scanners such as Open Vulnerability Assessment Language (OVAL) [70] or Nessus [142], which scans for vulnerabilities in a network and then maps them to NVD database [53] for CVSS score. We also need to scan for problems with the protocols being utilized by the target system and the policies enforced by the system. We can also analyze software source codes or binaries to discover software vulnerabilities or program bugs leading to attack surfaces as well as potential vulnerability exploits [50]. This could also represent the data collected from an individual node in the network or intra-node inputs.

- Network metadata such as network definition or descriptor files to understand the network topology [142], which contributes to how the attack is carried out such as the Coremelt attack [131], which relies on the attacker's knowledge of the target network topology. These topology information dictates the connectivity of all such individual nodes in the system/network and hence we could call these input as inter-node input data.

3.5.2 Quantifying Vulnerabilities

While it is important to understand the sources of risks i.e., understanding different vulnerabilities as well as other sources, it is equally important to quantify them to make them usable in risk assessments and metric derivation. Such quantification can be represented using metrics such as the Common Vulnerability Score System (CVSS) [121], and DREAD [124]. CVSS scores are the most widely used system to rank vulnerabilities and thus many academic proposals are centred around this scoring system. CVSS scores are considered as an essential input to security metrics. Singhal et al. [125] proposed a system where CVSS scores are one of the inputs taken for security metrics calculation. Keramati et al. [59] proposed a CVSS-based security metrics that takes in CVSS scores as inputs to calculate the impact of exploiting the vulnerability. Wang et al. [139] proposed a security metric to measure the trustworthiness for software systems, using the Common Vulnerabilities and Exposures (CVE) and the CVSS. Jiang et al. [54] proposed *VRank* to address the limitation of certain scoring systems such as the CVSS which fails to consider contextual information, a very unique characteristic of Service Oriented Architecture (SOA). VRank not only considers the inherent properties like exploitability, but also takes into account the context of the vulnerability, e.g., what roles the service plays and how critical the vulnerability is to the security objective of the service.

While almost all vulnerabilities are seen from a technical point of view, e.g., the direct damage they can cause to the system, an important aspect remains obscure, i.e., the economic aspect. Vulnerability scoring systems such as the CVSS do not take into account the economic loss that can be caused when a vulnerability is exploited. Security budget is limited in industry thus software vulnerabilities need a set approach when it comes to patching. To this end, Ghani et al. [36] employed the Multiple Criteria Decision Analysis (MCDA) methods to prioritize existing vulnerabilities and proposed a cost factor analysis for the economic damage of successful vulnerability exploitation. Fruhwirth et al. [35] showed a method to improve vulnerability prioritization by combining CVSS scores with a context dimension, which means that each vulnerability can possess different severity levels depending on the context of its existence. Thus, patching a vulnerability in one deployment context

might not be as urgent as in another deployment context. Alternative to CVSS, we also have Industrial Vulnerability Scoring System (IVSS) [30], which takes into account the impact of vulnerabilities in an industrial environment as opposed to an IT environment. IVSS acts as a calculation mechanism and the metrics and factors it considers are: Base Severity Level (BS), Base Exploitation Level (BEX), accessibility, impact, and consequences. IVSS calculates the base score and IVSS final score, by aggregating all previous values.

3.5.3 Modelling and Aggregation

After the vulnerability information have been collected, it is necessary to model the attacks to see what sort of attack paths can rise from exploiting these vulnerabilities. In the context of security metrics, attack graphs and attack defense trees are two examples on how such modellings are done. Nodes may represent states of attacks/defenses while edges can show the transition of states. An overall security indicator can be derived, e.g., by solving the shortest path problem.

Attack Graph

There have been various types of attack graphs (AG) proposed since its first proposal by Phillips and Swiler [108]. Despite the types, the nodes of an attack graph generally represent exploits (of vulnerabilities, e.g., software bugs) and conditions (e.g., certain privilege of a resource). Conditions leading to exploits are pre-conditions and conditions as a consequence of exploits are post-conditions. Edges (transitions) connecting nodes form different attack paths. An attack graph always concerns a target, the ultimate post-condition, which all attack paths lead to. In order to clearly convey the vulnerabilities and how it can be exploited, all the paths that are available to the attacker are shown. The attack graph contains many nodes with edges acting as connections between them to show the direct influence/control that one aspect of the system may have on the other. The nodes are representative of the many aspects of the system, such as each physical device that the user uses to interact with the system, the databases that store sensitive information, and the firewalls that defend against unwanted access along with others. The graph is shown in a way such that the

user can use one exploit present in one of the nodes and directly or indirectly control a group of nodes/systems. The attack graph is an effective tool to study the chained effects of multiple exploits on the same or different systems. With the existence of multiple exploits being present in a network of systems, the attacker is present with opportunities to employ one attack path in which the attack deems reasonable to use Since the attack graph is often filled with multiple of the same attack paths, in order to clarify the options that the attacker has present, there are also options of breaking up a huge graph of interconnected nodes and edges into simple non-cyclic paths where it is clearly shown how an attack can be carried out.

Attack Defense Tree

A slightly different graph technique is the attack defense tree (ADT) [62], which describes the exploits from the perspective of the attacker's goal. The root of the ADT represents the attacker's ultimate goal, and down the branches the ultimate goal is to iteratively break down to sub-goals till the leaves. To also capture the defense perspective, the main difference between an attack graph and an ADT is that the ADT has both attack nodes and defense nodes. While AG and ADT have seemingly fundamental differences, it is also possible to combine them to model attacks. Enoch et al. [27] proposed a composite analysis for network security metrics. Their proposed Hierarchical Attack Representation Model (HARM) consisted of a two layer model in which the upper layer, made of attack graphs represents the network reachability information and the lower layer, made of attack trees, representing the vulnerability information.

Aggregation

There are various aspects which could reflect the security of an individual/single system such as its authentication mechanism, its network setup (any vulnerable ports), its access control policies and so on. We can aggregate all these aspects to reveal the security aspect of that system termed as single-system aggregation. When we have a collection of those systems (a network), we need multi-system aggregation, which refers to aggregating values from several individual systems/entities to one single

value (by setting a node as a target) to represent the security state of the network. This single value represents the security state of that node i.e., how vulnerable that node is to an exploitation. All present works of security metrics involve assigning a target and then aggregate towards that. However, an owner or stakeholder might be interested to know the security state of the environment without having a set target (when the target is not just one component but any component inside the system). Such targeted aggregation can be done in the following ways:

- Structural metrics: These metrics use the structure of the attack graph to aggregate the security properties of individual systems in order to quantify network security. Each entity in the network is assigned a probability of exploitation and these probabilities are then aggregated to quantify the security state. Examples include Attack-graph based probabilistic security metrics [140], based on Bayesian Networks [34]. The Shortest Path (SP) [100] metric measures the shortest path that an attacker needs to reach the target. The Number of Paths (NP) [100] metric measures the total number of paths that the attacker needs to reach the target. The Mean of Path Lengths (MPL) metric [77] measures the mean of the length of all paths to the target in an attack graph.

- Time-Based Metrics: An owner might also be interested to quantify security in terms of time i.e., how quickly the network can be compromised. Some examples are Mean Time to Breach (MTTB), Mean Time to Recovery (MTTR) and Mean Time to First Failure (MTFF).

3.5.4 Automating the Process of Metric Derivation

Input collection and graph generation are a time-consuming and error-prone process. The target system needs to be thoroughly analyzed for any potential bug or flaw that an attacker may exploit, i.e., vulnerability scanning. In order to correctly identify an exploit, a database to compare with is usually involved (e.g., the NVD database). The challenge increases as the number of systems/nodes/VMs increases as it leads to a more complex environment. There have been academic/industrial projects and tools that aim to facilitate or automate this process, such as DOCTOR [135], MulVal

[101], Meta Attack Language (MAL) [56] and OpenSCAP [110],[1] covering automated vulnerability scanning, input collection and graph generation.

- MulVAL: MulVAL is a network vulnerability analysis framework and reasoning system that conducts multihost, multistage vulnerability analysis. For analysis, MulVAL employs Datalog as the modelling language (bug specification, configuration description, reasoning rules, operating-system permission and privilege model, and so on). The scanning is done using current vulnerability databases and the output is fed in Datalog and sending it to the MulVAL reasoning engine. Once the data is gathered, the analysis, which takes the form of attack graphs, may be scaled up for networks with thousands of workstations.

- DOCTOR: The DOCTOR architecture includes a correlation engine that may infer attack occurrences and notify orchestration components based on the micro-detector alerts exposed. MulVAL now has new rules for Named Data Networking (NDN) and NFV settings, added by DOCTOR, allowing it to do proactive security analysis based on the evaluation of attack graphs. The authors then devised and tested specialised detection algorithms capable of detecting the major attack scenario they are considering against NDN. The DOCTOR architecture can consider all possible attacks, and the DOCTOR Security Orchestration can provide a variety of defences.

- OpenSCAP: OpenSCAP is an implementation of Security Content Automation Protocol (SCAP), a way for automating vulnerability monitoring, measurement, and policy compliance evaluation of systems deployed in an organisation utilising specified standards, such as FISMA (Federal Information Security Management Act, 2002) compliance.

- Meta Attack Language (MAL) presents a formal method by designing a domain-specific attack language, that enables the semi-automated generation as well as the efficient computation of very large attack graphs. The attack languages are designed based on the generic characteristics of particular domains such as

[1] https://www.open-scap.org/features/scap-components/

e.g. the man-in-the-middle attacks in IT networks). Attack language designed based on generic features could then be applied to any instance of that domain for security analysis.

In recent times, Machine Learning (ML) techniques have also become prevalent in helping with automating some parts of the process of calculating the security metrics. Saman et al. [152] presented Elimet, a framework that collects information from different sources and analyzes how the security goals are being met against a set guideline and a generic security metric to calculate security measure estimates of every security state. To begin with, Elimet observes actions taken by system users against different security incidents and accordingly adjusts the calculated security value. If, at any stage, sufficient information could not be gathered regarding system actions, Elimet requires manual input from system users. In the last stage, Elimet compiles a ranking of counter measures that the system administrators can deploy against security threats. Rizi et al. [115] made use of vector embeddings and trained a deep learning model to approximate the shortest paths distances in large graphs. Pope et al. [109] proposed the usage of Hyper-heuristic Techniques (HHT) to produce graph-level security metrics, by employing hyper-heuristic machine learning techniques, which are trained on network attack simulation training data. The security metrics can then serve as an approximation for simulation when measuring network security in real time. Francia III et al. [33] used Machine Learning techniques to predict the security status of automotive vehicles, where the inputs to the model were known vulnerability attributes of the automotive vehicles.

We summarize the different tools and approaches available for modelling, aggregation and automation in Table 3.1. The table is an attempt to list primitives related to security metrics. These primitives represent the factors involved in deriving or approximating security metrics using various tools: collecting data such as vulnerability information, representing or modelling attacks, aggregating the data, deriving the metrics, automating parts of this process to reduce complexity or human interaction etc. The table lists the interactions of the state-of-the-art security metrics tools and proposals with the primitives. Each row represents the interaction between the particular primitive and the proposals. The first column lists the security metrics

primitives and the subsequent columns lists the tools and proposals.

- In second row first column, we have the Data Collection primitive. The rest of the columns in the second row lists if the tools or proposals have data collection involved in the execution of the said tool or proposal.
- In the third row, we have the Information Repository primitive. This primitive represents if the tools/proposals interact with any information repository like National Vulnerability Database (NVD), Common Vulnerabilities and Exposures (CVE) or Common Platform Enumeration (CPE).
- In the fourth row, we have the Attack Representation primitive. This primitive represents if the tool/proposal uses any formal language or framework for attack modelling or representation. For example, MulVAL uses Datalog language to formally model attacks.
- In the fifth row, we have Aggregation primitive. This primitive represents if the tool/proposal employs any category of aggregation of attack paths or patterns.
- In the sixth row, we have Metrics or Approximation primitive. This primitive represents if the tool/proposal outputs any metric value or approximates any particular metric like shortest path etc.
- In the final row, we have Machine Learning primitive. This lists the tools/proposals that employ machine learning techniques to automate security analysis or assessment.

The checkmark (✓) represents whether the tool or proposal makes use of the corresponding primitive, for example, MulVAL interacts with NVD, which satisfies the Information Repository primitive. For data collection primitive, we break it down further by determining if the collection is done automatically (A) or manually (M) given as input by human intervention.

