CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Transferring Resilience Metrics: Evaluating the Adaptability of Resilience Metrics from Different Domains for Assessing Vehicle Resilience

Master's thesis in Computer science and engineering

Dino Pasalic
Mattias Retteli

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Transferring Resilience Metrics: Evaluating the Adaptability of Resilience Metrics from Different Domains for Assessing Vehicle Resilience

Dino Pasalic
Mattias Retteli

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Transferring Resilience Metrics: Evaluating the Adaptability of Resilience Metrics from Different Domains for Assessing Vehicle Resilience
Dino Pasalic Mattias Retteli

Supervisor: Magnus Almgren, Department of Computer Science and Engineering
Examiner: Magnus Almgren, Department of Computer Science and Engineering
Advisor: Pierre Kleberger, RISE Research Institute of Sweden

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Transferring Resilience Metrics: Evaluating the Adaptability of Resilience Metrics from Different Domains for Assessing Vehicle Resilience
Dino Pasalic
Mattias Retteli
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Recent advancements in the automotive domain have made vehicles more dependent on the Internet to make the driving experience smoother. This increases the attack surface as this new connectivity can be targeted and it cannot be assumed that these attack surfaces are secure. To still build secure systems resilience can be used. Resilience is, the extent a system can sustain attacks and still be functional, to a certain degree.

Currently there exists no metrics for cyber security resilience in vehicular design. This thesis aims to explore existing metrics in related domain to investigate the possibility of adapting those metrics into the automotive domain. To this end, a literature study is performed with 10 investigated papers with 244 metrics found. Attack-injection experiments are performed using the MODIFI tool and Simulink models on the adapted Health Index metric to evaluate the difficulty and practicality of adapting metrics. We found that it is possible to adapt resilience metrics from related domains into the automotive domain. However, depending on the complexity of the metric it can be difficult to adapt it as some modifications may be necessary to be able to use it within the automotive domain.

Keywords: cybersecurity, resilience, survey, automotive security, safety-critical systems.

Acknowledgements

We want to thank Magnus Almgren for being an excellent supervisor and supported us with valuable feedback in writing the report and how to properly focus it. We also want to thank Pierre Kleberger and Peter Folkesson from RISE for sharing their knowledge and wisdom of the automotive resilience sphere and for making sure that we are doing a proper job of understanding it.

Dino Pasalic, Gothenburg, 2024-06-09
Mattias Retteli, Gothenburg, 2024-06-09

Contents

List of Figures

List of Tables

1

Introduction

The current trend among vehicle manufacturers is to utilize electronic systems to enhance the driving experience and make driving safer. Using data generated by the car and its surroundings enables the use of safety features such as automated braking and automatic crash detection. Such systems can function well with little thought to cybersecurity when run in isolation, but current trends indicate that vehicles are becoming increasingly more connected. These external communication possibilities open up new attack vectors for malicious actors. Miller and Valasek [1] describe and demonstrate remote attacks against vehicles, some of which included increasing the volume of the stereo and shutting down the car. Furthermore, there have also been cases involving semi-autonomous cars being attacked, leading to devastating crashes [2] [3]. This highlights a problem which if left unattended may lead to property damage, injury to humans or in the worst case even lead to loss of life.

Thus, not only is it needed to prevent attacks on a vehicle, but it is also required that the vehicle continues to function under attacks. Critical systems, such as automated braking, shall still work in the presence of an attack. Hence, a resilient vehicle design is required where safety-critical functions can sustain cybersecurity attacks and continue to function, to a certain degree, to prevent injuries to drivers, passengers, and surrounding people.

There already exists research regarding resilience [4] in the cybersecurity domain. However, more research is required in the embedded safety-critical systems domain. There exists lots of suggestions and frameworks [4]–[6] for resilience mechanisms, although the actual implementations are not provided in these frameworks. This makes it difficult to properly implement the resilience mechanisms these frameworks propose. Furthermore, several resilience mechanisms for cybersecurity and safety have been identified [7]–[9]. However, the efficiency, interplay of these mechanisms, and their overall effect on the vehicle's resilience have not been sufficiently studied.

1.1 The need for resilience

Many common cyber attacks can be prevented if the system is up to date. This builds upon the fact that the attack has been seen before and can thus be mitigated via known countermeasures. This is a common method of protecting against attacks. However, when a zero-day exploit is found, systems are usually wide open with little in terms of protection.

When an attack eventually breaks a system's protections and affects the system, it can have major consequences. This is why resilience is important. Resilience enables the system to withstand and mitigate attacks (to a certain degree) that are affecting the system. The mitigation needs to be strong enough so that the system continues to function, even if that entails some degraded functionality. There is, of course, a limit to what attacks are reasonable for the system to survive. It is up to the system architect to design the resilience requirement and then up to the engineers to implement systems to fulfill them. A system will of course lose availability if it undergoes a severe malfunction, such as a power outage. But it is up to the system architect to design the system so that such an event is unlikely, and to foresee and prepare for such scenarios, for example a backup generator ready in the event of a power outage.

Resilience is very important in a safety-critical system like a vehicle as attacks on such system can have disastrous results [2], [3]. It is not as easy as just implementing resilience to achieve safety, as some traits that increase safety could potentially harm cybersecurity and the other way around [7]. This exemplifies how intricate of a problem this is.

Increasing resilience in a vehicle against attacks will help the system survive attacks, so that if an attack does go through, the potential damage such an attack can cause is reduced. This is why it is an important area to further research.

1.2 Definitions

Resilience spans multiple different domains such as economical, biological, infrastructure, cybersecurity, and safety, each with their own requirements and definitions [10], [11]. Even though all these domains share similarities, there are differences that require clarification. To properly understand this thesis, it is required to define what we mean with resilience. In this work, we use the *Cybersecurity for automotive systems in a changing environment (CyReV)* project definition of resilience in an automotive context:

> "Property of a system with the ability to maintain its intended operation in a dependable and secure way, possibly with degraded functionality, in the presence of faults and attacks. *Note to definition*: Dependable and secure refer to attributes such as safety, confidentiality, integrity, privacy and maintainability." [5]

Metrics are useful tools that can be used to gain insight into complex problems and systems. Resilience is an intangible property, and thus, it is difficult to measure without a proper approach. The right metrics can be used to specify resilience more concretely and clarify resilience. In this thesis, we specifically define a metric to be a measurement that developers and stakeholders can use to measure how well the system fulfills the resilience criteria. These metrics show how well a system can handle unexpected scenarios or if the system needs improvements in order to handle adverse conditions.

There exists multiple general design implementations that increase resilience called resilience mechanisms. A resilience mechanism is a strategy or implementation that accomplishes some sort of design requirement. Some examples of mechanisms have been identified in [7]–[9].

1.3 Goals and challenges

Currently, there is a need for further resilience research in the automotive area. As the interconnectivity and communication increases in vehicles, so do the attack surfaces. This increases the risk of attack/failure, and such malfunctions in vehicles could lead to severe consequences. This calls for an increase of resilience in vehicles.

1. Firstly, we aim to investigate the possibility of a relevant metric for resilience. This is done by investigating the viability of a metric for resilience in vehicular design and determine the effectiveness of resilience mechanisms using such a metric.
2. If it is found that such a metric can be determined, then the aim is to adapt that metric to quantify the effectiveness of some resilience mechanisms.

This means that we will investigate frameworks for creating metrics and explore already existing metrics in order to understand what is needed to create an effective metric. This is a non-trivial problem as resilience is an interconnected area. Thus, the deciding factors of the evaluation system will have to be thoroughly thought out. For example, a metric's measuring point needs to be precisely defined in order to measure relevant data. Furthermore, adapting metrics from other domains pose a problem because different domains have different demands. For example, the automotive domain requires high response time because of small error margins when driving in traffic. This is not as prevalent in some other cyber-physical systems where the response time might be larger.

1.4 Methodology

This project is divided into three parts. First we begin with a literature study, followed by an investigation of the found metrics. Lastly we evaluate some metrics by implementing some resilience mechanisms and examining the metrics behaviour.

1. **Surveying of metrics for quantifying the efficiency of resilient mechanisms in vehicle designs**

   We begin by exploring both past scientific literature and current research in resilience, as well as how current resilience mechanisms are evaluated. This helps us build on what has already been done as well as give us insight into important details we should take into consideration for possible metrics. We use different methods during the survey to make sure we find as many relevant papers as possible.

2. **Draw conclusions and potentially propose a metric**

After performing our survey, we want to see what conclusions can be drawn. With the gathered information we identify different categories for the collected metrics. We also investigate patterns that we have found in our survey and discuss why the patterns exists. It's important to take time to evaluate the results of the survey properly, as a poorly designed metric will yield poor results.

3. **Adapt a metric and evaluate its efficiency.**

   Lastly, we either chose a metric found from bullet item **2** to attempt to adapt into the automotive domain or if there are no metrics that can be adapted, creating a new metric will be explored. To evaluate how well the metric in question performs we use a modeling tool (Simulink[1]) to model a simple infotainment system of a car. This model will get injected with attacks with the help of the MODIFI tool [12]. The metric will then be evaluated with the help of the experiment results in order to draw conclusions.

1.5 Delimitations

Resilience is a concept in multiple domains, but this thesis only focuses on resilience in the context of the automotive area. Currently, the automotive area needs further resilience research, and as such, we use related domains, like Cyber-physical System (CPS) that we believe are relevant to the automotive setting in our thesis.

There exist many resilience metrics that may be adapted to an automotive context. However, we focus on testing one metric and its ability to measure in different implementations and mechanisms. This allows us to compare the resilience of the metric in different scenarios.

The experiments will be performed through simulations as we do not have the resources to test automotive systems in production.

1.6 Related works

This section discusses resilience research in the automotive domain that is relevant to our thesis. There exists a lot of research regarding cybersecurity resilience, despite that there exists no consensus on how it should be effectively measured. Furthermore, resilience in different domains needs to adhere to different requirements. Thus, one can not just use the same implementations for resilience in all domains. This means that there exists general resilience knowledge but applying this to specific areas like the automotive domain is difficult. This leads to a wide base knowledge level but a deficit in specified research in the desired domain of resilient vehicular design.

[1] https://se.mathworks.com/products/simulink.html

1.6.1 Literature surveys' and taxonomies

Some previous works have been done in order to summarize and review different resilience mechanisms but also categorize and define different terminology. Papers like [13][8] are literature surveys that compile and list different research in the area. [13] presents resilience metrics in the cybersecurity area while [8] presents different resilience approaches in the cyber-physical domain. Others, [14][15], aim to be technical sources of information regarding resilience metrics. There are also some papers that propose taxonomies on resilience in order to better define and categorize it. The paper [10] proposes a taxonomy over general resilience strategies. They mention that even though resilience is domain dependent there are some strategies that can be used for multiple domains for example, redundancy. Other papers investigate the stakeholder perspective. The paper [16] is a study of challenges that stakeholders face regarding resilience. It discusses, among other things, the lack of standardized frameworks and no standard practices.

Kleberger *et al.* [5] present a framework over how to define and determine weaknesses intended to help analyze attacks and faults. The framework defines different potential states that may occur during an error or compromised system, such as if an error is detected or handled.

Qurashi *et al.* [17] present different resilience countermeasures of autonomous automotive attacks. They mention in the paper that the autonomous automotive industry is predicted to grow substantially in the future, although there is little research regarding resilience in autonomous cars as it is a fairly new concept. This means that the literature regarding resilience in autonomous cars is currently lacking. This paper claims to be the first that reviews autonomous vehicle's resilience approaches and gives a taxonomy for naming them. This is partially done by defining resilience and levels of autonomy (SAE J3016) in vehicles. It also presents their own approach and taxonomy for defining resilient approaches. Since this paper claims to be the only of its type in the automotive resilience area, it further reinforces the lack of resilience research within autonomous automotive domain.

1.6.2 Frameworks

There exists a number of different frameworks created to help engineers and decision-makers make their systems more resilient. These frameworks focus on the process of getting the most out of the metric, how to pick the right metric for the job, and some common problems that may arise.

Bodeau *et al.* [4] present a framework for engineers, managers, and product owners to establish cyber resilience metrics to uphold the cyber resilience of their organization. The framework builds on the National Institute of Standards and Technology (NIST) cyber resilience goals defined in publication 800-160 v2 [18]. Bodeau *et al.* discuss the importance of establishing the goal of the cyber resilience metrics and their impact on the effectiveness of the end result. To that extent, Bodeau *et al.* provided some already defined metrics in [4]. However, to be applicable in a diverse range of systems, the abstraction level of the proposed metrics leaves the implementation

details unspecified. Furthermore, [4] highlights the challenge of a single figure of merit and the problems that it brings. Bodeau *et al.* also note there are different use cases for metrics, and some metrics may be used to reveal improvements to the system and provide areas to focus on. Others are used to prioritize investments, yet other metrics measure the ability to perform cyber resilience activities and to achieve cyber resilience sub-objectives.

Snyder *et al.* [6] propose a framework for developing and scoring metrics of cyber resilience from an organizational and engineering perspective. The paper builds this framework on a two-player game where *red* is an adversary trying to, to some extent, disrupt or intercept information from *blue*, and where *blue* is trying to stay ahead of *red*. The framework builds on the fact that an adversary can take one of four actions, *Access*, *Knowledge*, *Capability* and *Impact*. Snyder *et al.* highlight that breaking down each action into possible ways *red* can achieve its adversarial goal will showcase the necessary countermeasures that *blue* has to implement. For example, *Access* can be targeted, either the supply chain or by an insider threat. By doing this threat analysis *blue* can identify these attack paths as well as measure the effectiveness of *blue*'s countermeasures. The authors note that this list will be non-exhaustive just by the nature of how complicated and intricate resilience is. Still, it will serve as a guide to focus *blue's* resources effectively. Snyder *et al.* suggest measurements should be done via a three-tiered assessment. Firstly, responsible leaders shall estimate how well each of *red*s actions are countered. Secondly, each countermeasure found in the self-assessments should be externally verified and validated, the authors suggest subject matter experts should perform this assessment. Snyder *et al.* note testing is a good approach, however, it has limitations. Tests are only as good as the team performing the test, not finding anything is not indicative of a faultless system. Lastly, Snyder *et al.* suggest the last step be a mission-wide roll-up of all of the countermeasures.

Linkov *et al.* [19] propose a framework to better understand an organisation's needs in terms of cyber resilience. The framework combines the four proposed domains in [20] with the proposed domains in [21], which creates a generic matrix of resilience metrics. Linkov *et al.* highlight that the metrics should (1) be broad enough to be used in a diverse range of systems and (2) precise enough to measure specific system processes and components. Unfortunately, this means that the granularity of the resulting metrics is vague to the point of them being more guidelines than metrics.

1.6.3 Contributions

As shown above, there exists cyber resilience surveys [13][8], however none of them have the automotive area as the main focus. Thus, we have created a literature survey with automotive cyber resilience metrics as the focus.

Furthermore, we have created a practical implementation comparing the different implementations, as opposed to [14][15] that take a theoretical approach. In contrast to the above papers we have adapted a metric from one cyber resilience domain to the automotive domain.

1.7 Outline

This thesis is divided into six chapters. First is the introduction, where we outline the general problem as well as our approach and delimitations. In Chapter 2, we present a short background chapter which contains different terminology and definitions that is important for our thesis. The third chapter contains the literature survey methodology and result of our literature study, where the most interesting and relevant papers will be discussed. In the fourth chapter, we discuss the patterns found in our literature survey and what we believe cause these patterns. Finally, Chapter 6 provides a conclusion, ethical and sustainability considerations and future work.

2

Background

This chapter introduces necessary materials and concepts for the readers of this thesis.

2.1 Metric and measure

The two words metric and measure are very similar and are sometimes used synonymously. However, they are different, Oxford Learner's Dictionary defines a metrics as:

> "a set of numbers or statistics used for measuring something, especially results that show how well a business, school, computer program, etc. is doing" [22]

There are a few different interpretations of what a metric should do ([6], [14], [23], [24]). They all boil down to giving insight into a problem by creating objective, reproducible and quantifiable measurements. The important distinction that needs to be made is that a metric is the function gathering the data while a measure is the result of said function. An example of this is a thermometer, where the thermometer is a metric that outputs degrees. The degrees that you can read of a thermometer becomes the measure the thermometer is outputting.

2.2 Safety and cybersecurity

Safety and cybersecurity are two terms that are very involved in Cyber-Physical Systems (CPS). According to Sabaliauskaite and Mathur [25] safety protects the system against accidental failures that could cause hazards while security protects the system against intentional attacks. While we agree that safety is about accidental failures and security is about intentional attacks, since we are working with cybersecurity this definition is slightly changed. We define safety as more concerned about physical safety. Security on the other hand is more focused on the digital security with network and data security. For clarity we will refer to this property as cybersecurity.

2.3 Faults and attacks

Faults and attacks are two very similar concepts that sometimes even have the same consequences, although there is an important distinction we want to make clear when we use these words. When we use the word faults, we mean a natural fault the resource is causing. An example could be running out of battery or a pump eventually breaking down because of use, which also applies to poorly built software. On the other hand, an attack would be someone intentionally breaking or sabotaging a resource. An attacker is intentionally causing the resource to behave abnormally. Some examples of this would be an attacker hijacking a resource or collecting unauthorized data, but can also be physically sabotaging a pump. The consequences of faults and attacks can look very similar, and the critical difference is the cause of the mishap.

2.4 Defining resilience

In this chapter we define what we mean with resilience and how we use it. We also define metrics and mechanisms that are relevant to resilience.

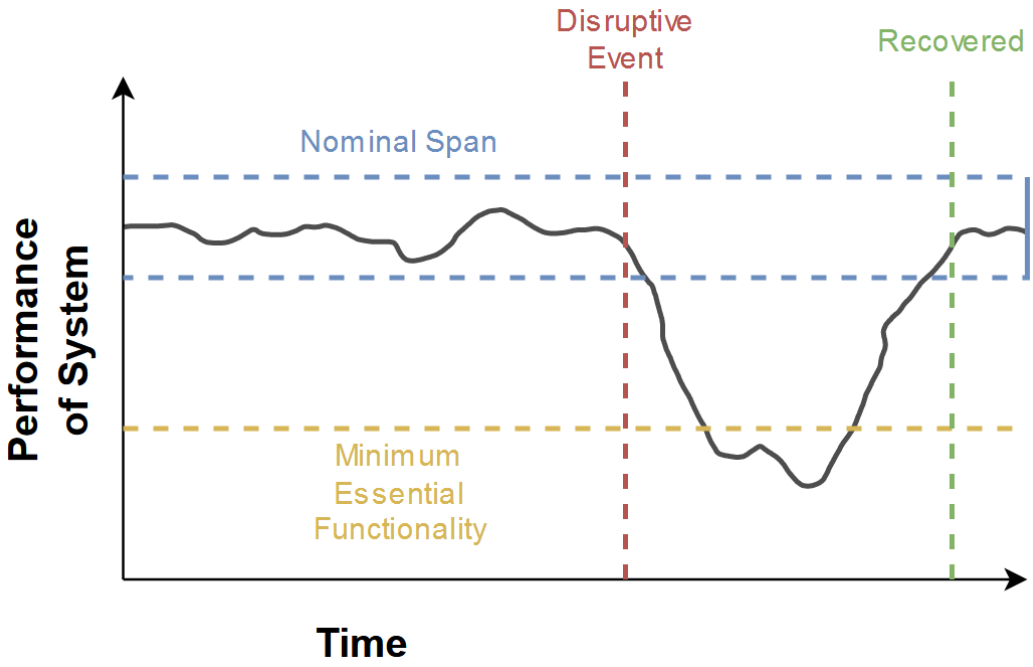


Figure 2.1: System performance according to a metric. Picture redrawn from the imagine in [4]

A metric measures an attribute or a mechanism's effects on the system. This can be represented by a line graph as seen in Figure 2.1. The gray line in the figure represents the measured system performance. The nominal span indicates the system's standard performance to the relevant attribute under normal conditions, indicated by the span with the blue outline.

