The DiSIEM project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 700692.
Editor

Kizito Salako, City

Contributors

Ana Respicio, FCiencias.ID
Pedro Dias Rodrigues, EDP

<table>
<thead>
<tr>
<th>Version</th>
<th>Author</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>City</td>
<td>First version of the document, submitted to the EC</td>
</tr>
</tbody>
</table>
Executive Summary

This deliverable furthers the modelling, evaluation and forecasting work first reported in deliverable D3.2 [1]. There, we highlighted the potential of employing diverse redundancy – the combined use of different, but complementary, network defence tools performing similar functions – as an affordable means of improving cyber-security through the use of SIEMs. While the effectiveness of such diverse configurations of tools has been studied and successfully demonstrated in a number of other contexts [2, 3], its application to improving the security of a SIEM-monitored network presents unique challenges.

In this deliverable, by way of formal modelling, numerical analyses and a case-study, we demonstrate that the potential benefits from using diversity can be significant – either in terms of the reductions one can expect in the losses (when failures occur) upon employing diverse configurations, or in terms of how diverse configurations more reliably discriminate between benign and malicious network activity. But, to use diversity effectively is to thoughtfully navigate along a path that is fraught with uncertainty and signposted with many trade-offs. Uncertainty about 1) the nature of activities on the network (i.e. whether benign or malicious), 2) the vulnerabilities and faults in cyber-defence tools, 3) whether the defences will soon fail, 4) what sort of failure it will be if they do fail, and 5) the losses that accrue due to failure. These all conspire to make the task of optimally configuring network defences a non-trivial one. We give much guidance in this regard, using statistical approaches to demonstrate the trade-offs that can arise, and to explore what diverse configurations have to offer. We also provide theoretical and practical guarantees for determining the maximum benefits, and detriments, diversity can bring, all the while pointing out those configurations that achieve these.

We report on a case study that demonstrates these ideas using network data from an airline booking application. With data provided by Amadeus (an industrial partner of the DiSIEM project), diversity is used to improve upon the detection capabilities of two detection tools configured to detect malicious web-scraping activities. Rather strikingly, there are cases where the improvement is shown to be very significant. Moreover, those optimal diverse configurations that guarantee the smallest expected losses when used are deduced. This is done for different scenarios with various loss sizes associated with the failures of the tools. The results of the study are consistent with the theoretical guarantees outlined in this deliverable.

Of course, evaluating the security that a diverse configuration of tools brings to a network now is important, and perhaps just as important is accurately anticipating how secure the network will be, to aid operational planning. The dynamically changing nature of IT networks – such as when the user base expands and user patterns of behaviour change, or when network devices’ vulnerabilities are patched or exploited, or when network technologies undergo maintenance or upgrades – means that, for a SOC operator to effectively anticipate problems that can arise in the future and plan accordingly, they require statistically principled approaches.
for building and objectively evaluating forecasts of future events, based on past ones. We build and demonstrate such an approach, inspired by similar ideas that have found success in predicting the future reliability of software-based systems. In particular, we define a model that extrapolates, from past publicly available data, the future occurrences of when known vulnerabilities will get patch releases, or when exploits will be announced. Based on such a model, we also show how to use the assessed, systematic shortcomings of previous forecasts, to improve future ones. These techniques – for evaluating and improving forecasts – have been implemented as part of DiSIEM’s Diversity Forecasting and Analytics platform.

We conclude the deliverable with some general comments about the statistically dependent nature of network data, and the challenges this brings for security assessment. Future work will turn to conservative Bayesian inference methods to overcome these challenges.
Contents

1 Introduction 8

2 Probabilistic Model of Failure Diversity amongst Cyber-defence Configurations 12

3 Loss-size vs Reliability Trade-offs amongst Diverse Cyber-defence Configurations 16

4 Optimal Adjudication 24

5 The Efficacy of Diversity in Anomaly Detection: A Case-study 31
   5.1 An Application of Diverse Redundancy 33
   5.2 An Application of Optimal Adjudication 34

6 Security Risk Metrics 36
   6.1 Classical Security Metrics 36
   6.2 Multi-level Risk manager 37

7 Statistical Models for Forecasting Security Risk 38
   7.1 “Single Event” vs “Multiple Event” Forecasting Models 39
   7.2 Model Fitting via Non-parametric Statistical Inference 42
   7.3 Evaluating and Improving Forecast Accuracy: U-plots and Recalibration 44

8 Statistical Dependence in Network Data: Implications for Security Assessment 50

9 Conclusions 53
List of Figures

1. SIEM data collection and fusion process ............................ 8
2. Probability distribution of loss due to failure .......................... 13
3. False-positive (FP) and false-negative (FN) regions ....................... 15
4. The effect of diversity on FP and FN errors ............................. 16
5. The 3 extreme loss distributions with the same mean .................... 17
6. Numerical example of 2 loss distributions with the same mean ....... 19
7. The loss distribution for the most reliable configuration: two proofs 20
8. Average loss vs standard-deviation example plots .......................... 21
9. ROC Convex Hull Theorem ............................................. 23
10. Tool responses partition the set of network-data $\Omega$ into subsets ... 25
11. Optimal adjudication and (un)reliability comparison .................. 30
12. Optimal adjudication and (un)reliability comparison (contd.) .......... 30
13. Optimal adjudication and (un)reliability comparison (contd.) .......... 31
14. Sensitivity and specificity: comparisons of single tools vs diverse 33
    configurations .................................................................. 33
15. Total error cost based on FP:FN cost ratio for first study .............. 35
16. Total error cost based on FP:FN cost ratio for second study .......... 36
17. “Single event” point process .............................................. 39
18. Vulnerability point process .................................................. 40
19. A hierarchy of point processes .............................................. 41
20. U-plots explained ............................................................. 45
21. “Recalibration” explained ..................................................... 46
22. Some u-plots and forecasts for time-to-next-patch: a comparison of 47
    3 datasets .................................................................. 47
23. Some u-plots and forecasts for time-to-next-exploit: a comparison 48
    of 3 datasets .................................................................. 48
24. Some u-plots and forecasts for Windows 10 ............................. 49
25. Constraints on feasible prior distributions of FP/FN probabilities .... 52
List of Tables

1  The 16 adjudication functions for 2 IDSs .................................. 26
2  The 8 expected losses for the adjudication functions in Table 1 size error 26
3  Two examples of Table 2 ................................................................. 28
4  The two probability distributions used in the examples of Table 3 size error 28
5  Sensitivity and specificity for diverse configurations ................. 33
6  Optimal adjudication functions from two studies: uniform losses . 34
7  Optimal adjudication functions from first study: varied losses .... 35
8  Optimal adjudication functions from second study: varied losses . 35
9  Parameters for data-generating processes and the resulting MLE fits 43
1 Introduction

This deliverable reports on the research carried out under WP3, aimed at informing the design of the “diversity assessment and forecasting” components of WP6, and diversity visualisation in WP5. This research furthers the initial, more speculative, statistical modelling work reported in deliverable D3.2 [1], but now with added analyses, deeper theoretical results and multiple demonstrations of the practical implications of these ideas in a number of examples. There is a particular focus on addressing 3 main challenges with using statistical methods to configure network tools for improved security – 1) deciding on the most appropriate measure of risk, 2) determining those configurations that give the largest security improvement above current levels, and 3) assessing, and possibly improving, forecasts of how security levels will change in the future. To appreciate these challenges in some detail, it will be helpful to keep the following basic scenario in mind.

The backdrop for our analyses is that of a Security Incident and Events Management (SIEM) system – a collection of software-based tools that aid a Security Operations Centre (SOC) operator in their defence of an IT/OT-network. The SIEM receives network data from various sensors and cyber-defence tools across the network\(^1\), it analyses and transforms/combines this data into relevant formats, and finally presents the results to the operator for consideration. Common examples of cyber-defence tools in use include firewalls, intrusion-detection/prevention-systems (IDS/IPS) and anti-virus software.

One may envisage a SIEM’s operation as depicted in Figure 1. Here, a SIEM

---

\(^1\)Collectively, these sensors and tools provide the SIEM with network data in varied, yet related, forms; for instance, the firewalls list the connections that were "dropped”/not allowed into the network, IDSs send both alarms/alerts and details of the abnormal network activity that triggered the alarms/alerts etc.

\(^2\)Such analysis can range significantly – from sophisticated algorithms to look for correlated behaviour between data streams from seemingly distinct sources, to merely forwarding the data received as-is onwards to the operator.

Figure 1: An example, conceptual depiction of the flow of data in an IT-Network monitored by an SOC operator using a SIEM.
is viewed as transforming data about activities on the network into a form that the SOC operator finds useful in rooting out vulnerabilities or malicious activity. Certainly, to be effective, the operator cannot be inundated with all of the network data that a SIEM can, potentially, present. In practice, indications of attacks or likely attacks under way are produced by various sensors, like the above-mentioned IDSs and antivirus, or possibly advance alert services of which the organisation is a client, or even open-source intelligence (OSint) analysis tools that monitor online informal alerting systems like social networks. Each sensor produces a reading, e.g. a Boolean flag meaning “there’s an attack on”, and/or a number “there’s an attack on with 67% probability”, and/or red/amber/green lights, and possibly also descriptions of which kinds of attack it “guesses” to be on. Instead of the SIEM simply presenting all these raw alarm readings, disparate and often discordant, to the SOC operator, it is desirable to use the knowledge available to the SIEM designer about the quality of the alarms to give the operator a more refined reading, e.g. a flag or percentage or red/amber/green light that takes into account all the alarm readings from the various sensors. We call this operation of extracting one “opinion” from many an adjudication (others call it “voting”, “alarm fusion”, etc.). This adjudged alarm information could be presented to the operator instead of, or together with, the raw alarm readings from which it is distilled.

So, as depicted in Figure 1, the SIEM can be designed to have an adjudication function that is the final arbiter of what information the operator is presented with from the SIEM and its myriad data sources. The precise form of this adjudication may vary – e.g., if all the alarms are Boolean, it is typical to use 1-out-of-N (an alarm is raised if anyone of the attack sensors raises it), or k-out-of-n “voting”. Of the possible adjudication functions (and there can be many possibilities), what is best depends on the quality of the attack sensors, the details of how they differ in their strengths and weaknesses, and the preferences about the risk of erroneous adjudged readings being given to the operator (e.g., loss trade-offs in terms of false-positive and false-negative detection errors).

There is some flexibility here, in how one might configure cyber-defence tools to monitor and control activities on the network, as well as to control the network data received by a SIEM. For instance, cyber-defence tools can be configured to monitor for certain forms of network activity, barring them from taking place (e.g. blocking ports or protocols, dropping packets from “blacklisted” IP addresses etc). As a consequence, a whole raft of attacks (and legitimate user activity) are made impossible. The point is that what the SIEM ultimately presents to the SOC operator “upstream” is affected by both the adjudication function and what the SIEM receives from suitably configured layers of cyber-defence tools “downstream” in the process, as depicted in Figure 1. Certainly, one can take the view that, in this sense, the physical configuration of the layers of cyber-defence tools, itself,

---

3A consequence of this is that the SIEM’s adjudication function won’t be receiving copious amounts of data that would have been generated had the forbidden network activities been allowed to take place.
is the implementation of an adjudication function. And, together with the SIEM adjudication function, both of these ultimately determine what is allowed to take place on the network and what information the operator is presented with.

Both the layers of cyber-defence tools “downstream” and the SIEM adjudication function “upstream” can be mathematically modelled in the same way, since there are many cases where their functionality – the data they produce as output or the action they take, given the data they receive as inputs – can be reproduced, and possibly improved upon, by using diverse redundancy. This is the idea that components performing similar functions, but doing so in different ways, can fail in different ways too, and such differences may be harnessed in the combined use of these components, resulting in more reliable functioning than any of the individual components can provide. “More reliable” would mean a larger probability of the function being correctly provided when needed. Of course, in a security context, failures (of, say, a cyber-defence tool deployed on a network to simply flag suspicious activity) can occur in roughly one of two ways – either the tool has treated benign activity on the network as malicious, or it has failed to detect an ongoing cyber-attack. Reliability, in this context, is intimately linked to the level of security provided by the tool. Admittedly, this is a rough, simplistic view of these failures: in practice, more than two failure modes are possible. For instance, tools will typically provide a reason for flagging activity as suspicious, leading to the possibility that malicious activity can be flagged but for the wrong reason, say. However, such added complications merely expand the number of failure modes to consider, and require no new mathematical ideas to gain insight into the implications of diverse redundancy.