To the best of our knowledge, the process of automating some parts of metric derivation gained popularity with the advent of Nessus in 1998. Although it was not the first security scanner of its type, Nessus was the first widely-accepted open

Table 3.1: Intersection of Security Metrics Primitives and Existing Tools/Projects

Primitives \ Tools & Proposals	MulVAL [101]	DOCTOR [135]	OpenSCAP [110]	Elimet [152]	HHT [109]	Nessus [142]	HARM [27]	MAL [56]	Stan et al. [128]	Rizi et al. [115]	OVAL [70]	Francia et al. [33]
Data Collection	A	A	A	M	A	A	✗	✗	A	✗	A	M
Information Repository	✓	✗	✗	✗	✗	✓	✗	✗	✓	✗	✓	✗
Attack Representation	✓	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗
Aggregation	✓	✓	✗	✗	✓	✗	✓	✓	✓	✗	✗	✗
Metrics or Approximation	✗	✗	✓	✓	✓	✗	✓	✓	✗	✓	✗	✓
Machine Learning	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓	✗	✓

source scanners while its peers were locked behind commercial paywalls. Gradually, many more academic proposals regarding automation were made, such as the OVAL scanner, MulVAL (based on Nessus) etc. Most of these tools/projects are results of academic research, with the exception of Nessus, OpenSCAP and DOCTOR. Nessus started off as an academic proposal and later turned into an industrial product. OpenSCAP and DOCTOR can be termed as hybrid, with joint contributions from both industry and academia. Apart from Nessus and OpenSCAP, none of the existing automation tools and projects are actively maintained or updated, which makes it doubtful regarding the applicability of these tools in modern day systems, barring some exceptions such as Stan et al. [128] recently extended MulVAL's ruleset to incorporate a wider range of protocol vulnerabilities, which were originally missing in MulVAL's ruleset and model attacks in advanced communications such as wireless communications etc. Machine Learning (ML) approaches have gained traction in recent years, employing real-time learning of network characteristics, attack patterns etc., making them more useful to be used in variety of modern day systems. Such machine learning proposals have been discussed in Subsection 3.5.4 and summarized in Table 3.1. Although we have not identified one single tool or proposal that could

achieve full automation of all primitives, ML approaches seem as the most promising option to fulfill that criteria, due to the volume of recent proposals working to address the issue of automation.

3.6 Challenges and Benefits

While significant researches have been done on security metrics for various environment setups, it has also continued to receive skepticism and criticism. Security is difficult to measure in practice due to their inherent existence of uncertainty. The uncertainty can be derived from multiple reasons. First, unknown attack behaviors are hard to be accurately predicted by a defender. For example, vulnerabilities are dynamically discovered while an attacker may identify some zero day vulnerabilities unknown to the defender. Moreover, the defender is uncertain about what exploits the attack possess. Some attacks are never detected by the defender. These indicate that uncertainty is inherent to the threat model the defender is confronted with. Verendel [137] presented arguments that "quantified security is a weak hypothesis" because the validity of results was not clear due to the lack of statistical evidence to either support or contradict them. The skepticism rose because of the lack of security data to validate the methods used to derive quantified metrics. It is also difficult to measure security because of the direct correlation between security and context. What is not harmful in one context might be a security issue in another context [107]. Some might also argue that it is very difficult to encapsulate security with just one quantified value.

However, despite the criticisms, security metrics helps us answer several questions. Some decision making processes rely on just a number to determine which path to take when numerous possibilities exist (such as multiple potential deployments at hand) [10]. Security metrics also help with prioritizing which vulnerabilities to patch when its not possible to patch them all. When it comes to security control placements, security metrics are vital because only through quantification are we able to identify the most vulnerable sections of the system [7]. Every organization strives for a more efficient system, and security metrics can provide insights on how to make the system for efficient in terms of security measurements [7]. Economically, a company stands to

gain more by using different sets of security metrics to their security posture updated against most potential exploits, as without security metrics it is hard to predict what the company tends to lose or gain from an upcoming security change, as wrong changes could end up costing millions to fix [15]. Added to the economic benefits, an organization also saves itself from loss of sensitive data, which could have spiralling effects such as affecting stock market prices [134] and future goodwill of the company.

3.7 Current Proposals regarding Security Metrics

A system could be individual/standalone or an interconnection of several systems. This is significant because one entity in an interconnected system has the ability to influence the security of its neighbouring entities present. For example, Computer B might not have any vulnerabilities but because Computer A does and Computer B is connected to Computer A, B is also at the risk of attacker incursion. Existing state of the art covers both these systems (standalone and interconnected). Depending on such setup and scopes (what we want to measure etc.), various security metrics proposal exists. Rodes et al. [116] presented a novel framework for assessing security arguments to generate and interpret security metrics that eliminates the needs of engineers to figure out what to measure for each system and instead provides comprehensive documentation of all evidence and rationales for justifying belief in a security claim about a software system. Ahmed et al. [3] proposes a novel security metric framework that identifies and quantifies objectively the most significant security risk factors, which include existing vulnerabilities, historical trend of vulnerability of the remotely accessible services, prediction of potential vulnerabilities for any general network service and their estimated severity and finally policy resistance to attack propagation within the network. Cheng et al. [20] proposed a new approach for aggregating CVSS scores in a way so that the semantics is not lost during the aggregation process in a network security metrics. Discussion regarding aggregation are done in a later section. LeMay et al. [75] defined the ADversary View Security Evaluation (ADVISE) method, where the approach is to create an executable state-based security model of a system and an adversary that represents how the adversary is likely to attack the system and the results of such an attack. It is important for well designed

software systems to meet certain Quality-of-Service (QoS) requirements, such as reliability, availability and performability. At the same time, most such software systems are network accessible through public networks, such as the Internet. As a result, these applications and systems have become prone to security intrusions. Madan et al. [85] systematically analyzed how to quantify security attributes of software systems and have them at par with the QoS requirements. In the last decade, cloud computing has undergone rapid expansion and systems are being deployed/hosted more and more in cloud, including recent telecommunication technologies like 5G. Hence, it is also important to take a look at the existing literature to investigate the existing security metrics touching the cloud. Alhebaishi et al. [5] conducted a comprehensive threat modelling for cloud, which will aid cloud providers in better evaluating the risks in cloud infrastructures. Le et al. [73] provided a review of capability maturity models and security metrics for cloud and proposed a cloud security capability maturity model (CSCMM) that extends existing cyber security models with a security metric framework. Liu et al. [82] proposed a framework, in which a module measures executables running in virtual machines (VMs) and transfers the values to a trusted VM and a comparison is done between those values to a reference table containing the trusted measurement values of running executables, which then verifies the executable/s status. To evaluate multi-tenancy threats in the cloud, Madi et al. [87] introduced multi-level distance metrics based on compute, physical and network distances, taking into account the level of resource sharing between tenants. The metrics are reliant upon the deployment and configuration of the cloud so as to make it easier for the cloud provider to evaluate the risks.

3.8 Gaps between the State-of-the-Art Security Metrics and Emerging Technologies

Despite the various models of security metrics, there are some places which the current security metrics have not yet touched. For example, the Fifth-Generation (5G) technologies are rapidly expanding but there are no security metrics that have been tailored for application in a 5G environment. We take a look at the gaps preventing the application of existing metrics to 5G environments. While it is difficult to

encompass all the gaps, we discuss some of them to showcase that the gaps do exist.

3.8.1 Gap 1: Threat Model Shift

Existing security metrics models consider the infrastructure owner to be trusted [4]. But as we have already seen from the structure of 5G environments, we have a combination of several stakeholders/owners and there exists a mutual distrust about one another. Stakeholders, by nature, can be malicious in nature or curious, for example, wants to take a peek what sort of data other stakeholders are holding or simply careless not to implement own security measures, which in turn affects the security of other parties. A customer (owner of user equipment in a 5G environment) might become affected or malicious in nature and affect the whole environment. Thus, we need to revisit this assumption that owners can be trusted and modify existing security metrics to fit in this new threat model.

3.8.2 Gap 2: Input Collection Barriers

When peers are connected and can influence each other's security, security metric computation needs data from all involved parties. However the input collection here is not straightforward as the stakeholders are isolated from one another through technical and institutional techniques (as discussed in 5G Preliminaries chapter) which prevents data collection, which in turn affects the security metric computation. Existing security metric proposals, by default, consider that the information required for security metric computation is either available or easily accessible through usage of various tools such as scanners etc. Such information barriers then pose a significant challenge because the information collection requirement stated by the security metrics cannot be fulfilled due to the conflicting scenario, where the security metric approach considers information to be already available but the barriers block the information from being accessible.

3.8.3 Gap 3: Localized Security Metrics

A stakeholder, for example a virtual operator, might want to evaluate the level of security of their own environment. We need a multi-system aggregation with the

virtual operator set as target. We can also perform a single-system aggregation just by taking the components of the virtual operator into account, but that would not be sufficient as the virtual operator is part of a larger network, and will have threats incoming from other systems surrounding it, for example, from the MEC or from RAN. The key thing to note here, is that we have a target which is local to the computing stakeholder but the threats are going to be global i.e., they can be from other stakeholders such as the cloud service provider or the MEC module as well.

3.8.4 Gap 4: Dynamicity

One of the defining factors of 5G is its ability to dynamically allocate resources to entities in the network that require additional resources. For example, a network slice might be assigned to a stakeholder dynamically or a VM might be spawned, which affects the topology of the environment and in turn, affects the attack paths, which may have been different previously. Security metrics only takes into account static data as it is expensive to monitor data continuously, requiring resources which some organizations might not be willing to budget for. Hence it becomes difficult to compensate for the dynamic data and thus affects the security score, as it won't reveal the actual scenario.

3.9 Survey Summary and Discussion

As we conclude the survey, we want to summarize the contents of the survey. Our goal was to conduct a survey that would avoid any limitation regarding the dimensions and instead take a holistic look into the domain of security metrics. We wanted to explore the topic of security metrics from the perspective of risk management, irrespective of setting or setup of the system being studied on such as single systems, networked setup etc. This enabled us to explore many diverse threats and risk assessment techniques, that would have been otherwise not possible if we were limited by a scope. Figure 3.1 summarizes all the topics that have been discussed and covered by this survey, demonstrating the process of how to derive security metrics, as part of a bigger umbrella encompassing the risk management process.

Some key lessons from the survey:

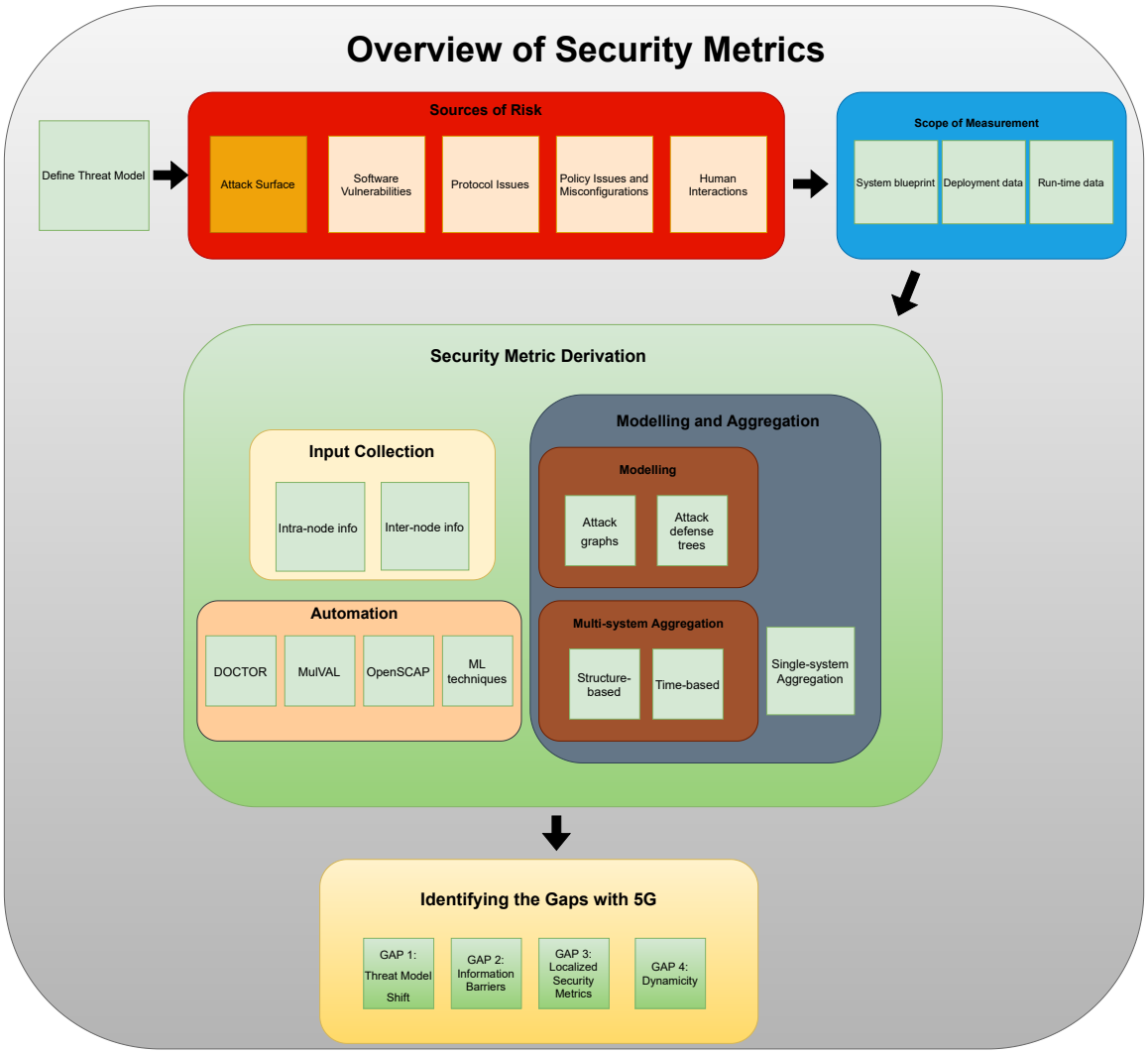

Figure 3.1: Summary of the Survey

- Every security metrics proposal must define a comprehensive threat model as the process of deriving the metrics depends on what the owner considers as "threat".
- Security metrics proposals must also consider "human users" as a widespread source of threats as in our exploration we have observed most proposals consider "technical" vulnerabilities or weaknesses to compute metrics and ignore threats introduced through the action of the users.
- State-of-the-art security metrics tools and frameworks must do more extensive work to consider inputs other than software vulnerabilities during security metrics analysis as the threat landscape span much wider.
- Despite some drawbacks, security metrics have significant advantages that could benefit everyone: from the owner to the end-user and everything in between.
- Existing security metrics must be adapted to make it applicable in a 5G/NFV environment.