Handling methods should be activated if the system's performance drops below the nominal span. The system will still be able to accomplish its goal while under the nominal span, although with potentially degraded functionality. When the system's performance is under the minimum essential functionality, it indicates compromises in the system's resilience, and as such system output can not be guaranteed. In Figure 2.1, the minimum essential functionality line is noted by a yellow line.

There are multiple disruptive events that can cause the system performance to decrease, for example, a cyberattack or a hardware fault. The start of the disruptive event is marked by the red line in Figure 2.1. Logging when a disruptive event occurs is interesting as it aids in finding when and where the weaknesses of the system lie. If the handling mechanism is successful, the system will eventually regain performance and be within the nominal span again. The green line in Figure 2.1 indicates when the mechanism successfully recovered.

2.4.1 Defining resilience metrics

A metric is a helpful tool used to gain insight into complex problems. Metrics measuring resilience does not have a unanimous definition. Even if not clearly defined, a resilience metric follows a similar structure as other software metrics. According to Snyder *et al.* [6], the purpose of a metric is to support decisions that are made to the software. This is done by helping a developer understand, control and therefore improve the software[23]. With multiple existing definitions of resilience metrics, we want to be clear how we define resilience metrics in this thesis.

In this thesis, we specifically mean metrics that developers and stakeholders can use to measure how well the system fulfills the resilience requirement of a particular system. We, therefore, focus on metrics that measure resilience. These metrics will show how well the system can handle unexpected scenarios or if the system needs improvements in order to handle adverse conditions.

Two types of metrics exists.

1. Metrics that can be objectively measured without the need for any human input. An example is measuring the outside temperature.
2. Metrics are metrics that require some sort of human/expert in the area in order to evaluate it properly. An example of this would be if the outside temperature is perceived to be hot or cold. The expert goes out and makes a judgement which is based on multitude of factors like the wind, humidity and their clothing to then give an answer. The expert then compiles all these factors and outputs an answer.

1 is referred to as data driven metrics while **2** is referred to as expert driven metrics

2.4.2 Defining mechanisms

A mechanism is a strategy or implementation that accomplishes some design requirements. There exist many different mechanisms. However, as defined above, we will

only focus on mechanisms that contribute to resilience. Sangchoolie *et al.* [7] have identified 17 unique mechanisms for automotive that provide resilience in some way. Some examples are access control, encryption, operating system, and signatures.

All resilience mechanisms can be further divided into three sub-categories: Prevention, Detection, and Handling/Reaction [8]. Mechanisms that are part of prevention are the first line of defense and increase the necessary resources needed to attack the system successfully. The goal of detection mechanisms is to identify that an attack is happening and, when it does, put the system on high alert by triggering handling mechanisms. When/if an attacker makes it into the system or otherwise can influence the system, that is when handling mechanisms come into play. A handling mechanism is supposed to ensure the system can still function correctly under attack/fault (possibly with degraded capacity). Furthermore, the handling mechanism is also responsible for ensuring that other parts of the systems function correctly during the attack and are not affected by the ongoing attack.

3

Metrics in literature

Chapters 1 and 2 serve as an orientation of the current state of resilience in research and practice. In this chapter, we conduct a survey of the current literature to establish a basis of current resilience research. This entails gathering information on existing solutions as well as problems with no current solution.

This chapter discuss the method used in the survey as well as the most relevant papers we have surveyed. First we explain the method used to gather all the papers, and then at the end we present the most relevant metrics found from our survey along with a discussion of the domains.

3.1 Method

A literature survey helps build on what has already been done and gives insight into important details that should be taken into consideration for possible metrics. This literature survey consists of three different surveying methods. First, we do a database search to find a wide area of differing relevance. Then, we search some of the most attended conferences relevant to cybersecurity and CPS. Lastly, we do a search engine search which catches any missed papers from the previous two methods. An overview of the results can be seen in Table 3.1.

The identified papers are screened for relevance. The sorting process is divided into a first screening and a second screening. The first screening is done by reading the title and abstract. If the abstract and title are not in the scope of the thesis, the paper is not furthered considered. When all papers have undergone the first screening, the second screening is conducted. The second screening consists of a cursorily reading of the papers. This step provides a better understanding of the contents and relevance of the paper. If it is found to be outside of the scope of the thesis, the paper is discarded. In this step, we also note interesting quotes and images or ideas for further investigation. If a paper passes both screenings, it is then thoroughly read.

3.1.1 Database search

The first part of the literature survey is a database search. We search through the Scopus[1] database to start with a large set of papers. We decided on Scopus because it allows for the filtering of keywords, domains, article types, languages, and abstracts. Furthermore, it is easy to use because we have access through our institution. In our search, we decided to only search for English papers because readers of this thesis should also be able to read the sources. We also decided to search only for papers in the computer science domain. Even though resilience exists in other domains, they are considered as outside our scope. We started searching for *resilience* in the keywords, title, or abstract of the papers. However, this resulted in a large number of papers (around 22,000 papers), so we narrowed the search. We added more keywords: *metric* and *software*. We also searched for papers that may contain *recovery*, *error handling*, *fault tolerance*, or *survivability*. This gave around 600 returns. Many of the papers were focused on wireless communication or hardware. We therefore excluded *hardware*, *wireless*, *memory* from the keywords. The final number of found papers was 311. The entire Scopus search query can be found in appendix C. The papers were then sorted according to the two-step screening process described above.

3.1.2 Conferences search

We also searched through some of the largest conferences for papers related to resiliency. The reasoning behind also searching for conference papers is that conference papers tend to be more concrete and thus can shed some light on how certain metrics can be implemented. Knowing how other metrics should be implemented can be beneficial to consider when it is time to adapt the metric. The conferences searched are: NDSS, ACM CCS, IEEE Security and Privacy, Euro Security and Privacy, Usenix security, ACSAC, and DSN. These were chosen because they are all cybersecurity and fault tolerance conferences and thus will attract relevant papers for us. The specific conferences were chosen based on their popularity rating where we decided on primarily conferences that are considered A - A*. Furthermore, both security and fault tolerance focused conferences were chosen as they can provide unique insights into their respective area. Many of these conferences have been running for a while, and this means that there are many papers to search for. Since we can not search them all, a cutoff point was set. We will exclude conferences earlier than 2017 for all the conferences. The reasoning behind this is that a cursory search of papers before 2017 contained few to no resilience papers.

3.1.3 Search engine search

Lastly, we perform a search engine search through Google Scholar[2]. This means that we do a snowball search, catching any papers we might have missed with the other methods. This involves both searching the Internet for papers and using the

[1] https://www.scopus.com/home.uri
[2] https://scholar.google.com/

forward and backward citations feature in Google Scholar. This allows us to follow the cites, both backward in time and forward, to find papers using the original as a source. We use this method to find papers relevant to the topic without having to screen resilience papers outside our scope. The process is as follows:

1. We pick a paper as a starting point. This could be a paper that we find interesting in one of the previous search strategies or a paper we find through searching the Internet.
2. Using Google Scholar, we can see papers that use the starting paper as a reference as well as follow the citation trail forward and backward.
3. Using this method, we can find more papers with the same or neighboring topics as the original paper.

3.1.4 Strategy evaluation

While doing the search, we noticed some traits about how successful each method was and the quality of the found papers for each method. Our findings, as well as a short discussion, are presented below.

- **Database:** The database search gave about 15 papers that passed both screenings, but most of the papers from the search were outside our scope. The number of papers that underwent the first screening was 311. From that about 30 proceeded to the second screening, where we ended up with 15 papers. The resulting papers were of varying degrees of relevance to our thesis. Some were very useful and helpful, while others touched on the relevant points but lacked a satisfactory depth to them. This might be because we picked the wrong search terms and keywords. It could also be that there is a lack of research on the details of resilience.
- **Conferences:** We searched multiple highly rated conferences. However, they gave little returns in terms of relevant papers. We had hoped that conferences would have some concrete implementations of metrics measuring resilience. Unfortunately, we found none. Out of all the papers that were gathered from the conferences, none of the conference papers were used in the final literature survey as they were all still on a quite high level. This could mean either of two things: (1) we did not search for the correct conferences, or (2) we might have overlooked some papers when screening our found conference papers. The other possibility is that resilience is not suitable for conference papers at this current time. This argument is supported by the lack of available concrete implementations of resilience that can be found in general. It can thus be hard to present concrete implementation when most of the resilience area is still rather abstract.
- **Search engine:** Most of the papers found via the search engine search passed the screenings. This might be because this is the method we are most familiar with or that search engine search is better suited for finding papers relevant to cybersecurity resilience.

3.2 Metric attributes

After we collected papers from our survey, we read through them all and noted down the metrics they presented. Some papers presented only one metric while other papers presented multiple metrics.

The reason we collect the metrics we found in the papers is to create a catalogue of metrics we could potentially adapt into the automotive domain and to deepen our understanding of resilience metrics. By sorting all the metrics we found by what they measure, we can gather some information on popular methods of measuring certain aspects. We can also see similarities and differences in how different metrics measure the same or similar properties. This was done by identifying reoccurring themes among the found metrics. For each metric we looked at, among other things, how the measurement is taken, the unit of the result and how general the metric is. Then a judgement had to be made of how many different categories were to be presented. The four categories presented below where chosen as they encapsulated all metrics and highlights the measurement type clearly.

- **Time:** Metrics that result in some measure of time or that use time as a measure to create the metric fall into this category. A general example of this are metrics that measure the availability of a system.
- **Amount:** Metrics that measure something countable, for example, number of attacks, or number of accounts assigned to personnel that have left the company, are categorised as Amount metrics.
- **Score:** The score metric has been defined as a measurement that results in one or multiple values. The result of a score metric is seen as a value without a unit. Score metrics combine different attributes of a system and with the help of weights, combine their results into a single value. What the value mean is decided by the score metric implementation and can change between implementations.
- **Throughput:** Throughput is the rate at which something is processed, thus metrics which are a combination of *amount* and *time* attributes are characterised under this category. A example of this metric is the number of packets sent per second in a communication channel.

3.3 Metric classification

The metrics identified in the papers can be classified into three different implementation levels. This implementation level describes to what extent the metric has been implemented in the respective paper. The three levels are defined as follows:

- **Framework:** A framework paper describes how to implement a metric, but without doing it itself. It gives a course of action or the steps needed to create a metric for a given system.
- **Theoretical:** A theoretical paper describes the metric, discusses the metric

and its capabilities. However, this type of paper does not use the metric in a simulation or practical setting to show the metric working.

- **Practical:** A practical paper describes as well as implements a metric.

3.4 Dependability attributes

According to Avizienis *et al.* [26] dependability consists of the attributes of:

- Availability: readiness for correct service.
- Reliability: continuity of correct service.
- Safety: absence of catastrophic consequences on the user(s) and the environment.
- Integrity: absence of improper system alterations.
- Maintainability: ability to undergo modifications and repairs

And, when discussing security an additional attribute is that of confidentiality i.e., the absence of unauthorised disclosure of information. However, when it comes to cybersecurity the papers mention how availability should be seen as "availability for authorized actions only" while improper should be read as unathorized for the definition of integrity. Thus, the papers identified during the survey are mapped to those dependability attributes that they refer to and presented in Table 3.1.

3.5 Results of literature survey

The result of our classification of the papers is presented in 3.1. The summarized results of each metric type can be seen in Table 3.2.

Table 3.1: Results of surveyed papers

Title	Time	Amount	Throughput	Score	Implementation	Domain	Dependability Attributes
On the Resilience of Machine Learning-Based IDS for Automotive Networks [27]	0	4	0	0	Practical	Automotive	Na
A Resilience Metric and Its Calculation for Ship Automation Systems [28]	2	3	0	0	Practical	Cyber-Physical	Availability, Reliability
An Effective Semantic Security Metric for Industrial Cyber-Physical Systems [29]	0	0	0	4	Theoretic	Cyber-Physical	Na
An empirical study of software architecture resilience evaluation methods [30]	1	3	2	2	Practical	Cyber-Physical	Availability, Reliability
Measurement and Analysis of Cyber Resilience for Control Systems: An Illustrative Example [31]	0	0	0	4	Practical	Cyber-Physical	Confidentiality, Integrity, Availability, Reliability
Structure-based resilience metrics for service-oriented networks [32]	0	2	0	0	Practical	Cyber-Physical	Availability
Towards New Metrics for High-Performance Computing Resilience [33]	1	1	0	0	Practical	Cyber-Physical	Integrity, Availability
AWR: Anticipate, Withstand, and Recover Resilience Metric for Operational and Planning Decision Support in Electric Distribution System [34]	0	0	0	3	Practical	Cyber-Physical	Availability, Reliability
Complete Guide to Security and Privacy Metrics: Measuring Regulatory Compliance, Operational Resilience, and ROI [13]	7	84	0	2	Framework, Theoretical	Organisational	Confidentiality, Integrity, Availability, Reliability
Cyber Resiliency Metrics, Measures of Effectiveness, and Scoring [4]	52	58	6	3	Framework, Theoretical	Organisational	Integrity, Availability, Reliability

Table 3.2: Summary of the investigated metrics

Domain	Time	Amount	Throughput	Score	Paper
Automotive	0	4	0	0	[27]
Cyber-Physical Systems	4	9	2	13	[28]–[34]
Organisational	59	142	6	5	[4], [13]

We will now present some observations that can be gathered from Table 3.1. The number of organizational metrics vastly surpassed the number for any other domain. This is because the two sources for cybersecurity were large books that focused on metrics. There were a decent amount of CPS metrics that measured specific situations and were, in general, the most in-depth metrics that we found from any category. CPS is quite a large area, and we noticed a lot of electrical power grid metrics and network architecture resilience metrics. Some of these were still interesting and useful for us, although it was difficult to use any electrical power grid metrics as we felt that a lot of what makes those resilient were not applicable to the automotive domain. We also found that quite a few metrics in the CPS category resulted in some kind of score. This might be because many of the papers discussing metrics were interested in evaluating a larger system.

We only managed to find one paper talking about automotive metrics. This could be because our literature search was poor, there are not many studies on automotive metrics or a combination of both.

We wanted to showcase some of the more interesting metrics and explain them as they give insight into what we should think about when deciding metric.

The following section will review some of the more interesting metrics in more detail. We will reason why we believe these to be interesting metrics and explain how they can help us create our own metric. This does not mean the metric itself or what it is measuring is interesting, but rather how it measures and what variables and angles used that might be applicable for us? We will do this in order of the closest domain, starting with automotive, following up with CPS, and ending with the organisational domain.

3.6 Metrics found in the automotive domain

While the automotive domain is the most relevant domain for us, we failed to find a lot of metrics here. All the metrics presented in this section come from the paper "On the Resilience of Machine-Learning (ML) - Based Intrusion Detection System (IDS) for Automotive Networks" by Zenden *et al.* [27]. These metrics were more focused on ML evaluation rather than automotive resilience because of the nature of the paper.

3.6.1 False-Positive-Rate

False-Positive-Rate is a volumetric metric that counts every false positive encountered by the ML algorithm. This means that every time a negative is flagged as positive, this number goes up. This is useful to measure what weaknesses a ML-IDS has, if this number is high, it could indicate a successful evasion attack. This is a valuable metric to keep in mind for eventual ML implementations for the automotive system.

3.6.2 False-Negative-Rate

Similar to False-Positive-Rate, False-Negative-Rate measures the amount of positives that get marked as negatives.

3.6.3 Accuracy

Accuracy gauges how accurate the ML guesses are. It does this by measuring how many of the ML decisions are correct with its test data. This has a problem of being misleading in unbalanced data, though, as in the absence of any errors, the accuracy number will be highly inflated. To counter this, a F1-Score is implemented as described in 3.6.4. A metric like this, measuring how well a task is doing what it is supposed to, could be a useful one, although it serves as a reminder that depending on the situation, it could be biased and might require another metric in tandem with itself.

3.6.4 F1-Score

The F1-Score is a score that helps evaluate how biased the accuracy metric is. This is done by computing the harmonic mean of precision and recall. Precision is calculated by diving the true positives with all positives while recall is calculated by diving true positives with true positive added with false negatives. If we have an accuracy score of 80% but an F1 score of 0, we can say that the model did not detect any attacks. While this metric is very specific to ML, it is still interesting if we decide to look into ML for automotive even further. This also helps us realize that some metrics might need to rely on other metrics in order to be more useful.

3.6.5 Discussion: automotive metrics

The presented metrics evaluated the ML-based IDS but did so in a very ML-focused way, as all of these are metrics to gauge how well the ML did its part. While that is good for the purpose, it is not quite helpful in the general automotive domain. We did not manage to find any more concrete metrics in the automotive domain. This could be the result of poor usage of Scopus or a lack of automotive metrics, or perhaps a combination of both. Even though the metric evaluates something within the automotive industry, it evaluates the ML part more than anything related to the automotive sphere. The concluding remarks from the paper were that this

technology is still very immature, as the ML model can be affected by adverse data without any real knowledge of how the internal network looks.

3.7 Metrics found in the cyber-physical systems domain

CPS is quite a wide area that automotive falls under. The reason the two domains are separated in our thesis is because we want to specifically highlight the automotive domain separately. CPS are different systems that combine the use of computing and hardware in their task. By combining software and hardware, areas like robotics and electrical power grids can be made more efficient and better. Here, we present the most interesting metrics we found under the CPS domain.

3.7.1 Normalized health index

Song *et al.* [28] proposes a resilience metric designed for a ship's cooling water supply system. The system is built up out of valves and pumps, which are represented by nodes in a graph. The metric is normalized to a value from 0 to 1 so as to be easily compared with other metrics. The metric is computed by:

$$r_h = \frac{t_{MTTF}}{t_{Life}} \tag{3.1}$$

The metric measures the Mean Time to Failure (MTTF) and is normalized against the lifetime of the measured device. This can serve as an indication if the component experiences failures due to incorrect implementations, for example. We think this is a useful metric to consider if one is interested in the uptime/lifespan of a component.

3.7.2 Normalized network connectivity

Normalized network connectivity, proposed in [28] measures network connectivity on a ship's cooling water supply system. The value goes from 0 to 1 to be easily compared with other metrics. The metric is computed by:

$$r_c = \frac{\lambda_+(L(G))}{N_n} \tag{3.2}$$

The metric measures the worst-case connectivity scenario in the graph normalized against the maximal connectivity in the graph.