Of course, reliability – quantified as the probability of correct functioning – is only part of the story. Just as important are the actual losses incurred by stakeholders when a system experiences failures. Such loss comes in various forms, like the sudden unplanned demands placed on an organisation’s workforce during a cyber-attack, or the resulting loss of business from disgruntled customers, or the added capital expenditure needed to restore and upgrade a compromised network, and so on. The expected loss when using a component/configuration is a quantitative measure of the cost of failures of that component/configuration. The value of expected loss is a consequence of both the uncertainty of whether/when/how failures happen and the losses incurred if they do. In principle, one may decide on the best configuration of cyber-defence tools by, first, estimating the expected loss in using each alternative component/configuration, and then comparing these to see which ones give the smallest expected loss.

But “smallest expected loss” does not equate to “most reliable”, and it is here that the first of our 3 challenges rears its head. In deciding on the best component/configuration, one should be aware of the trade-offs that can occur in reliability if, say, multiple alternatives all give very similar levels of expected loss. Even if one decides that “best” does mean “smallest expected loss”, there is also an assessment challenge to overcome. Because, potentially, there can be a very large number of possible alternative configurations. How does one ensure that,
in choosing the best, one has really considered all of the possible alternatives? Moreover, if it is infeasible to assess the expected loss for all these alternatives, can one still determine that alternative which is best?

The dynamic nature of network security presents our final challenge. One may have assessed how secure the system is “now”, using expected loss say, but this value will surely change in the future. New vulnerabilities are always being discovered and exploited. Patch releases are issued that attempt to correct old vulnerabilities without introducing new ones. And the timings of all these events – i.e. the order in which they occur – can necessitate very different response actions by a SOC operator trying to defend their network against cyber attack. Statistical forecasts about when vulnerability-related events will occur in the future, based on past evidence of similar occurrences, can be a useful aid for planning defence strategies and resource allocation. We propose that such forecasts be in the form of probability distributions for how long before, say, the next critical patch is issued. But how does one conduct a statistically principled assessment of the accuracy of such predictions? How does one determine one forecast to be “better” than another forecast? And, is there scope for improving future predictions based on the shortcomings of past ones?

The outline of the deliverable is as follows. Section 2 defines a conceptual model that clarifies how failure diversity amongst network detection tools can positively impact the likelihood of some types of detection errors, while negatively impacting others – a trade-off. Yet another trade-off is explored in section 3 – that of how diversity can bring about a more reliable detection capability, but at the cost of an increased loss when failures occur. This is followed in section 4 with guidance on how to determine a diverse configuration that guarantees the smallest possible expected loss when used. Section 5 highlights a case-study conducted using network data from Amadeus – a DiSIEM industrial partner – and applying the ideas of previous sections. Applications of security metrics are detailed in section 6. And statistical forecasting of security-related events is outlined and implemented in section 7 along with methods for evaluating and improving forecasting accuracy. While section 8 discusses a statistical challenge presented by the sorts of correlation one would expect in typical network data. The deliverable concludes with a summary in section 9.
2 Probabilistic Model of Failure Diversity amongst Cyber-defence Configurations

In this section, we outline a conceptual probabilistic model of the process of detecting malicious activity in a network. This model abstracts the essential aspects of the scenario in Figure 1 in a form that makes it amenable to the formal analysis done in this report. In particular, the model shows the different ways in which diverse redundancy can impact the probability of correctly detecting the presence, or absence, of malicious activity.

During an ongoing attack, a cyber-defence tool – such as a firewall, IDS or Anti-virus software – detects the presence of suspicious activity by inspecting data from the network. The activities spawned by an attack generate network data. One may envisage a set, $\Omega$, consisting of all the possible network data fragments that could be generated by the activities of users of the network – both legitimate and nefarious alike. In particular, if data $\omega \in \Omega$ was generated by an attack, a cyber-defence tool can make a determination as to whether $\omega$ should be deemed evidence of malicious activity or not. If, for instance, the tool fails to raise an alarm upon analysing malicious $\omega$, then the tool will be considered to have failed in alerting the SOC operator via the SIEM.

Now, a cyber-defence tool inspects data it receives from the environment within which it operates, where both the time over which the data is generated and the content of the data are random. It is easy to see how such data-related uncertainty can arise: in general, the data is randomly generated by users – both legitimate and nefarious – as they interact with network resources. Even for legitimate users alone, the data they generate arises at random (because of uncertainty in when different users interact with network resources), the data is generated over random lengths of time (because of the uncertainty in the durations of user-interactions), and the content of the data will be a characterisation of the random behaviour/activities of users. And, similar randomly generated data comes from attackers as they carry out their attacks and actively try to ensure their illegal actions go undetected.

---

4Of course, once an ongoing attack is detected, an evaluation may be needed of the alternative ways in which to stop or mitigate the attack. We don’t focus on this extra evaluation step, since it involves identical steps to those taken in evaluating detection-efficacy.

5A more realistic characterisation of the set $\Omega$ requires more mathematical exposition than we are willing to give here. In fact, the same is true for much of the modelling exposition which follows – these models, as presented, are intended to be sufficiently insightful and relatively simple to grasp the relevant principles being discussed.

6We require that each $\omega \in \Omega$ can be identified, after sufficient forensic investigation, as uniquely originating either from benign or malicious activities on the network.

7For simplicity of exposition, we are ignoring cases where an alarm is given by the tool, correctly identifying that malicious activity is afoot, but for the wrong reason. If a tool gives an alarm when an attack is ongoing, we consider this correct behaviour.

8Although, one can imagine scenarios where user habits mean that certain behaviours are much more likely for a given user than other behaviours.
Imagine the environment within which a cyber-defence tool operates as a black-box, spewing forth data fragments (from $\Omega$) generated at random by the activities of network users, where the data is required to be analysed by cyber-defence tools for evidence of nefarious activity. Despite the uncertainty in the occurrence of this data and whether it arose from malicious activity, one might expect the tool, itself, to be deterministic in its functioning – i.e., if it declares a data fragment it receives as being of suspicious origin (or not), it will always declare that data to be of suspicious origin (or not) whenever that data is submitted to it for re-inspection.

Due to the interplay of this deterministic behaviour of the tools and the aforementioned data uncertainty, these induce uncertainty about when a tool commits detection errors, such as raising an alarm on data generated by a legitimate user. This is a *false-positive* (FP) error. Similarly, if an attack produces the data but the tool does not trigger an alarm, a *false-negative* (FN) error has occurred.

![Figure 2: The distribution of loss $L$ due to an error made by a cyber-defence tool in analysing data-samples from the network it protects. The expected loss $E[L]$ and variance $V[L]$ for this distribution are defined in eqs. (2.1) and 2.2.](image)

Each of these error-types has associated costs $l_{fp}$ and $l_{fn}$ when they occur. For simplicity, we assume that the cost associated with an error-type is always the same whenever the error occurs\(^9\). The actual value of these costs will be determined by the economics of the network being protected. Typically, there is a significant difference between the cost of a false-positive and a false-negative error. Now, suppose a data-sample is received by a tool for analysis. During operation, the tool has an associated conditional probability $q_{fn}$ of failing to raise an alarm when malicious activity is under way, and conditional probability $q_{fp}$ of raising an alarm when there is no ongoing attack. The conditional probabilities $(1 - q_{fn})$ and $(1 - q_{fp})$ are referred to as *sensitivity* and *specificity* respectively. Define the following function:

$$1_{fn} := \begin{cases} 
1 & \text{if the solution commits an FN error} \\
0 & \text{otherwise}
\end{cases}$$

\(^9\)It might help to view this as being the conditional expected loss, conditioned on the relevant error occurring.
and 1_{fp} is similarly defined for false-positive errors. With these functions, the loss resulting from a tool’s failure is the random variable \( L := l_{fn}1_{fn} + l_{fp}1_{fp} \). Let \( \gamma = P(\text{attack}) \) and \( 1 - \gamma = P(\text{no attack}) \). The distribution of \( L \) is of the form depicted in Figure 2. And the formulae for the expected loss and variation in loss, resulting from the tool incorrectly analysing data, are:

\[
E[L] := E[l_{fn}1_{fn} + l_{fp}1_{fp}] = l_{fn}E[1_{fn}] + l_{fp}E[1_{fp}] = l_{fn}q_{fn} + l_{fp}q_{fp}
\]

\[
\text{Var}[L] := \text{Var}[l_{fn}1_{fn} + l_{fp}1_{fp}] = l_{fn}^2q_{fn}(1 - q_{fn}) + l_{fp}^2q_{fp}(1 - q_{fp}) - 2l_{fn}l_{fp}q_{fn}q_{fp}
\]

Conceptually, if one considers the set \( \Omega \) of all data fragments that, potentially, could be generated by activities in a network and analysed by a cyber-defence tool, the tool induces a partition of \( \Omega \) into 4 subsets: 1) a true-positive region, 2) a true-negative region, 3) a false-positive region and 4) a false-negative region. This is depicted in Figure 3.

We may now illustrate how diverse redundancy can affect the probabilities of FP and FN errors. Consider two functionally equivalent cyber-defence tools, \( s_1 \) and \( s_2 \), that produce alarms for suspect network data. The tools have respective parameters \( 1^1_{fn}, 1^1_{fp}, q^1_{fn}, q^1_{fp} \) and \( 1^2_{fn}, 1^2_{fp}, q^2_{fn}, q^2_{fp} \). Suppose an alarm-filtering adjudication function is defined on the outputs for these tools so that an alarm is relayed by the SIEM to the operator if, and only if, both tools raise an alarm – effectively a 2-out-of-2 alarm system. An alarm raised in this fashion is a false alarm according to the indicator function \( 1^{	ext{sys}}_{fp} := 1^1_{fp}1^2_{fp} \) – it satisfies the inequality \( 1^1_{fp} \geq 1^1_{fp}1^2_{fp} \). Therefore, by definition, the probabilities of false alarms are ordered as

\[
\{(1 - \gamma)q^1_{fp}, (1 - \gamma)q^2_{fp}\} = \{E[1^1_{fp}], E[1^2_{fp}]\} \geq E[1^1_{fp}1^2_{fp}] = q^{	ext{sys}}_{fp}(1 - \gamma) \quad (2.3)
\]

That is, our choice of diverse redundancy can result in a system comprised of the two tools with an improved probability \( (1 - \gamma)q^{	ext{sys}}_{fp} \) of false alarms, compared with if only one of the tools were used instead. Given the probabilities \( q^1_{fp} \) and \( q^2_{fp} \), notice that, since both tools are required to fail for a system false positive error, there is a range \( 0 \leq q^{\text{sys}}_{fp} \leq \min\{q^1_{fp}, q^2_{fp}\} \) possible from using diversity.

There are shortcomings to using diversity in this way, however, since the indicator function for false negatives is \( 1^{	ext{sys}}_{fn} := 1^1_{fn} + 1^2_{fn} - 1^1_{fn}1^2_{fn} \). \( 1^1_{fn}1^2_{fn} \) and, analogous to (2.3), the probabilities of false negatives are ordered as

\[
\{\gamma q^1_{fn}, \gamma q^2_{fn}\} = \{E[1^1_{fn}], E[1^2_{fn}]\} \leq E[1^1_{fn} + 1^2_{fn} - 1^1_{fn}1^2_{fn}] = \gamma q^{\text{sys}}_{fn} \quad (2.4)
\]

So, the 2-out-of-2 system might possess a worse probability \( \gamma q^{\text{sys}}_{fn} \) of a false negative than the corresponding probabilities for the individual tools. Like before, given \( q^1_{fn} \) and \( q^2_{fn} \), since the false negative failure of any of the tools ensures the combined
D3.3

Figure 3: Conceptually, a cyber-defence tool induces a partition of the set $\Omega$ of all of the data that could be generated by activities in a network and analysed by the tool. The data would be randomly generated by either legitimate users (underlying green region) or malicious actors (underlying red region). When the solution raises an alarm, the detection might be correct (tp-region) or incorrect (fp-region). When no detection is made, this might either be because there is no evidence (tn-region) or evidence was missed (fn-region). Of interest are various probabilities over the depicted set $\Omega$, whose masses are not explicitly depicted. Related to the fp and fn regions are the probabilities $q_{fp}$ and $q_{fn}$, while benignly and maliciously generated data-samples occur according to $P(\text{no attack})$ and $P(\text{attack})$ respectively.

system experiences a false negative failure, there is a range $\max\{q_{fn}^1, q_{fn}^2\} \leq q_{sys}^{fn} \leq 1$ promised by our use of diverse redundancy. An illustration of a partition of the set $\Omega$ for a 2-out-of-2 alarm filtering configuration is shown in Figure 4a.