We pursue some of these takeaways in the future direction of our research.

Chapter 4

Addressing the Gaps

Based on the gaps identified in the previous chapter, we want to see how we can overcome those. We looked into existing literature to import some solutions that can be used to address the gaps.

4.1 Gap 1: Changes in Threat Model

Because of the threat model shift, trusting a potentially malevolent or curious third-party with access to one's own resources for input collecting is difficult. We also cannot rely on self-reporting i.e., expect another stakeholder to honestly report the data in an untarnished manner. For example, a dishonest supplier may simply report inaccurate software versions to hide security flaws in his environment. When mutual confidence is eroded, hardware help becomes more vital, especially for more privileged parties, such as cloud service providers, provided that hardware vendors are still trusted. There have been multiple works to protect VNF executions and its data through hardware assisted models [102, 123], such as Intel SGX. Hence, when we do not trust the data supplied by other parties, we leverage hardware support to do the collection for us.

4.2 Gap 2: Data Sharing through Information Digest

To address Gap 2, data could be reported in such a way that not too much is revealed, just enough for the requesting party to use in security metric computation, to bypass the information sharing barriers, termed as *Information Digest*, where the entire information is boiled down to contain essential details only and a digested version is generated. For example, a stakeholder might want to know how a VM is created. So instead of exposing the original operation logs, an infrastructure provider may expose

meaningful semantics for the guest, e.g., VM create, init, start and terminate. Local security score based on metrics like bayesian network or shortest path could be shared with the interested party instead of the whole metric derivation process to safeguard own security.

We also have standardization to help us in this process. Standardized protocols designed by various agencies like NIST [39], [49] unifies what data can be collected and in what format so that there exists a parity amongst the stakeholders. For example, if different parties use different metrics (one party uses bayesian and one party uses shortest path) to derive their score, aggregation of those scores become impossible.

We want to acknowledge an issue here: information barriers installed to ensure privacy, pose considerable challenge to security metric computation, causing a security vs privacy dilemma. While keeping things private, the security computation gets hampered. If we favour security, privacy is hampered. However, in this process, critical data might not be shared, which doesn't reveal the actual condition of the party supplying the information, hence affects everyone in the environment. Information Digest provides a viable solution to this dilemma: instead of exposing everything and losing privacy, we summarize and reveal certain or essential details (such as local security score), keeping the privacy intact to a certain degree. The falloff here is that this won't reveal the actual security scenario as a lot of details are being removed or shortened, however this is the trade-off between privacy and security that will help to ensure some security (as opposed to having no security by having 100% privacy).

4.3 Gap 3: Modular Security Metrics (MSM)

To address the issue of localized metric computation, we propose Modular Security Metrics (MSM), an adapted security metric computation approach based on ownership boundaries. We divide the entire environment into different modules, where each module belongs to distinct owners such as the cloud provider or virtual operator. Each module is capable of independently computing their security scores by considering risk factors present in their own module. The target module first sets the target component and computes their security score. The other modules in the environment perform a similar operation and compute their security score. The scores from the

neighbouring modules are then shared with the target module, who aggregates all the individual scores into one compound value. This will achieve two-purposes: first, it will allow the local owner to compute the security of their own environment and second, they will also be able to take into account the attack paths that are coming from other owners.

Modular Security Metric is based on attack graphs. At first, we perform an initial threat analysis of the 5G landscape to study the threat scenarios and possible attack paths, as shown in Figure 4.1.

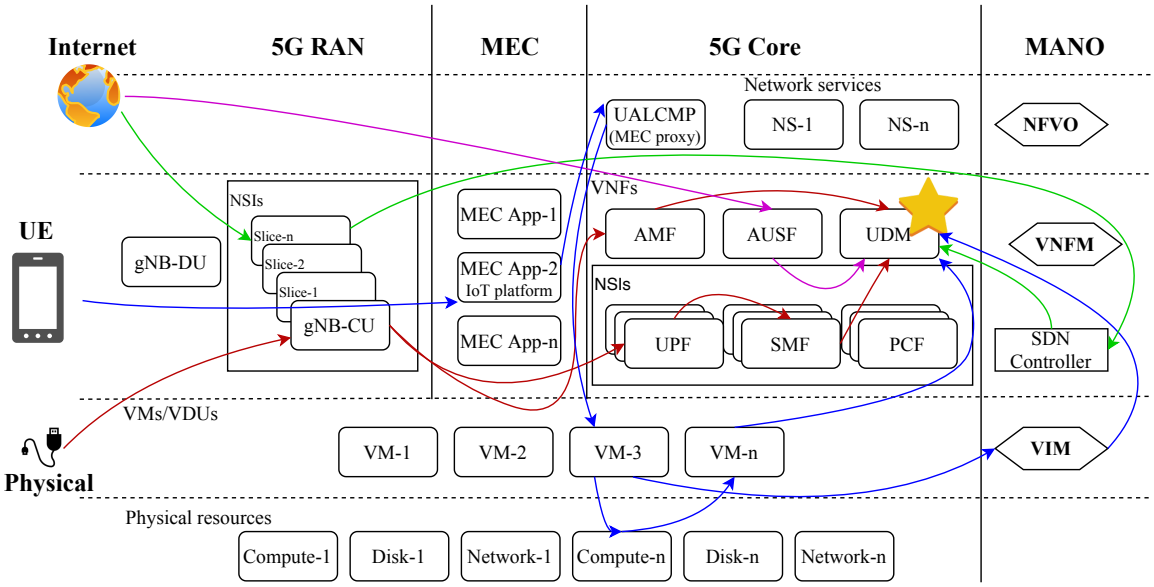

Figure 4.1: 5G Threat Landscape

The threat scenario can then be utilized to compute an attack graph for the whole environment, with ownership boundaries, referred to as modules, pertaining to different stakeholders. For demonstration purposes, we will assume UDM is the attacker's target component. By observing the threat scenario of 5G, we consider three possible attack paths for our demonstration purpose.

- From a user equipment connected to the MEC module. An UE can connect to the MEC module for various applications such as IoT applications etc. The UE compromises a MEC-hosted IoT application, and the UE then exploits the core network's UALCMP (User Application LifeCycle Management Proxy). An

attack path is constructed if the UALCMP is co-located on the same VM as the target VNF.

- From the base station, referred to as gNB in 5G, which are spread throughout several locations to provide signal coverage. Such base stations are generally physically accessible to the public and the attacker can compromise the base station with little resistance [148]. gNBs, which are part of 5G-RAN, have direct connections to the 5G core through interfaces. Hence, a compromised gNB leads to a direct attack path to the UDM.

- From the internet. Some VMs in the environment will have internet connectivity and based on protocol vulnerabilities and/or software vulnerabilities discussed in previous section, an attacker can gain access to the environment. For example, AUSF, responsible for authentication in 5G network, can have accessibility to the internet. Once connected to internet, it can form attack paths through the N13 interface to the UDM.

The components in 5G environment are connected via interfaces, as characterized by 3GPP Release 15 [1], for example, we have N2 interface connecting gNB with AMF, N13 interface connecting AUSF with UDM. These interfaces also contribute to the formation of the attack paths, as they serve as the connection between components.

Based on the attack originating points and the connections between the components, we generate the attack graph, as shown in Figure 4.2. We assign some arbitrary probability values to the vulnerabilities in the attack graph, for demonstration purposes. This probability values (assigned as number inside the boxes) represent the likelihood that an attacker will be successful with the exploit. The graph is divided into separate modules (boxes marked by dark black solid lines) based on ownership boundary.

Notations used in the attack graph are described as follows:

- $\langle root, gNB1 \rangle$: refers to a pre or post condition, having root privilege in gNB1.

- $\boxed{\underset{0.44}{\langle N2, gNB1, AMF \rangle}}$: refers to scenario where by exploiting the N2 interface, $\langle root, gNB1 \rangle$ can lead to $\langle root, AMF \rangle$ with the probability of $p() = 0.5$.

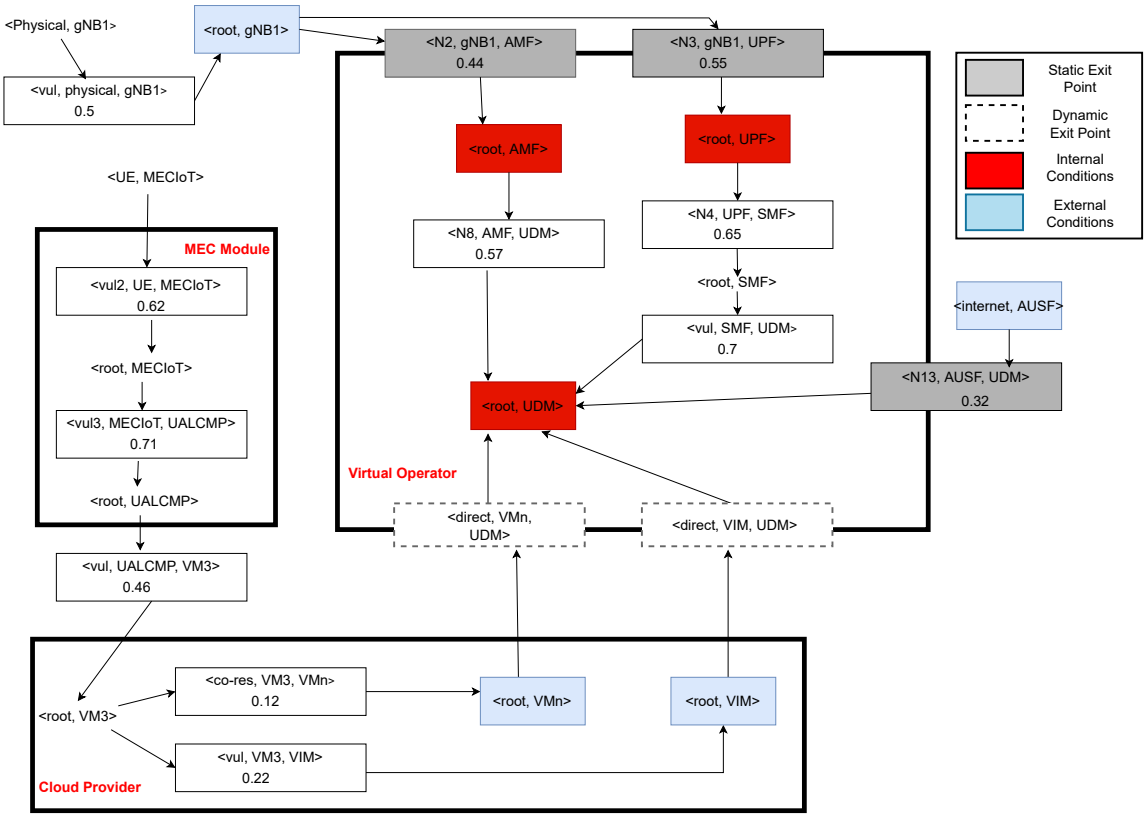

Figure 4.2: Attack Graph based on Ownership boundary

We use Bayesian Networks (BN) and Shortest Path (SP) metrics to derive the quantified security value $P()$, a cumulative probability of an attacker following the attack paths and executing the exploit. We calculate $P()$ with the Bayesian conditional probability using individual $p()$s, done by Frigault et al. [34].

Our goal is to enable the virtual operator module owner to be able to calculate the security value for UDM using the BN and SP metrics.

As is the nature of attack graphs, it consists of pre-conditions, post-conditions, connected by exploits. Normally this is straightforward. We have a pre-condition, leading to the exploit, which leads to the post-condition. However, when it comes to MSM, we have boundaries separating modules/owners. This leads to the scenario where we have exploit nodes that have a pre-condition belonging to one module (such as a Virtual Infrastructure Manager (VIM) that belongs to Cloud Service Provider) and the post-condition belonging to another module (such as a UDM belonging to Virtual Operator). We term these nodes as Border Exploit Points (BEP). In terms of data transfer needed for metric computation, this is where the data exchange happens between the two parties. The pre-conditions of the BEP are termed as External Conditions (marked as blue) and post-conditions of the BEP are termed as Internal Conditions (marked as red). The modules compute their own security score based on their attack graphs and data handoff happens through the BEP. This process happens in three steps.