3.7.3 Normalized remaining network utilization

Normalized remaining network utilization metric is presented in [28] and measures the remaining network utilization in the same ship's cooling water supply system. The metric is computed by:

$$r_u = \frac{p_1}{p_2} \tag{3.3}$$

The metric measures the remaining network utilization normalized against the total network capacity. The authors note that even though they measure gallons per minute, the metric itself is general and can be applied in other areas, such as a power system measuring the remaining kW after the current load demands have been met.

3.7.4 Normalized redundancy

Normalized redundancy, proposed in [28], is a metric to measure redundancy in the system. It is normalized to provide a value from 0 to 1. The metric is computed by:

$$r_r = \frac{\sum_i n_b(i)}{n_c + \sum_i n_b(i)} \tag{3.4}$$

The metric measures the sum of the critical devices where n_b is the number of redundant devices for a device i and is normalized against all such devices in addition to the number of critical devices without redundancy (n_c).

3.7.5 Score Security Metric

Aigner *et al.* [29] compile a score metric that measures and combines pre-, post, and run-time security into a score. This is done by calculating each area separately and multiplying it with a system security factor in order to get a score for that area. Each area is calculated using different Boolean rules, which decide the score for the area, which is presented in Table 3.3. If none of the Boolean rules for an area are true, then the digit of that area will be set to 0. The two left columns define what rule and digit the table is handling. The digit-value column is a metric to calculate that area's value, and the rightmost column contains the rules for the area. If none of the rules are followed, the value of the digit will be set to 0.

Table 3.3: The table of basic metric rule set

Rule	**Digit**	**Digit-Value**	**Rule Set**
R_1	X_1	100 * SSF(Pre-Proc.)	CPU > CAP
			IREQ = 0 **AND** ERR > MERR
		0	else
R_2	X_2	10 * SSF(Proc.)	ERR > MERR
			CPU > (RREQ − IREQ)
			CPU ! = VRES + ERR
		0	else
R_3	X_3	1 * SSF(Pre-Proc.)	ORES ! = VRES
			ORES = CPU **AND** ERR > 0
			ORES ! = (RREQ − IREQ − ERR)
		0	else

These rules could be, for example, if more processing power is used than expected / supposed to. While the specific rules could be interesting and are metrics of their

own, we found the way they combined all rules into a score to be more interesting. It is interesting to see that sometimes multiple metrics are used to create one final one, and this is something we will keep in mind when adapting our own metric. many of the smaller Boolean statements presented in this paper were interesting but small enough to not warrant putting into this report.

3.7.6 Attack surface analysis resilience metric

Pan *et al.* [30] evaluate five different methods to evaluate architecture resilience. One of the methods is an attack surface analysis where an activity diagram is used to calculate a nodes probability of surviving/recovering from an attack, which looks like this:

$$resilience = \frac{\sum_{n \epsilon N} prob(n)}{|N|} \tag{3.5}$$

This metric uses the sum of that probability for all nodes and divides it by the size of the activity diagram. The resulting measure would look akin to $\frac{survivability\ percentage}{number\ of\ nodes}$ This is quite a unique way of measuring availability and, therefore, is good to keep in mind for availability metrics.

3.7.7 Static Bayesian network

Static Bayesian Network is presented in [30]. This metric is based on a static Bayesian network method, which uses a component diagram. This metric produces a range from 0 to 1 that defines how resilient the system architecture is. A 0 indicates poor resilience, and a 1 indicates strong resilience. It does so by evaluating different conditions and then summing them all together to output a final score. This metric is quite different in execution than the previous score metric presented in 3.7.5, although there are still some similarities, like taking into account multiple aspects. It is helpful for us to know different implementations of scores if we want to create a similar metric.

3.7.8 Rapidity Minimal Path Method

The minimal path method is a method presented in [30] that contains a couple of metrics, one of which is the Rapidity metric. The Rapidity metric measures rapidity in a minimal path method according to this equation:

$$Rapidity_i = \frac{\sum_{u=1}^{U_i} V_{i,u}}{U_i} \tag{3.6}$$

Rapidity measures the average recovery time through the amount of disruptions. The result from this is later used in order to generate a score for resilience in the minimal path method. This is a very unique metric that measures throughput, and while it is used in combination with other metrics to calculate resilience, we believe that this is an interesting metric for measuring availability time.

3.7.9 Availability Minimal Path Method

The Availability in the minimal path method presented in [30] is calculated with this equation:

$$Availability_i = \frac{\sum_{t=1}^{T} P_m(t))}{T} \tag{3.7}$$

This metric calculates the availability in a minimal path method by taking the amount of normal state time for services and dividing it by total simulation time. In other words, it checks how long services were working as normal. This metric uses a common pattern of measuring availability, where you take the time it is in a state of regular or irregular function and divide it by the total time it was measured for. This can be useful to keep in mind if we decide to create an availability metric.

3.7.10 Markov modeling resilience

A Markov modeling method is presented in the paper [30].

$$Resilience = (-1)^{k+1} R_k \frac{|E|}{|I-M|} \tag{3.8}$$

This metric calculates the resilience with the help of Markov modeling. This means that this metric uses transition matrices to predict when nodes should transition into a normal, success, or failed state. This transition matrix is generated from the activity diagram.

3.7.11 Infrastructure Resilience Analysis Methodology

Nicholas *et al.* [31] measure the resilience of a load frequency control system for an electrical network. They use the Infrastructure Resilience Analysis Methodology (IRAM), which consists of three metrics, each focusing on different parts. Systemic Impact (*SI*) measures of performance (magnitude and duration) loss from an attack, Total Recovery Effort (*TRE*) measures the costs to overcome an attack, and Recovery Dependent Resilience (RDR) which is a combination of *SI* and *TRE*.

$$SI(S_n) = \Sigma_{i=1}^{3} \left[\int_0^{100} ACE_i^2(t, s_n)dt - \int_0^{100} ACE_i^2(t, s_1)dt\right] \tag{3.9}$$

$$TRE(S_n) = \Sigma_{i=1}^{3} \left[\int_0^{100} u_i^2(t, s_n)dt - \int_0^{100} u_i^2(t, s_1)dt\right] \tag{3.10}$$

$$RDR(S_n) = SI(S_n) + TRE(S_n) \tag{3.11}$$

The metrics measure the squared error ACE^2 and the amount of control expended u^2 (both relative to the value in the absence of an attack), which represents the performance and cost of the system for responding to an attack. The result of these is a score, which can then be used when comparing different mechanics against each other.

3.7.12 Resilience factor for high power computing

A resilience factor for high power computing is found in the paper "Towards New Metrics for High-Performance Computing Resilience" by Hukerikar *et al.* [33]. This paper investigates and presents new resilience metrics within the High-Performance computing domain that measure the system's ability to produce correct results during faults.

$$RF_{PE} = \frac{time{-}to{-}solution_{event-free}}{time{-}to{-}solution_{events}} \tag{3.12}$$

This metric takes the time for a solution during a golden run, meaning a run with no faults, and divides it by the time of the same solution but during a run that contains faults. This creates a resilience factor for performance evaluation. A similar metric can be found in the same paper that evaluates variable values instead of time. This metric makes use of a golden run and combines it in the metric formula, which can be used for other metrics that we have found as well.

3.7.13 Anticipate, Withstand & Recover metric

We present three very similar metrics that measure different attributes depending on what input data is used. The base equation for the metric is the same across all three metrics.

Kandaperumal *et al.* [34] propose a resilience metric based on the Anticipate, Withstand, and Recover principle (AWR). It is modeled on an electric distribution system. Each metric is a combination of multiple smaller factors.

3.7.13.1 Anticipate metric

This is the Anticipate metric, and it is calculated by:

$$R_A = \Sigma_{i=1}^{N} T_i W_i^T + \Sigma_{i=1}^{N} P_i W_i^P + \Sigma_{i=1}^{N} R_i W_i^R \tag{3.13}$$

Where T is a list of factors with regards to threats and vulnerabilities with associated weights W^T, P is factors in power delivery and load domain with associated weights W^p and R is the factors for restoration and recovery domain with associated weights W^R. What is noteworthy about this metric is that the metric can be calculated without the need to do a simulation. The final score can be directly calculated from the factors and weights.

3.7.13.2 Withstand metric

The second part of the AWR [34] metrics is the Withstand metric. It is also modeled on an electric distribution system and is a combination of multiple smaller factors. This is the Withstand metric, and it is calculated by:

$$R_i = \Sigma_{j=1}^{N} \rho_{i,j} W_i \tag{3.14}$$

Where $\rho_{i,j}$ is a linear transformation of a pairwise comparison matrix based on the resilience factors: *Critical load not lost*, *Total available generation*, *Critical load demand*, *Topological robustness* and *Threat impact factor*.

3.7.13.3 Recover metric

The last part of the AWR [34] metrics is the Recover metric. It is also modeled on an electric distribution system and is a combination of multiple smaller factors. This is the Recover metric, and it is calculated by:

$$R_i = \Sigma_{j=1}^{N} \rho_{i,j} W_i \tag{3.15}$$

Where $\rho_{i,j}$ is a linear transformation of a pairwise comparison matrix based on the resilience factors: *Critical load restored*, *Path redundancy*, *Generation redundancy*, *Switching operations* and *Switching time*.

3.7.14 Discussion: cyber-physical systems metrics

The presented CPS metrics were the more interesting ones we found. All of the found metrics are presented in Appendix A. We found many different kinds of metrics for different areas within CPS, like software architecture, high-performance computing, and electrical systems. The areas in CPS are very different. Some are closer to an automotive system, and some are further away, but we believe many metrics and concepts from the CPS area could be applied to the automotive area. We noticed similar methods and metrics in the CPS papers we read. A common method was using some sort of prediction analysis in order to simulate a system or a network. With this, graph metrics or graph pathing algorithms were common within the CPS domain. This could be because quite a few systems in CPS could be abstracted to a network of connected nodes. There were a few power grid papers which felt the hardest to apply to automotive systems, this could be due to a lack of electrical knowledge and should be investigated by people with the proper domain knowledge.

3.8 Metrics found in the organisational domain

The organisational domain contained the majority of our found metrics, even with the low number of found papers. This is because the found papers in this domain were libraries containing many different metrics. Many of the metrics are similar, with all of them looking more like a question instead of a mathematical model. This usually means that they do not contain a specific implementation and are sometimes more vague because of it. We will present some of the found metrics from the papers to give an idea on how the metrics look. It is important to state that this domain had the most varied metrics and the presented metrics is a demonstration of how they are formulated rather than the contents of all the metrics. The rest of the organisational metrics can be found in Appendix A.

3.8.1 Example of metrics in the organisational domain

- "Percentage (%) of organizational units that verify the origin of software patches and upgrades before acting upon them, by system risk and IT asset category."[13]

- "Percentage (%) of communication channels controlled by the organization that have been secured via encryption, by information sensitivity."[13]
- "By asset criticality and system risk, average amount of time it takes to transition to a controlled failure mode: Fail safe/fail secure, Fail operational."[13]
- "Percentage (%) of operational time that critical services were unavailable (as seen by users and customers)."[13]
- "Length of time to bring a backup server online."[4]
- "Percentage of hardware components to which tamper-evident technologies have been applied."[4]

3.8.2 Discussion: organisational metrics

The organisational category has the majority of our found metrics. Most metrics in the organisational metrics section follow a similar-looking pattern, i.e.

$$\text{Check the quantity of items that follow rule } X. \tag{3.16}$$

This is the only domain in which we found expert driven metrics. Both previous domains of automotive and CPS only had data driven metrics. However, even with this being the case, a majority of metrics in the organisational domain were still data driven.

4

Metrics Evaluation

This chapter will build on the information we learned from Chapter 3 and present our findings and found patterns.

4.1 Cascading effects

We have noticed that some papers from the literature survey discuss how a small error or attack in a sub-system can have great consequences on the system as a whole. We conclude that it is thus important for a metric to capture the root cause of the problem rather than the resulting consequence that can be cause by such cascading effects. We have seen two different types of metrics, one that measures mechanism resilience and one that measures structural resilience. The literature survey show that these two types of resiliences are disjoint form each other which means that two different metrics are required. Measuring structural resilience can be important in order to figure out how susceptible a system is from a cascading effect or chain reaction of faults. Combining the two different types of metrics could be interesting to assess the criticality and health of a system's components and might be something to investigate in the future.

4.2 Redundancy

Through our literature survey, we saw that redundancy is popular within the cyber-physical domain and we believe that it is most likely a good method for increasing resilience. Redundancy can be implemented without or with diversity. This means that the redundant devices implemented without diversity will be identical. This works great for safety resilience where random faults are the main threats. However, cybersecurity redundant devices that are identical does not majorly improve the resilience, as a smart attacker will be able to break any identical redundant devices if one is vulnerable. In the case of cybersecurity resilience, redundant devices need to implement diversity. These redundant devices will perform the same task in a different way in order to avoid having the same vulnerabilities.

Another issue with redundancy is the resource costs of adding additional similar devices which is not always feasible for many systems. We believe that one such system may be vehicles and that is why we have not seen many papers suggesting redundancy as a resilience mechanism for vehicles.

4.3 Lack of metrics in the automotive domain

The main part of this thesis is finding resilience metrics for the automotive domain, and the search results can confirm that the area is currently lacking research. The only found paper in our survey that mentions resilience metrics in an automotive context was more focused on ML metrics rather than automotive resilience metrics[27]. Other papers were found that discussed automotive resilience but they focused on attacks and resilience mechanism instead of metrics. As these papers did not involve metrics they were discarded during our two step sorting process.

4.4 Difficulties using metrics from different domains

When searching for papers relevant to our field, we found a lot of interesting papers from other fields. A common example of this was papers regarding resiliency in power grids for electricity. Our initial thoughts were that papers relating to electrical power grids could be helpful to draw parallels and be used in automotive resilience. What we found was that these papers usually had similar properties but on a broader level. For example, power grids use electricity as a core measure, while network papers use availability. Despite this, it was difficult to translate power grid resilience into any meaningful way to automotive resilience besides very broad aspects of resilience. This limited the areas of relevant papers more than we initially thought because of the difficulty of relating different resilience areas at times.

4.5 Metrics' time of application

We have found that metrics can be applied at three different times during a system's runtime. These are Before, During, and After a software's runtime. Most resilience metrics that measure a system before its runtime seem to involve some sort of graph model. Usually, these graph model metrics involve analyzing a given graph to determine the resilience of the resulting system in a predictive manner. For example, it could be used to measure the architectural resilience of a system.

There were very few resilience metrics that ran during runtime. Bondavalli *et al.* [35] mention how it is crucial that a metric that measures during runtime should not interfere with the system itself while measuring. This could cause the metric to give skewed results and, as such, not accurately measure the system.

Lastly, the most common of the three types were metrics that are evaluated after the facts when all data have been collected. This can be done by having specific parameters to look out for or an expert look through simulation data logs or audit logs that the system produces. The benefit of post processing is that all data already exists and can be examined. This means that the precise state of the system is known and can be used in the evaluation. Furthermore, post processing can show the trail of the attack as well as all affected components and how severely they are affected.

Since data is only collected during the run-time there are no impact on the live system. However, this means that all the data has to be collected and stored, this might be resource intensive.

4.6 Analysis of data driven and expert driven metrics

A majority of the metrics found were data driven metrics. However, there were some expert driven metrics found in our literature survey but they were exclusively found in the organisational domain. Even in the organisational domain these expert driven metrics were few and we believe that metrics that are expert driven could be difficult to implement as the knowledge level of the expert may differ which in turn could give different results to the same system. There is also the consideration that having an expert on hand is not always an option.

4.7 All encompassing metric

When it comes to resilience, we would ideally want to measure an entire system's resilience. This would require measuring every individual part, such as every mechanism and every physical part. This would be a difficult task, but even when dialed down to only cybersecurity, creating a system-wide resilience metric for cybersecurity is difficult. This is because there are many different mechanisms and parts of a system to measure the resilience of. Not only do proper metrics for every mechanism not currently exist, but all the metrics would need to be added together to form an all-encompassing metric for the entire system.

4.8 Domain patterns

From our survey, we managed to find patterns that correlated to the domain groups, which we will present in this section.

4.8.1 Automotive patterns

Only one paper was identified in the automotive domain. The paper implements a ML IDS for vehicles and evaluates its efficiency with common ML metrics like accuracy. Thus, the metrics provided in [27] were more relevant to ML than to cybersecurity. Considering the low number of papers we found in the automotive domain and the metrics used in the paper, it is difficult to recognize any patterns from our findings within the automotive domain. Another observation we noticed is a lack of papers that use resilience metrics specifically in the automotive domain, which indicates that this is an area that requires further research.

4.8.2 Cyber-physical systems patterns

We found that the CPS papers contained the most specialized metrics. All metrics found in the CPS domain were precise and clear. No metrics were vague or loosely defined such as they are in the organisational domain.

From the papers in the CPS domain, we noticed a majority focused on resilience measured in graphs. This includes networks, power grids, and architectural resilience. Usually, this is some sort of metric to evaluate how the structure or design of the target is evaluated. It is common that resilience metrics that measure these kinds of things are evaluated before running the program/system itself. We found that these metrics were usually made with one intended use case in mind and usually contained mathematical models. This category had a wide mix of metrics that could potentially be seen in an automotive context, but some that feel more or less incompatible with automotive.

We noticed that these appeared to be harder to transfer between domains because of their use of domain-specific properties. Transferring a specific metric to another domain often involves changing or modifying some parameters to fit the new domain, depending on how different the domains are, the changes can be small or substantial. There are papers that have such specific points of interest that it is almost impossible to transfer them to another domain. It could be because the metrics rely on attributes that are specific to the domain in which they are being used.

4.8.3 Organisational patterns

The organisational domain contained the majority of our found metrics, although many of them were deceptively similar. Our study of the organisational domain papers revealed that the presented metrics are not formulated with a mathematical model. They were rather formulated similarly to questions where the result of the question is the resulting measurement of the metric. These are designed to be generic so that they can be applied in many different circumstances. This allows the organisational metrics to be flexible and could apply to many different areas.

Organisational metrics consider the company level aspect more than the metrics from the other domains. This means that some of these metrics are not clearly convertible to a system level, which makes some metrics difficult to apply to the automotive domain. These organisational metrics serve more as suggestions or inspiration, and that more precise metrics have been tailor created for that scenario.

This, however, means that there are not many implementation details for a majority of the found metrics. Problems may arise when it is unclear how to measure the necessary area in order to implement one of these metrics. If implemented improperly, there might be lost data skewing the result, which is counterproductive and goes against the reason the metric exists in the first place. If a metric does not have implementation suggestions it is left up to the individual engineers to decide, then there is room for errors and subjectivity may influence the resulting measurement.

4.9 Non-Domain Patterns

Like the domain patterns, we managed to find correlations and similarities in the attributes of the papers that were not strictly correlated to the domain of the paper. These attributes were the papers' implementation method and cybersecurity focus.