Analogous considerations show that for a 1-out-of-2 alarm system – an alarm is sent if, and only if, at least one of the tools triggers an alarm – the probability of a false negative may reduce while that of a false-positive may increase. An illustration of a partition of the set $\Omega$ for such a 1-out-of-2 alarm filtering configuration is Figure 4b.

This simple example regarding alarm filtering illustrates a general principle: diverse redundancy can be used to either combine the functionality of cyber-defence tools for improved dependability, or to combine data from various sources for improved situational awareness for an SOC operator. And, with the inclusion of more tools or data sources in any given diverse configuration comes more opportunities that diversity presents us with.
The effect of diversity on false-positive and false-negative errors. A pair of tools, $s_1$ and $s_2$, in a) a 2-out-of-2 configuration, and b) a 1-out-of-2 configuration. Notice that, compared to the subsets induced by the individual tools, the configurations are such that in a) the subset of false-positive cases is significantly smaller while that for false-negatives is significantly larger. And, in b) the subset of false-negatives is smaller while the false-positive subset is larger.

3 Loss-size vs Reliability Trade-offs amongst Diverse Cyber-defence Configurations

The expected loss $\mathbb{E}[L]$ can be a useful estimate of the level of security provided by a given configuration of cyber-defence tools. However, in deducing the implications of this estimate for the occurrence of $FP/FN$ failures, it is helpful to also consider other measures of security – such as the probability of detection devices correctly discriminating between genuine users and malicious actions. Given the losses incurred by an organisation when $FN$ and $FP$ errors occur, as well as an estimate of the expected loss $\mathbb{E}[L]$, one may ask the following questions. What can be conservatively claimed about the probability of the defence tools operating correctly? And what are the implications of such claims for choosing an appropriate configuration of defence tools? Indeed, for a cyber-defence configuration with a given estimate $\mathbb{E}[L]$ for its expected loss, there exists a range of possible discrete, 3-point, loss distributions that could characterise this cyber-defence configuration, all of which share this common value for expected loss. And, in practice, which particular one of these distributions would be preferred/undesirable as characterising the cyber-defence tool must ultimately be a judgement call that depends on the specifics of a particular situation. There is a trade-off to consider here – between how reliable the defence tools are and how small the losses are when failures occur. That is, with expected losses remaining fixed, the more reliable a cyber-defence configuration is, the more likely it is that large losses are incurred when failures occur. Arguably, faced with such a trade-off, there is usually no clear winner to pick amongst these distributions, as one is forced to make a choice between “a rock” and “a hard place”.

We can exemplify this trade-off. Figure 5 shows the 3 extremes of this range of possible loss distributions, using a normalised scale for the losses (i.e. the losses have been divided by the largest possible loss due to a failure, typically the loss
Figure 5: The 3 extremes of the range of all loss distributions that share the same expected loss $\mathbb{E}[L]$. All of the distributions in the range are defined over the normalised loss values $\{0, l_{fp}, 1\}$. In particular, the distributions in (a) and (b) above give the smallest values for the probability $\theta$ of correct detection, depending on whether $l_{fp}$ is larger or smaller than $\mathbb{E}[L]$, respectively. These also give the smallest values for the probability $q$ of an FN error. The largest values for both $\theta$ and $q$ are given by the distribution in (c).
due to an \textit{FN} error). The smallest probability \( \theta \) of correct detection is either \( \theta = \frac{l_{fp} - \mathbb{E}[L]}{l_{fp}} \), when the loss due to \textit{FP}s is larger than \( \mathbb{E}[L] \) (see Figure 5a), or it is \( \theta = 0 \) otherwise (see Figure 5b). Contrastingly, the largest value of \( \theta \) is \( \theta = 1 - \mathbb{E}[L] \), given by the loss distribution in Figure 5c.

What do these extremes represent in practice? Figure 5a represents a situation where the only failures possible are \textit{FP}s with associated loss \( l_{fp} \). While with the distribution in Figure 5b no correct detection ever occurs – only \textit{FP} and \textit{FN} failures occur with associated losses \( l_{fp} \) and 1. Of course, in practice, it is fairly easy to determine that one is not in the extreme situation of Figure 5b once, for instance, the defence tools have been observed to have correctly determined the activities of at least some legitimate network users to be benign. In contrast, determining that Figure 5a is not the situation one faces in practice can be more of a challenge, especially since \textit{FN}s due to sophisticated attacks can be difficult to diagnose. Moreover, while it seems clear that Figure 5a is to be preferred over Figure 5b, the choice between Figure 5a and Figure 5c is less clear. Of course, lest one get too excited at the possibility of the most reliable configuration that produces distribution Figure 5a, it is a sobering thought that this possibility might also be easily excluded in practice, once \textit{FP}s have been observed. Still, the usefulness of thinking in terms of these extremes is not that they, themselves, are “achievable” in practice, but that they clearly scope the range of those possibilities that \textit{are}. And, thereby, they scope the range of possible failure behaviours one has to consider in deciding which cyber-defence configuration to employ.

Here is another way to view this trade-off. Consider the, perhaps stylised, situation when \( l_{fp} = \mathbb{E}[L] \). In this case, the distributions in Figures 5a and 5b both collapse to the same deterministic function – where only failures with associated losses of size \( l_{fp} = \mathbb{E}[L] \) occur with probability 1. That is, with the system not under attack, the defence tools declare all user activity as suspicious. There is no uncertainty here, as only \textit{FN} failures will occur with accompanying losses of value \( \mathbb{E}[L] \), unlike the uncertainty of the distribution in Figure 5c, which happens to possess the largest amount of variation amongst all of these discrete distributions. In this sense, the trade-off between the extreme distributions can be viewed as exchanging the certainty of small losses (i.e. losses due to \textit{FP}s) for an increase in the reliability of the defence tools, but at the added cost of an increased probability of incurring much larger losses when failures occur (i.e. losses due to \textit{FN}s).

Of course, whether the losses due to \textit{FP}s and \textit{FN}s are, indeed, “small” and “large”, will depend on the circumstances one finds in practice. So, the reader should take our use of these monikers with a pinch of salt – they merely reflect our view that, for many cyber-security applications, the cost associated with malicious actors successfully accessing network resources tends to be significantly higher than the cost of the inconvenience to a legitimate user that is, incorrectly, denied access to the network. Despite this note of caution to the reader, nevertheless, the trade-offs we are outlining here still hold true even when “small” and “large” losses are associated \textit{FN}s and \textit{FP}s instead. Only a mere relabelling of the variables in the distributions (e.g. \( FP \leftrightarrow FN, l_{fp} \leftrightarrow l_{fn} \)) is required.
Two illustrative numerical examples are shown in Figure 6, for two distributions having the same mean $E[L] = 0.2$, but with very different standard deviations – 0.07 and 0.35 – for the (a) and (b) distributions respectively. In the language of decision theory, while a unimodal distribution like (a) might appeal to a so-called risk-averse\(^\text{10}\) SOC operator in charge of the network for a government intelligence agency – they are willing to undergo the inconvenience and cost of ruling out a large number of false alarms in order to keep out enemy agents – a “twin-peaked” distribution like (b) might appeal to the risk-tolerating SOC operator in charge of a network for whom ease-of-access by users might be of paramount importance. In going from (a) to (b), although the system becomes 25 times more reliable, it also becomes 25 times more likely to experience large losses when failures occur. In fact, with probability 1, when a failure occurs it results in large losses.

These caricatures of what a SOC operator’s preferences might be under different scenarios is meant slightly tongue-in-cheek. Of course, it might be impractical to have to rule out every false-positive if one is swamped with these, just as a false-negative probability of $\frac{1}{4}$ could be, perhaps, too much of a risk to tolerate, even if this saves legitimate users the hassle/inconvenience brought about by falsely regarding their actions as nefarious. The point is that, in general, where an operator’s preferences lie between the extremes exemplified in Figure 6 will depend greatly on the nature of the network being protected, what resources are available to support network management, and the cost of the loss events.

So far our discussion has focused on the extremes of a range of discrete, 3-point, loss distributions, remarking that the distribution Figure 5c – the distribution for the most reliable cyber-defence tool – possesses the maximum variation amongst all of these distributions. But the following stronger claim is also true. Amongst all loss distributions over any (normalised) loss values in $[0, 1]$ – where the distri-

\(^{10}\)The particular notion of risk-aversion used here is borrowed from finance/economics, to mean a simultaneous preference for smaller expected loss and smaller variation of loss [4].
Figure 7: A depiction of two dominating functions over the interval \([0, 1]\), either of which proves the form of the distribution that achieves both the largest probability of successfully executing \(n\) tests, and the largest probability of failing on all \(n\) tests. The dominating functions are the lines \(f(x) = x\) in (a) and \(f(x) = 1 - x\) in (b). So, in either case, the expected value of the red curve is smaller than the expected values of these lines. And the expected values of these lines are, in turn, equal to the expected values of the red curves if, and only if, one uses the depicted loss distributions. This unique loss distribution is also the loss distribution with the largest variance for the given value of expected loss.

Distributions all share the same expected loss \(E[L]\) – Figure 5c is the loss distribution for the most reliable cyber-defence configuration, and this is also the distribution with the largest possible variation. The proof relies on dominating functions, either \(f(x) = x\) or \(f(x) = 1 - x\), as in Figure 7. So, “increased variance” of a loss distribution becomes synonymous with “increased reliability and increased losses when failures occur”.

The trade-off only becomes more complicated when considering a choice amongst cyber-defence tools that possess loss distributions with different expected losses and variances. Suppose, for instance, that one of these tools has an associated loss distribution of the kind in Figure 5a, while another one has a distribution of the kind in Figure 5c, but with larger expected loss than the first distribution. Then this could be a situation where the first system has a smaller expected loss, but the second system is noticeably more reliable during operation, but more prone to making FN errors. Consequently, one’s preference may lie somewhere between these two configurations, if this can be achieved by suitably combining them for operation.

Attempting to simultaneously reduce expected loss and reduce the probability of failures (i.e. increase the variance) involves the use of multi-objective optimisation techniques. Using expectations and variances together in making multi-objective choices is not a new idea – Harry Markowitz and William Sharpe applied such an approach in finance to the problem of selecting “efficient” investment portfolios, for which they were awarded the 1990 Nobel prize in Economics [4–6]. What is novel here is the application of these ideas to the problem of choosing optimal cyber-defence tools and configurations.
Figure 8: Four plots – four different simulated scenarios – for which we have plotted the standard-deviation of loss distributions vs expected loss. The simulations differ in the number of cyber-defence configurations used to create convex-combinations, the values of $q_{fp}$ and $q_{fn}$ for these configurations, and the values of $P$(no attack), $l_{fp}$ and $l_{fn}$. While (a) and (b) involve 3 configurations each, (c) uses 4 configurations and (d) 5 configurations. In each case, the dark region approximately represents all of the cyber-defence configurations that lie in the convex-hull of the initially available configurations. The non-dominated configurations will be found at the vertices/“tips” of the “legs” of these regions.
One of the notable lessons from modern portfolio theory is the notion that combinations of diverse investments can lower the risk associated with the returns of the portfolio, compared with the risk of returns when making these investments in isolation. In that context, with a fixed budget to invest, a desirable portfolio is constructed as a collection of varied investments in carefully chosen proportions. Here, analogous constructions can be made out of cyber-defence solutions, resulting in a hybrid solution that has properties not wholly possessed by any single constituent solution. When used as part of a decision-theoretic framework, Gaffney et al. have even argued for this as a way of creating preferred IDSs out of unsatisfactory, available IDSs \cite{7,8}, generalising the seemingly more restricted, but still useful, observations of Provost et al. \cite{9,10}.