Step 1: Local Attack Graph

The target module (Virtual Operator in this case) at first identifies all the nodes (AMF and UPF) that are connected to the neighbouring modules. The virtual operator sets UDM as the target and the internal conditions as the sources and computes its own attack graph. The virtual operator then calculates its local security value using BN:

$$P(\langle root, UDM \rangle) = Bayesian(N)$$

where N represents the local graph of the virtual operator. The virtual operator derives a score of $P(\langle root, UDM \rangle) = 0.73$ based on the conditional probability table

of the network, as demonstrated by Frigault et al. [34]. For the SP metric, the virtual operator receives a score of $Path_{min}(\langle root, UDM \rangle) = 1$. The other modules compute their local attack graph in a similar manner, setting the external conditions of the BEP as their targets.

Step 2: Connecting the attack graphs through Static Exit Points

Once each of the modules compute their local attack graphs, the attack graphs must be connected to each other to transfer data through projection, where the external scores are passed along through the interfaces that connect the components to one another. Static exit points refers to the static connection channels between the different modules, for example, N2 interface connecting gNB1 and AMF. The term "static" here denotes the connections that are not changing or constant. These static connections or exit points are responsible for the BEP between the two modules. Once the virtual operator receives the scores of other modules through the static exit points and constructs the BEP and assigns probabilities. For example, from the RAN owner, the virtual operator receives $\langle root, gNB1 \rangle$ gets $P(\langle root, gNB1 \rangle) = p(vul) = 0.1$, and its shortest path is 1. Once the scores are received from other modules, the virtual operator updates the calculation by taking the external scores into account.

$$P_{\text{static}}(\langle root, UDM \rangle) = Bayesian(N_{\text{static}}) = 0.17$$

where N_{static} refers to the new graph where the local graph is connected to attack graph of other modules through static channels.

The SP metric $Path_{min}(\langle root, UDM \rangle)$ is still 1 as the shortest path from the AUSF to UDM is 1.

Step 3: Connecting the attack graphs through Dynamic Exit Points

In addition to the static channels, there exists the dynamic connectivity through which the modules can be connected. The Cloud Provider (CP) provides physical machines (VMs) to host the VNFs of the virtual operator. The placement of the VNFs in different VMs can render different scores due to the fact that not all VMs

have the same security score. For example, the CSP might choose to host UDM in one VM and AMF in another, or might choose to host UDM and AMF both in one VM. The virtual operator then probes different placements to see which placement will lead to optimal security score. Since there are four components: AMF, SMF, UPF, UDM, it leads to 16 possible combinations in a VM (VMn in this case), it leads to 16 possible combinations, as shown in Figure 4.3

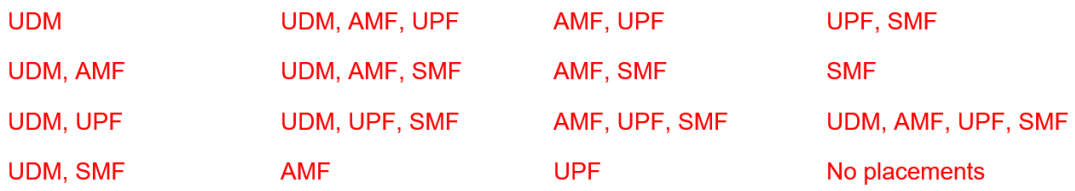

Figure 4.3: Possible placements inside VMn

A key thing to note here is that the virtual operator will not have the knowledge about the actual placement combination as the Cloud Provider won't divulge that information, leading to loss of information, whose significance will be discussed in details in a later section.

Final Metric

After the static and dynamic connections, eventually the virtual operator arrives at the final metric computation, which is:

$$P_{\text{Consolidated}}(\langle root, UDM \rangle) = \max_{i=1,2,3,\ldots,n} (P_{\text{static,dynamic}(i)})$$

where $P_{\text{static,dynamic}(i)}$ is calculated with $Bayesian(N_{\text{static}})$ combined with the 16 placement combinations $(N_{\text{dynamic}(i)})$.

Since each placement will lead to different scores, the virtual operator chooses the maximum value from the static and dynamic computation, depicting the least secure scenario, which will help the virtual operator to understand its security posture better. Hence the score stands at:

$$P_{\text{Consolidated}}(\langle root, UDM \rangle) = 0.153$$

The SP metric is: $Path_{min}(\langle root, UDM \rangle) = 1$.

Chapter 5

Argus-5: Automated Resource and Attack Graph Generator and Simulator for 5G and beyond

While we validated MSM through manual calculation as we formulated the theory, we needed a tool for generation of large number of graphs and perform the calculation in an automated manner, as we wanted to perform MSM calculation on a sufficiently large amount of graphs, of different sizes, so that we could test its validity and see how it performs against the best or optimal scenario and other measures and also have a look at the pattern by which the graph changes as we alter different conditions and topology. We looked into existing literature to identify tools that could accommodate our simulation parameters and conditions. Some of the main features we were looking for is support for modules, support for layered resource graph generation to represent the different layers existing in the current 5G environment, and be able to convert the resource graph to an attack graph by connecting the vulnerabilities present in the network. We examined resource generation tools like:

- Waxman [96], one of the first topology generators which produces random graphs based on the Erdos-Renyi random graph model.
- Tiers [25] is a multi-tier network topology generator that generates models imitating the structure of the Internet.
- Inet [55] is a resource generator, reproducing the connectivity properties of Internet topologies.
- KOM ScenGen [47] is a topology generator that supports the manual and automatic creation of experimentation scenarios for network research from the topology creation over traffic generation to evaluation.

After careful examination of various such resource generation tools, we were unable to identify any such tool that satisfied all the requirements and features we were looking for. This lead us to design our own tool, to support different parameters and scenarios. Hence, we introduce Argus-5,[1] an automated resource and attack graph generator and simulator for 5G and other networks. In the following sections, we propose the design and implementation of our tool. Our goal was to come up with a multi-scenario tool, i.e., we want to have a tool that can support various scenarios and use cases, in addition to MSM validation. The tool generates resource graph (a graph consisting of all the resources, each represented by a node, in the network and their connectivity) and converts the resource graph to an attack graph by drawing out paths containing vulnerabilities present in the network and uses Bayesian calculation and Shortest Path approach to derive the security scoring.

5.1 Design Choices

The tool is designed based on Python programming to make use of the extensive libraries that support the design of the networks, such as Pandas to handle the dataframes required to preserve data, PySMILE to do the Bayesian calculation, NetworkX to visualize the graphs. This tool has been designed with one main feature in mind: scalability. We want to be able to simulate networks on a large scale to encapsulate various scenarios and don't want to be limited by the capacity of the number of nodes or modules. One design challenge of scalable networks is performance, which is to say that as we scale up, there is a possibility the system might fail to keep track of the nodes or lose performance in terms of generation time. It is easier to simulate and keep track of 10 nodes than 100 nodes or 1000 nodes. To handle this issue of node information loss and keep easier track of nodes, we treat each node as an individual object and bind them to modules. We also divide our tool into various sub modules so that each module has independent task and can be called on independently. This also helps in scalability as each part has their own goal and hence performance doesn't become an issue, which would have been rather problematic if everything was done in one place.

[1] In Greek mythology, Argus is a giant with multiple eyes

5.2 Graph Generation Parameters

To generate our graphs, we make use of parameters that control the graph generation features and allow us to introduce certain restrictions to facilitate various test cases.

Percentage of secure/unsecure nodes. We define secure/unsecure nodes based on the probability of their vulnerabilities. We assume that nodes containing vulnerabilities with probabilities less than 0.39 are considered secure. This assumption is based on the fact that vulnerabilities with CVE score less than 3.9 are considered to have low severity. So we set 0.39 (derived from converting 3.9 into a severity percentage by performing 3.9/10, where 10 is the highest severity) as the secure threshold. This parameter controls what percentage of nodes can be secure. For example, with a parameter value 0.6, 60% of the nodes will have vulnerabilities less than 0.39 assigned to them. This secure/unsecure approach allows us to simulate what is the impact on the overall metrics as we scale secure nodes up and down. An infrastructure owner might be interested in simulating the scenario where a certain portion of the resources in the network are secure through defensive mechanisms like security patches etc.

Two simplifications need to be acknowledged here:

- The secure threshold of 0.39 is based on our assumption. We chose this simplification as a way to demonstrate probable scenarios where an administrator might be interested to simulate security posture after patching a set of vulnerabilities (we assume patching here refers to having less severity than before as a result of a firewall or software patch etc).

- While there exists repositories (such as NVD, CVE) that assign severity scores to vulnerabilities, we do not have any universal database with a collection of vulnerability probabilities. We have CVSS scores, obtained by combining several base metrics, representing the severity of the vulnerability. However metrics like Bayesian require a single probability value assigned to each node/vulnerability for calculation. Hence, we needed a way to convert the CVSS score (combination of multiple base values) to a single probability value. This is another simplification we chose by attempting to convert severity to probability to assist

in calculation by following the proposal of Zhang et al. [149].

Number of nodes per modules. This parameter controls the number of nodes that we want to instantiate in each module. These allows us to simulate the effects of the number within any module and how it impacts the overall calculation. For example, an owner might want to re-evaluate their security after introducing some new nodes to their module.

Number of exit nodes per module. Through this parameter, we are able to control how many exit nodes each module can have.

The target node. While the target node is assigned randomly, we are able to control which node can be the target. This parameter allows us to simulate the security score for one particular target.

Connectivity between the nodes within each module. While the connectivity is assigned in random, there might be situations where more closely connected environments might need to be simulated, especially how the connectivity between the nodes affect the overall scoring.

Template. Controls what template the graph is generated on. If nothing is set, the graph follows a normal resource graph generation, consisting of typical network equipment such as routers, computers, firewalls. The tool is also capable of following a 5G and cloud template, discussed in Section 5.3 and Section 5.5.

These parameters are used to perform simulations with various inputs and examine how the number of nodes, their combinations and connectivity can affect the security score.

5.3 5G specific Resource Generation

We also want to introduce 5G specific scenarios, with 5G specific resource generation. Few of the major highlights of 5G are network slicing, that allows for the dynamic allocation of resources, standalone and non-standalone architecture, accommodating different use cases. A module or infrastructure owner might be curious as to how the security score changes when a new slice is instantiated hence we want to be able to factor network slicing into our tool or simulate different use cases when choosing to opt for standalone or non standalone architecture.

We take 3GPP [38] documentation into account to head into the direction of 5G resource generation. There are two approaches that can be taken:

1. We keep our random generation model and introduce rules that control the generation in a more 5G specific manner, such as what node can be connected to which node and the connecting edges can be defined as interfaces, as per the 3GPP reference architecture.

2. We start with a seed graph from the 5G reference architecture and add additional resources to it.

We have chosen option 2 to proceed with as it gives us the flexibility to parameterize it to scale up and down as per need. Since each owner requires graphs of different scales, we give preference to scalability to start with.

The generated nodes will have characteristics representing the 5G Protocol Stack.

To go towards that goal, we define a template, which we can employ with the help of the **Template** parameter of our tool. The template contains a set of rules and constraints that govern the generation of 5G resources like 5G specific nodes (AMF, SMF etc), 5G specific interfaces, for example N4 interface that connects UPF with SMF and lastly, generation contraints, i.e. which node can be connected to which node, for example, an UE can only be connected to RAN and AMF and so on. Once this template is called by Argus-5, the generated resource graph will have representations of 5G resource nodes (compared to the default generated resource nodes like pc, firewall, routers etc.) and the edges will represent the interfaces. The generation constraints will ensure that the generated resource graph is an actual representation of the 5G network, blocking any arbitrary connectivity between the nodes.

For 5G specific resource graph generation, the following modules are needed:

- MEC module to represent the nodes pertaining to multi edge access computing. The MEC consists of the antenna units, routers interconnecting the AAU units and IoT applications.
- 5G Core module to represent the 5G core components.
- 5G RAN module to represent 5G RAN components

We generate MEC module using 5GEN (discussed in subsection 5.3.1) and preserve the information in dataframes. We import the dataframes to our tool for further preprocessing and used in our network creation. We generate the other required modules though our tool and set up the resource graph, appending the MEC module. Each node in the entire network is assigned a vulnerability along with a probability of exploitation i.e. the probability that the vulnerability will be exploited by an adversary. A target node is chosen at random. We plot the attack graph module by module i.e. each module independently computes their attack graph through to the target node (which can be the target node in the entire graph or setting the exit node as the target).

Figure 5.1 summarizes the graph generation steps.

5.3.1 Overview of 5GEN Tool

5GEN [90] creates graphs based on the MEC module's 5G network characteristics. The created graph comprises six M1 switches, each with six Active Antenna Units (AAUs) attached, which offer communication to devices at the network's end point, such as motherboards, surveillance cameras, and robots. The aggregation ring, which contains six M2 switches connected to four distinct access rings, sends traffic from the access rings to the core. Every pair of aggregation rings is then interconnected by 5GEN using two redundant M3 switches. To obtain the graphs, 5GEN uses a clustering approach. It divides the AAUs into clusters of six and connects each to an M1 switch. Then, as groups of M1 and M2 switches, access and aggregation rings are constructed.