4.9.1 Implementation patterns

What is evident from Table 3.1 is that no framework paper was found in the CPS/Automotive domain, while all the papers in the organisational domain were of type framework. In contrast, the CPS/Automotive domain lacked framework-type metrics, and most of the metrics were applied in a practical manner. One hypothesis is that theoretical metrics are potentially easier to adopt in different use cases. This is because theoretical metrics tend to be more general and abstract in their construction. However, due to the small sample size of theoretical papers, no significant conclusion about the theoretical implementation types can be made.

We have seen that there are some patterns that relate to both implementation style and domain. In particular, we did not manage to find any framework paper in the CPS/Automotive domain while all the papers in the organisational level were some kind of framework. In contrast, many papers in the CPS/Automotive domain usually tested their metric in a practical way which was lacking in the organisational/higher level metrics that we found.

4.9.2 Dependability attributes patterns

We assigned dependability attributes to every paper we read that we felt that we could. It was difficult to give some papers these attributes as some had no obvious connection to cybersecurity. For the papers that had cybersecurity attributes, we noticed that many of them had availability as one of their attributes. As such, we believe availability is an important part of resilience and it is not surprising that many of the found metrics would focus or include availability. Availability is also very wide of a metric group and covers important aspects of resilience which makes it important for measuring resilience. However, because of the prevalence of availability other metric might potentially not get enough attention. This could be interesting to investigate in the future.

4.10 Concluding remarks of the survey

After a concluded survey we have found multiple patterns in current resilience metric research. We have also understood that creating a proper metric is difficult and requires to account for multiple things. Many metrics were found during the survey, although not many for the automotive domain. Creating an entirely new metric would be too great of an undertaking. As such we have decided that we will attempt to adapt a metric that we found in the literature survey. This way, we can investigate

how difficult it is in practice as well as properly evaluate the metric to see if it is performing as expected.

5

Experimental results

This chapter describes how we performed our experiments to test the chosen metric. The chapter starts with a description and motivation of the chosen metric to implement. Then, the tools used in the experiments are presented, followed by a description of the experiment limitations we had to make. Afterward, the experimental setup and process of performing the experiments are explained. Lastly, the result of the experiment are presented and discussed.

As mentioned in Section 1.3, the aim of this thesis is to quantify the effectiveness of resilience mechanisms. To that end, we have conducted a literature study and analyzed current resilience metrics. But to further expand current research regarding resilience in the automotive domain, we have conducted experiments to understand what possible shortcomings might exist when adapting existing resilience metrics from other domains. The experiments were performed through a simulation using the fault injection tool MODIFI [12] and Simulink[1] to model a system and test the metric. We will investigate how our metric measures a system in a normal state and under the effects of a handling mechanism.

5.1 The metric to investigate

The chosen metric for investigation was chosen based on the results of the literature study in Chapter 3. The results from the literature study indicate that most of the papers have a metric that measures some form of availability, as discussed in Section 4.9.2. It can thus be concluded that availability is an important part of resilience. Because of this, we felt it important to study these kinds of metrics, and as such, the metric has, to some extent, measure availability.

There are some qualities we would like to see in the availability metric we choose. It would be interesting if the metric came from another domain than the automotive domain, as it would give us insight into the difficulties of adapting metrics from different domains. It is also essential that we choose a metric that can be used inside of a simulation, as we do not have access to any physical test tools.

We also have to choose some mechanisms to be used when testing the metric. After reviewing existing resilience mechanisms three main types where identified. The types are *prevention*, *detection* and *handling*. Out of these, we have decided on

[1] https://se.mathworks.com/products/simulink.html

handling as it is very interesting, and we feel that prevention and detection are more developed compared to handling.

After the discussion and the overview of the found metrics presented in Chapter 3, we created a mind map (see Appendix B) over different handling mechanisms that are connected to different metrics. We created the mind map from the mechanisms presented in [7]. The handling mechanisms *Host Configuration Management and Automated Software Management*, *Intrusion Detection Systems*, *Operating Systems* and *Virus/Malicious code Detection Systems* were chosen based on[36] as the backbone. In this mind map, we linked the mechanisms to different possible metrics. This helped us find relevant metrics and mechanisms combinations from the metrics we found in our literature survey. When deciding on which found availability metric we should try and implement, we referenced the mind map (see Appendix B) for metrics.

By referencing the mind map the most suitable metric was chosen for our experiment, this was the health index metric. The health index metric is defined as: $r_h = \frac{t_{MTTF}}{t_{Life}}$ [28], where t_{MTTF} is the Mean Time To Failure and t_{Life} is the expected lifetime of the component (see Appendix 3.7 for a more detailed explanation), as it checks the requirements described above. It is also simple, which makes it easier to adapt to another domain. We also believe that the health index metric has the most unique way of measuring availability, which we think is interesting to investigate further. Furthermore, the health index metric is interesting to study as it is a safety-oriented metric rather than a cybersecurity resilience metric. As such, investigating it will give insight into whether it is possible to have the same metric measure both parts of the resilience dichotomy.

This metric comes with some inherent definition difficulties that need to be sorted out before we use it in our experiment. When it comes to *MTTF*, we need to make sure it works in the context of our experiment. As per our experiment setup, nodes that have been attacked once will always be deactivated. This means that *MTTF* will essentially measure the time it takes for a node to be attacked since even if further attacks were to happen, they would not affect the node/the system any further.

There does not exist any accurate expected lifetime for the components in the models, and since the components in our simulation will not randomly encounter faults (as per Section 5.2), having a component's expected life to be less than simulation time is not logical. Having a longer time to live than the simulation time will only result in the resulting measure being stretched. This results in that the component's time to live will be the same as the simulation time, in this case, only one second. This way, we can make sure that *MTTF* and Time are on a similar scale (as we cannot create more attacks outside of the simulation time.)

Overall, the metric will measure the expected time a node is healthy. If an attack were to happen early, we would get a low health index score (minimum of 0), which indicates poor health. If an attack were never to happen, the result would become a 1, which indicates a fully healthy node during the simulation period.

5.2 Tools and software

MODIFI [12] is the tool chosen to perform the attack injections into our simulation. These attacks are treated as premeditated and intentional attacks on the system by a malicious entity, that we call a smart attacker. As such, no faults are injected or occur in the simulations. We will use MODIFI's ability to collect data to view when attacks are happening as well as the effect of the attack. MODIFI uses a "golden run" to calculate whether an attack was successful or not. The golden run is created in the beginning by simulating the model once without any faults or attacks. The values are then recorded and a simulation is conducted. A simulation requires injection points to be set. These are points in the model where an attack can be injected. Injection points may be specifically selected, or they can be set to inject attacks randomly. For our experiment, a simulation starts at time step 0 and ends at time step 1, approximately 1 second of run time. An attack can occur anytime between time step 0 and 1 in 0,02 time step intervals. Multiple attacks can occur in the same time step.

The model is an abstraction of a car's infotainment system. The reason that we are investigating a car's infotainment system is that it has been shown that it can be a target for attacks [1]. Thus, each node in our model represents a service the infotainment system provides, such as radio, back camera, or parking assistance. Due to limitations (see Section 5.3), each service will be abstracted to a node that outputs a constant value simulating the service's actual output. This output will be what is being attacked, and changes to the value will indicate if a node is attacked. Three different states were made to indicate whether a node is available or not. These are: *Available*, *Malicious*, and *Isolated*. A node can only be in one state at a time. Nodes that do not respond with the expected value are counted as unavailable. This means that *Malicious* and *Isolated* nodes are unavailable. Nodes that are *Available* (i.e., function correctly) are defined as $N = ev$, where N is the output value of the node, and ev is the expected value received from the golden run. A node is available as long as it can still fulfill its purpose, i.e., outputs the ev. A node is *isolated* when $N = 0$, indicating that it is not available. If the outputted value N deviates from the set constant and the node is not isolated (i.e., $N \neq ev \wedge N \neq 0$), we consider the node to be *Malicious* and hijacked by an attacker. A node becomes malicious as a result of an attacker sabotaging/hijacking the node and isolated as a result of a handling mechanism.

5.3 Experiment limitations

In the interest of time, we performed an experiment on an abstracted system rather than a real life counterpart. Yet we believe that even this limited experiment can reveal trends in how resilience in other domains can be translated to the automotive domain. As such, we only tested one metric from our literature survey. However, as it turns out, the chosen metric is not clearly defined in [28]. It is not stated how the $MTTF$ is calculated and on what terms, neither is the t_{Life} term. Thus, we have to assume its intended functionality and try to adapt it to have the same functionality

in our context. This means that we can not test the exact implementation of the metric but rather our interpretation of it. However, we still believe that this will show how well metrics can adapt to different domains.

Another limitation is the abstraction and simplification of the models used in the experiment. The implemented models are a simplification of how a car's infotainment system works. We do not have the necessary tools nor time to test this metric on a model that is closer to life, and therefore, we decided to make simpler models.

As already discussed in Section 5.1, we only focus on metrics that measure the performance of the handling mechanism, as we have noticed a lack of said metrics. As such, the detection mechanisms are abstracted away and will always catch an attack. This means that every attack that is injected will be detected instantly. However, this is something that is worth investigating further and will be discussed in more detail in Section 5.6.6.

Another limitation in our experiments is due to the fault and attack injection tool we use. MODIFI is used for the fault and attack injections. However, it unfortunately can not insert attack into Matlab function blocks. Only the in and output points may be targeted. This means that models using these have a limited attack surface.

5.4 Experiment method

When performing the experiment, a scientific method was followed to make sure the different models have the same conditions. As previously mentioned, the MODIFI tool is used to perform the experiments. MODIFI uses models created in Simulink[2] to perform the injections. It is via MODIFI that it is possible to know where, when, and how an attack may occur.

We created three models in Simulink with different handling mechanisms. The model represents a car's infotainment system. The three different models are:

- Base model
- Isolation model
- Triple-Modular Redundancy (TMR) model

The base model is a control and has no handling mechanism implemented. The isolation model isolates a node that is deemed malicious. This model was created in order to see if the metric could accurately measure the effects of a mechanism that only handles the attack without any concern for availability. Lastly, the TMR model implements a Triple-Modular redundancy for each node. The TMR model tries to preserve availability while handling attacks. This allows us to observe how the metric behaves with a mechanism that handles attacks with regards to availability.

These models were injected with 10 different attack compositions and then evaluated with the health index metric. Each attack composition is simulated 1000 times. This is to make sure the data we capture is consistent as the attack time (and in some

[2]https://se.mathworks.com/products/simulink.html

attack compositions also the attack points) are randomly chosen. The data from the simulations are also used to create a confidence interval of our result.

The attack model used is MODIFI's "Stay at value" attack which causes a node to change and stay at a certain value. Since attacked nodes stay attacked for the remainder of the simulation period the node will stay at the attacked value for the rest of the simulation period.

In Simulink and MODIFI it is possible to pick the simulation time and sample time for the model. Some outcomes may be obscured depending on how often and for how long the model is allowed to run. Therefore we should be careful when deciding these values. We ran each model for one second of simulated time, with a sample time of 0.02, and 50 time steps.

5.4.1 The experiments attack compositions

Ten different attack compositions are performed on each model. The timing of when a certain attack occurs is a random point during the entire simulation period. The randomness in the random experiments refers to picking the attack point to be attacked while the fixed experiments guarantee which nodes will get attacked. We ran the following experiments

- *1 - 3 Random*: these tests involved a fault injection on any attack surface. For the 1 Random experiment, only one attack was injected per run. In the 2 Random, two attacks were injected. Similarly, in 3 Random, three attacks were injected.
- *1 Fixed*: injected one fault into specifically one node. This means that in every simulation of this experiment, exactly one node is attacked. Furthermore, the handling mechanism is never attacked, only the nodes.
- *2 Fixed*: is similar to the previous test, but two distinct nodes are always attacked. The combinations on this one are node 1 & 2, 1 & 3 and 2 & 3
- *3 Fixed*: is the experiment where all three nodes are attacked.

These experiments reveal how the metric behaves in different situations and models. This allows us to evaluate the health index metric performance and find out what changes are needed in order to make it a proper metric for stakeholders and engineers to use. The models we use are simple enough so that their expected behavior can be hypothesized and then checked against the results. The consequences of a change in a simple model are easier to spot, unlike a more complex model where a small change can have larger cascading consequences. This makes it easier to notice if the models are implemented poorly and behave incorrectly. Even though the models are simple, they allow us to notice nuances in how well the metric measures a system.

Another decision for the attack composition is to have the malicious nodes output the same value. This is to simulate smart attackers. Smart attackers can control the TMR and influence the outcome to an arbitrary value if they can compromise two of the nodes, and as such, we can simulate more purposeful attackers. A similar

example can be seen in the isolation mechanism model. MODIFI can inject an attack at both the node and the node target to alter the behavior of the isolation mechanism. Attacks on these nodes may happen one at a time or simultaneously. By injecting simultaneously on the node and node target value, an attacker can trick the isolation mechanism into thinking the value is correct when, in fact, the value has been altered.

Lastly, in our experiment the detection mechanism is perfect. This means that once an attack is injected it is immediately known to the system although how the system handled it depended on the model.

5.4.2 Base model

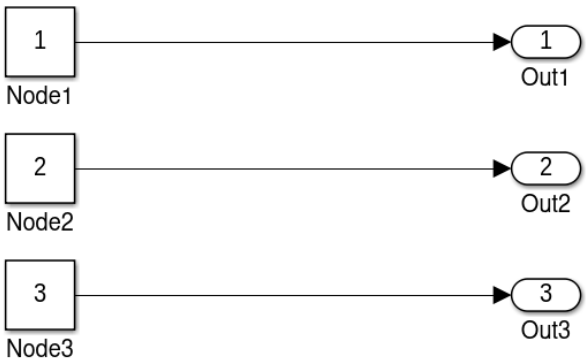


Figure 5.1: Base model in Simulink

The base model (Figure 5.1) is the simplest of them all. It serves as a control that only contains three nodes and has no handling mechanisms, which means that it is not able to mitigate any attacks it receives. The purpose of this model is to see how well the health index metric can display the effectiveness of different handling mechanisms compared to a system without any mechanisms implemented. Each node produces an integer value of 1 which shows that the system is working as intended. If the node gets attacked the value will change to whatever value the attack puts it to.

5.4.3 Isolation model

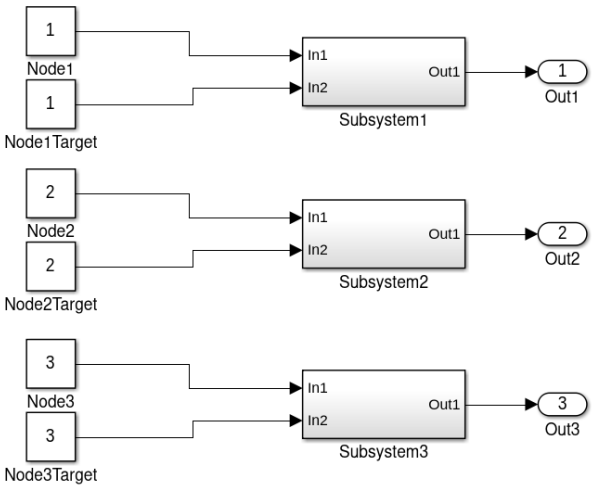


Figure 5.2: Isolation model in Simulink

The isolation model (Figure 5.2) follows a similar structure to the base model but with an added subsystem after the node. The subsystem will detect if a node is behaving abnormally (does not produce the expected value determined by the "Node Target") and isolate them for the rest of the simulation run. This results in the node in question never being able to perform its task but also protects the system from any malicious or erroneous information that a malicious node could potentially send. In order to apply this to our simulation the model will change the output of the node to 0 which indicates that it is isolated.

5.4.4 TMR model

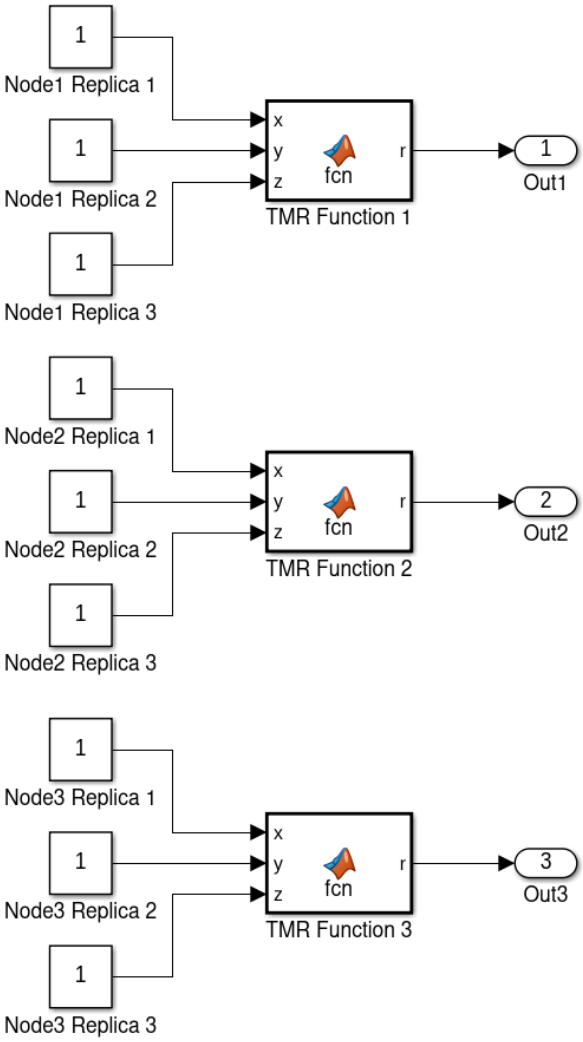


Figure 5.3: TMR model in Simulink

The Triple-Module Redundancy model (Figure 5.3) makes use of redundancy to increase resilience. Instead of only having one node perform the action, there are two additional nodes performing the same action. This is because if one node is attacked/fails, the other two still outputs the correct value, and by voting, the majority result is then passed on. This means that if only one of the nodes gets attacked, then the TMR will still output the correct answer. If the TMR cannot decide a majority, then the TMR will output 0. But depending on system specification, it may be acceptable to randomly pick one of the inputs and chance it. However, we assume a smart attacker and as such, the attacker would rather take control of the TMR than disable it.

5.5 Results

The results from our experiments are presented here. As a reminder, the score is the Health Index (HI) or, in our case, the mean time to a successful attack. A score of 1 means the node in question never had its expected output altered, while a lower score indicates (in percentage) the time from the start of the simulation when it lost its function. We begin by presenting the base model, afterward the isolation model, and lastly, the TMR model.

5.5.1 Base model

Table 5.1: Results of experiments with no mechanism

Attacks	Node 1 HI	Node 2 HI	Node 3 HI	System HI
1 Random	0.82562	0.84924	0.83476	0.83654
2 Random	0.69498	0.71962	0.70376	0.70612
3 Random	0.60758	0.62324	0.59266	0.60783
1 Fixed (Node 1)	0.49882	1.0	1.0	0.83294
1 Fixed (Node 2)	1.0	0.5145	1.0	0.83817
1 Fixed (Node 3)	1.0	1.0	0.4885	0.8295
2 Fixed (Node 1 & 2)	0.51096	0.52064	1.0	0.6772
2 Fixed (Node 1 & 3)	0.52382	1.0	0.5031	0.67564
2 Fixed (Node 2 & 3)	1.0	0.49622	0.50364	0.66662
3 Fixed	0.59098	0.59866	0.59616	0.59527

The results from the base model is presented in Table 5.1. The HI for all nodes lowers with the increase in the number of attacks for random surfaces. When it comes to the fixed experiment cases, while the nodes HI might differ, the overall system HI is consistent across all 1 fixed attacks and similar for 2 fixed. Every node in 3 fixed has a very different HI compared to its 1 fixed counterpart, which is unexpected as a 3 fixed attack could be seen as three 1 fixed attacks on different nodes.