The receiver operating characteristic (ROC) approaches of Gaffney et al. and Provost et al. are each based on particular forms of the following idea (see Figure 9). Suppose there are \( n \) alternative cyber-defence configurations, each with their respective probabilities of false-positives, \( q_{i_{fp}} \), and probabilities of false-negatives, \( q_{i_{fn}} \). If a data fragment is randomly selected from the set \( \Omega \), let it be inspected by configuration \( i \) exclusively, with probability \( p_i \). This defines a hybrid configuration with expected loss and variance given as

\[
\mathbb{E}[L] = \sum_{i=1}^{n} \mathbb{E}[L^i] p_i
\]

\[
\mathbb{V}[L] = \sum_{i=1}^{n} \mathbb{V}[L^i] p_i^2 - 2 \mathbb{I}_{f_{n_{fp}}} \left( \sum_{i<j} \left( q_{i_{fn}} q_{j_{fp}} + q_{i_{fp}} q_{j_{fn}} \right) p_i p_j \right)
\]

where, for each \( i \), \( \mathbb{E}[L^i] \) and \( \mathbb{V}[L^i] \) are given by eqn.s (2.1) and (2.2) respectively. It might not be immediately deducible from eqns (3.1) and (3.2) what the set of those convex-hull cyber-defence solutions that are not dominated by any other solutions looks like. This is obtained as the solutions to relevant optimisation problems\footnote{For the set \( D \) of discrete distributions over \( \{1, \ldots, n\} \), one solves \( \min_{D} (\mathbb{V}[L]) \) subject to each possible value for \( \mathbb{E}[L] \).} – solutions which we illustrate in Figure 8.

The approaches of Gaffney et al. \cite{7} and Provost et al. \cite{9,11} should be viewed as complementary, stemming from the fact that the approach by Gaffney et al. is more explicit in at least two regards: 1) actions taken by an ISO – whether to act in accordance with the alerts of a cyber-defence configuration or act contrarily – allows for yet more diverse configuration possibilities, and 2) it explicitly admits that \( n \) cyber-defence tools can be combined in \( k \text{-out-of-} r \) configurations (where \( 1 \leq k \leq r \leq n \)), thereby also providing more diverse configurations for a SOC operator. That is, conceptually, the actions of the SOC operator amount to defining a new “configuration” whose outputs are a deterministic, binary aggregation of the outputs of diverse cyber-defence tools – an adjudication function. And, there
is no reason why, say $k$-out-of-$r$ configurations, cannot also simply be considered as resulting from appropriately defined adjudication functions as well. Altogether, these two adjudication possibilities merely define extra points in ROC-space – i.e. extra points in the so-called ROC convex-hull (ROCCH) approach of Provost et al. (see Figure 9).

**Figure 9:** Multi-objective optimisation for detection tools can usefully be depicted using Receiver Operating Characteristic (ROC) plots. Consider five alternative cyber-defence configurations, $a$, $b$, $c$, $d$ and $e$. The polygon $abcde$ is the convex-hull of the five vertex points, itself contained in the larger convex-hull abfge. A configuration that lies anywhere within abfge can be constructed by choosing the appropriate convex combination of the five vertex configurations. The ROC convex-hull theorem (ROCCH) states that those configurations that will have the lowest values of expected loss must lie on the boundary from $g$ to $f$ through $e$, $a$ and $b$ [9, 11]. Two dashed lines representing two different scenarios of minimum expected loss are depicted. The configurations $b$ and $e$ give the preferred choices of cyber-defence in each case.

The “optimal adjudication” algorithm – of which the approach in [7] is a special case – is discussed next in section 4.
4 Optimal Adjudication

It is rather tempting to take the popular view that the smaller the loss to be expected in employing a tool, the stronger the argument for deploying said tool on the network. And if there are multiple, diverse defence tools to hand, then the situation only gets better. Because, in principle, if trustworthy estimates of expected loss can be obtained for defence tools, then there is a way of combining the responses of multiple detection tools into responses that, when headed by a SOC operator during network operation, minimise the expected loss. A given rule that defines combinations – for each set of responses given by the tools – is a so-called adjudication function. Conceptually, minimising expected loss by using an “optimal” adjudication function is usually feasible, for the following two reasons: 1) the collective responses of the defence tools divide up Ω – the set of all possible data fragments that the tools can receive for inspection – into disjoint subsets, and 2) on each of these subsets, a response can be chosen that minimizes the expected loss, when the data fragments received by the tools specifically come from this subset. In what follows, we illustrate how this works by way of an example, using the responses from two detection devices. We then discuss the algorithm in greater generality and highlight research employing the algorithm.

Consider then, the case of only two intrusion detection systems (IDSs). We assume each of these systems gives a binary response – alert “1” or no alert “0” – upon inspecting a data fragment it receives from the network. Whatever the response turns out be, in actuality, the data fragment will either be from benign users or malicious agents, thus dividing the set Ω into the two disjoint subsets as illustrated in Figure 10. The precise content of the next data fragment inspected by the IDS from the network, whether this will constitute evidence of benign or malicious intent, and what response the IDS will give upon analysing the fragment, are all unknown before the fragment is produced on the network. Let X₁ be the response given by IDS 1 upon inspecting the next data fragment it receives at random from the network. The two events, that of no alert issued [X₁ = 0] and that of an alert [X₁ = 1], further divide Ω into more disjoint subsets. Similarly, IDS 2’s responses also split up Ω into disjoint subsets, according to the events [X₂ = 0] and [X₂ = 1]. Altogether, the pairs of responses (X₁, X₂) divide Ω into 4 notable regions – the parts of Ω that trigger the responses (1, 1), (1, 0), (0, 1) and (0, 0). In Figure 10 these 4 regions are labelled R₁₁, R₁₀, R₀₁ and R₀₀ respectively.

On each of these “R” regions – so, given a unique pair of responses from the IDSs – the IDS responses may be combined to give a binary response in one of only two ways: either issue an alert or give no alert. An adjudication function is a choice of responses on each of these 4 regions. There are 16 possible adjudication functions, f₁ … f₁₆, as outlined in table 1. So, for instance f₇(0, 1) = 1 and f₁₁(1, 1) = 0. Such a range of possibilities subsume well known architectures for

---

12This set is very large but ultimately finite. The reader may visualise any given data fragment that is analysed by a defence tool as a binary string of some maximum length. Then Ω is the set of all such strings to which the IDSs can give responses upon their analyses.
Figure 10: In this example, consider two detection tools that each give a binary response – alert “1” or no alert “0” – upon receiving data from the network. Let $\Omega$ be the set of all possible network data fragments that these tools could receive for analyses. There are two ways in which $\Omega$ may be partitioned into subsets: 1) The possible malicious and benign data fragments divide the set $\Omega$ into two disjoint subsets; 2) The pair of responses $(X_1(\omega), X_2(\omega))$ that detection tools 1 and 2 give for each data fragment $\omega \in \Omega$ also partitions $\Omega$ into 4 disjoint subsets labelled ‘$R$’. These 4 subsets are defined as $R_{i,j} := \{ \omega \in \Omega \mid (X_1(\omega), X_2(\omega)) = (i, j) \}$ where $i, j = 0, 1$. 
combining binary responses, such as \( k\)-out-of-\( n \) voting schemes. Of these adjudication functions, an “optimal” one is that which, when employed during network operation, minimises the expected loss. In this sense, optimality is with respect to how \( \Omega \) has been divided up by the responses of the tools – that is, given the 4 subsets that make up \( \Omega \), what are the 4 responses to data from these subsets that give the smallest expected loss due to these responses being wrong. And, if the number of subsets of \( \Omega \) changes due to a change in the number of tools being considered – but everything else remaining the same – then the optimal responses required changes as well.

### Table 1: The 16 adjudication functions for 2 IDSs

<table>
<thead>
<tr>
<th>( R )</th>
<th>( f_1 )</th>
<th>( f_2 )</th>
<th>( f_3 )</th>
<th>( f_4 )</th>
<th>( f_5 )</th>
<th>( f_6 )</th>
<th>( f_7 )</th>
<th>( f_8 )</th>
<th>( f_9 )</th>
<th>( f_{10} )</th>
<th>( f_{11} )</th>
<th>( f_{12} )</th>
<th>( f_{13} )</th>
<th>( f_{14} )</th>
<th>( f_{15} )</th>
<th>( f_{16} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>((1,1))</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>((1,0))</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>((0,1))</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>((0,0))</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Now which of these adjudication functions is optimal – i.e. which of these, if followed during operation, results in the smallest expected loss – will depend on two sources of uncertainty: 1) how likely it is for the next data fragment received by the IDSs to be of benign or malicious origin, and 2) upon analysing the data, what the responses of the IDSs – and, therefore, what the prescribed response by the adjudication function – will be. This uncertainty, together with the losses incurred when the adjudication function gives the wrong response, determines the expected loss as follows.

Let \( l_{fp} \) be the average loss due to a false positive (FP) error. Without loss of generality, assuming this is much smaller than the average loss due to a false negative (FN) error, consider a normalised scale of loss with \( 0 < l_{fp} < l_{fn} = 1 \).

The expected loss \( \mathbb{E}_f[L] \) resulting from using adjudication function \( f \) is

\[
\mathbb{E}_f[L] = l_{fp} \cdot P(f(X_1, X_2) = 1, \text{benign data}) + 1 \cdot P(f(X_1, X_2) = 0, \text{malicious data})
\]  

So, different forms of adjudication \( f \) have different expected loss values \( \mathbb{E}_f[L] \) according to (4.1), and those specific \( f \) with the smallest expected loss are considered optimal. To determine the optimal adjudication \( f \), one does not have to compute the expected loss for all 16 possibilities in table 1 in order to see which is smallest. Instead, one computes two expectations for each region \( "R" \) of \( \Omega \) – the expected loss if alerts are issued for data from \( R \) and this causes FPs, and
the expected loss if no alerts are issued and this causes FNs. In total, over the 4 regions, 8 expectations will have to be computed\textsuperscript{13}. And with these, the optimal adjudication will merely be a choice of which responses – for each region – give the smallest expected losses. In general, if there are \( n \) tools with binary outputs to be combined optimally, while the number of possible adjudication functions is \( 2^{2^n} \), the actual number of expectations that need to be computed in order to determine the optimal adjudication function is \( 2^n + 1 \). Admittedly, this still represents an exponential growth in the number of expectations that need to be computed. As such, computing these can be pursued while it remains practical to do so.

\[
\begin{array}{l}
| R | P(f(1, 1) = 1 \& \text{benign data}) \cdot l_{fp} \quad P(f(1, 1) = 0 \& \text{malicious data}) \\
| (1, 1) | P(f(1, 0) = 1 \& \text{benign data}) \cdot l_{fp} \quad P(f(1, 0) = 0 \& \text{malicious data}) \\
| (1, 0) | P(f(0, 1) = 1 \& \text{benign data}) \cdot l_{fp} \quad P(f(0, 1) = 0 \& \text{malicious data}) \\
| (0, 1) | P(f(0, 0) = 1 \& \text{benign data}) \cdot l_{fp} \quad P(f(0, 0) = 0 \& \text{malicious data}) \\
| (0, 0) | \\
\end{array}
\]

Table 2: The 8 expected losses for the adjudication functions in Table 1.

An optimal adjudication function is then deduced from a table such as table 2 by choosing the response, for each \( R \), that gives the smaller of the two possible expectations for the region. Incidentally, the opposite procedure gives the worst-case adjudication possible. Two numerical examples are give in table 3, both with \( l_{fp} = 0.3, l_{fn} = 1 \), and the two probability distributions in table 4 for the various \( R \) regions in Figure 10.

Notice how, for the region \( R_{1,1} \), the optimal adjudication function in scenario 1 suggests one risks FN failures because the chances of this kind of malicious data arising is very small (at approx. 0.008) compared to that of legitimate user data (at approx. 0.113). However, for scenario 2, the probabilities are roughly of the same order of magnitude (at approx. 0.07 and 0.23 respectively), and since losses due to FPs are much smaller than losses due to FNs in both scenarios, one can expect to lose more from an FN error than from an FP error in scenario 2. This reversal illustrates how optimal adjudication is dependent on how likely the various kinds of data fragment regions in Figure 10 are during operation, and the relative

\textsuperscript{13}Of course, from these 8 expectations, the expectation for any of the 16 adjudication functions can be easily computed by appropriately summing together the relevant 4 of these 8 expectations.
Table 3: Two example scenarios of Table 2 with expected losses related to each $R$ subset of $\Omega$. In each scenario, for each subset, the smallest (so, resulting from the optimal response) and largest (so, resulting from the worst response) expected losses are given. Note that these expected losses are computed using normalised losses associated with FP and FN errors, so the expected losses are all smaller than 1.