5.4 Tool Performance

From importing dataframes from any third party software for additional topology information to deriving security score of a graph containing 50 nodes, it takes approximately 3.6 seconds for the whole process to complete. We were able to generate 700 graphs consisting 40 nodes on average and derive security score within a time frame of 20 minutes. The time taken to generate a single resource graph can be described using function T (measured in miliseconds), where T is a combination of the

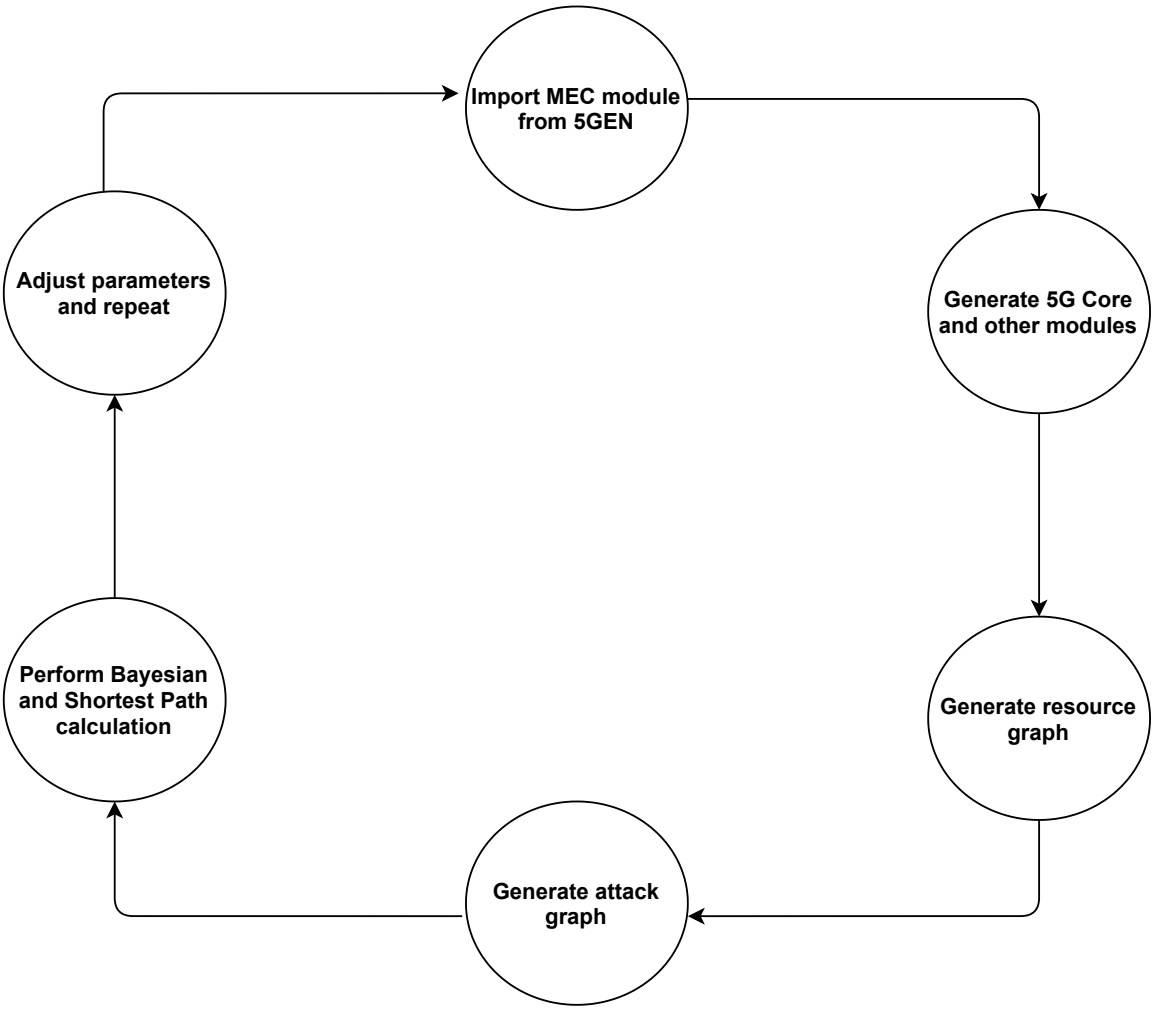

Figure 5.1: Summary of Graph Generation

time taken to generate the nodes (N) and edges (E) in the graph and can be expressed as:

$$T = N + E \tag{5.1}$$

For any given graph, the most complex scenario is when every node is connected to every node in the graph. For a graph with 40 nodes, each node can have an edge with 39 other nodes (assuming 1 to 1 connectivity between each pair of nodes), so the maximum number of edges each node can have is 39. Hence, if we apply equation 5.1, the time required to generate would be $T = 40 + 40 \times 39 = 1600ms$. This is an estimation of the most complex scenario of any given graph. The actual number of edges will depend on random assignment or the degree of connectivity value given as input by the user.

The time required to derive security score (T_S) of a single graph would be a combination of the time required to generate the resource graph T_R and the time required to generate the attack graph T_A and can be expressed as:

$$T_S = T_R + T_A \tag{5.2}$$

Since our tool facilitates generating graphs in bulk, the total time (T_t) required to generate X number of graphs can be expressed as:

$$T_t = T_{S_1} + T_{S_2} + ... + T_{S_X} \tag{5.3}$$

where T_{S_1} is the time required to generate and derive the security score of the first graph in the iteration and T_{S_X} is the time required to generate and derive the security score of the last graph in the iteration.

The equations provide us an estimated time required to generate the graphs, based on their complexity, which is dependent on the parameters of the tool. The actual generation time depends on the complexity as well as the configuration of the host machine the tool is running in.

5.5 Applications of Argus-5

Based on the design of our tool, there are several applications that can take benefit of our tool.

5.5.1 MSM Simulation

To validate our proposed Modular Security Metric (MSM) approach, our tool can generate simulations that can facilitate design and principles of MSM. Support for modules based on ownership boundaries are already an integrated part of the tool's design. Simulation design, conditions and results are further discussed in Chapter 6.

5.5.2 Machine Learning for Security Metrics calculation

Our tool can serve as a data set generator for machine learning models, designed to predict the security scoring or security label of a graph. We have automated the graph generation for our tool, which means mass number of graphs can be generated with given inputs. As machine learning models require sufficiently large amount of data for training, testing and validation, our graph generation and automation will aid in that task by generating sufficiently large number of graphs and additional data as part of the data set. Our tool is capable of producing large number of graphs with any given number of nodes and can be modified to suit the needs of the machine learning model. Our tool is already divided into separate functional compartments, with each part assigned to tasks of their own, which makes it easier to modify the tool to generate data sets as per requirement, as every model have their distinct inputs. The dataset is generated graph by graph. We generate a graph, record its data in text format and proceed to generate the next graph.

5.5.3 NFV/Cloud based Resource Generation

While we have strategies in place to facilitate 5G specific generation, as discussed in Section 5.3, our tool can also serve as a resource generator for cloud/NFV environments with layers. The design of our tool can accommodate the layered generation of resources, where we have the representations of physical machines in the bottom

layer, hosting VMs/Network functions/applications in the upper layer. This resembles the cloud/NFV architecture to an extended degree. Hence, we are also able to do simulations specific to cloud/NFV environments, by using the **Template** parameter to employ a cloud template.

Chapter 6

Simulation Discussion and Results

To validate our MSM approach, we have designed a simulation based on attack graphs. We utilize the 5GEN tool to generate nodes representing the MEC module, in addition to the CORE/RAN nodes generated through our tool.

For the experiments, we make use of 3 distinct sets of data: Local or target module's graph, Optimal scenario, which has no information loss as there is a free flow of information between the modules and Modular approach, which shows the scoring with information loss due to barriers.

6.1 Calculation Procedure

We perform the calculation in three steps, once for Bayesian and once for shortest path. At first, we compute the security score of the local graph, pertaining to the target module. In the next step, we perform calculation using static interfaces, by considering the path coming from other modules to the target module. In the final step, we perform dynamic calculation by probing the placements of the VMs by applying various permutations i.e. we probe by changing placements of VMs in different hosts across modules and see which placement leads to optimum score for the local module. Since we have multiple scores from the second and third step, we opt for the maximum of all the scores, which represents the most vulnerable state of the environment.

6.2 Simulation Results

We conducted 4 distinct simulations on multiple graphs (ranging from 90-110 graphs for each experiment), which we generated using different parameters, with the help of Argus-5.

Experiment 1: Simulate MSM on default operating conditions based on Bayesian calculation

In the first experiment, we want to investigate the feasibility of our modular security metric. We do not introduce any restrictions in the generation of the resource graph, other than constraining the range of vulnerability probability to 0.5-0.6, to control the degree of randomness in the graphs. While this may not properly accommodate the real scenario, where vulnerabilities are not assigned a restricted range, we focus on our original goal, which is to demonstrate how MSM performs against the Optimal case.

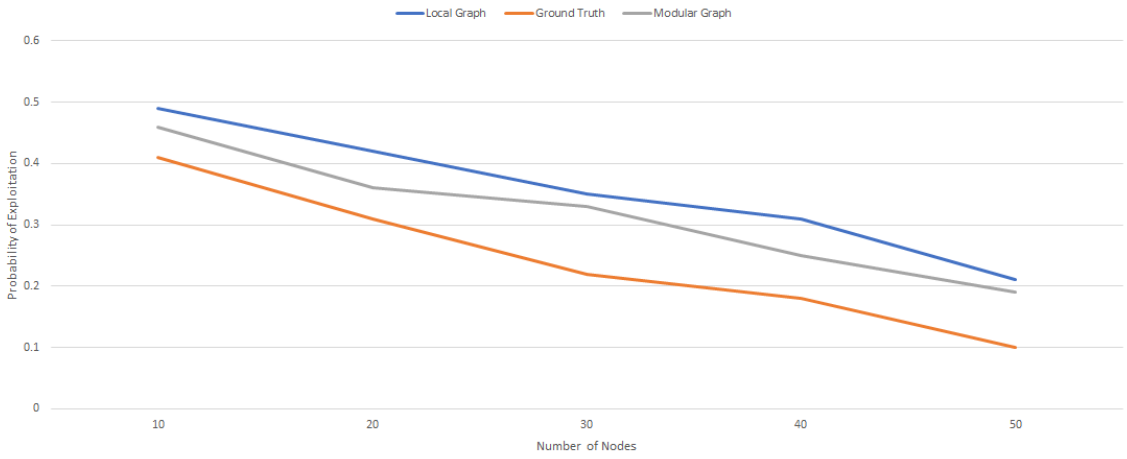

Figure 6.1: Experiment 1 result showing performance of MSM

As we can see in Figure 6.1, our MSM approach has a better score than local only approach and close to the ground truth. The difference in scores between the modular approach and the ground truth is due to the information loss caused by the exit points. The probability of exploitation goes down from local to modular to ground truth because as we start considering vulnerabilities from outside the local module, the difficulty for the attacker goes up, which is to say, for local only approach, the attacker needs to compromise less number of vulnerabilities to exploit the target, which indicates less effort. Then, as we start connecting vulnerabilities from other modules, which is the actual scenario, the attacker needs to put in more effort to compromise the target.

Experiment 2: Simulate MSM with increased degree of connectivity based on Bayesian calculation

In the second experiment, we want to study the effects on the probability of exploitation as we gradually increase the connectivity between the nodes.

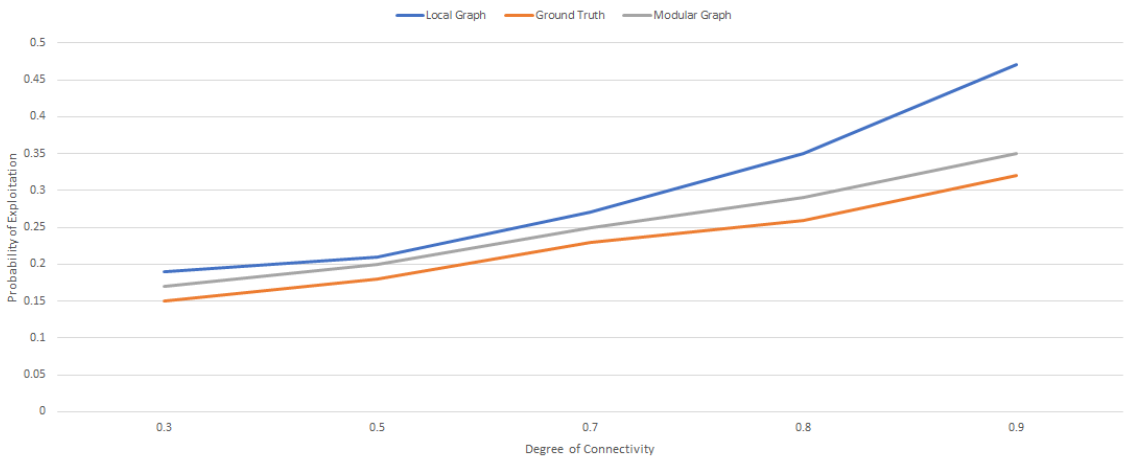

Figure 6.2: Experiment 2 result showing performance of MSM with increased connectivity

What we can observe from the results, as seen in Figure 6.2, is that, the target node become more and more vulnerable as the probability of exploitation increases as we increase the connectivity between them. This can be interpreted as more attack paths come into play when we introduce more connections between the nodes.

Experiment 3: Simulate MSM with Shortest Path calculation

In the third set of experiment, we wanted to explore the shortest path calculation as the metric, with regards to our modular security metric.