5.5.2 Isolation model

Table 5.2: Results of experiment with isolation mechanism

Attacks	Node 1 HI	Node 2 HI	Node 3 HI	System HI
1 Random	0.86168	0.84912	0.85352	0.85477
2 Random	0.73188	0.73214	0.72408	0.72937
3 Random	0.64172	0.64492	0.61788	0.63484
1 Fixed (Node 1)	0.4986	1.0	1.0	0.83287
1 Fixed (Node 2)	1.0	0.515	1.0	0.83833
1 Fixed (Node 3)	1.0	1.0	0.49798	0.83266
2 Fixed (Node 1 & 2)	0.50842	0.47976	1.0	0.662727
2 Fixed (Node 1 & 3)	0.5051	1.0	0.49968	0.66826
2 Fixed (Node 2 & 3)	1.0	0.5159	0.5005	0.67213
3 Fixed	0.49976	0.50676	0.50274	0.50309

The results from the isolation model is presented in Table 5.2. The isolation results are very similar to the base model. This means that the previous segment also applies to this model.

5.5.3 TMR model

Table 5.3: Results of random experiment with TMR mechanism

Attacks	Node 1 HI	Node 2 HI	Node 3 HI	System HI
1 Random	0.9793	0.97874	0.9777	0.97858
2 Random	0.93516	0.94166	0.94718	0.94133
3 Random	0.88676	0.87924	0.88394	0.88331

Table 5.4: Results of fixed experiment with TMR mechanism

Attacks	System HI
1 Fixed (Node 1 Replica 1)	1.0
1 Fixed (Node 1 Replica 2)	1.0
1 Fixed (Node 1 Replica 3)	1.0
2 Fixed (Node 1 Replica 1 & 2)	0.65898
2 Fixed (Node 1 Replica 1 & 3)	0.6795
2 Fixed (Node 1 Replica 2 & 3)	0.66612
3 Fixed (Node 1 Replica 1, 2 & 3)	0.5044

The results from the TMR model is presented in Table 5.3 and Table 5.4. The results are divided into two different tables. The first, Table 5.3 presents the results from the $1-3$ random experiments with the model depicted in Figure 5.3. However, because of the increase of nodes with the redundancy in TMR, we elected to perform the fixed experiments on only one of the three nodes in Figure 5.3. We can, however, show that this is enough to cover all the possibilities as the three nodes are not inter-connected.

The table for TMR looks a little different than the previously presented tables. This is because TMR can mask one malicious node from the input of redundant nodes. All 1 fixed attack output 1, which is to be expected as a TMR with only one faulty node will give the correct result every time. At two or more attacks, the table for the TMR model starts looking similar to the previous two tables, although with noticeably higher values. It is important to note that for 2 or more attacks, the systems HI will be decided when the second attack hits a different node as that is when it will begin outputting an incorrect value.

5.6 Discussion

The results of the experiments are discussed in this section.

As can see in the result tables, the higher number of attacks a model received the lower the health index metric becomes. This follows our hypothesis and makes sense as a system, no matter how resilient, can not withstand an infinite number

of attacks. There is a limit to the severity and quantity of attacks any system can manage although this limit increases with the help of resilience mechanisms. This gives us confidence that the models we set up are working correctly as we can see an decrease in HI every time there is an increase in attacks.

5.6.1 TMR outperforms other mechanisms

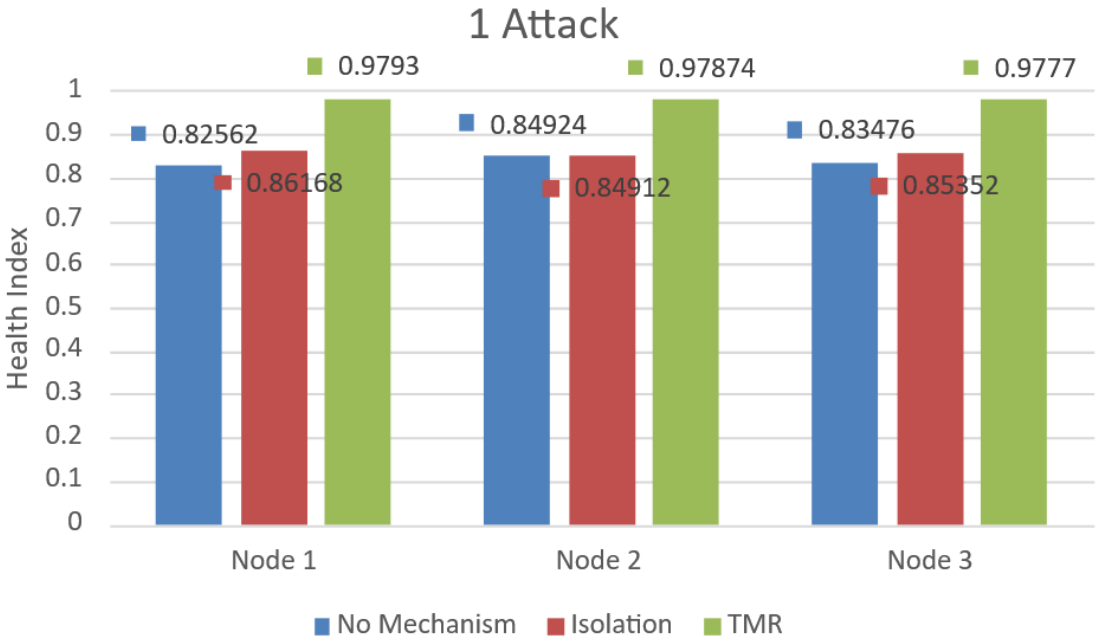


Figure 5.4: Results of 1 attack for the mechanisms

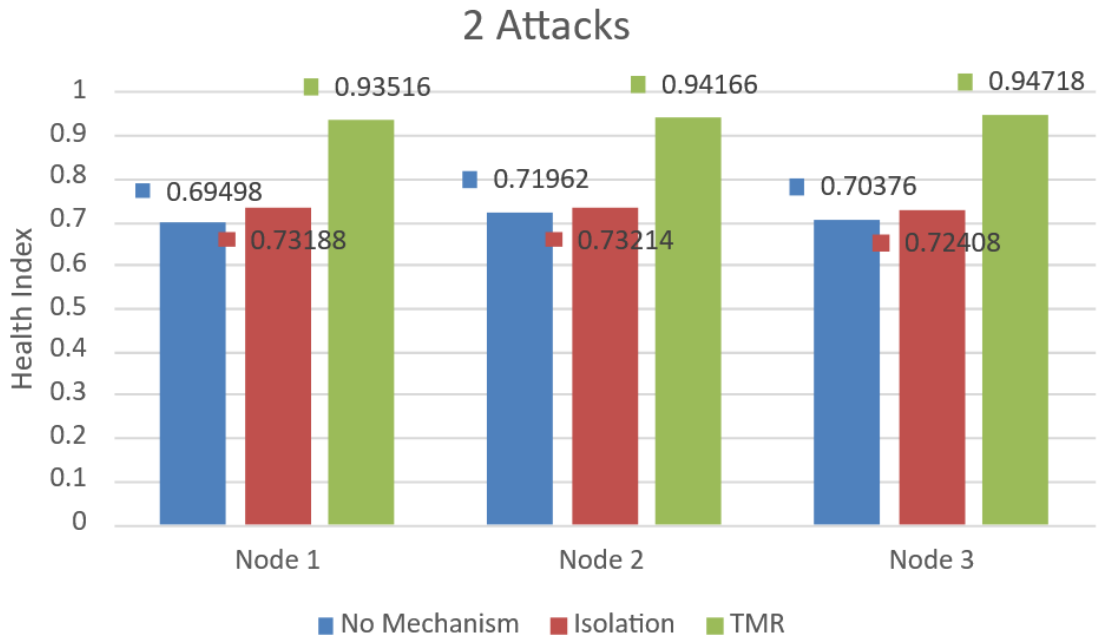


Figure 5.5: Results of 2 attacks for the mechanisms

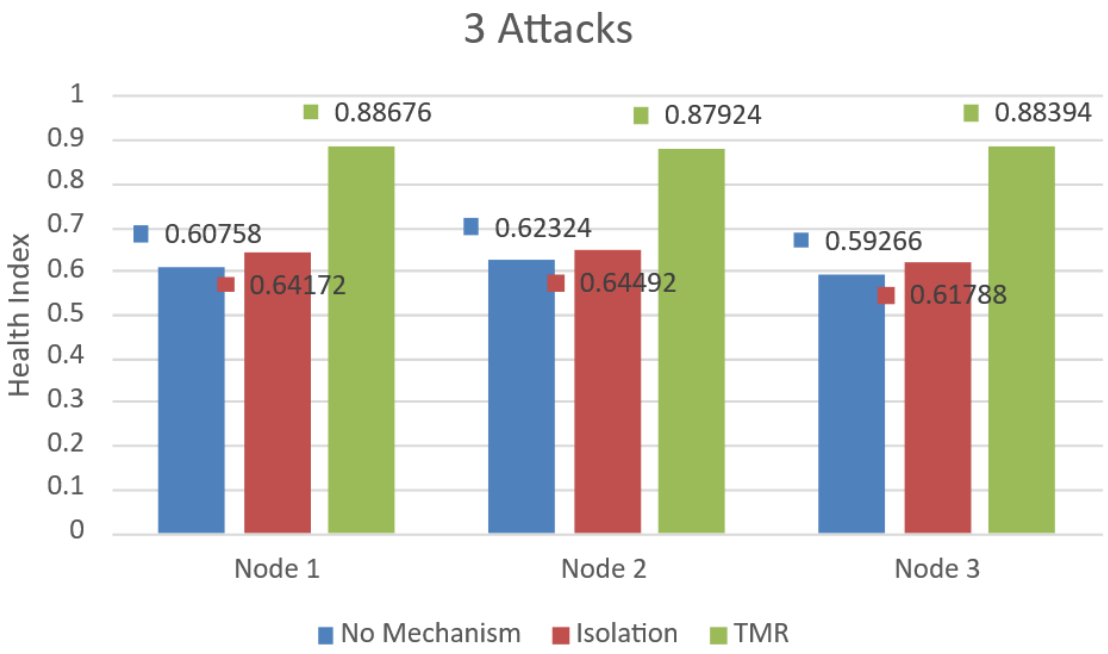


Figure 5.6: Results of 3 attacks for the mechanisms

The results from the $1-3$ random experiments (Fig 5.4, 5.5 and 5.6) show a clear dominance in the performance of the TMR mechanism. This is expected as the health index metric mainly measures the availability of the system. Since only the TMR mechanism conserves availability when acting, it is expected to outperform the other implementations.

It can also be seen that the isolation mechanism slightly outperforms the base model. This is believed to be due to the fact that the isolation mechanism has more attack points than the base model. Injected attacks in parts of the subsystem that are responsible for handling does not affect the output if the mechanism receives the expected values. Thus, even though an attack occurred on the mechanism, it was not propagated.

Another interesting note is that the isolation mechanism just barely outperforms the base model according to the health index metric. It can be argued that it is as secure as the TMR implementation since any malicious node is deactivated when detected. However, as the health index primarily focuses on the availability of the system, this robustness is not captured. This shows that even though the system may be strengthened against attacks, the health index metric does not capture this aspect of the isolation mechanism. This "blindness" calls for either a wider metric that can capture multiple aspects of the system or a compound metric that summarizes multiple metrics.

5.6.2 TMR performance on one attack

The TMR model has excellent performance when there is only one attack present. The reason TMR is not fully resilient in the one attack random experiment is because

there is a chance that the attack is injected after the TMR mechanism. However, when it's just one fixed attack, it will always produce a HI of 1 or, in other words, never fail. That is the highest HI measured with the health index metric and clearly achieves better scores than the other mechanism.

The measured HI for TMR only slightly decreases for each new attack, whereas there is a larger decrease in the other mechanisms' performance. We can see that the quantity of attacks decreases the HI as expected, although there is no substantial decrease when increasing the number of attacks as seen in the other two implementations. This is because TMR triples the number of nodes and, as such, also the number of possible attack points. This creates a disproportionate amount of possible attack points compared to the number of attacks, and as such, skews the distribution of attacks that are required to fool the TMR mechanism. By increasing the number of nodes that can be attacked, the likelihood of a certain node being attacked is lowered. This added with the fact that the TMR mechanism requires two nodes to be attacked before the mechanism fails, minimizes the amount of successful attacks. However, this method does have its drawbacks as well. Implementing redundancy requires more resources and increases complexity. Certain systems may not allow for resource-intensive computation or higher complexities, which makes this method not suitable for all cases.

5.6.3 Choice of simulation time

Incorrect choices of simulation time, time steps and sample time could lead to obscured and incorrect results. As such, multiple configurations of simulation time, time steps and sample time where tested on the models. We were looking for a change in the measured HI score when increasing or decreasing each variable. The measured results from these runs showed no impact on the measured HI value when increasing or decreasing the simulation time, time steps or sample time. All measured HI scores were inside the established confidence interval.

5.6.4 Confidence of results

For every experiment carried out a confidence interval is calculated. Throughout all the simulations the results was constant and no major outliers were found. The confidence interval is calculated using a normal distribution with a 95% confidence. The 95% confidence interval reveals that the HI values have a confidence interval of $mean \pm 0.0217$ at the most. This makes us confident that the results are accurate.

5.6.5 Taking the Health Index metric one step further

We have noticed that the health index can not differentiate between safely shutting down a node via the isolation mechanism and a node that is unavailable because it has turned malicious. By only measuring the availability of a node, we cannot measure if the system is hijacked or the extent of the system's compromise. This means that the isolation mechanism and the base model might look similar because, from the perspective of availability, both of them will be similarly available. It does

not take into consideration that the isolation model will contain any threat to the system by isolating the node, while the base model will let the hijacked node still act in the system.

A method to combat this is by adding another data vector since measuring availability is shown to not be enough to determine the difference between an isolated and malicious node. We also propose measuring each node's produced performance. We define produced performance as a positive or negative value representing the perceived impact on the system. When a node has a positive produced performance, it means that the node is working as intended and impacting the system positively. If this produced performance is 0, then the node is disabled, dead, or isolated. It will indicate that the node is not impacting the system in any way. Finally, if the produced performance is negative it means that the node is hurting the system. This would be the case if a node gets hijacked or corrupted and has the capability of affecting the system.

If every node produces a value like the proposed produced performance, it can be used in tandem with the health index metric to figure out how much unavailability is malicious and how much is non-malicious. This could be implemented in a simulation, although we believe it may still be difficult to differentiate properly between isolated and malicious nodes in a real system since it may be hard to detect if a node is malicious or just unavailable. This means that a measure such as the proposed performance will not be able to solve the problem of marking what nodes are malicious. If, however, malicious and isolated nodes are already differentiated, produced performance can help a user see how much of the system becomes damaged or hijacked compared to safely shutting down.

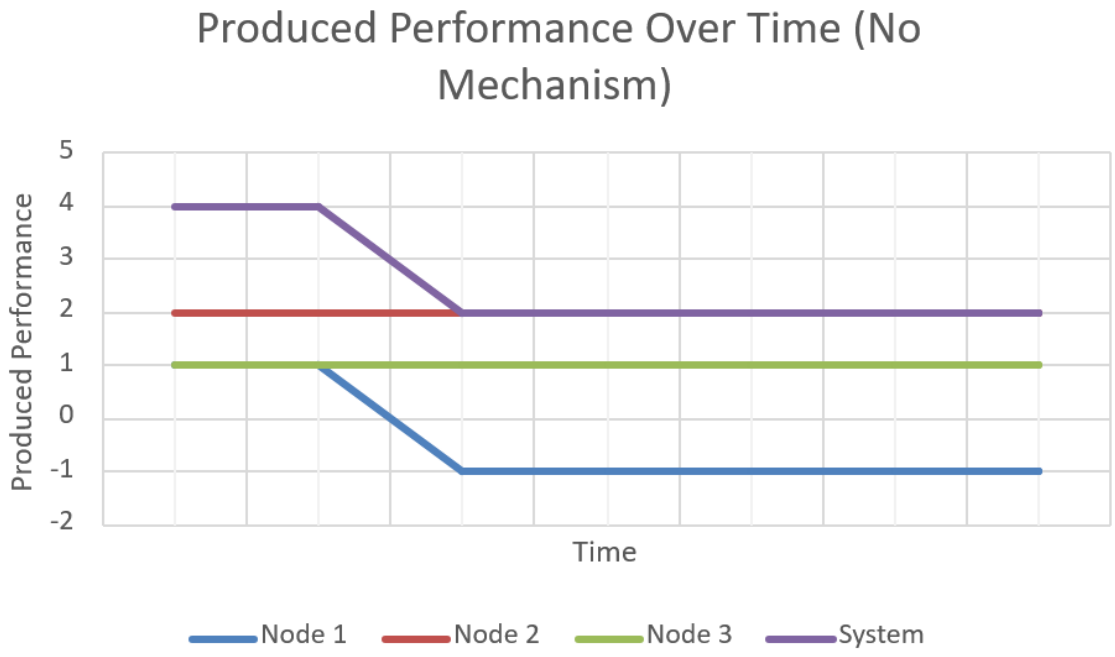


Figure 5.7: Produced performance plot of the no mechanism model

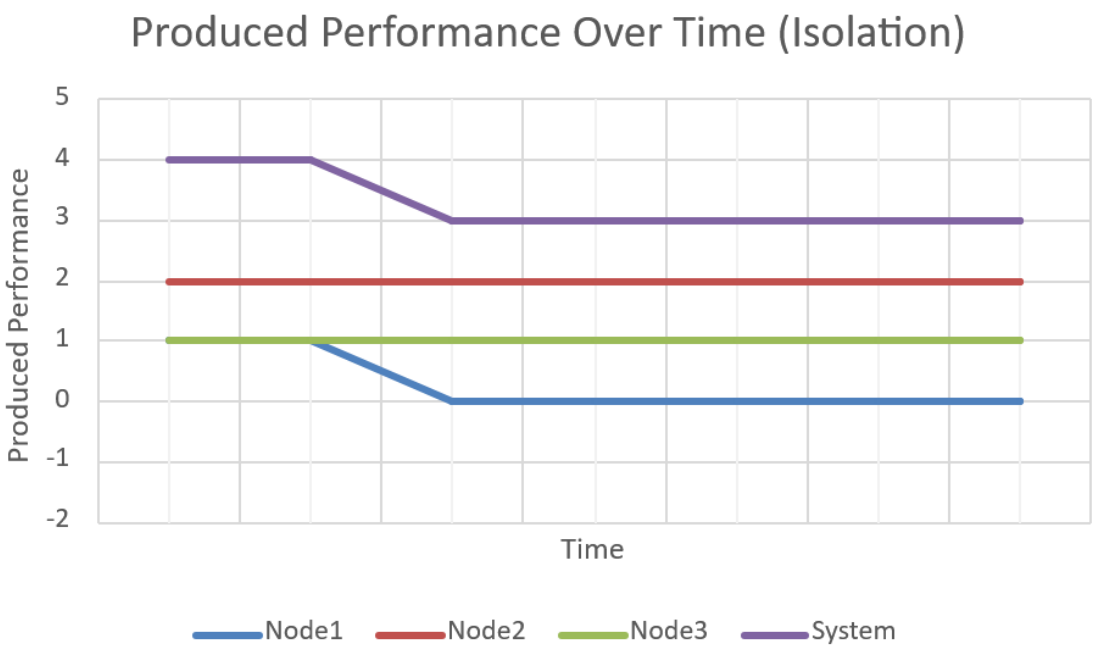


Figure 5.8: Produced performance plot of the isolation model

Figure 5.7 shows a run of the no mechanism model but measured with produced performance. Here the system impact is shown and the produced performance is evaluated to 2 where as in Figure 5.8 the produced performance is 3. On its own the HI does not reveal the difference between these runs but as show the produced performance highlights an impact from the isolation mechanism.