<table>
<thead>
<tr>
<th>$R$</th>
<th>scenario 1</th>
<th>scenario 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 1)</td>
<td>$3.36e-2$, worst response = 1</td>
<td>$6.781e-2$, optimal response = 1</td>
</tr>
<tr>
<td></td>
<td>$8.018e-3$, optimal response = 0</td>
<td>$7.408e-2$, worst response = 0</td>
</tr>
<tr>
<td>(1, 0)</td>
<td>$1.302e-2$, optimal response = 1</td>
<td>$2.081e-2$, optimal response = 1</td>
</tr>
<tr>
<td></td>
<td>$2.836e-1$, worst response = 0</td>
<td>$3.248e-2$, worst response = 0</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>$6.436e-2$, worst response = 1</td>
<td>$9.374e-2$, worst response = 1</td>
</tr>
<tr>
<td></td>
<td>$4.696e-2$, optimal response = 0</td>
<td>$8.729e-2$, optimal response = 0</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>$5.865e-2$, optimal response = 1</td>
<td>$1.553e-2$, optimal response = 1</td>
</tr>
<tr>
<td></td>
<td>$9.599e-2$, worst response = 0</td>
<td>$1.465e-2$, worst response = 0</td>
</tr>
</tbody>
</table>

Table 4: Two probability distributions, used respectively to compute the expectations in Table 3 of the various regions in Figure 10.

<table>
<thead>
<tr>
<th>$R$</th>
<th>scenario 1</th>
<th>scenario 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 1)</td>
<td>$P(1, 1$, benign data) $0.112005$</td>
<td>$0.226038$</td>
</tr>
<tr>
<td></td>
<td>$P(1, 1$, malicious data) $0.008017$</td>
<td>$0.074076$</td>
</tr>
<tr>
<td>(1, 0)</td>
<td>$P(1, 0$, benign data) $0.043412$</td>
<td>$0.069367$</td>
</tr>
<tr>
<td></td>
<td>$P(1, 0$, malicious data) $0.283563$</td>
<td>$0.032477$</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>$P(0, 1$, benign data) $0.214545$</td>
<td>$0.312464$</td>
</tr>
<tr>
<td></td>
<td>$P(0, 1$, malicious data) $0.046961$</td>
<td>$0.087287$</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>$P(0, 0$, benign data) $0.195507$</td>
<td>$0.051752$</td>
</tr>
<tr>
<td></td>
<td>$P(0, 0$, malicious data) $0.095986$</td>
<td>$0.146535$</td>
</tr>
</tbody>
</table>
sizes of losses from FPs vs FNs.

In a natural way, these ideas can be extended beyond tools that give binary responses to tools that can give any of a finite number of distinct responses upon analysing network data. Then the collective responses of such defence tools on each possible network data fragment enable the set of all such fragments, \( \Omega \), to be partitioned into disjoint subsets. In essence, the analogue of Figure 10 merely contains more subsets, but the considerations remain the same. And so, in principle, expected losses from FN/FP failures in each of these regions can be computed, and the optimal adjudication function deduced.

The reader may have deduced the scope for much greater generality that is being hinted at here. \textit{Any} discrete properties of the tools or system – which could be their responses to data inputs, or some discrete characterisation of their operational states, or histories over time of both of these properties, e.t.c. – that allow an observer to “label” a data fragment in \( \Omega \) using a binary word capable of ever greater precision, will allow for better performing optimal adjudication functions. This is because this allows for \( \Omega \) to get divided up into ever smaller chunks, allowing for ever more precise optimal responses to be deduced. And this can still be the case if the values for these properties for the different tools in a configuration are separated in time (e.g. responses to data inputs received at different times of the day/week by the different tools) and space (e.g. responses about network activities from different locations on the network, or even different networks entirely).

Of course, deducing optimal adjudication may not be the end of the story, if the trade-offs of section 3 are taken into account. Is there a clear relationship between the loss distribution resulting from employing an optimal adjudication function, and the extreme loss distributions of Figure 5? For instance, are optimal configurations very reliable ones, or do they trade-off reliability in favour of making FNs very unlikely? Does optimal adjudication give the same reliability as the most reliable configuration with the same expected loss? Figure 11 shows a situation where the worst adjudication configuration is similar to the theoretical most reliable system with the same expected loss, while the optimal adjudication configuration lies in-between the most reliable and least reliable systems with the same expected loss. The situation in Figure 12 – where \( l_{fn} \) is an order of magnitude larger than \( l_{fp} \) – is similar for the worst adjudication configuration. However, there, the optimal adjudication function is now identical to the least reliable system. But how likely is it that the optimal adjudication scheme will not be the same as the most reliable system with the same expected loss? Simulation suggests in Figure 13 that, for benign-traffic heavy applications, when \( l_{fn} \) is orders of magnitude larger than \( l_{fp} \), the most reliable system can be more than two orders of magnitude more reliable than the optimal adjudication configuration with the same expected loss.

Optimal adjudication has appeared in the literature in different forms \[7, 12, 13\]. Section 5 summarizes the results of a DiSIEM case-study, showcasing the efficacy of diverse redundancy and an application of “optimal adjudication”.

29
Figure 11: Loss distributions for both the worst-case and optimal adjudication functions, compared with the extreme loss distributions (from Figure 7) for the theoretical least reliable and most reliable configurations with the same expected losses. Scenario (a), with an expected loss of 0.36 shows that the worst-case adjudication is similar to the most reliable configuration possible with the same expected loss. While scenario (b), with an expected loss of 0.24, shows the optimal adjudication loss distribution is “inbetween” the extreme loss distributions. Normalised FP/FN losses are used in both (a) and (b), with $l_{fp} = 0.3$ and $l_{fn} = 1$.

Figure 12: Scenario (a), with an expected loss of 0.67, shows an even greater similarity (compared with the similarity in Figure 11a) between the worst-case adjudication and the most reliable configuration. However, scenario (b), with its expected loss of 0.04, now shows an optimal adjudication loss distribution identical to the least reliable configuration. Normalised FP/FN losses are used in both (a) and (b), with $l_{fp} = 0.1$ and $l_{fn} = 1$. 
Figure 13: For a randomly chosen, benign-traffic heavy, distribution over the \( \mathbf{R} \) subsets of \( \Omega \), if the loss due to FPs, \( l_{fp} \), is very much smaller than that due to FNs, then (a) the worst-case adjudication gives the same reliability as the most reliable configuration with the same expected loss. However, (b) shows that the most reliable configuration is “orders of magnitude” more reliable than the corresponding optimal adjudication configuration.

5 The Efficacy of Diversity in Anomaly Detection: A Case-study

DiSIEM-related studies have been undertaken, investigating the extent to which diverse redundancy can improve network security \[14, 15\]. In this section, we detail the results of a study conducted using network data from Amadeus, one of the industrial partners of the project \[16\]. In particular, how diversity between two detection tools was used to improve the detection of malicious web-scraping.

Significant diversity in the detection results of the tools was observed in a first study. Using this diversity the tools were reconfigured – became more similar – and the amount of observed diversity in the tools was reduced in a second study, but the overall performance of both tools was significantly improved. Further investigations into the causes of diversity between the tools were done, related to the design and configuration of the tools. And “optimal adjudication” \[12, 13\] schemes that achieve the lowest cost of alerting errors were computed.

The dataset consisted of Apache HTTP Access logs for an e-commerce application provided by Amadeus. Two tools were used to detect scraping activities based on the HTTP requests: a commercial tool, and an in-house Amadeus tool called Arcane. Preliminary results suggest there is considerable diversity in the alerting behavior of these tools.

Amadeus uses a version of a commercial tool we shall refer to as CommTool, deployed in the cloud in front of the web servers of the web application to protect.
This means that all HTTP requests coming from users are first inspected by CommTool. Legitimate requests are forwarded to the web application and requests deemed from bots are blocked. CommTool (as well as many other bot detectors) use different techniques to detect scrapers, including:

- **client-side fingerprinting**: A JavaScript file is downloaded from the protected website and run on the clients browser. This script extracts many device attributes to create an accurate fingerprint of the clients system. The fingerprints are shared worldwide amongst CommTool products, creating a global database of known violator fingerprints. Bots and scrapers are detected based on session attributes, such as session length, pages per session, pages per minute, e.t.c.;

- **Javascript tests** – For suspicious user sessions, CommTool can run further client-side JavaScript tests, such as inspecting the consistency of device attributes;

- **Machine learning** – CommTool uses evolving behavioural user models based on the data collected from different domains protected;

- **custom rules** – For advanced scraping activities that are not detected using the above methods, CommTool can implement custom rules based on user device attributes. Custom rules can also be created on request by the customers of the tool (for example to monitor a particular domain of interest to the customer);

- **known violator databases** – CommTool uses a worldwide database of known violators for easy identification of bots. The known violators can be IP addresses, subnets, ISPs and countries.

Amadeus’ in-house tool is called Arcane, used to detect scraping activities. Arcane is used to monitor CommTools performance for domains which are already protected by CommTool and assess robotic activities for non-protected domains. Arcane uses only Apache HTTP access logs, and the information these contain, to detect scraping activities. HTTP access records are grouped into HTTP sessions via a unique session identifier. The unique session identifier is stored in the client browser cookies and logged in the Apache audit trails. The features are collected from these sessions are used to detect bot activity. The session features used are shown in the table below.

In the datasets analysed, each data fragment represents an HTTP request session and contains HTTP request data (useragent, method, etc.), HTTP response data (status, content-type, etc.), as well as metadata about the connection, such as duration and bytes sent and received. The application is a fairly typical electronic retail application in the travel industry. The first dataset covers a period of 5 days, from May 7th to May 12th, 2018. The second dataset was captured during a period of 6 days, from September 13th to September 19th, 2018.
5.1 An Application of Diverse Redundancy

The tools were configured into r-out-of-n (RooN) adjudication schemes, where an r number of systems, out of a total of n systems, need to raise an alert for it to be raised as an alert by the system. In particular, for n=2, 1oo2 and 2oo2 adjudication systems can be configured, which require just one or both systems (Arcane and CommTool) to raise an alert, respectively. In Table 5 is detailed the sensitivity and specificity rates for both adjudication systems. And how these compare to the sensitivity/specificity of the single systems are depicted in Figure 14.

Table 5: Adjudication Systems Sensitivity and Specificity

<table>
<thead>
<tr>
<th></th>
<th>First study</th>
<th>Second study</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1oo2</td>
<td>2oo2</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>61.07%</td>
<td>28.38%</td>
</tr>
<tr>
<td>Specificity</td>
<td>78.37%</td>
<td>99.83%</td>
</tr>
</tbody>
</table>

Figure 14: Sensitivity and specificity changes from single tools to adjudication systems in datasets from studies (1) and (2).

Clear signs of diversity benefits were observed in the first study. In a second study, using these results, the weaknesses of the tools were highlighted, and both of the tools were reconfigured to improve their detection capabilities. As a result, the detection rate of each tool improved significantly. This improvement in performance gave less room for improvements resulting from diversity – since the individual tools have already become very good. Nevertheless, some diversity in the tools’ improved behaviour was still observed. These results are also detailed in table 5 and Figure 14.

In particular, Figure 14 shows how changing from either Arcane or CommTool to a 1oo2 or 2oo2 configuration changes the observed sensitivity and specificity. The show the results of changing from a single tool to an adjudication scheme for both studies (identified in the Figure by study (1) and (2)).

The results are exactly what our conceptual model of section 2 and Figure 3 indicate should happen: changing from any system individually to a 1oo2 adjudication
scheme increases sensitivity, while lowering specificity. The inverse occurs when changing any individual system to a 2oo2 adjudication scheme. What is important is how much better, or how much worse, a diverse pair would perform in these setups, and the results in Table 5 and Figure 14 give us some indicators about this.

There are two major factors for differing alert patterns between Arcane and Comm-Tool. Firstly, the mode of operation between the two tools is substantially different from one another. CommTool works primarily on the client-side, and most of its detection capabilities are due to JavaScript tests run on the clients device. This translates to a generally faster decision as to whether a request is malicious or not but prevents CommTool from correlating between various different connections. Arcane is exclusively server-side, and thus is capable of looking and correlating between multiple different client requests. This means that Arcane can more easily detect the presence of botnets. It also means that Arcane outputs more accurate alerts for longer connections. Secondly, the use of different known violator databases leads both tools to detect different connections. This, however, is not inherent to their operation modes.