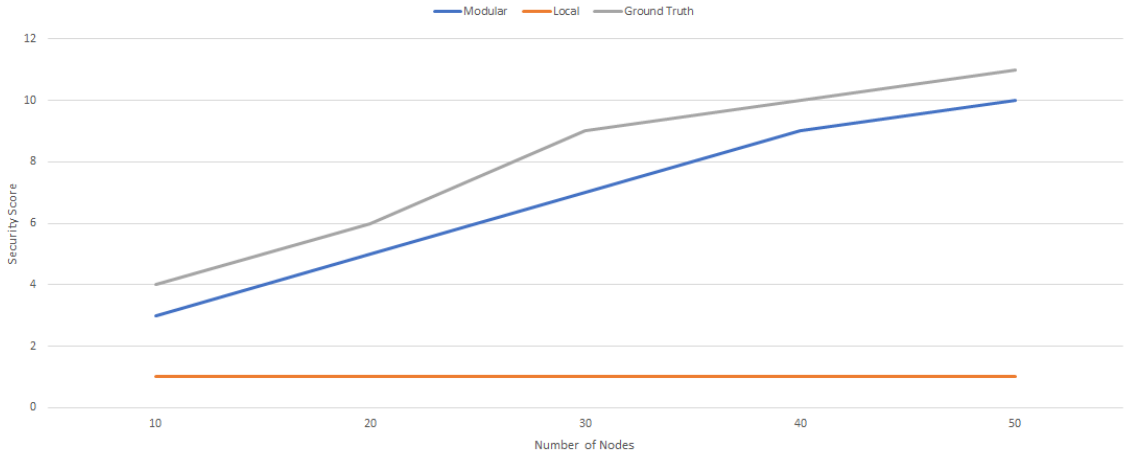

Figure 6.3: Experiment 3 result showing performance of MSM in Shortest Path approach

As we increased the number of nodes, as seen in Figure 6.3, the security score goes up as the shortest path increases. This is in relation to the complexity of the attack graph as more number of nodes in the resource graph will naturally lead to a more complex attack graph, hence the path count increases. However, for local graph, the shortest path remains constant because in the local graph, even when we added new nodes, the shortest path didn't change from 1.

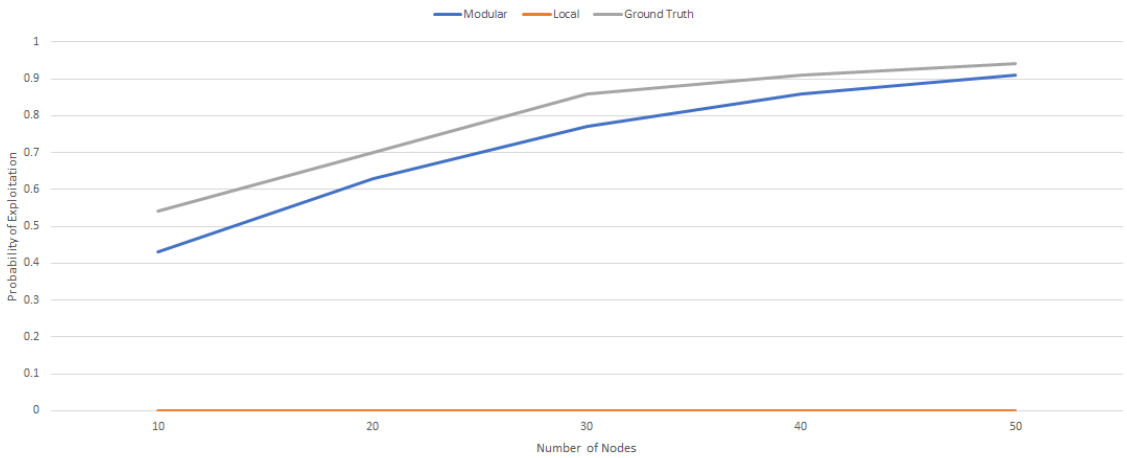

Figure 6.4: Experiment 3 result with normalized scoring

For comparison with previous experiments, we normalized the y-axis of Figure 6.4 using $\log_{0.08}(X)$, converting the shortest path count to probability of exploitation, which indicates the number of zero day vulnerability needed to reach the target. We saw a similar trend with Experiment 2.

Experient 4: Simulate MSM with increased degree of connectivity based on Shortest Path calculation

In the final set of experiments, we investigate the shortest path approach as we increase the connectivity between the nodes.

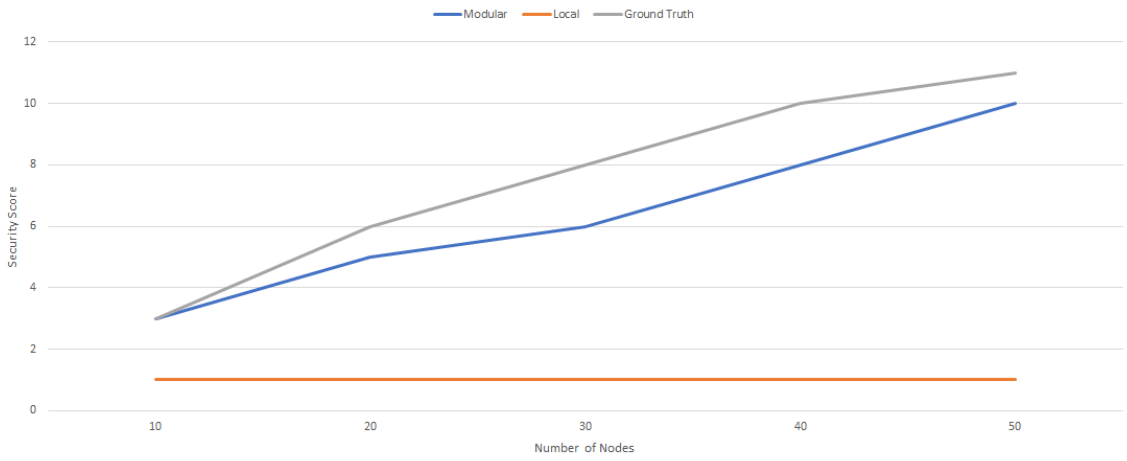

Figure 6.5: Experiment 4 result showing performance of MSM in Shortest Path approach with increased connectivity

As we increased the connectivity keeping the number of nodes consistent at each point and then taking the average, as seen in Figure 6.5, we see that there is hardly any noticeable difference with experiment 3, which leads us to believe that increasing connectivity does not have much effect on the shortest path calculation.

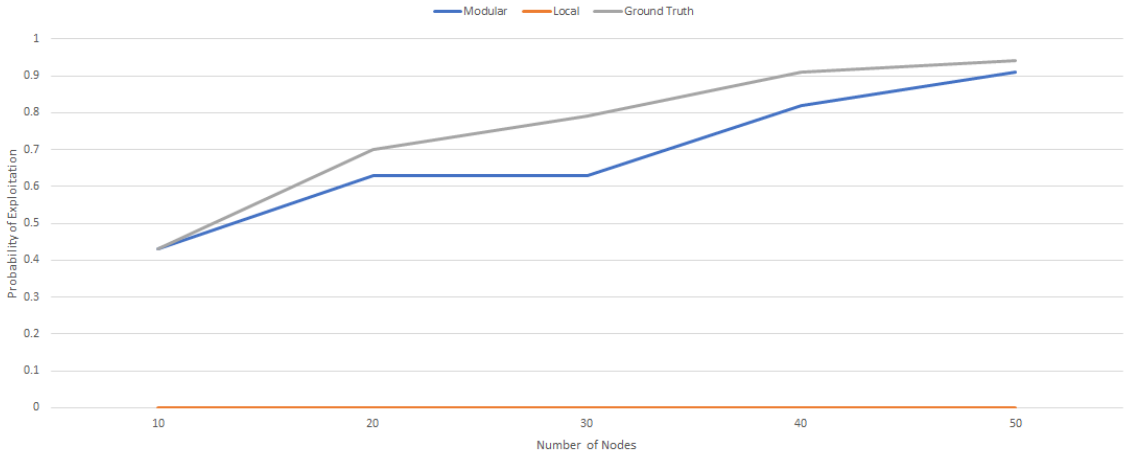

Figure 6.6: Experiment 4 result with normalized scoring

Chapter 7

Conclusion and Future Direction

As we reach the conclusion of this thesis, we want to revisit our stated contributions to confirm that they have been fulfilled and discuss future directions of this research

7.1 Revisiting Contributions

We conducted a survey of state-of-the-art of security metrics with two components: investigating the process of security metric derivation and the factors that can be addressed to make existing security metrics a better fit for 5G environment. While there have been numerous surveys conducted on security metrics, all of them were focused on particular scopes: system, networks etc. Our goal was to gain a better understanding of the domain of security metrics as a whole, irrespective of the scope. Hence, our survey was conducted from the perspective of risk management: we take a look at the risks in cybersecurity coming from different directions such as vulnerabilities, misconfigurations etc. We revisit the initial questions we wanted to answer with security metrics: which vulnerability to prioritize while deploying patches, how vulnerable we are regarding future attacks, where to place security controls etc. We have seen various proposals that demonstrate how security metrics could be the answer to those questions. This survey also supplies us with the knowledge and set the foundation stone for our next two contributions: first, how we can adapt security metrics to assess one of the major aspects of 5G: multi-ownership. 5G substantially moves away from traditional model of single owner, owning the whole operation stack, to accommodate multiple stakeholders. Hence, an adaptation of the existing security metrics were needed to make them applicable to 5G. Once the adapted security metrics was formalized, we needed a tool for an automated verification of MSM on sufficiently large number of graphs. Our exploration of existing tools of that purpose did not yield any fruitful result, leading us to design our own tool, Argus-5, which

went on to be our third contribution. We have demonstrated the design, purpose and analysis of our tool through various simulations, to show its effectiveness and capability.

7.2 Future Directions

While our research served as an assessment of one of the aspects of 5G, there remains some aspects that also need to be taken into account for security metrics to be more effective, such as how dynamic resource allocation could affect security, discussed as Gap 4 in Section 3.8, which needs more work to be addressed properly. 5G will continue to evolve everyday as more nations continue to roll 5G networks in their telecom sector, hence it is a very promising incentive to continue the research on security metrics for 5G. Our research was also limited to only taking software vulnerabilities into account for assessment, due to the limitations of existing tools and frameworks. We believe this creates a blind screen for stakeholders and users as only taking software vulnerabilities information will not be enough to reveal the whole security posture as other factors like compliance also affect how secure an environment can be. Lastly, we believe our designed tool can be further developed and improved to be more effective for security simulations, as its existing capability of handling multiple scenarios could be further enhanced.

References

[1] 3GPP TS 38.413. *NG-RAN; NG Application Protocol (NGAP)*, 2020. Rel. 15.

[2] Muhammad Abedin, Syeda Nessa, Ehab Al-Shaer, and Latifur Khan. Vulnerability analysis for evaluating quality of protection of security policies. In *Proceedings of the 2nd ACM workshop on Quality of protection*, pages 49–52, 2006.

[3] M. S. Ahmed, E. Al-Shaer, and L. Khan. A novel quantitative approach for measuring network security. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 1957–1965, 2008.

[4] Nawaf Alhebaishi, Lingyu Wang, and Sushil Jajodia. Modeling and mitigating security threats in network functions virtualization (NFV). In *DBSec'20*, pages 3–23. Springer.

[5] Nawaf Alhebaishi, Lingyu Wang, Sushil Jajodia, and Anoop Singhal. Threat modeling for cloud data center infrastructures. In *International Symposium on Foundations and Practice of Security*, pages 302–319. Springer, 2016.

[6] Hussain AlJahdali, Abdulaziz Albatli, Peter Garraghan, Paul Townend, Lydia Lau, and Jie Xu. Multi-tenancy in cloud computing. In *SOSE'14*, pages 344–351. IEEE.

[7] Andrea Atzeni and Antonio Lioy. Why to adopt a security metric? a brief survey. In *QoP'06*, pages 1–12. Springer.

[8] Lujo Bauer, Scott Garriss, and Michael K Reiter. Detecting and resolving policy misconfigurations in access-control systems. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–28, 2011.

[9] Jennifer L Bayuk. Security as a theoretical attribute construct. *Computers & Security*, 37:155–175, 2013.

[10] Yolanta Beres, Marco Casassa Mont, Jonathan Griffin, and Simon Shiu. Using security metrics coupled with predictive modeling and simulation to assess security processes. In *2009 3rd International Symposium on Empirical Software Engineering and Measurement*, pages 564–573, 2009.

[11] Leyla Bilge and Tudor Dumitraş. Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 833–844, 2012.

[12] Matt Bishop and Carrie Gates. Defining the insider threat. In *Proceedings of the 4th annual workshop on Cyber security and information intelligence research: developing strategies to meet the cyber security and information intelligence challenges ahead*, pages 1–3, 2008.

[13] Tim Bray. Measuring the web. *Computer networks and ISDN systems*, 28(7-11):993–1005, 1996.

[14] David Brumley and Dan Boneh. Remote timing attacks are practical. *Computer Networks*, 48(5):701–716, 2005.

[15] Shawn A Butler. Security attribute evaluation method: a cost-benefit approach. In *Proceedings of the 24th international conference on Software engineering*, pages 232–240, 2002.