5.6.6 Perfect detection mechanism in simulation

As previously mentioned, the detection mechanism is perfect in our experiment. This means that once an attack is injected it is immediately known to the system although how the system handled it depended on the model. However, there are some aspects that might not be fully realized in our experiment because of this. One such case is that we cannot properly measure the time from injection to detection. This may be interesting to examine, especially for larger models as an attacks propagation time can differ depending on the model.

There is also the time from an attack injection to measured failure, meaning the time it takes the attack to propagate through the system before it actually affects the system. In our experiment, the attack will always propagate through the system after one time step. That is because we have a very simple model but a more complex model may experience alternate behaviours or delayed symptoms. This means that we are not properly testing against attacks that are hard to detect or stay silent yet still affect the system, such as man in the middle or a worm attack. There is also the fact that some attacks take a while before any noticeable effect is noticed on the system. These attacks would result in a higher HI value as the attack would leave the node available for longer.

5.6.7 Absence of recovery mechanism in simulation

It was decided not to implement any recovery mechanisms in the experiment, meaning that if a node is attacked early on, it is gone for the rest of the simulation. While this choice was intentional, it is hard to say if it favored some mechanisms more than others. There is no doubt that the availability of the models would increase if a recovery mechanism is implemented, but the question is if it disproportionately helps some models compared to others. This is something to keep in mind and is an interesting addition for future work.

5.6.8 The Health Index metric misses vital attributes of mechanisms

When comparing the three models, the TMR outperforms the other two models. While it may seem obvious that it should outperform the base model, it is also noteworthy that the isolation model did not outperform the base model significantly. This could be because even though isolation is a good handling mechanism to prevent an attacker further access to the system, it also does not inherently increase availability. This attribute of the isolation mechanism is unfortunately lost in the health index metric. TMR will increase availability as even if one of its redundancies is attacked, the node continues to function correctly. The difference, however, is when two replicas get attacked, the node will be hijacked. This allows the attacker to send data freely without the handling mechanisms' intervention, which could be very dangerous to the system. As we previously mentioned, having a measure like the proposed produced performance could help alleviate this discrepancy and assist the availability metric so that it can give a clearer picture of availability.

5.6.9 Difficulties applying the Health Index metric

We implemented the health index metric for the experiments. We decided to choose a metric that is in a similar domain to automotive, and that is not complex. This made it easier to modify it to suit our domain and use case, and even then, it still required some work to get the metric properly modified for our case. The adaptability of metrics depends heavily on the metric and domain it originates from. While there is merit in looking for similar metrics to adapt to the automotive domain for more complex metrics, this could be difficult to achieve.

After some modification, the metric satisfied our prerequisites, and by looking at the results, we can see that it performed favorably in measuring availability. While the metric does successfully measure availability, it does not work as a sole measure of resilience. This is partially because availability is just one part of resilience, and in the paper for the health index metric[28], it was used as a part of a compound metric. This makes us believe that the health index metric was never meant to measure the resilience of an entire system. However, the metric still measures the availability of a system, and as discussed earlier, availability is an essential part of resilience. This makes it suitable to be used in conjunction with other metrics in order to get a better resilience measure of a system.

5.6.10 Impact of experiment definitions

Depending on how we choose to categorize a malicious node as either available or unavailable affects the result. Both of these scenarios reveal some inherent issues when measuring the resilience of a system with this metric alone. If we instead define malicious nodes as available we can get the scenario where the metric outputs a worse result for a system that handles malicious nodes by isolating them compared to allowing the attacks through. If we define malicious nodes as unavailable a system that has a malicious unmitigated node and a malicious mitigated node will produce the same result. This does not give a proper indication of the health of the systems as a system that is compromised is in a worse state than one that has an unavailable node.

5.6.11 Experiment reflection

The experiment went well and produced a somewhat expected result. TMR performed surprisingly well compared to the other two mechanisms regarding availability. If we could develop the simulations further, it would be interesting to see how TMR would look and behave if we introduced more random attacks. It would also be interesting to implement recovery and detection mechanisms into the models and compare how the Health index value changes in their presence. In our experiments we only performed a maximum of three attacks, however, with how TMR works it would be interesting to evaluate its behaviour when it is exposed to more attacks.

After performing this experiment, we discussed the definition of availability and how to measure it properly. One way of measuring availability is to determine it in a binary way, where a resource is either available or unavailable. As long as the system can complete its task, it is available even if some of the parts of the system are not responsive. Another way of viewing availability is percentage based, where the system can be anywhere between 0 and 100 percent and, as such, may be partially available. This could be more appropriate in cases where resources could gradually decrease in functionality, such as in graceful degradation. This would give a more precise overview of the system's availability, although it could potentially be difficult to measure properly.

6

Conclusion

We present our concluding remarks of our thesis here as well as some ethical and sustainability considerations of our work and future directions this work can be continued in.

This master thesis investigated the current state of resilience metrics in the automotive domain by performing a literature study surveying 311 papers, where we found that very few cybersecurity resilience metrics specifically for the automotive domain currently exist. Therefore, we decided to take a metric from the cyber-physical domain instead and adapted it to fit within the automotive domain. We perform multiple experiments simulating a very simplified car infotainment system in order to evaluate how well the adapted metric performed in an automotive context. The experiment was performed using Simulink[1] as a modelling tool and MODIFI[12] as an attack injection tool. The result of our experiment showed us that it is possible to do so with simple metrics and that it might also be possible to do so with metrics of higher complexity. Because there is very little research in cyber resilience metrics for vehicles this thesis serves as a first step to further research cyber resilience metrics within the automotive domain.

6.1 Ethical & Sustainability considerations

This thesis focuses on metrics for cyber resilience in a safety-critical system. This means that a poor analysis or portrayal of the metrics discussed in this thesis could give readers or users of the metrics the wrong impression leading to misuse of metrics. We have thoroughly analyzed the metrics discussed in order to prevent this to the extent we are able to. The analysis is done by reading the papers which presents the metrics as well as reading papers in the area of cyber resilience in order to grasp the concepts. These papers are cited when presented in the thesis in order to show what we have though of on our own and what we have read and learned from other papers and articles.

The analysis of our literature survey could potentially hide or obscure patterns that we do not notice by not bringing attention to them. While we have performed a through research of these papers there is no guarantee that we have understood

[1] https://se.mathworks.com/products/simulink.html

everything perfectly. We have kept this in mind while reading and collecting the papers in order to prevent any misrepresentation.

This thesis focuses on the automotive domain which means that it touches on vehicles which is a safety critical system. If any information is misrepresented in this thesis it could potentially have disastrous effects as accidents in safety critical systems can be dangerous. While this report mostly focuses on metrics these metrics could be a deciding factor of how to prevent such accidents and a poor representation of thees metrics could increase the likelihood of accidents not being prevented. We believe that the metrics showcased in this report is sufficiently explained so that this is avoided.

As previously mentioned this thesis focuses on the automotive domain and a topic that has to be discussed is this thesis works and its impacts of the carbon dioxide emissions of vehicles. It is unlikely that the implementation or use of cyber resilience metrics will affect a vehicles emission levels although this is something that should be taken into consideration when implementing some of these metrics.

6.2 Future works

Further research is required into resilience metrics for vehicles and the automotive domain. More specifically, automotive metrics as they are is currently a lacking area. It would be interesting to investigate the conversion of other cyber-physical metrics into the automotive domain as a possible way of increasing the number of resilience metrics in the automotive domain. Investigating resilience and metrics for different security attributes that are not availability might also further advance current resilient research, as there is a lot of focus on availability currently in the area. There exists areas in our experiment that could be developed further. Such developments could, for example, be introducing more metrics as this could show new aspects that can influence the resilience of a system. Increasing the size and complexity of the models used in the simulation could reveal how cascading effects may influence the system and how these effects can be measured. Further developing the tests with more attacks and a wider range of attacks as our attacks where limited and we believe there is more to be explored.

Bibliography

[1] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, no. S 91, pp. 1–91, 2015.

[2] *Collision between a car operating with automated vehicle control systems and a tractor-semitrailer truck*, Oct. 2017. [Online]. Available: `https://www.ntsb.gov/investigations/AccidentReports/Reports/HAR1702.pdf`.

[3] *Collision between vehicle controlled by developmental automated driving system and pedestrian*, Nov. 2019. [Online]. Available: `https://www.ntsb.gov/news/events/Pages/2019-HWY18MH010-BMG.aspx`.

[4] D. J. Bodeau, R. D. Graubart, R. M. McQuaid, and J. Woodill, "Cyber resiliency metrics, measures of effectiveness, and scoring: Enabling systems engineers and program managers to select the most useful assessment methods," *Mitre Corp Bedford Ma Bedford United States*, 2018.

[5] P. Kleberger, P. Folkesson, and B. Sangchoolie, "An integrated safety and cybersecurity resilience framework for the automotive domain," in *CARS-Critical Automotive applications: Robustness & Safety*, 2022.

[6] D. Snyder, L. A. Mayer, G. Weichenberg, *et al.*, *Measuring cybersecurity and cyber resiliency*. RAND Corporation, 2020.

[7] B. Sangchoolie, P. Folkesson, P. Kleberger, and J. Vinter, "Analysis of cybersecurity mechanisms with respect to dependability and security attributes," in *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, IEEE, 2020, pp. 94–101.

[8] M. Segovia-Ferreira, J. Rubio-Hernan, A. R. Cavalli, and J. Garcia-Alfaro, "Cyber-resilience approaches for cyber-physical systems," *arXiv preprint arXiv: 2302.05402*, 2023.

[9] T. Rosenstatter, K. Strandberg, R. Jolak, R. Scandariato, and T. Olovsson, "Remind: A framework for the resilient design of automotive systems," in *2020 IEEE Secure Development (SecDev)*, IEEE, 2020, pp. 81–95.

[10] H. Maruyama, R. Legaspi, K. Minami, and Y. Yamagata, "General resilience: Taxonomy and strategies," in *2014 International Conference and Utility Exhibition on Green Energy for Sustainable Development (ICUE)*, IEEE, 2014, pp. 1–8.

[11] S. Hosseini, K. Barker, and J. E. Ramirez-Marquez, "A review of definitions and measures of system resilience," *Reliability Engineering & System Safety*, vol. 145, pp. 47–61, 2016.

[12] R. Svenningsson, J. Vinter, H. Eriksson, and M. Törngren, "Modifi: A model-implemented fault injection tool," in *Computer Safety, Reliability, and Se-*

curity: 29th International Conference, SAFECOMP 2010, Vienna, Austria, September 14-17, 2010. Proceedings 29, Springer, 2010, pp. 210–222.

[13] D. S. Herrmann, *Complete guide to security and privacy metrics : measuring regulatory compliance, operational resilience, and ROI*. 2007, ISBN: 978-0-8493-5402-1. [Online]. Available: https://search.ebscohost.com/login.aspx?direct=true&db=edsbvb&AN=edsbvb.BV022270392&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds.

[14] European Network and Information Security Agency, "Measurement frameworks and metrics for resilient networks and services: Technical report," 2011. [Online]. Available: https://www.enisa.europa.eu/publications/metrics-tech-report/at_download/fullReport.

[15] A. Kott, *Cyber Resilience of Systems and Networks*. Jan. 2019, ISBN: 978-3-319-77491-6. DOI: 10.1007/978-3-319-77492-3.

[16] European Network and Information Security Agency, "Measurement frameworks and metrics for resilient networks and services: Challenges and recommendations," 2011. [Online]. Available: https://www.enisa.europa.eu/publications/metrics-survey/at_download/fullReport.

[17] J. M. Qurashi, K. Jambi, F. Alsolami, F. E. Eassa, M. Khemakhem, and A. Basuhail, "Resilient countermeasures against cyber-attacks on self-driving car architecture," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[18] R. Ross, V. Pillitteri, R. Graubart, D. Bodeau, and R. Mcquaid, "Developing cyber-resilient systems," 2021.

[19] I. Linkov, D. A. Eisenberg, K. Plourde, T. P. Seager, J. Allen, and A. Kott, "Resilience metrics for cyber systems," *Environment Systems and Decisions*, vol. 33, pp. 471–476, 2013.

[20] I. Linkov, D. A. Eisenberg, M. E. Bates, *et al.*, *Measurable resilience for actionable policy*, 2013.

[21] D. S. Alberts, "Information age transformation: Getting to a 21st century military (revised)," *Monograph. Command And Control Research Program (CCRP), Office of the Assistant Secretary of Defense, Washington DC*, 2002.

[22] *Oxford Learner's Dictionary*. Oxford University Press, 2024, Metric.

[23] N. Fenton and J. Bieman, *Software metrics: a rigorous and practical approach*. CRC press, 2014.

[24] E. Doynikova, A. Fedorchenko, and I. Kotenko, "Ontology of metrics for cyber security assessment," in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, 2019, pp. 1–8.

[25] G. Sabaliauskaite and A. P. Mathur, "Aligning cyber-physical system safety and security," in *Complex Systems Design & Management Asia: Designing Smart Cities: Proceedings of the First Asia-Pacific Conference on Complex Systems Design & Management, CSD&M Asia 2014*, Springer, 2015, pp. 41–53.

[26] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE transactions on dependable and secure computing*, vol. 1, no. 1, pp. 11–33, 2004.

[27] I. Zenden, H. Wang, A. Iacovazzi, A. Vahidi, R. Blom, and S. Raza, "On the resilience of machine learning-based ids for automotive networks," in *2023 IEEE Vehicular Networking Conference (VNC)*, IEEE, 2023, pp. 239–246.

[28] Z. Song, G. Ren, L. Mirabella, and S. Srivastava, "A resilience metric and its calculation for ship automation systems," Aug. 2016, pp. 194–199. DOI: 10.1109/RWEEK.2016.7573332.

[29] A. Aigner and A. Khelil, "An effective semantic security metric for industrial cyber-physical systems," in *2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS)*, IEEE, vol. 1, 2020, pp. 87–92.

[30] J. Pan, Z. Liu, D. Li, L. Wang, and B. Li, "An empirical study of software architecture resilience evaluation methods," *Journal of Systems and Software*, vol. 202, p. 111 726, 2023.

[31] N. Jacobs, S. Hossain-McKenzie, and E. Vugrin, "Measurement and analysis of cyber resilience for control systems: An illustrative example," in *2018 Resilience Week (RWS)*, 2018, pp. 38–46. DOI: 10.1109/RWEEK.2018.8473549.

[32] M. D. C. M. Kaâniche and A. Pataricza, "Dependable computing–edcc-5,"

[33] S. Hukerikar, R. A. Ashraf, and C. Engelmann, "Towards new metrics for high-performance computing resilience," in *Proceedings of the 2017 Workshop on Fault-Tolerance for HPC at Extreme Scale*, 2017, pp. 23–30.

[34] G. Kandaperumal, S. Pandey, and A. Srivastava, "AWR: Anticipate, withstand, and recover resilience metric for operational and planning decision support in electric distribution system," *IEEE Transactions on Smart Grid*, vol. 13, no. 1, pp. 179–190, 2022. DOI: 10.1109/TSG.2021.3119508.

[35] A. Bondavalli, A. Ceccarelli, L. Falai, and M. Vadursi, "Foundations of metrology in the observation of critical systems," *Resilience Assessment and Evaluation of Computing Systems*, pp. 205–211, 2012.

[36] P. Kleberger, P. Folkesson, and B. Sangchoolie, "Interplay between error/intrusion detection/handling," *CyReV D3.3*, 2021.

[37] A. Farooqui, D. Pascalic, and M. Retteli, "Metrics for evaluating resilient designs," *CyReV D2.3*, 2023.

A

List of Resilience Metrics

This appendix presents all identified metrics, paper by paper. Each metric is shortly described. For more detailed information about each metric, we refer to the original paper. These are the same as in [37].

A.1 On the Resilience of Machine Learning-Based IDS for Automotive Networks

The following metrics are presented in [27].

- **Accuracy**
  This metric measures the accuracy with which a machine learning model classifies the provided input. This is calculated by taking the total correct classifications and dividing it by all classifications.
- **False Positive Rate (FPR)**
  This metric counts every result that was marked positive that was in fact a negative.
- **False Negative Rate (FNR)**
  This metric counts every result that was marked negative that was in fact a positive.
- **F1-Score**
  This metric balances accuracy out in the case of unbalanced data. This is done by taking the harmonic mean between precision and recall. A high F1-Score of 1 means that the accuracy is also good while a low F1 score indicates that the accuracy might be misleading.

A.2 A Resilience Metric and Its Calculation for Ship Automation Systems

These metrics are presented in [28] and measures the resilience of a cooler system on a ship. Each metric is normalised to give a value between 0 and 1, these are then compiled in a vector as the final output for the system.

- **Normalised health index**
  The Normalised health index metric measures the ratio of mean time to failure of a component to the total lifetime of the component. The normalised health index metric takes the mean time to failure of a component and divides it by the total lifetime of the same component. This gives a unitless result but as the result is relevant to time it has been categorised as a time metric. This is an alternative way of measuring availability by measuring the health of the system instead.

- **Normalised network connectivity**
  Normalised network connectivity measures the worst case connectivity in a graph and compares it to the maximal possible connectivity in the graph. This metric is dependant on the number of connections and thus was categorised as an Amount metric.

- **Normalised remaining network utilisation**
  Normalised remaining network utilisation measures the remaining capacity of the system. The paper exemplifies this as "For power system, it can be measured by the remaining power capacity, in terms of kW, that remains available after the current load demands have been met.". But continues that it can be substituted for any type of unit depending on application. This metric is the remaining amount of the measured unit and thus was categorised as an Amount metric.

- **Normalised redundancy**
  Normalised redundancy measures the number of critical devices that have backup (redundant) devices to take over during a failure. It measures the redundancy by simply counting the amount of redundant devices there are for any critical device.