5.2 An Application of Optimal Adjudication

Table 6 shows what an optimal adjudicator would output, in regards to the first and second study, where a black cell represents an instance where an alert would be raised, and a white a cell represents an instance where an alert would not be raised.

**Table 6: Optimal Adjudication Output for First and Second Studies. Empty cell: no alert; Ticked cell: alert**

<table>
<thead>
<tr>
<th></th>
<th>None</th>
<th>Arcane only</th>
<th>CommTool only</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>First study</strong></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td><strong>Second study</strong></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Both studies possessed an attack-heavy dataset. For the first study, this meant that an optimal adjudicator always raised an alarm, regardless of the syndrome this included the situation when neither Arcane nor CommTool raised an alarm themselves. In the second study, the optimal adjudicator acted as a 1oo2 system, raising an alert if either Arcane or CommTool raised an alert.

In Table 6, a 1-to-1 value for the losses incurred from alerting or not alerting is assumed. This means accepting that FPs and FNs have the same cost associated with them. Tables 7 and 8 show how the optimal adjudication changes if the costs associated with FPs and FNs change. So, in Table 7, when valuing true positives against FNs between 100 to 1 and 33 to 1, an optimal adjudication scheme alerts if either CommTool alerts, or both CommTool and Arcane alerts, but never when only Arcane alerts. This changes when valuing true positives between 32 to 1 and
12 to 1 against false negatives, where the optimal adjudicator would act as a 1oo2 system, raising an alert if either Arcane and/or CommTool alert.

Table 7: Optimal Adjudication Output with Varying FP and FN Costs for First Study. Empty cell: no alert; Ticked cell: alert

<table>
<thead>
<tr>
<th>FP:FN ratio</th>
<th>None</th>
<th>Arcane only</th>
<th>CommTool only</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>100:1 to 33:1</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>32:1 to 12:1</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>11:1 to 1:100</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 8: Optimal Adjudication Output with Varying FP and FN Costs for Second Study. Empty cell: no alert; Ticked cell: alert

<table>
<thead>
<tr>
<th>FP:FN ratio</th>
<th>None</th>
<th>Arcane only</th>
<th>CommTool only</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>100:1 to 83:1</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>82:1 to 16:1</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>15:1 to 1:3</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>1:4 to 1:100</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

These are expanded upon in figs. 15 and 16 where the total losses incurred is shown – on the y-axis – when assuming a value of 1 for each FP or FN, multiplied by the FP to FN ratio indicated by the x-axis. This is depicted for an optimal adjudicator, as well as for Arcane, CommTool, 1oo2 and 2oo2.

Figure 15: Total error cost based on FP:FN cost ratio for first study.

Optimal adjudicator always incurs the least amount of losses, which is the expected behaviour. But this analysis quantifies how much better an optimal adjudicator would perform, for different costs due to failures. For example, when valuing the cost associated with FNs higher than that of FPs (which is usually the case for most real setups), an optimal adjudicator can achieve up to three orders of magnitude reduction in costs for the first study, and up to two orders of magnitude in the second study.
6 Security Risk Metrics

6.1 Classical Security Metrics

Deliverable 3.1 - Security Metrics and Measurements [17] focused on security metrics. We identified, in Chapter 2, a set of classical security metrics and gathered information from the industrial partners in the DiSIEM project on metrics that their security managers and operators used in their Security Operation Centers. Based on the analysis of the responses from the industrial partners, we identify some metrics candidates to be integrated in the components developed in DiSIEM. In the context of the DiSIEM project, only EDP implemented a specific dashboard to display and analyse a subset of the identified metrics. The metrics selected were the following:

- MTTR - Mean Time to Remediate (a known vulnerability and a reported incident);
- MTTRV - Mean time to resolve a vulnerability;
- MTTRI - Mean time to resolve an incident;
- Number of vulnerabilities cases by month in each severity category;
- Number of reported incidents by month;
- Number of reported incidents at each region of operation;
- Number of resolved incidents and vulnerabilities by month; and
- SHS - Security Health Score.
Including these metrics in the EDP SOC dashboard allows the operations team to have a continuously updated viewpoint over day-to-day performance measurements, as well as a global risk evaluation. With this readily available information, the SOC efforts are directed to the most prominent issues. By providing an immediate comparison with past performance, the SOC management is also alerted to any abnormal activity, acting immediately instead of only being made aware of issues at the end of the month, when the operational report is produced.

**6.2 Multi-level Risk manager**

Deliverable 3.1 [17] proposed a multi-level risk assessment model. This model was further improved and refined, and its implementation resulted in the Multi-level Risk Manager (MRM) component. The MRM was designed to enhance the security risk assessment capability of SIEM and improve its monitoring and threat detection ability. Moreover, the MRM aims at producing information concerning different levels of risk assessment and support decision-making at C-level management. The implementation of this component was not originally planned (in the proposal) and the early stage prototype demonstrator was not included in Deliverable 6.2 [18]. Nevertheless, the component was integrated in industrial context. Deliverable 6.3 [19] includes a description of the MRM component and of its demonstrator, while Deliverable 7.2 [20] describes and discusses the integration of the MRM component.
7 Statistical Models for Forecasting Security Risk

While previous sections illustrate benefits from diversity for making a network secure now, and addressed the problems of evaluating and optimally configuring cyber-defence tools for the present, this section’s focus is to effectively anticipate changes to security in the future, based on similar changes in the past. But can one forecast, in a statistically principled way, how a system’s level of security evolves over time? Naturally, the evolution of system security – specifically, by which we mean the random discovery, patching or exploitation of vulnerabilities in time – can be viewed as a sequence of security-relevant events occurring randomly over time. We say “occurring randomly” because, for a hypothetical observer of such a system/network, they do not know when the next event will actually occur, nor do they know what this event will be (i.e. whether a discovery, or a patch, or an exploit). Therefore, this event sequence is a stochastic process, comprised of events occurring according to some unknown probability law. Knowing this law enables an observer to make predictions about events they haven’t seen yet. For instance, computing the probability that the next successful breach occurs within some time “t” into the future. Or, whether the next attack will be successfully thwarted. Or, further still, the expected number of breaches to occur by some time t.

But herein lies the difficulty: for typical systems, one would not know this process law, nor would there be available to-hand enough evidence to uniquely identify it. It is quite handy, then, that the mathematical literature provides many candidate stochastic processes (with their associated probability laws) to choose from, for characterising the occurrence of security events in time – an observer “merely” needs to use their past observations to choose amongst these alternatives with some confidence\(^{14}\). To this end, our observer can take a two-stage approach, roughly speaking: first, they can whittle down the set of candidate stochastic processes provided by the literature, by using “plausibility” arguments (based on properties that our observer has gleaned from seeing past events) to exclude models. Then, from the resulting restricted collection of candidate models, they can perform statistical inference\(^{15}\) to choose amongst these alternatives. In this section of the report, we aid the “whittle down”-step by developing and using a suitable processes.

Our goal is to develop bespoke processes for forecasting the evolution of system security over time. In certain respects, this work extends processes found useful in another context – that of software reliability forecasting \([21, 22]\). The primary extension being that while the processes for software reliability forecasting involve, essentially, only 1 type of event – the occurrence of software failure due to a triggered fault during operation – our proposed process will account for the 3 aforementioned events associated with each vulnerability. The following section

\(^{14}\)“merely” is meant tongue-in-cheek here; of course, there is a lot of non-triviality to be found in making such a choice confidently!

\(^{15}\)That is, apply techniques for choosing a probability law based on past observations.
makes this extension clear, and shows how these different models relate to one another.

### 7.1 “Single Event” vs “Multiple Event” Forecasting Models

![Figure 17: Points on a timeline are a useful representation of the occurrence of security-related events as they occur. The simplest case has only one type of event to keep track of, and there are two ways of describing such a process. Before the current time $t^*$, one can either consider the “$x$” time points at which events were observed to occur, or the amount of time inbetween these time points given by the “$t$”s.](image)

*Point processes* – a very general class of stochastic process – are well-suited for modelling the evolution of system properties over time. Conceptually, imagine events as being represented by points on a timeline (see Figure 17). As time passes and progresses to the right on the timeline, new points begin to appear, representing when new events have been seen to occur. So intuitively appealing is this family of models that they have gained wide-spread application in various contexts [23–26]. In particular, they have been employed in predicting how the reliability of software grows during its use over time, as bugs in the software are found and patched. Many examples of this special class of point processes – so-called *software-reliability growth models* (SRGMs) – exist, including a particular subset of these models called *exponential order-statistics models* [27].

The underlying structure of these SRGMs is the following. A piece of software is assumed to contain a countable number, $I$, of bugs, each of which is triggered by “nature” and causes failure at a rate $\lambda_i$ (for $i \in I$) when the software is in use. Each bug is triggered independently of all other bugs. An observer watching this software in operation will see bug-triggered failures as they randomly occur in time. When a failure occurs, the bug which caused the failure is removed. The time to the first, and only, observed failure due to a bug $i$ is assumed to be $X_i \sim \text{Exp}(\lambda_i)$ – an exponentially distributed random variable with rate $\lambda_i$. The times at which these observed failures occur are then assumed to be the so-called *order statistics* of these random variables. That is, each $X_i$ is sampled (independently of the rest) according to their respective exponential distribution, and these samples are ordered from smallest to largest. This ordering is the order in which the failures from each bug are observed, and the sampled times are the times at which these failures occur – the smaller “$X$”s are the times at which their respective bugs
cause failure, and these happen before the larger “X” times at which the other bugs cause failure. So, for example, in Figure 17, the sampled exponential times are $x_3 < x_1 < x_2$, so a “bug 3” failure occurs at time $x_3$ first, followed by a “bug 1” failure at time $x_1$, and then a “bug 2” failure at time $x_2$, all before current time $t^*$. This is the basic underlying model for a wide class of SRGMs.

This basic model consists of a single kind of event – that of a software failure due to some bug. With suitable caveats, this is sufficient for forecasting how software reliability tends to improve over time. But perhaps not for forecasting how secure a system will be because, over time, more than one kind of event impacts system security, as vulnerabilities in the system are discovered, hopefully patched, and often exploited by malicious actors. So, for forecasting future security, one needs more – a stochastic process with multiple event types. It is convenient therefore, that there is a natural way of extending the basic “order-statistics” approach used in SRGMs, and thereby define a point process that caters for multiple event types.

**Figure 18:** A configuration of cyber-defence tools contains $n$ vulnerabilities. Over time, each vulnerability is discovered and subsequently patched or exploited, and these events form a random sequence in time. In this example, there are 7 events that have been seen before the present time $t^*$, and all of the other analogous events are still to be seen. Notice that in this process each vulnerability is discovered before it is either patched or exploited. So, for instance, vulnerability “$n$” is discovered at time $x_{n1}$, patched at $x_{n2}$, and exploited at $x_{n3}$, where $x_{n1} < x_{n2} < x_{n3}$.

To see how to do this, first consider the following simple case, of a cyber-defence tool that contains a single undiscovered vulnerability. At some unknown point in time $T_v$ in the future, the vulnerability will be discovered. We assume that this random time $T_v$ is distributed according to some density $f_v(t_v)$. Furthermore, once the vulnerability has been discovered at some time $t_v$, then a race begins for which of the next two possible events will occur first – at some further unknown time $T_\pi$ a patch will be developed, and at some unknown time $T_\epsilon$ an exploit will be developed. These times are assumed to follow density functions $f_\pi(t_\pi)$ and $f_\epsilon(t_\epsilon)$, respectively. These times $T_\pi$ and $T_\epsilon$ are assumed statistically independent of each other, given the time of the discovery of their associated vulnerability.

These times “$T$” are times inbetween events. To show how similar this model is to the SRGM model, it will be conceptually more appealing to reason in terms of the time from the start of the observation period up until when the vulnerability was
either discovered, patched or exploited. That is, we reason in terms of random times $X_1, X_2, X_3$ defined by

<table>
<thead>
<tr>
<th>Times for vulnerability discovery, patching and exploitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1 = T_v$</td>
</tr>
<tr>
<td>$X_2 = T_v + T_\pi$</td>
</tr>
<tr>
<td>$X_3 = T_v + T_\epsilon$</td>
</tr>
</tbody>
</table>

These “$X$” random times are very similar to the random times of the basic SRGM model in Figure 17 with one notable difference – by definition (7.1), we must have $X_1 < X_2, X_3$. So, in this model, vulnerabilities are discovered before they are either patched or exploited.