[16] Eddy Caron, Anh Dung Le, Arnaud Lefray, and Christian Toinard. Definition of security metrics for the cloud computing and security-aware virtual machine placement algorithms. In *CyberC'13*, pages 125–131. IEEE.

[17] S. Chandra and R. A. Khan. Software security metric identification framework (ssm). In *Proceedings of the International Conference on Advances in Computing, Communication and Control*, ICAC3 '09, page 725–731, New York, NY, USA, 2009. Association for Computing Machinery.

[18] Stephen Checkoway, Lucas Davi, Alexandra Dmitrienko, Ahmad-Reza Sadeghi, Hovav Shacham, and Marcel Winandy. Return-oriented programming without returns. In *CCS'10*, pages 559–572.

[19] Wei Chen, Hongyi Lu, Li Shen, Zhiying Wang, Nong Xiao, and Dan Chen. A novel hardware assisted full virtualization technique. In *2008 The 9th International Conference for Young Computer Scientists*, pages 1292–1297. IEEE, 2008.

[20] Pengsu Cheng, Lingyu Wang, Sushil Jajodia, and Anoop Singhal. Aggregating cvss base scores for semantics-rich network security metrics. In *2012 IEEE 31st Symposium on Reliable Distributed Systems*, pages 31–40, 2012.

[21] Zachary A. Collier, Daniel DiMase, Steve Walters, Mark Mohammad Tehranipoor, James H. Lambert, and Igor Linkov. Cybersecurity standards: Managing risk and creating resilience. *Computer*, 47(9):70–76, 2014.

[22] Frédéric Cuppens, Nora Cuppens-Boulahia, and Joaquin Garcia-Alfaro. Misconfiguration management of network security components. *arXiv preprint arXiv:1912.07283*, 2019.

[23] Savino Dambra, Leyla Bilge, and Davide Balzarotti. Sok: Cyber insurance–technical challenges and a system security roadmap. In *IEEE S&P'20*, pages 293–309.

[24] Ron Davidson. The fight against malware as a service. *Network Security*, 2021(8):7–11, 2021.

[25] Matthew B Doar. A better model for generating test networks. In *Proceedings of GLOBECOM'96. 1996 IEEE Global Telecommunications Conference*, pages 86–93. IEEE, 1996.

[26] Frank Doelitzscher. Security audit compliance for cloud computing. 2014.

[27] Simon Yusuf Enoch, Jin B Hong, Mengmeng Ge, and Dong Seong Kim. Composite metrics for network security analysis. *arXiv preprint arXiv:2007.03486*, 2020.

[28] Ericsson. Ericsson mobility report, 2021. https://www.ericsson.com/en/reports-and-papers/mobility-report.

[29] David Ferraiolo, Janet Cugini, and D Richard Kuhn. Role-based access control (rbac): Features and motivations. In *Proceedings of 11th annual computer security application conference*, pages 241–48, 1995.

[30] Santiago Figueroa-Lorenzo, Javier Añorga, and Saioa Arrizabalaga. A survey of iiot protocols: A measure of vulnerability risk analysis based on cvss. *ACM Computing Surveys (CSUR)*, 53(2):1–53, 2020.

[31] Xenofon Foukas, Georgios Patounas, Ahmed Elmokashfi, and Mahesh K Marina. Network slicing in 5g: Survey and challenges. *IEEE ComMag'17*, 55(5):94–100.

[32] Xenofon Foukas, Georgios Patounas, Ahmed Elmokashfi, and Mahesh K. Marina. Network slicing in 5g: Survey and challenges. *IEEE Communications Magazine*, 55(5):94–100, 2017.

[33] Guillermo A Francia III. Vehicle network security metrics. In *Advances in Cybersecurity Management*, pages 55–73. Springer, 2021.

[34] Marcel Frigault, Lingyu Wang, Anoop Singhal, and Sushil Jajodia. Measuring network security using dynamic bayesian network. In *Proceedings of the 4th ACM workshop on Quality of protection*, pages 23–30, 2008.

[35] C. Fruhwirth and T. Mannisto. Improving cvss-based vulnerability prioritization and response with context information. In *ESEM'09*, pages 535–544.

[36] H. Ghani, J. Luna, and N. Suri. Quantitative assessment of software vulnerabilities based on economic-driven security metrics. In *CRiSIS'13*, pages 1–8.

[37] Ioannis Giannoulakis, Emmanouil Kafetzakis, George Xylouris, George Gardikis, and Anastasios Kourtis. On the applications of efficient nfv management towards 5g networking. In *1st International Conference on 5G for Ubiquitous Connectivity*, pages 1–5, 2014.

[38] gpp. 3gpp release 15, 2018. https://www.3gpp.org/release-15.

[39] Arthur Griesser. Ensuring high quality data transfer standards. 2004.

[40] Nabeel Hadaad, Luke Drury, and Ronald G Addie. Protecting services from security mis-configuration. In *2015 International Telecommunication Networks and Applications Conference (ITNAC)*, pages 120–125. IEEE, 2015.

[41] Hazem Hamed and Ehab Al-Shaer. Taxonomy of conflicts in network security policies. *IEEE Communications Magazine*, 44(3):134–141, 2006.

[42] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97, 2015.

[43] Yi Han, Jeffrey Chan, Tansu Alpcan, and Christopher Leckie. Virtual machine allocation policies against co-resident attacks in cloud computing. In *2014 IEEE International Conference on Communications (ICC)*, pages 786–792, 2014.

[44] HU Hao, LIU Yuling, and Hongqi ZHANG Yuchen ZHANG. Survey of attack graph based network security metric. *Chinese Journal of Network and Information Security*, 4(9):1, 2018.

[45] MGM Mehedi Hasan and Mohammad Ashiqur Rahman. Protection by detection: A signaling game approach to mitigate co-resident attacks in cloud. In *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, pages 552–559, 2017.

[46] Liuxing He, Xiao Li, Wenchang Shi, Zhaohui Liang, and Bin Liang. Visolator: An intel vmx-based isolation mechanism. In *Proceedings of the International Conference on Human-centric Computing 2011 and Embedded and Multimedia Computing 2011*, pages 245–257. Springer, 2011.

[47] Oliver Heckmann, Krishna Pandit, Jens Schmitt, and Ralf Steinmetz. Kom scengen the swiss army knife for simulation and emulation experiments. In *International Workshop on Multimedia Interactive Protocols and Systems*, pages 91–106. Springer, 2003.

[48] Peter Hedman. Description of network slicing concept. Technical report, NGMN Alliance, 2016.

[49] Michael Hogan, Fang Liu, Annie Sokol, and Jin Tong. Nist cloud computing standards roadmap. *NIST Special Publication*, 35:6–11, 2011.

[50] Aram Hovsepyan, Riccardo Scandariato, Wouter Joosen, and James Walden. Software vulnerability prediction using text analysis techniques. In *Proceedings of the 4th international workshop on Security measurements and metrics*, pages 7–10, 2012.

[51] Hong Hu, Shweta Shinde, Sendroiu Adrian, Zheng Leong Chua, Prateek Saxena, and Zhenkai Liang. Data-oriented programming: On the expressiveness of non-control data attacks. In *IEEE S&P'16*, pages 969–986. IEEE.

[52] GSMA Intelligence. Understanding 5g: Perspectives on future technological advancements in mobile. *White paper*, pages 1–26, 2014.

[53] Sushil Jajodia, Steven Noel, and Brian O'berry. Topological analysis of network attack vulnerability. In *Managing cyber threats*, pages 247–266. Springer, 2005.

[54] J. Jiang, L. Ding, E. Zhai, and T. Yu. VRank: A context-aware approach to vulnerability scoring and ranking in SOA. In *SERE'12*, pages 61–70.

[55] Cheng Jin, Qian Chen, and Sugih Jamin. Inet: Internet topology generator. 2000.

[56] Pontus Johnson, Robert Lagerström, and Mathias Ekstedt. A meta language for threat modeling and attack simulations. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, pages 1–8, 2018.

[57] Sami Kekki, Walter Featherstone, Yonggang Fang, Pekka Kuure, Alice Li, Anurag Ranjan, Debashish Purkayastha, Feng Jiangping, Danny Frydman, Gianluca Verin, et al. Mec in 5g networks. *ETSI white paper*, 28:1–28, 2018.

[58] James Kempf, Bengt Johansson, Sten Pettersson, Harald Lüning, and Tord Nilsson. Moving the mobile evolved packet core to the cloud. In *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 784–791. IEEE, 2012.

[59] M. Keramati, A. Akbari, and M. Keramati. Cvss-based security metrics for quantitative analysis of attack graphs. In *ICCKE 2013*, pages 178–183.

[60] Hisham A. Kholidy, Andrew Karam, James L. Sidoran, and Mohammad A. Rahman. 5g core security in edge networks: A vulnerability assessment approach. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6, 2021.

[61] Keith Kirkpatrick. Software-defined networking. *Communications of the ACM*, 56(9):16–19, 2013.

[62] Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Patrick Schweitzer. Foundations of attack–defense trees. In *FAST'10*, pages 80–95. Springer.

[63] Igor Kotenko and Elena Doynikova. Security metrics for risk assessment of distributed information systems. In *2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*, volume 02, pages 646–650, 2013.

[64] Igor Kotenko, Elena Doynikova, and Andrey Chechulin. Security metrics based on attack graphs for the olympic games scenario. In *2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 561–568, 2014.

[65] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2014.

[66] Nir Kshetri and Jeffrey Voas. 5g, security, and you. *Computer*, 53(3):62–66, 2020.

[67] D Kuipers. Common cyber security vulnerabilities observed in control system assessments by the inl nstb program. *Idaho National Lab.(INL), Idaho Falls, ID (United States), Tech. Rep*, 2008.

[68] K Anitha Kumari, G Sudha Sadasivam, S Shymala Gowri, Sebastin Arockia Akash, and EG Radhika. An approach for end-to-end (e2e) security of 5g applications. In *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing,(HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)*, pages 133–138. IEEE, 2018.

[69] Anil Kurmus, Reinhard Tartler, Daniela Dorneanu, Bernhard Heinloth, Valentin Rothberg, Andreas Ruprecht, Wolfgang Schröder-Preikschat, Daniel Lohmann, and Rüdiger Kapitza. Attack surface metrics and automated compile-time os kernel tailoring. In *NDSS'13*.

[70] Youngmi Kwon, Hui Jae Lee, and Geuk Lee. A vulnerability assessment tool based on oval in linux system. In *IFIP International Conference on Network and Parallel Computing*, pages 653–660. Springer, 2004.

[71] Fanny Lalonde Levesque, Jude Nsiempba, José M Fernandez, Sonia Chiasson, and Anil Somayaji. A clinical study of risk factors related to malware infections. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 97–108, 2013.

[72] Matthew J. Lanoue, James Bret Michael, and Chad A. Bollmann. Spoofed networks: Exploitation of gnss security vulnerability in 4g and 5g mobile networks. In *2021 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, pages 1–8, 2021.

[73] Ngoc T Le and Doan B Hoang. Capability maturity model and metrics framework for cyber cloud security. *Scalable Computing*, 2017.

[74] Juho Lee, Younsun Kim, Yongjun Kwak, Jianzhong Zhang, Aris Papasakellariou, Thomas Novlan, Chengjun Sun, and Yingyang Li. Lte-advanced in 3gpp rel-13/14: An evolution toward 5g. *IEEE Communications Magazine*, 54(3):36–42, 2016.

[75] Elizabeth LeMay, Michael D. Ford, Ken Keefe, William H. Sanders, and Carol Muehrcke. Model-based security metrics using adversary view security evaluation (advise). In *2011 Eighth International Conference on Quantitative Evaluation of SysTems*, pages 191–200, 2011.

[76] Tingting Li and Chris Hankin. Effective defence against zero-day exploits using bayesian networks. In *International Conference on Critical Information Infrastructures Security*, pages 123–136. Springer, 2016.

[77] Wei Li and Rayford B Vaughn. Cluster security research involving the modeling of network exploitations using exploitation graphs. In *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, volume 2, pages 26–26. IEEE, 2006.

[78] Guanjun Lin, Sheng Wen, Qing-Long Han, Jun Zhang, and Yang Xiang. Software vulnerability detection using deep neural networks: A survey. *Proceedings of the IEEE*, 108(10):1825–1848, 2020.

[79] Hao Lin and Pierre Siohan. Major 5g waveform candidates: overview and comparison. In *Signal processing for 5G: algorithms and implementations*, pages 170–187. Wiley, 2016.

[80] Richard P Lippmann and James F Riordan. Threat-based risk assessment for enterprise networks. *Lincoln Lab. J*, 22(1):33–45, 2016.

[81] Richard Paul Lippmann, JF Riordan, TH Yu, and KK Watson. Continuous security metrics for prevalent network threats: introduction and first four metrics. Technical report, Massachusetts Inst of Tech Lexington Lincoln Lab, 2012.

[82] Qian Liu, Chuliang Weng, Minglu Li, and Yuan Luo. An in-vm measuring framework for increasing virtual machine security in clouds. *IEEE Security Privacy*, 8(6):56–62, 2010.

[83] Xin Liu, Min Jia, Xueyan Zhang, and Weidang Lu. A novel multichannel internet of things based on dynamic spectrum sharing in 5g communication. *IEEE Internet of Things Journal*, 6(4):5962–5970, 2018.