- **Normalised response speed**
  Normalised response speed is an indication of the time constant used in the system. This metric is a measure of time in the system and thus was categorised as a Time metric.

A.3 An Effective Semantic Security Metric for Industrial Cyber-Physical Systems

These metrics are found in [29]. In this paper, a set of rules are identified for the pre-processing, processing and post processing steps of a software's runtime that help calculate security.

- **Pre-Processing metric rule set**
  The pre-processing metric has rules that can be calculated before running the program. It contains rules such as if the calculated processing power needed is higher than the max capacity of computing power. This eventually outputs x_1.

- **Processing metric rule set**
  The processing metrics has rules that can be calculated during run time of the software. It contains rules such as if the amount of process requests is larger than received valid requests. This outputs x_2.
- **Post-Processing metric rule set**
  The Post-Processing metric contains rules that can only be calculated after runtime of the software. It contains rules such as checking if outgoing responses and valid responses are not equal to each other. This eventually outputs x_3.
- **Table metric rule set**
  The overarching metric concatenates the result from the pre-processing, processing and post-processing metrics into a single metric that looks like: $x_1x_2x_3$.

A.4 An empirical study of software architecture resilience evaluation methods

These metrics are presented in [30] and come from different methods based on an activity and component diagram. All these methods incorporate simulation to some degree in order to calculate resilience.

- **Attack surface analysis method**
  The Attack surface analysis method uses the activity and component diagram to calculate the probability that a node survives or recovers from an attack. This is calculated by measuring the resilience based on all nodes survivability divided by the amount of nodes. resilience = $\frac{\sum_{n \epsilon N} \text{prob}(n)}{|N|}$
- **Static Bayesian network method**
  This method calculates resilience in a range of 0 to 1, where 0 is the worst and 1 is the greatest resilience. A Bayesian network is created with the component diagram dependencies. Then some dependencies are classified which will be used in the equation to calculate the resilience. The classifications are as following: $P(\text{Resilience} = \text{True}) = \sum p(\text{attacks})$
  $\times P(\text{components not affected by any others})$
  $\times P(\text{components directly attacked}|\text{attacks})$
  $\times P(\text{components affected by one or more components}$
  $|\text{components affecting this component})$
  $\times P(\text{reliability}|\text{components that affect reliability})$
  $\times P(\text{restoration}|\text{reliability, components that affect restoration})$
  $\times P(\text{resilience} = \text{True}|\text{reliability, restoration})$
- **Dynamic Bayesian network method**
  A Bayesian network is constructed by analysing the activity diagram. Then the method implements node transition equations that handle nodes switching state between time slices. Thereafter a formula which takes the steady-state availability and divides it by the number of attack which results in the resilience. $\rho = \frac{A}{nln(t_1)} \sum_{i=1}^{n} \frac{A_2^i A_3^i}{ln(t_3^i - t_2^i)}$

- **Minimal Path Method**
  In order to calculate this methods resilience three other metrics of robustness, rapidity and availability needs to first be calculated. Thereafter, those three metrics, each combined with a particular weight, is used to compute the resilience. This is done by analysing the activity diagram and using a minimal path tree algorithm to feed into the equations that results in the resilience. Resilience $= \alpha_1 \cdot \text{Robustness}\alpha_2 \cdot \text{Rapidity}\alpha_3 \cdot \text{Availability}$

- **Minimal Path Method (Robustness)**
  Robustness calculates the lowest performance value in a simulation with regards to certain disturbances. $\text{Robustness}_i = \min\left(\frac{Q_{\min_{i,u}}}{100\%}\right)$

- **Minimal Path Method (Rapidity)**
  Rapidity calculates average recovery speed during a specific disturbance in a simulation. $\text{Rapidity}_i = \frac{\sum_{u=1}^{U_i} V_{i,u}}{U_i}$

- **Minimal Path Method (Availability)**
  Availability calculates how long a service has been in normal state time compared to the simulation period. $\text{Availability}_i = \frac{\sum_{t=1}^{T} P_m(t))}{T}$

- **Markov Modelling method**
  The Markov modelling method uses both activity and component diagram to create a Markov model and a state transfer matrix. With the help of last state node resilience and the state transfer matrix the architectural resilience can be calculated. Resilience $= (-1)^{k+1} R_k \frac{|E|}{|I-M|}$

A.5 Measurement and Analysis of Cyber Resilience for Control Systems: An Illustrative Example

These metrics originates from [31] and measure the resilience of a load frequency control system. The security and cyber resilience are measured with two methods, first via an examination of confidentiality, integrity and availability (CIA). Secondly via an infrastructure resilience analysis methodology. Only the infrastructure resilience analysis methodology metrics will be shown here.

- **Infrastructure Resilience Analysis Methodology**
  The infrastructure resilience analysis methodology consists of three sub metrics, Systemic Impact (SI), Total Recovery Effort (TRE) and Recovery Dependent Resilience (RDR). Systemic Impact measures the scale and duration of the performance loss related to an attack. Total Recovery Effort measures the resources and costs of returning the system to an acceptable performance level after an attack has occurred. Lastly, the Recovery Dependent Resilience is a combination of both SI and TRE and measures the impact on the system.

A.6 Structure-based resilience metrics for service-oriented networks

These metrics presented in [32] to calculate node and edge resilience of a undirected graph. They do this by defining what sub networks are created that does not include a specific service. This sub network becomes deficient of that service. Both metric is performed similarly although uses different cutsets.

- **Edge resilience**
  The minimum weight edge cutset is used and a edges weight becomes infinite if it is connected to a necessary service and 1 if it is deficient of said service. This metric calculated the minimum number of edges that needs to be deleted in order to remove a path between a target node and the target service. $G = \min\{\Gamma_j : s_j \in S\} - 1$

- **Node resilience**
  The minimum weight node cutset is used and every node has a weight of 1. This metric calculated the minimum number of nodes that needs to be deleted in order to remove a path between a target node and the target service. $G = \min\{\sigma_j : s_j \in S\} - 1$

A.7 Towards New Metrics for High-Performance Computing Resilience

Metrics to evaluate a high performance computing system are presented in [33].

- **Resilience factor performance efficiency**
  The first presented metric is about performance efficiency. This is done by comparing the time of a solution while the system is not under an attack and while it is under an attack. This can also be done with two different implementations although it is important that both solutions have the same attack frequency. $RF_{PE} = \frac{\text{time-to-solution}_{\text{event-free}}}{\text{time-to-solution}_{\text{events}}}$

- **Resilience factor value efficiency**
  Similar to the previous metric although compares variable values while the system is attack free and during attacks. $RF_{VE} = \frac{\text{ProgramValue}_{\text{event-free}}}{\text{ProgramValue}_{\text{events}}} = \frac{V_x}{V_x + |\sigma|}$

- **Resilience factor yield**
  The last metric is a way to combine resilience factors for a group of tasks. This is done by taking the geometric mean of a group of resilience factors that all share the same type, this means that all values of resilience factors that is combined needs to be either about performance efficiency or value efficiency.

  $RY = \sqrt[n]{RF_{T1} RF_{T2} \dots RF_{Tn}}$

A.8 AWR: Anticipate, Withstand, and Recover Resilience Metric for Operational and Planning Decision Support in Electric Distribution System

These metrics originates from [34] and measures the resilience of an electric distribution system. The metric is a three stage process based on system characteristics factor before, during and after a disruption. Each of the metrics calculates a resilience score from some factors.

- **Anticipate metric**
  The anticipate metric evaluates the resilience of the electrical distribution system in three main areas. First is the threat and vulnerability domain where seven factors are measured. Examples of these are: threat identification, threat analysis and documentation and Threat warning time. The second area that is measure is the power deliver and loads domain. Some factors in this domain are: critical load identification, average runtime of backup generation and fault prevention plan. Lastly is the restoration and recovery domain. Factors in this domain are: Energy storage, automatic restoration plan and standby repair crew.

- **Withstand metric**
  The withstand metric measures the resilience based on five factors which are then used to develop a pairwise comparison matrix. Examples of factors that are taken in to account are: critical load not lost, total available generation and topological robustness. Then depending on the factors impact on the system, the withstand resilience metric score is calculated.

- **Recover metric**
  The recover metric is calculated similar to the withstand metric where five factors are used to develop a pairwise comparison matrix. Some factors that are measured are: critical load restored, generation redundancy and switching operations. The final resilience score is then computed using the comparison matrix and the factors impact on the system.

A.9 Complete Guide to Security and Privacy Metrics: Measuring Regulatory Compliance, Operational Resilience, and ROI

Herrmann [13] collected a library of resilience metrics. They are categorised in to different subcategories. Below are some metrics from each different subcategory related to cyber resilience shown. The displayed metric below are not the full extent of the metrics identified in [13].

A.9.1 Logical Access Control

- Percentage (%) of inactive user accounts that have been disabled in accordance with policy this reporting period, by sensitivity of the assets accessed.
- Percentage (%) of user accounts assigned to personnel who have left the organisation or no longer have need for access that have been closed this reporting period, by sensitivity level of the assets accessed.
- Percentage (%) of workstations with session time-out/automatic log-out controls set in accordance with policy, by information sensitivity.
- Percentage (%) of user accounts that have been reviewed within the past quarter for justification of current access rights and privileges in accordance with policy, by information sensitivity.
- Percentage (%) of systems and applications that assign user access rights and privileges according to role-based access control, by asset criticality.
- Percentage (%) of mobile and personally owned devices that are required to verify compliance with approved security and configuration policies prior to being granted access to IT and information assets, by asset criticality.

A.9.2 Data Authentication, Non-Repudiation

- Percentage (%) of organisation units that employ data authentication mechanisms to verify the exchange of information, by information sensitivity category: Internally within in facility, Internally within a distributed corporate IT infrastructure, Externally with business partners, Externally with outsiders.
- Percentage (%) of organisational units that verify the origin of software patches and upgrades before acting upon them, by system risk and IT asset category.
- Length of time non-repudiation evidence and related reports are kept, by organisational unit and asset criticality.
- Distribution of organisations that employ mandatory or manual data authentication for: Critical IT assets, Essential IT assets, High sensitivity information, Moderate sensitivity information.
- By asset criticality and information sensitivity, percentage of organisational units that have tested their data authentication mechanisms this reporting period and found that they worked as specified under normal and abnormal conditions: Distribution by type and severity of errors found, this reporting period and the previous three reporting periods.

A.9.3 Encryption, Cryptographic Support

- Percentage (%) of communication channels controlled by the organisation that have been secured via encryption, by information sensitivity.

- Percentage (%) of critical and essential information assets stored on network accessible devices that are encrypted with widely tested and published cryptographic algorithms, by information sensitivity category.
- Percentage (%) of mobile computing devices that use encryption for critical and essential assets while they are stored and transmitted.
- Percentage (%) of passwords, PINs, and other authentication data that are encrypted.
- Percentage (%) of information assets for which encryption is implemented, by organisational unit, asset criticality, and information sensitivity: Data related to an organisation's operations and mission, Proprietary data, Financial data, Data covered by privacy regulations, Authentication data, Access control rules, Corporate communications.
- The strength of encryption used, by organisational unit, asset criticality, and information sensitivity: Ultra high, High, Moderate, Low.

A.9.4 Flow Control

- By system risk and asset criticality, percentage of IT assets for which permitted and prohibited operational control flows have been specified.
- By system risk and asset criticality, percentage of systems and networks that have been implemented and configured to: Enable permitted operational control flows, Prevent prohibited operational control flows, Generate alarms when an attempt is made, or successful, to bypass operational control flow mechanisms.
- By system risk and asset criticality, percentage of systems and networks that have been tested this reporting period and the results indicated that all permitted and prohibited operational control flows worked as specified under normal and abnormal conditions: Distribution by type and severity of errors found, this reporting period and the previous three reporting periods.
- By asset criticality and information sensitivity, percentage of IT assets for which permitted and prohibited data control flows have been specified.
- By asset criticality and information sensitivity, percentage of systems and networks that have been implemented and configured to: Enable permitted data control flows, Prevent prohibited data controls flows, Generate alarms when an attempt is made, or successful, to bypass data control flow mechanisms.

A.9.5 Identification and Authentication

- Number and percentage of active user IDs assigned to only one person, by criticality of assets accessed.
- Percentage of systems and applications that perform password policy verification, by system risk.

- Number of system accesses by unauthorized users through channels protected by strong identification and authentication, by type of failure.
- Number of active user passwords that are set to expire in accordance with policy, by system risk.
- Percentage of systems with critical and essential information assets that use stronger authentication than IDs and passwords.
- Percentage of systems where vendor-supplied or default accounts and passwords have been disabled or reset, including maintenance back doors, by system risk.
- Percentage of systems and networks that lock user accounts or the point of entry after two or three failed log-on attempts, by system risk and asset criticality.

A.9.6 Maintainability, Supportability

- By system risk and asset criticality, percentage of system development acquisition initiatives that: A: Include and evaluate supportability and maintainability requirements. B: Analyse supportability and maintainability options. C: Identify and reserve adequate funding for supportability and maintainability throughout the life of a system or product. D: Develop and coordinate a comprehensive supportability plan that specifies support levels, support agents, and support scenarios, before a product or system is fielded. E: Monitor the execution of the supportability plan at a senior level of management.
- Extent to which a product or system has been designed and developed with supportability attributes, using the following scale (high 5, medium 3, low 1, none 0), maximum total points 50: A: Modularity. B: Testability. C: Easy to install, configure, and operate. D: Easy to update and modify, such as changing interfaces. E: Design and implementation adheres to national or international consensus standards. F: Lack of dependence on specific hardware or software platforms or external interfaces. G: Adaptability to different operational profiles and duty cycles. H: Percentage of support activities that can be performed without disrupting operations. I: Expected useful life span with respect to technology refresh and obsolescence. J: Product maturity.
- Number and percentage of security incidents, by incident severity, that were precipitated by inadequate or ineffectual supportability and maintainability.

A.9.7 Privacy

- Number and percentage of personal data elements collected that are beyond the scope needed for the stated purpose of collection: Distribution by type of data, Disposition of extraneous personal data.
- Number and percentage of personal data elements that were collected without: The individual's prior knowledge and consent, Telling the individual what the

data would be used for, Telling the individual who the information will be disseminated to, Telling the individual how long the data would be kept.

- Frequency with which the accuracy, completeness, and currency of personal data held by an organisation is verified: A: Date of most recent verification activity. B: Percentage of data found to be accurate, complete, and current during most recent verification activity.
- Number of times personal data was put to a new use without first obtaining the individual's approval.
- Distribution of times personal data was and was not disposed of in a secure manner after the end of the stated usage time interval.
- Speed with which an organisation is able to respond to an individual's request to: Receive copies of their personal data, Correct inaccurate data.

A.9.8 Residual Information Protection

- By information sensitivity and system risk, percentage of information assets for which residual information protection mechanisms are invoked upon: Allocation of dynamic reusable resources, De-allocation of dynamic reusable resources, Both allocation and de-allocation of dynamic reusable resources.
- Percentage (%) of systems for which residual information protection has been implemented consistent with rollback requirements, by system risk.
- By asset criticality and information sensitivity, percentage of organisational units that have tested their residual information protection mechanisms this reporting period and found that they worked as specified under normal and abnormal conditions. Distribution by type and severity of errors found, this reporting period and the previous three reporting periods.
- Number and percentage of security incidents, by incident severity, related to: Failure of a residual information protection mechanism, Faulty implementation or configuration of a residual information protection mechanism, Lack of residual information protection, A combination of two or more of the above.

A.9.9 Security Management

- Percentage of security attributes that were verified this reporting period and found to be accurate, valid, and current.
- Percentage of security attributes that are linked to a specific expiration date.
- Percentage of security attributes for which the action to be taken upon expiration is defined.
- Percentage of security attributes for which revocation rules have been defined and verified, by attribute type.

- Average time required to revoke security attributes, by attribute type: Number of security attributes revoked this reporting period.
- Percentage of security attribute revocation rules that specify the action to be taken upon revocation of an attribute: Percentage of security attribute revocation rules that specify what protective measures are to be taken between the time revocation is requested and completed.
- Percentage of security attribute revocation rules that specify what evidence is to be kept about why, how, and when a security attribute was revoked and attempts to use the attribute after it was revoked: Length of time this evidence is kept.

A.9.10 Audit Trail, Alarm Generation

- By asset criticality, percentage of IT assets that are covered by more than one type of audit mechanism.
- By asset criticality, percentage of IT assets for which audit data is captured and reported: Locally, Regionally, Enterprise wide.
- By asset criticality, percentage of IT assets for which audit data can easily be sent to a centralised security audit management function: Locally, Regionally, Enterprise wide.
- By asset criticality, percentage of IT assets for which audit data can easily be merged with data from other audit mechanisms: Locally, Regionally, Enterprise wide.
- Percentage of systems for which event and activity logs are monitored and reviewed in accordance with policy, by system risk and asset criticality.
- By asset criticality, percentage of IT assets for which the audit capability distinguishes: Potential security violations, Imminent security violations, Actual security violations.

A.9.11 Availability

- Percentage (%) of operational time that critical services were unavailable (as seen by users and customers).
- Percentage (%) of IT assets for which availability: Requirements have been specified or Is routinely monitored, measured, and reported.
- Percentage (%) of critical and essential IT assets for which redundancy is implemented: Active redundancy, Standby redundancy, Monitored redundancy.
- Percentage (%) of critical and essential IT assets and services for which diversity is implemented: Hardware, Software, Telecommunications equipment, Telecommunications paths.

- Percentage (%) of critical and essential IT assets and services for which fault tolerance is implemented: Hardware, Software, Application systems, Telecommunications networks.
- Percentage (%) of critical and essential IT assets for which block recovery is implemented.
- By asset criticality, percentage (%) of IT assets for which recovery procedures have been defined and implemented.

A.9.12 Error, Exception, and Incident Handling

- By asset criticality, percentage of IT assets that have a robust error, exception, and incident handling capability.
- By asset criticality, percentage of IT assets for which robust error, exception, and incident handling is implemented: A: Within operating systems and utilities. B: Within custom software applications. C: Within COTS software applications. D: Within hardware components. E: Between hardware and software components. F: Between operating systems and software applications. G: Between different COTS software applications. H: Between COTS and custom software applications. I: Within networks. J: At interfaces to external systems and networks. K: Such that it is able to detect erroneous conditions in the operational environment.
- By asset criticality, percentage of IT assets that have a robust error, exception, and incident handling capability that: A: Defines normal system behavior, modes, states, and parameters. B: Defines abnormal and illegal behavior, modes, states, and parameters and the specific corrective action to be taken for each. C: Traps unknown and undefined erroneous conditions and specifies the appropriate corrective action to be taken.
- By asset criticality, percentage of IT assets for which error, exception, and incident handling capabilities cover the entire threat control chronology.
- By asset criticality, percentage of critical checkpoints throughout the IT infrastructure for which error, exception, and incident handling has been implemented.

A.9.13 Fail Safe/Fail Secure, Fail Operational/Fail Soft/Graceful Degradation/Degraded Mode Operations

- By asset criticality and system risk, percentage of assets, systems, and networks for which failure modes have been thoroughly identified.
- By asset criticality, system risk, and failure mode severity, percentage of identified failure modes that are not mitigated by fault tolerance, for which fail safe/fail secure and/or fail operational controls and procedures have been implemented.