![Figure 19: A hierarchy of useful stochastic processes for modelling the occurrence of events in time.](image)

With this ordered triplet of event-times for each vulnerability, one then defines a finite collection of such triplets $\{X_{i1}, X_{i2}, X_{i3}\}$ for $i \in I$ – where each triplet is statistically independent of the rest – and one considers the order-statistics of this collection to be a point process that generates triplets of events for each vulnerability in time (see Figure 18). That is, all of these random “$X$” times are sampled according to their respective probability distributions, and then ordered. The resulting ordering, for instance if the samples turn out to be such that $x_{n1} < x_{11} < x_{n2} < x_{31} \ldots$ and so on, is the order and times at which the 3 types of events for each vulnerability occur. This is our multiple-event point process for modelling vulnerability-related events. We shall refer to this as a vulnerability process. The hierarchy of point processes we have outlined so far is depicted in Figure 19.

In passing, we remark that this choice of point process for forecasting — based on an order-statistics model — is a considered choice for two reasons: 1) order-statistics models have been successfully used to characterise other point-processes in similar contexts (e.g. software reliability growth) and a number of mature statistical techniques exist for their analyses, 2) order-statistics models have been shown to be sufficiently general in capturing stochastic models that possess the desirable Markov property\(^{16}\). In the next section, we demonstrate fitting this vulnerability process to data and obtaining forecast distributions.

\(^{16}\)The property that uncertainty about future events in a random process is only dependent on the current state of the process, and independent of all past states.
7.2 Model Fitting via Non-parametric Statistical Inference

Using the vulnerability process we have defined, we now illustrate the fitting of this model to vulnerability data. For the inference, we use non-parametric, maximum likelihood estimation (MLE). This was implemented using the R statistical modelling language [31]. We also show how the predictive accuracy of a sequence of model fits is evaluated and improved, using rather general statistical techniques.

The goal of inference is to use publicly available data, consisting of the 3 kinds of vulnerability-related events, to determine 3 distributions according to which each vulnerability’s associated event times $T_v, T_\pi, T_\epsilon$ are distributed. All of the vulnerabilities share the same triplet of distributions for their associated event times. These distributions will be approximated as kernel density estimates, where we have elected to use Gaussian kernels as our choice of non-parametric models. That is, we seek kernel density estimates $(\hat{f}_v, \hat{f}_\pi, \hat{f}_\epsilon)$ based on historical observations. In this sense, one regards the inference as being non-parametric since, in a well-defined approximate sense, all continuous distributions consistent with the historical data are considered. The density estimates, given the history of the process up to the present $t^*$, are as follows (for any $h_v, h_\pi, h_\epsilon > 0$):

\[
\hat{f}_v(t) = \frac{1}{h_v} \sum_{j=1}^{n} K\left(\frac{t - x_{j1}}{h_v}\right) \frac{1_{x_{j1} < t^*}}{\left(\sum_{i=1}^{n} 1_{x_{i1} < t^*}\right)}
\]

\[
\hat{f}_\pi(t) = \frac{1}{h_\pi} \sum_{j=1}^{n} K\left(\frac{t - x_{j2} + x_{j1}}{h_\pi}\right) \frac{1_{x_{j2} < t^*}}{\left(\sum_{i=1}^{n} 1_{x_{i2} < t^*}\right)}
\]

\[
\hat{f}_\epsilon(t) = \frac{1}{h_\epsilon} \sum_{j=1}^{n} K\left(\frac{t - x_{j3} + x_{j1}}{h_\epsilon}\right) \frac{1_{x_{j3} < t^*}}{\left(\sum_{i=1}^{n} 1_{x_{i3} < t^*}\right)}
\]

(7.2)

where the symbol $1_S$ equals 1 when the predicate $S$ is true, and equals 0 otherwise, and we have used Gaussian kernel $K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$.

Now, the triplets (7.2) change with varying “$h$”s so that, given the historical event times $x_{11} < \ldots < x_{kl}$, MLE determines a triplet $(h^*_v, h^*_\pi, h^*_\epsilon)$ for an optimal triplet of “$f$”s – these “$f$”s give the largest value of the likelihood function for our vulnerability process. MLE was performed numerically, using a variant of the simulated annealing algorithm.

Once optimal $(h^*_v, h^*_\pi, h^*_\epsilon)$ have been obtained for the distributions of $T_v, T_\pi, T_\epsilon$, Monte-Carlo simulation techniques are used to obtain distributions of the corresponding times $X_v, X_\pi, X_\epsilon$ (see (7.1)), and thereby produce conditional distributions (that are conditional on the events seen thus far) for how long it will be

\[42\]
before the next patch, or the next exploit, is observed. To test the robustness of MLE fits of the vulnerability processes, we generated test-data from 3 simulated point processes of increasing sophistication, in each case using a predetermined number of vulnerabilities. These generative processes are:

1. **order-statistics of exponentially distributed times**: triplets of exponentially distributed times are sampled in accordance with the statistical independence structure of our vulnerability point process, and the order-statistics of these constitute dataset “a” for the MLE fit;

2. **order-statistics of log-normal times with uniformly distributed scaling**: log-normally distributed times are independently sampled and the order-statistics of these are obtained, representing data about when vulnerabilities have been discovered. For each of these discovery times, two uniformly distributed fractions are used to rescale the discovery time. The resulting pair of rescaled times represent the times after discovery when a vulnerability is patched and exploited. Altogether, these triplets of randomly generated times for each vulnerability constitute dataset “b” for the MLE fit;

3. **order-statistics of gamma, log-normal and exponentially distributed times**: triplets of rimes are sampled (without replacement) from the order-statistics of independently sampled gamma, lognormal and exponentially distributed times. These triplets constitute the times at which each vulnerability is discovered, patched and exploited – dataset “c” for the MLE fit.

The parameters for these test-data generating processes are given in Table 9 along with the resulting “h” parameters from the MLE fits in each case. In the
next section, we outline how to evaluate, and possibly improve, the predictive accuracy of forecasts using these MLE fits.

7.3 Evaluating and Improving Forecast Accuracy: U-plots and Recalibration

The true test of a statistical model of the future is in how well it predicts future uncertainty. From the MLE fits of our vulnerability process, based on past observations of security-related events, we can produce a sequence of conditional distributions of how long one would have to wait before observing the next patch or exploit event. One evaluates how good these forecast distributions are by, essentially, analysing how good they have been so far – that is, goodness is judged by analysing the statistical performance of these distributions in forecasting events that have been observed so far. A well-defined separation of past observations from future ones keeps this evaluation from being unsound and circular (i.e. one does not “fit” and “evaluate” a statistical model using the same data). We will make this separation clear.

A very useful, non-parametric technique for evaluating predictive accuracy is the U-plot. It is based on this theorem by Rosenblatt [33]: for any finite collection of continuous random variables $X_1, \ldots, X_n$, they jointly satisfy

$$ (F(X_1), F_{X_1}(X_2), \ldots, F_{X_1,\ldots,X_{n-1}}(X_n)) \sim U[0,1] \times \ldots \times U[0,1] $$

(7.3)

where, for $j = 1, \ldots, n$, we have the sequence

$$ F_{X_1,\ldots,X_{j-1}}(t) := P(X_j \leq t \mid X_1 = x_1, \ldots, X_{j-1} = x_{j-1}) $$

of conditional cumulative distribution functions (CDFs) for a sequence of past observed events $x_1, \ldots, x_n$. This theorem says that these conditional CDFs, when regarded as functions of their respective random variables $X_1, \ldots, X_n$, are really a collection of independent, uniformly distributed (on the interval $[0,1]$) random variables. Consequently, we can define the following necessary statistical criterion for what a good sequence of forecasts should look like. Suppose one has a statistical procedure that generates a sequence of fitted forecast CDFs, say $\hat{F}(t), \hat{F}_{x_1}(t), \ldots, \hat{F}_{x_1,\ldots,x_{n-1}}(t)$, from an increasing sequence of observations – the most recent of these being $x_1, \ldots, x_n$. If these fitted CDFs are close to the true unknown CDFs for the observations, then the $\hat{u}$-variates defined as

$$ \hat{u}_1 := \hat{F}(x_1), \hat{u}_2 := \hat{F}_{x_1}(x_2), \ldots, \hat{u}_n := \hat{F}_{x_1,\ldots,x_{n-1}}(x_n) $$

(7.4)

will be close to being $n$ independent samples from the uniform distribution $U[0,1]$. That is, a plot of the empirical CDF of these $\hat{u}$-variates will consist of points close to the 45°-line in the unit square (see Figure 20). This is a U-plot.
forecasted times are less likely than they should be

\[ \sum_j^n 1_{\hat{u}_j \leq u} \]

forecasted times are more likely than they should be

**Figure 20:** A U-plot is the plot of the empirical CDF of the \( \hat{u} \)-variates defined by (7.4). The empirical CDF is written in terms of the symbol \( 1_{\hat{u}_j \leq u} \), which equals 1 if the value of \( \hat{u}_j \) is not greater than \( u \), and equals 0 if it is. If the empirical CDF does not lie close to the diagonal, then the nature of its deviations from the diagonal can indicate systematic bias in the sequence of forecast CDFs being evaluated. Two types of deviation are depicted here: the case when the CDFs make the forecasted times more likely than they should be (when the points lie below the diagonal), or less likely than they should be (when the points lie above the diagonal).

Notice how, in definition (7.4) of, say, the \( \hat{u}_n \) variate, only the observations up until the \((n-1)\)th observation are used to fit the model, the resulting fit being the \( n \)th forecast CDF that is \( \hat{F}_{x_1,\ldots,x_{n-1}}(t) \). This makes the value of \( \hat{u}_n \) exactly what one would expect it to be if only the events seen “so far”, i.e. \( x_1,\ldots,x_{n-1} \), are used to fit a forecast CDF \( \hat{F}_{x_1,\ldots,x_{n-1}}(t) \). And the earliest time at which this forecast can be evaluated, i.e. the earliest time when \( \hat{u}_n \) can be computed, is when the next event \( x_n \) is eventually observed, and not before. This is why properly constructed U-plots of \( \hat{u} \)-variates are statistically principled and not circular. There is a clean separation between data for fitting the “\( \hat{F} \)”s and data for evaluating them.

The primary advantage of using Rosenblatt’s theorem for evaluation is the fact that only very weak assumptions about the true, unknown probability law being sought are needed. This is in contrast with other goodness-of-fit tests that require statistics from observations to satisfy certain properties e.g. assumptions of “normality” or statistical independence between the observations. Here, the generality of the theorem is also what makes it a very practical result.

For many years, U-plots have been successfully applied to evaluating the accuracy of predictions about how the reliability of software changes over time [34, 36]. The fact that using this technique does not require strong assumptions to hold in practice makes its use very appealing indeed. And its applicability to assessing our fitted vulnerability process is immediate, without any modifications needed to account for the “multiple events” in our model, which are absent from the “single
Figure 21: Recalibration uses the U-plot from a sequence of U-variates (e.g. those variates that give the points below the diagonal in Figure 20) to define a smooth, monotonic function $\text{Re}(u)$ that passes through all of the points in the U-plot. This function, when composed with a fitted CDF $\hat{F}(t)$, produces both a new CDF $\text{Re}(\hat{F}(t))$ and new U-variates (based on past data) that have a 45° U-plot.

Rosenblatt’s theorem also offers the following possibility for improving forecasts. Since a biased U-plot indicates how forecasts fall short of producing $\hat{u}$-variates that “look” uniformly distributed, the U-plot also approximately defines a function that can transforms these biased variates into new values that, then, produce a U-plot which lies on the diagonal. Conveniently, the relevant transforming function is precisely some continuous version of the U-plot itself – that is, a continuous version of the $\hat{u}$-variates’ empirical CDF, $\sum_{i=1}^{n} \mathbb{1}_{u_i \leq u} / n$. Let’s denote this function as $\text{Re}(u)$. This function is not unique – it only needs to be monotonic and pass through all of the points of the empirical CDF. However, once such an $\text{Re}(u)$ is defined, the new variates $\text{Re}(\hat{u}_1), \text{Re}(\hat{u}_2), \ldots, \text{Re}(\hat{u}_n)$ will approximately lie on the 45° diagonal. So, if the bias observed at the present time were to continue into the future – i.e. if the bias is systematic – then the adjustments made by the “Re” function can continue to keep new $\hat{u}$-variates looking like independent uniform samples. So, for subsequent forecasts after the “$n$”-th, one uses transformed conditional forecast CDFs, such as $\text{Re}(F_{x_1, \ldots, x_n+1}(t))$, as the forecast distribution. This procedure – of using the U-plot to define a suitable function that transforms our forecasts into better ones – is termed recalibration, and it is yet another technique that has proven useful in the software reliability context, specifically for improving forecasting performance. Of course, there is no need to assume that recalibration, using a given transform function “Re” that works, will continue to improve forecasts – one should continually use the technology of U-plots to evaluate all subsequent predictions, including the “recalibrated” ones! And, like it was for U-plots, the application of recalibration to improving forecasts from our “multiple event” processes is as straightforward as it is in the “single event” situations of software reliability prediction.