[84] Emil C Lupu and Morris Sloman. Conflicts in policy-based distributed systems management. *IEEE Transactions on software engineering*, 25(6):852–869, 1999.

[85] B.B. Madan, K. Gogeva-Popstojanova, K. Vaidyanathan, and K.S. Trivedi. Modeling and quantification of security attributes of software systems. In *Proceedings International Conference on Dependable Systems and Networks*, pages 505–514, 2002.

[86] Frank Mademann. The 5g system architecture. *Journal of ICT Standardization*, pages 77–86, 2018.

[87] T. Madi, M. Zhang, Y. Jarraya, A. Alimohammadifar, M. Pourzandi, L. Wang, and M. Debbabi. Quantic: Distance metrics for evaluating multi-tenancy threats in public cloud. In *CloudCom'18*, pages 163–170.

[88] Pratyusa Manadhata, Jeannette Wing, Mark Flynn, and Miles McQueen. Measuring the attack surfaces of two ftp daemons. In *QoP'06*, pages 3–10.

[89] Pratyusa K Manadhata and Jeannette M Wing. An attack surface metric. *IEEE TSE'10*, 37(3):371–386.

[90] Jorge Martín-Pérez, Luca Cominardi, Carlos J Bernardos, and Alain Mourad. 5gen: A tool to generate 5g infrastructure graphs. In *2019 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 1–4. IEEE, 2019.

[91] Stephanos Matsumoto, Samuel Hitz, and Adrian Perrig. Fleet: Defending sdns from malicious administrators. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, page 103–108, New York, NY, USA, 2014. Association for Computing Machinery.

[92] Alain Mayer, Avishai Wool, and Elisha Ziskind. Fang: A firewall analysis engine. In *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pages 177–187. IEEE, 2000.

[93] Guerrino Mazzarolo and Anca Delia Jurcut. Insider threats in cyber security: The enemy within the gates. *arXiv preprint arXiv:1911.09575*, 2019.

[94] Fabiao Miao, Liming Wang, and Zailong Wu. A vm placement based approach to proactively mitigate co-resident attacks in cloud. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 00285–00291, 2018.

[95] Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Niels Bouten, Filip De Turck, and Raouf Boutaba. Network function virtualization: State-of-the-art and research challenges. *IEEE COMST'15*, 18(1):236–262.

[96] Maurizio Naldi. Connectivity of waxman topology models. *Computer communications*, 29(1):24–31, 2005.

[97] Kartik Nayak, Daniel Marino, Petros Efstathopoulos, and Tudor Dumitraş. Some vulnerabilities are different than others. In *International Workshop on Recent Advances in Intrusion Detection*, pages 426–446. Springer, 2014.

[98] William Newhouse, Brian Johnson, Sarah Kinling, Jason Kuruvilla, Blaine Mulugeta, and Kenneth Sandlin. Multifactor authentication for e-commerce: Risk-based, fido universal second factor implementations for purchasers. Technical report, National Institute of Standards and Technology, 2019.

[99] Cyril Onwubiko. A security audit framework for security management in the enterprise. In *International Conference on Global Security, Safety, and Sustainability*, pages 9–17. Springer, 2009.

[100] Rodolphe Ortalo, Yves Deswarte, and Mohamed Kaâniche. Experimenting with quantitative evaluation tools for monitoring operational security. *IEEE Transactions on Software Engineering*, 25(5):633–650, 1999.

[101] Xinming Ou, Sudhakar Govindavajhala, and Andrew W Appel. Mulval: A logic-based network security analyzer. In *USENIX Security'05*, volume 8, pages 113–128.

[102] Nicolae Paladi and Linus Karlsson. Safeguarding VNF credentials with Intel SGX. In *SIGCOMM'17*, page 144–146, Los Angeles, CA, USA, 2017.

[103] Mukul Pareek. Standardized scoring for security and risk metrics. *ISACA Journal*, 2:1–6, 2017.

[104] Marcus Pendleton, Richard Garcia-Lebron, Jin-Hee Cho, and Shouhuai Xu. A survey on systems security metrics. *ACM Computing Surveys (CSUR)*, 49(4):1–35, 2016.

[105] Teresa Pereira and Henrique Santos. A security audit framework to manage information system security. In *International Conference on Global Security, Safety, and Sustainability*, pages 9–18. Springer, 2010.

[106] Abida Perveen, Mohammad Patwary, and Adel Aneiba. Dynamically reconfigurable slice allocation and admission control within 5g wireless networks. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, pages 1–7, 2019.

[107] Shari Pfleeger and Robert Cunningham. Why measuring security is hard. *IEEE Security Privacy*, 8(4):46–54, 2010.

[108] Cynthia A. Phillips and Laura Painton Swiler. A graph-based system for network-vulnerability analysis. In *NSPW '98*.

[109] Aaron Scott Pope, Robert Morning, Daniel R Tauritz, and Alexander D Kent. Automated design of network security metrics. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1680–1687, 2018.

[110] Martin Preisler and Marek Haicman. Security automation for containers and vms with openscap. 2017.

[111] Kiran Radhakrishnan, Rajeev R Menon, and Hiran V Nath. A survey of zero-day malware attacks and its detection methodology. In *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, pages 533–539. IEEE, 2019.

[112] Alex Ramos, Marcella Lazar, Raimir Holanda Filho, and Joel J. P. C. Rodrigues. Model-based quantitative network security metrics: A survey. *IEEE Communications Surveys Tutorials*, 19(4):2704–2734, 2017.

[113] Lars Richter, Johannes Götzfried, and Tilo Müller. Isolating operating system components with intel sgx. In *SysTEX'16*, pages 1–6.

[114] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In *CCS'09*, pages 199–212.

[115] Fatemeh Salehi Rizi, Joerg Schloetterer, and Michael Granitzer. Shortest path distance approximation using deep learning techniques. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1007–1014, 2018.

[116] Benjamin D Rodes, John C Knight, and Kimberly S Wasson. A security metric based on security arguments. In *Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics*, pages 66–72, 2014.

[117] Sudhir K Routray and KP Sharmila. Software defined networking for 5g. In *2017 4th international conference on advanced computing and communication Systems (ICACCS)*, pages 1–5. IEEE, 2017.

[118] Julian L Rrushi. Scada protocol vulnerabilities. In *Critical Infrastructure Protection*, pages 150–176. Springer, 2012.

[119] Konstantinos Samdanis and Tarik Taleb. The road beyond 5g: A vision and insight of the key technologies. *IEEE Network*, 34(2):135–141, 2020.

[120] Danish Sattar and Ashraf Matrawy. Towards secure slicing: Using slice isolation to mitigate ddos attacks on 5g core network slices. In *2019 IEEE Conference on Communications and Network Security (CNS)*, pages 82–90, 2019.

[121] Karen Scarfone and Peter Mell. An analysis of CVSS version 2 vulnerability scoring. In *ESEM'09*, pages 516–525. IEEE.

[122] Steve Sheng, Mandy Holbrook, Ponnurangam Kumaraguru, Lorrie Faith Cranor, and Julie Downs. Who falls for phish? a demographic analysis of phishing susceptibility and effectiveness of interventions. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 373–382, 2010.

[123] Ming-Wei Shih, Mohan Kumar, Taesoo Kim, and Ada Gavrilovska. S-nfv: Securing nfv states by using SGX. In *SDN-NFV Security'16*, pages 45–48.

[124] Ak Ashakumar Singh and K Surchandra Singh. Network threat ratings in conventional DREAD model using fuzzy logic. *IJCSI'12*, 9(1):478.

[125] Anoop Singhal and Xinming Ou. Techniques for enterprise network security metrics. In *CSIIRW'09*, pages 1–4.

[126] Guochao Song, Wei Wang, Da Chen, and Tao Jiang. Kpi/kqi-driven coordinated multipoint in 5g: Measurements, field trials, and technical solutions. *IEEE Wireless Communications*, 25(5):23–29, 2018.

[127] Adi Sosnovich, Orna Grumberg, and Gabi Nakibly. Finding security vulnerabilities in a network protocol using parameterized systems. In *International Conference on Computer Aided Verification*, pages 724–739. Springer, 2013.

[128] Orly Stan, Ron Bitton, Michal Ezrets, Moran Dadon, Masaki Inokuchi, Ohta Yoshinobu, Yagyu Tomohiko, Yuval Elovici, and Asaf Shabtai. Extending attack graphs to represent cyber-attacks in communication protocols and modern it networks. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2020.

[129] Sal Stolfo, Steven M. Bellovin, and David Evans. Measuring security. *IEEE Security Privacy*, 9(3):60–65, 2011.

[130] Gary Stoneburner, Alice Y Goguen, and Alexis Feringa. Sp 800-30. risk management guide for information technology systems, 2002.

[131] Ahren Studer and Adrian Perrig. The coremelt attack. In *European Symposium on Research in Computer Security*, pages 37–52. Springer, 2009.

[132] Kun Sun and Sushil Jajodia. Protecting enterprise networks through attack surface expansion. In *SafeConfig'14*, pages 29–32.

[133] Tarik Taleb, Konstantinos Samdanis, Badr Mada, Hannu Flinck, Sunny Dutta, and Dario Sabella. On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration. *IEEE COMST'17*, 19(3):1657–1681.

[134] Rahul Telang and Sunil Wattal. An empirical analysis of the impact of software vulnerability announcements on firm stock price. *IEEE Transactions on Software Engineering*, 33(8):544–557, 2007.

[135] Thalesgroup.com. the DOCTOR ANR project, 2018. `https://github.com/DOCTOR-ANR`.

[136] Leendert Van Doorn. Hardware virtualization trends. In *ACM/Usenix International Conference On Virtual Execution Environments: Proceedings of the 2 nd international conference on Virtual execution environments*, volume 14, pages 45–45, 2006.

[137] Vilhelm Verendel. Quantified security is a weak hypothesis: a critical survey of results and assumptions. In *Proceedings of the 2009 workshop on New security paradigms workshop*, pages 37–50, 2009.

[138] David Wagner, Bruce Schneier, et al. Analysis of the ssl 3.0 protocol. In *The Second USENIX Workshop on Electronic Commerce Proceedings*, volume 1, pages 29–40, 1996.

[139] Ju An Wang, Hao Wang, Minzhe Guo, and Min Xia. Security metrics for software systems. In *ACM-SE 47*, pages 1–6, 2009.

[140] Lingyu Wang, Tania Islam, Tao Long, Anoop Singhal, and Sushil Jajodia. An attack graph-based probabilistic security metric. In *DBSec'08*, pages 283–296. Springer.

[141] Lingyu Wang, Sushil Jajodia, Anoop Singhal, Pengsu Cheng, and Steven Noel. k-zero day safety: A network security metric for measuring the risk of unknown vulnerabilities. *IEEE TDSC'13*, 11(1):30–44.

[142] Lingyu Wang, Sushil Jajodia, Anoop Singhal, Pengsu Cheng, and Steven Noel. k-zero day safety: A network security metric for measuring the risk of unknown vulnerabilities. *IEEE Transactions on Dependable and Secure Computing*, 11(1):30–44, 2014.

[143] Bing Wu, Jianmin Chen, Jie Wu, and Mihaela Cardei. A survey of attacks and countermeasures in mobile ad hoc networks. In *Wireless network security*, pages 103–135. Springer, 2007.

[144] Gang Xiong, Jiayin Tong, Ye Xu, Hongliang Yu, and Yong Zhao. A survey of network attacks based on protocol vulnerabilities. In *Asia-Pacific Web Conference*, pages 246–257. Springer, 2014.

[145] George OM Yee. Security metrics: An introduction and literature review. *Computer and Information Security Handbook*, pages 553–566, 2013.

[146] Bo Yi, Xingwei Wang, Keqin Li, Min Huang, et al. A comprehensive survey of network function virtualization. *Computer Networks*, 133:212–262, 2018.

[147] Awad A. Younis, Yashwant K. Malaiya, and Indrajit Ray. Using attack surface entry points and reachability analysis to assess the risk of software vulnerability exploitability. In *2014 IEEE 15th International Symposium on High-Assurance Systems Engineering*, pages 1–8, 2014.

[148] Mohamed Younis, Aseem Lalani, and Mohamed Eltoweissy. Safe base-station repositioning in wireless sensor networks. In *2006 IEEE International Performance Computing and Communications Conference*, pages 8–pp. IEEE, 2006.

[149] Hua Zhang, Fang Lou, Yunsheng Fu, and Zhihong Tian. A conditional probability computation method for vulnerability exploitation based on cvss. In *2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, pages 238–241, 2017.

[150] Shunliang Zhang. An overview of network slicing for 5g. *IEEE Wireless Communications*, 26(3):111–117, 2019.

[151] Lianying Zhao, Muhammad Shafayat Oshman, Mengyuan Zhang, Fereydoun Farrahi Moghaddam, Shubham Chander, and Makan Pourzandi. Towards 5g-ready security metrics. In *ICC 2021 - IEEE International Conference on Communications*, pages 1–6, 2021.

[152] Saman Zonouz and Parisa Haghani. Cyber-physical security metric inference in smart grid critical infrastructures based on system administrators' responsive behavior. *Computers & security*, 39:190–200, 2013.