- By asset criticality and system risk, percentage of systems and networks for which controls and procedures are in place to execute controlled failure modes on short notice: Fail safe/fail secure, Fail operational, Both.
- By asset criticality and system risk, percentage of controlled failure mode implementations that address all sources of failures: Hardware failures, Software failures, Network failures.
- By asset criticality and system risk, percentage of systems and networks for which service priorities have been specified for all critical and essential functions, services, and tasks, while operating in a fail operational mode.
- By asset criticality and system risk, average amount of time it takes to transition to a controlled failure mode: Fail safe/fail secure, Fail operational.

A.9.14 Integrity

- Percentage of IT integrity security controls that protect against: Accidental malicious activity, Intentional malicious activity.
- Percentage of assets within the IT infrastructure for which integrity is continuously monitored, by asset criticality: A: Hardware. B: Telecommunications equipment and services. C: Operating systems software and utilities. D: Applications system software. E: User data (online and offline). F: System data (online and offline). G: External interfaces.
- Percentage of assets within the IT infrastructure for which the action to be taken (automatically and/or manually) upon the detection of an integrity error is defined, by asset criticality: A: Hardware. B: Telecommunications equipment and services. C: Operating systems software and utilities. D: Applications system software. E: User data (online and offline). F: System data (online and offline). G: External interfaces.
- By asset type, distribution of assets for which integrity measures have been implemented consistent with their criticality and sensitivity.
- By information sensitivity, percentage of stored data assets for which: A: The integrity of backup and archival procedures has been verified this reporting period. B: Environmental controls have been verified this reporting period. C: Access and distribution physical security controls have been verified this reporting period.

A.9.15 Domain Separation

- Percentage (%) of IT assets for which domain separation is implemented, by asset criticality.
- Percentage (%) of information assets for which domain separation is implemented, by information sensitivity.

- By asset criticality, number and percentage of systems and networks for which: A: User functions are separated from system management functions. B: Security management functions are separated from network and system management functions. C: Data and memory regions are partitioned.
- Number and percentage (%) of systems and networks that implement domain separation through: A: Logical partitioning. B: Physical partitioning. C: Information hiding. D: Security kernels. E: Encapsulation.
- Number and percentage (%) of application systems that monitor and control the execution of COTS products through a security kernel, by system risk.
- Number and percentage (%) of networks that protect data during transmission through encapsulation, by risk category.
- Number and percentage (%) of security management information parameters that are shared with or accessible by non-security related functions.

A.9.16 Resource Management

- By organisational unit and asset criticality, percentage of IT resources that are included in and covered by: Resource allocation procedures and controls, Service priority procedures and controls, Capacity management procedures and controls.
- By organisational unit and asset criticality, percentage of resource management procedures and controls that address: A: Normal operations. B: Degraded mode operations. C: Parallel mode operations. D: Graceful degradation. E: Fail operational. F: Fail safe/fail secure.
- By organisational unit and asset criticality, percentage of resource management procedures and controls that have the capability to: Preempt tasks or operations quickly, Change and communicate the resource allocation schema quickly, Change and communicate service priorities quickly, Change and communicate capacity management strategies quickly.
- By organisational unit and asset criticality, percentage of IT resource needs that have been identified for: Normal duty cycle, A peak loading scenario, A low load scenario, The minimum resources needed for degraded mode operations.
- By organisational unit and asset criticality, percentage of resource allocations, service priorities, and capacity management strategies that are consistent with an enterprise wide view toward achieving the organisation's mission.

A.10 Cyber Resiliency Metrics, Measures of Effectiveness, and Scoring

The following metrics are presented in [4].

A.10.1 Continue – Minimise degradation of service delivery

Perform mission damage assessment

- Elapsed time for mission damage assessment.
- Average, median, or maximum time required to validate the integrity and/or quality of mission-critical data.
- Percentage of mission-critical data assets for which data integrity / quality can be validated.
- Percentage of mission-critical data assets for which data integrity / quality has been validated since initiation of CCoA.
- Average, median, or maximum time required to validate the integrity and/or quality of security-critical data.
- Average, median, or maximum time required to validate the integrity and/or behaviour of mission-critical services or processes.
- Percentage of mission-critical applications for which integrity / behaviour has been validated since initiation of CCoA.
- Average, median, or maximum time required to validate the integrity and/or behaviour of security-critical services or processes.
- Percentage of security-critical applications for which integrity / behaviour has been validated since initiation of CCoA.
- Frequency of software/service integrity check.

Maintain acceptable levels of performance for mission critical, security-critical, and mission-supporting applications and services

- Degree of degradation of a specific mission-essential function (or set of functions).
- Length of time between initial disruption and availability (at minimum level of acceptability) of mission-essential functions.
- Percentage of mission-critical applications and services for which MOPs remain at or above their required levels [for the duration of the mission task they support | for the duration of the mission they support | for the (specified) time period].
- Percentage of pre-disruption availability / performance after disruption.
- Percentage of security-critical applications and services for which MOPs remain at or above their required levels over (specified) time period.
- Percentage of mission-supporting applications and services for which MOPs remain at or above their required levels [for the duration of the mission task they support | for the duration of the mission they support | for the (specified) time period].

Select and tailor Cyber Course of Action (CCoA)

- Time between selection of CCoA and completion of tailoring.
- Time between determination that a CCoA must be taken and initiation of tailored CCoA.

Dynamically reconfigure existing resources

- Percentage of cyber resources which can be reconfigured on demand.
- Time between decision to reconfigure resources and completion of reconfiguration.
- Percentage of cyber resources which can be [automatically, manually] reconfigured.

Dynamically provision by reallocating existing resources

- Percentage of cyber resources which can be reallocated on demand.
- Time between decision to reallocate resources and completion of reallocation.

Relocate resources to minimise service degradation

- Time between decision to relocate resources and completion of relocation.
- Percentages of services which can be relocated virtually (e.g., to another virtual machine).
- Percentage of services which can be automatically relocated virtually.
- Percentage of resources which can be relocated physically (e.g., to a backup facility).

A.10.2 Continue – Minimise interruptions in service delivery

Perform mission damage assessment

- Elapsed time for mission damage assessment.
- Average, median, or maximum time required to validate the integrity and/or quality of mission-critical data.
- Percentage of mission-critical data assets for which data integrity / quality can be validated.
- Percentage of mission-critical data assets for which data integrity / quality has been validated since initiation of CCoA.
- Average, median, or maximum time required to validate the integrity and/or quality of security-critical data.
- Average, median, or maximum time required to validate the integrity and/or behaviour of mission-critical services or processes.

- Percentage of mission-critical applications for which integrity / behaviour has been validated since initiation of CCoA.
- Average, median, or maximum time required to validate the integrity and/or behaviour of security-critical services or processes.
- Percentage of security-critical applications for which integrity / behaviour has been validated since initiation of CCoA.
- Frequency of software/service integrity check.

Coordinate response activities to ensure synergy rather than interference

- Percentage of responsible organisational entities which have established points of contact, primary and alternative lines of communication, and documented procedures for responding to a cyber incident.
- Time since last exercise.
- Frequency of joint exercises.

Deploy diverse resources rapidly (e.g., in near real time)

- Time between decision to redeploy resources and completion of redeployment.
- Number of differences between initial set of resources and redeployed set.

Fail over to replicated resources

- Average, median, or maximum time to fail over mission-critical functions over [specify period over which measurements are taken].
- Percentage of failovers which met required MOPs during [specify period over which measurements are taken].
- Time since last test of failover.
- Length of time to deploy redundant resources.
- Length of time to bring a backup server online.

Replace suspect hardware components with protected alternates

- Time to replace a mission-critical hardware component with a protected alternate.
- Confidence that alternate is not affected by similar issues.

Switch processing to use alternative processing paths (i.e., different sequences of services or applications used to respond to the same request)

- Average, median, or maximum time to switch a mission-critical function to an alternative processing path.
- Frequency of use/test of alternative processing paths.

Switch communications to use alternative communications paths (e.g., different protocols, different communications media)

- Average, median, or maximum time to switch a mission-critical connection to an alternative communications path.
- Frequency of use/test of alternative communications paths.

Locate and switch over to alternative mission data sources

- Average, median, or maximum time to locate and switch over to an alternative mission data source.

Locate and switch over to alternative information stores

- Average, median, or maximum time to locate and switch over to an alternative information store.

A.10.3 Continue – Ensure that ongoing functioning is correct

Validate provenance of mission-critical and system control data

- Average, median, or maximum time required to validate the provenance of mission-critical and system control data.
- Percentage of mission-critical and system control data for which provenance can be validated.
- Percentage of mission-critical and system control data for which provenance has been validated since the initiation of the CCoA.
- Average, median, or maximum time required to validate the provenance of security-critical data.
- Percentage of security-critical data for which provenance can be validated.
- Percentage of security-critical data for which provenance has been validated since the initiation of the CCoA.

Validate data integrity / quality to ensure it has not been corrupted

- Average, median, or maximum time required to validate the integrity and/or quality of mission-critical data.
- Percentage of mission-critical data assets for which data integrity / quality can be validated since initiation.
- Percentage of mission-critical data assets for which data integrity / quality has been validated since initiation of CCoA.
- Average, median, or maximum time required to validate the integrity and/or quality of security-critical data.
- Number of points in a mission thread where mission-critical data is validated in support of an operation.

Validate software and hardware / service integrity / behaviour to ensure it has not been corrupted

- Average, median, or maximum time required to validate the integrity and/or behaviour of mission-critical services or processes.
- Percentage of mission-critical applications for which integrity / behaviour has been validated since initiation of CCoA.
- Average, median, or maximum time required to validate the integrity and/or behaviour of security-critical services or processes.
- Percentage of security-critical applications for which integrity / behaviour has been validated since initiation of CCoA.
- Frequency of software/service integrity check.
- Software / service integrity check performed on operational systems [yes/no].

A.10.4 Reconstitute – Identify damage and untrustworthy resources

Identify resources that have been destroyed, damaged beyond repair, or otherwise made unavailable (e.g., network connections lost)

- Time to identify unavailable resources and represent damage in status visualisation.
- Time to notify services or mission / business functions which use damaged or unavailable resources that those resources are no longer available.

Identify corrupted, falsified, or suspect information

- Percentage of mission-critical data assets for which data integrity / quality is validated.
- Percentage of mission-supporting data assets for which data integrity / quality is validated.
- Average, minimum, or maximum time to identify suspect information.
- Number of locations where corrupted / falsified information checks occur.
- Data validation includes data format, data types, and ranges [yes/no]
- Time to notify services or mission / business functions which use suspect information to delete or disregard that information.

Identify compromised, faulty, or suspect processes or services (i.e., those which can no longer be trusted)

- Percentage of [mission-critical, security-critical, supporting] processes or services which are validated.

- Time to identify suspect [mission-critical, security-critical, supporting] processes or services.
- Time to notify services or mission / business functions which use or communicate with suspect processes or services to terminate interactions with those services.

Identify damaged, corrupted, or subverted components

- Percentage of hardware components to which tamper-evident technologies have been applied.
- Percentage of mission critical components that employ anti-tamper, shielding, and power line filtering.
- Percentage of such components which are checked.
- Time to identify damaged components.

A.10.5 Reconstitute – Restore functionality

Execute recovery procedures in accordance with contingency or continuity of operations plans

- Time between initiation of recovery procedures and completion of documented milestones in the recovery, contingency, or continuity of operations plan.
- Time between event or detected circumstances which motivated recovery procedures and achievement of [minimum acceptable, target] mission MOPs.
- Percentage of mission capabilities for which [minimum acceptable, target] MOPs are achieved within [minimum threshold, target] period of time since initiating event.
- Percentage of mission-critical cyber resources which are recovered from a backup.
- Size of gap between lost and recovered mission-critical resources (time service or connection was unavailable, number of records not recovered).
- Percentage of mission-essential processes and interfaces restored to predisruption state.
- Length of time to reconstitute a key information asset from a backup data store.

Restore non-critical functional capabilities

- Time between event or detected circumstances which motivated recovery procedures and achievement of [minimum acceptable, target] MOPs for supporting functional capabilities.
- Percentage of supporting functional capabilities for which [minimum acceptable, target] MOPs are achieved within [minimum threshold, target] period of

time since initiating event.

- Percentage of non-mission-critical resources which are recovered from a backup.
- Size of gap between lost and recovered non-mission-critical resources (time service or connection was unavailable, number of records not recovered).

Coordinate recovery activities to avoid gaps in security coverage

- Percentage of cyber resources for which access control is maintained throughout the recovery process.
- Percentage of cyber resources for which access controls at multiple levels or using different mechanisms are maintained consistently throughout the recovery process.
- Percentage of cyber resources for which auditing or monitoring is maintained throughout the recovery process.
- Duration of gap in auditing or monitoring for [mission-critical resource, non-mission-critical resource] during recovery.

Reconstruct compromised (i.e., destroyed, corrupted) critical assets or capabilities from existing resources

- Percentage of compromised critical information stores which are reconstructed from existing resources.
- Percentage of compromised critical information stores which are irretrievably lost.
- Percentage of compromised services or functions which are reconstructed from existing resources.
- Time to reconstruct an asset or capability from existing resources.
- Time to reconstruct an asset or capability from the current gold image.
- Time to reconstruct an asset or capability from a previous gold image.
- Minimum amount of information or service loss necessary to make the system inoperable.
- Time to locate tools, services, and data sources needed to repair or reconstitute an infrastructure that serves mission requirements.
- Time to combine tools, services, and data sources needed to repair or reconstitute the infrastructure that serves mission requirements.
- Length of time to put into operational use the tools, services, and data sources needed to repair or reconstitute the infrastructure that serves mission requirements.

A.10.6 Reconstitute – Heighten protections during reconstitution

Intensify monitoring of restored or reconstructed resources

- Percentage of cyber resources for which additional auditing or monitoring is applied during and after the recovery process.
- Length of time to bring online a backup network intrusion detection system.

Isolate or restrict access to or by restored or reconstructed resources

- Percentage of reconstituted cyber resources for which more stringent access controls are applied during and after reconstitution.
- Percentage of reconstituted cyber resources which are placed in a restricted enclave for a period after reconstitution.

A.10.7 Reconstitute – Determine the trustworthiness of restored or reconstructed resources

Validate data provenance of restored or reconstructed resources

- Percentage of restored or reconstructed [mission-critical, security-critical, supporting] data assets for which data provenance is validated.
- Validate data integrity / quality of restored or reconstructed resources to ensure they not been corrupted
- Percentage of restored or reconstructed [mission-critical, security-critical, supporting] data assets for which data integrity / quality is checked.
- Quality of restored / recovered / reconstituted data.

Validate software / service integrity / behaviour of restored or reconstructed applications, services, and processes to ensure they have not been corrupted

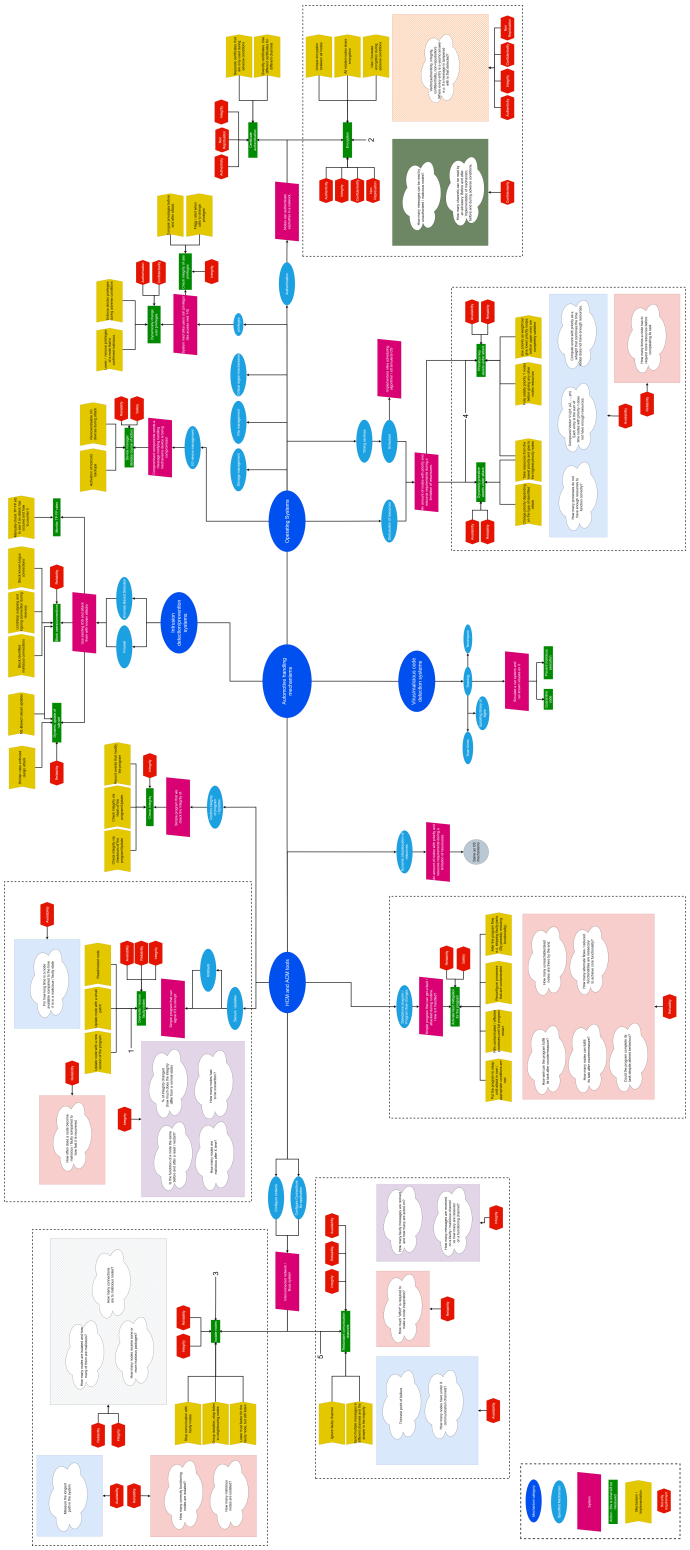
- Percentage of restored or reconstructed [mission-critical, security-critical, supporting] applications, services, and processes for which behaviour is checked.

General

- Level of trust in a system that has been restored to its pre-disruption capability.

B

Mind Map



C

Appendix: Scopus Query

(TITLE-ABS-KEY (resilience) AND TITLE-ABS-KEY (software) OR TITLE-ABS-KEY (survivability) OR TITLE-ABS-KEY (recovery) OR TITLE-ABS-KEY (error AND handling) AND TITLE-ABS-KEY (fault AND tolerance) AND NOT TITLE-ABS-KEY (memory) AND NOT TITLE-ABS-KEY (wireless) AND NOT TITLE-ABS-KEY (hardware) AND TITLE-ABS-KEY (automotive) AND TITLE-ABS-KEY (metric)) AND (LIMIT-TO (SUBJAREA , "COMP")) AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "ar")) AND (LIMIT-TO (LANGUAGE , "English"))