We now exemplify these ideas with a series of U-plots and recalibrated forecast CDFs in Figure 22, Figure 23 and Figure 24. In addition to the generated datasets
a, b and c, we have applied the model to publicly available vulnerability data for the Windows 10 platform, spanning almost a decade’s worth of patch releases and vulnerability exploits. Without recalibration, the vulnerability process produces largely pessimistic predictions for the time till the next patch release – for the most part, patch releases are predicted to take longer than they do. While exploits are less likely than they should be to take less than a day to be the next event to occur, but then are more likely than they should be to occur for times longer than a day. This symmetry surrounding a day is not surprising since, in the Windows data, announcements of exploits are correlated quite closely in time with announcements of vulnerabilities or patch releases – within a day or so (see Figure 24).

These ideas, for evaluating and improving forecast accuracy, and more, have been implemented as part of the diversity forecasting and analytics platform in DiSIEM, with some related studies in [37, 38].

Figure 22: Each column in this figure contains U-plots (above) and fitted CDF Forecasts (below), based on the 3 generated datasets, for the time taken (in days) for the next vulnerability to be patched. The points on the 45°-line in the U-plots are the recalibrated U-variates, and the black CDFs are the corresponding recalibrated fitted CDFs. The vulnerability process produces extremely good fits for both datasets a and b, with some optimistic bias beginning to show for b. However, the U-plot for c shows clear optimistic bias before recalibration.
Figure 23: U-plots and fitted CDFs for the time taken (in days) for the next vulnerability to be exploited. Again, the vulnerability process produces an extremely good fit for dataset a, with some optimistic bias for b. And again, the U-plot for c shows clear optimistic bias before recalibration. That the plots for the “patch” and “exploit” times are very similar, for both datasets b and c, is no surprise – these datasets were produced by processes that have similarly distributed times to next patch and times to next exploit. What is noteworthy is that the vulnerability process fits picked this symmetry up, despite the optimistic biases.
Figure 24: The vulnerability process is fitted to publicly available Windows 10 data spanning almost a decade, since 2010. The model fit has parameters $(h_v, h_\pi, h_\epsilon) = (-0.446, -1.906, 1.019)$. The forecasts (in days), for time-to-next-patch, are evaluated/recalibrated in (a), and those for time-to-next-exploit in (b). In both cases, U-plots show biases above and below the diagonal – small times are too unlikely and large times are too likely. Recalibration is used to mitigate these.
8 Statistical Dependence in Network Data: Implications for Security Assessment

This section reflects on a difficulty that stems from statistical dependence typically present in network data — a difficulty that one must face when attempting a statistical assessment of network security. We also give some proposals for how this difficulty might be addressed in future work.

To understand the underlying problem, suppose one attempts to estimate the probability, $\theta$, that a defence tool will correctly discriminate the next data fragment it receives for evaluation. And a proposed way of doing this would be to use “classical” statistical arguments. So, we might imagine the tool as receiving a sequence of data fragments $\omega_1, \ldots, \omega_n \in \Omega$ from the set of possible data fragments that the tool could evaluate. Ideally, this sequence is actual network traffic that the tool is required to evaluate. So the data is being produced randomly (i.e. one does not necessarily know what the next fragment of data will be) according to the uncertainty of ongoing network activities. On this sequence, the tool gives a sequence of responses $X(\omega_1), \ldots, X(\omega_n)$. These are either correct responses or not — one might require that the average number of correct responses, denoted $\hat{\theta}$, becomes a very good estimate for the unknown $\theta$ as $n$ becomes very large. That is, we define

\[
\hat{\theta} := \frac{1}{n} \sum_{i=1}^{n} 1_{X(\omega_i) \text{ is correct}}
\]  

(8.1)

But, to guarantee that $\hat{\theta}$ adequately estimates $\theta$ requires suitable statistical arguments. And this is where some difficulties arise. If the data sequence is statistically “nice” — for instance, if this is a sequence of independent and identically distributed (i.i.d.) samples — then a number of so-called “law of large numbers” results guarantee that $\hat{\theta}$ can be a good estimate when using a large enough data sample. However, it is unlikely that these data sequences will be comprised solely of statistically independent samples. We expect the activities of network users, as they produce such data sequences, to also produce correlations within these sequences. These correlations can be quite complicated, ranging over short and long periods of time. We expect this to be particularly true of malicious data produced in a cyber attack. So, data sequences are unlikely to exhibit an i.i.d. property, making it difficult to rely on our estimate $\hat{\theta}$ being statistically properly defined. Perhaps, at the proper level of granularity for the data and over varying, appropriate time-scales, one can verify that the i.i.d. property approximately holds. But even if this was done, it will only give guarantees for the estimate to really be an estimate of the probability of failure on the next data sample if the next data sample was also i.i.d., and we generally don’t expect this to be the case.
A step towards addressing this difficulty could be to try a Bayesian statistical approach, rather than the “classical” one just outlined. To use a Bayesian approach, one would have to characterise their beliefs about what the loss distribution (see Figure 2) really looks like. There are many possible values for the probabilities of FP and FN errors – let's label these probabilities \( P \) and \( Q \), respectively – and some of these will be more likely than others, based on the evidence one has. The technical requirement is that one needs to define a prior distribution over all possible loss distributions. This may be hard to do, and by analysing previous network traffic and using external sources of data (e.g., open-source intelligence and publicly available security reporting), one might form only some prior beliefs about how big the probabilities of FP and FN errors probably are. While this falls short of defining beliefs about all possible values of \( P \) and \( Q \), it is possible to use this limited information to conservatively bound the probability of success \( \Theta \) (where \( \Theta = 1 - P - Q \)) by testing the tool on the data sequence \( \omega_1, \ldots, \omega_n \). To be conservative, we need to find a prior distribution for the probabilities \( P, Q \) and \( \Theta \) that: 1) satisfies our limited, expressed beliefs about these probabilities, and 2) using the results of how the tool performed on the data sequence, it gives the biggest confidence that the probability of success is not very high.

So, for instance, suppose various sources of evidence support beliefs that the FP probability is no smaller than \( 10^{-3} \) with some confidence 0.9, the FN probability is no smaller than \( 10^{-7} \) with some confidence 0.65, and the probability of the tool correctly discriminating the data is no smaller than \( 10^{-5} \) with some confidence 0.6. Formally, we write these as \( P(P \geq 10^{-3}) = 0.9 \) for FPs, \( P(Q \geq 10^{-7}) = 0.65 \) for FNs, and \( P(\Theta \geq 10^{-5}) = 0.6 \) for correct responses. Furthermore, suppose that in analysing the \( n \) data fragments, the tool commits \( k_1 \) many FP errors and \( k_2 \) many FN errors. Then, to obtain conservative estimates for \( \Theta \) based on this testing evidence – such as how confident we can be that \( \Theta \) is smaller than \( 1 - 10^{-6} \) – we solve the following optimisation problem:

\[
\begin{align*}
\text{maximise} & \quad P(\Theta \leq 1 - 10^{-6} | n \text{ tests}, k_1 \text{ many FP errors}, k_2 \text{ many FN errors}) \\
\text{subject to} & \quad P(P \geq 10^{-3}) = 0.9, \\
& \quad P(Q \geq 10^{-7}) = 0.65, \\
& \quad P(\Theta \geq 10^{-5}) = 0.6,
\end{align*}
\]

where \( \mathcal{D} \) is the set of all possible prior distributions over values for the triplet of probabilities \( (P, Q, \Theta) \). Figure 25 illustrates the constraints over the possible values of \( (P, Q, \Theta) \) that any consistent prior distribution must satisfy. This is a more general form of conservative Bayesian inference – an assessment approach that has been successfully applied elsewhere, such as in addressing challenges faced when assessing the dependability of autonomous vehicles [39].

Although this is a welcomed first step, more is required. Because, in solving the optimisation problem above, one might do what is typically done in applications of Bayesian inference and still appeal to an i.i.d. structure for the test data sequence. But, at least now, one is computing estimates that are the most conservative one can have, given the operational evidence and this i.i.d. assumption.
Figure 25: All candidate prior joint distributions over values for \((P,Q,\Theta)\) – i.e. distributions over the probabilities of FP, FN and correct response, respectively – are all constrained in how they allocate probabilities over the triangular region in the figure for feasible values of \(P\) and \(Q\) (note that \((P,Q,\Theta)\) are probabilities, and so must sum to 1). In particular, the prior distributions must allocate the probability masses \(M_1 = 0.9\), \(M_2 = 0.65\) and \(M_3 = 0.6\) related to the triangular subset consisting of the disjoint regions \(R_1\), \(R_2\) and \(R_3\). The masses are not allocated, respectively, to these regions, but instead to the triangular union of these regions.

Perhaps a better further step is the following. Begin by acknowledging that, for each data fragment \(\omega_i\), the nature of the response \(X(\omega_i)\) – i.e. whether this is an FP, FN or correct response – is governed by unknown probabilities \(P_i, Q_i, \Theta_i\), where these probabilities depend on how the tool has performed on all previous data fragment in the sequence. Then, in principle, our optimisation problem to obtain conservative assessments looks the same, except \(D\) now becomes the set of prior distributions over all possible values for the vector \((P_1, Q_1, \Theta_1, \ldots, P_n, Q_n, \Theta_n)\). Of course, this optimisation problem is significantly more involved than the original version, partly because it involves specifying beliefs – even if it is only some beliefs – about values of the vector in a \(3n\)-dimensional hypercube (where \(n\) can get very large indeed). Future work will focus on exploring the extent to which this optimisation is feasible in practice.
9 Conclusions

Following on from deliverable D3.2 [1], this deliverable details research and guidance aimed at effectively using diverse redundancy to improve the security of a SIEM-managed IT network. By way of probabilistic modelling and statistical evaluation approaches, we illustrate the benefits that diversity can bring, and show how to achieve this by a clear understanding of the inherent trade-offs when employing diversity.

Over the course of this report we clarify how and why diversity can be helpful, and show a range of alternative diverse configurations, detailing how one may reason about the pros and cons of these. The existence of a diverse configuration that gives the smallest expected loss is guaranteed by the “optimal adjudication” algorithm. While in principle it is always possible to determine an optimal adjudication function, in practice this is feasible as long as the number of alternative diverse configurations is not exceedingly large, so as to allow for a restricted, combinatorial search for the best alternative.

These results are exemplified by a number of examples and demonstrated in a case-study using a real-world airline booking application. The benefits from diversity and optimal adjudication are clearly shown, and are quite significant in some cases. More studies beyond those in the DiSIEM project – studies into the efficacy of diverse redundancy for improving network security – have been carried out [40, 41]. And further work generalising, say, the “optimal adjudication” algorithm to apply to more situations with constantly evolving defence tools (e.g. tools based on AI/machine-learning algorithms that continually update) is ongoing.

Just as important as evaluating present security levels is effectively anticipating future ones, and we demonstrate statistical approaches to constructing, objectively evaluating, and possibly improving, forecasts of future security-related events, using data on similar past events. We illustrate these ideas using approximately a decades worth of publicly available data on vulnerabilities for the Windows 10 platform. These ideas, and more, have been implemented as part of the diversity forecasting and analytics platform in DiSIEM, with some related studies in [37, 38]. The deliverable concludes with some closing considerations on challenges to the statistical assessment of security, posed by correlations in network data. In particular, we indicate how Bayesian approaches (shown to be successful in other contexts [39]) might be useful in mitigating these problems – the feasibility of these will be investigated in future work.
References


detection systems: An empirical study. In 2017 IEEE 16th International