Investigating Episode Prioritisation in Alert-Driven Attack Graphs

Analysing PICA: A Novel Approach to Episode Prioritisation

Senne Van den Broeck

Supervisor(s): Sicco Verwer, Azqa Nadeem

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Senne Van den Broeck
Final project course: CSE3000 Research Project
Thesis committee: Sicco Verwer, Azqa Nadeem, Asterios Katsifodimos

Abstract

Intrusion Detection Systems (IDSes) detect malicious traffic in computer networks and generate a large volume of alerts, which cannot be processed manually. SAGE is a deterministic algorithm that works without *a priori* network/expert knowledge and can compress these alerts into attack graphs (AGs), modelling intruders' paths in the network. These AGs are too high in quantity/complexity for manual analysis, creating the necessity for prioritising individual attack stages (ASes). The existing prioritisation metric does not take into account graph properties and is not granular enough to function on a node level. We propose PICA, an urgency metric inspired by the CIA triad (Confidentiality, Integrity and Availability) and the graph properties. It works on a node level and an attack-stage level. PICA is evaluated by comparison with the current implementation, based on AGs generated by SAGE using open-source intrusion alert datasets. The evaluation is based on the number and the type of the discovered attack stages. Results show that PICA manages to discover ASes that contain nodes with a high in-degree but fails at discovering urgent ASes that contain many nodes with low in-degrees. Compared to the baseline, the ASes are distributed more evenly over the different urgency levels. Analysis of urgent node positioning revealed that sub-AGs lose information when objectives (final goal in a path) are also starting nodes. Changing the weights of the CIA triad showed a clear bias in results towards the larger weights, as was intended. Finally, further work is proposed for PICA and in the generation process of SAGE's AGs.

Index Terms: SAGE, Attack Graphs, Urgency, Prioritisation, Network Security

1 Introduction

Daily, Security Operations Centres (SOCs) receive over **one million alerts per day**. These alerts are generated by an Intrusion Detection System (IDS), and often the volume of alerts outweighs the capacity of the Security Analysts, leading to a process called 'alert fatigue' or information overload [1]. From these intrusion alerts, an attack graph (AG) can be created, which models the path and strategy the intruder takes to reach their objective. Creating these graphs requires network knowledge and is time-consuming [2]. Nadeem *et al.* developed SAGE (intruSion alert-driven Attack Graph Extractor), which can compress thousands of alerts into "alert-driven" attack graphs without pre-existing knowledge about the network and its vulnerabilities [3]. Although SAGE can heavily reduce the number of alerts into AGs, the quantity and size of the graphs are still too large for manual inspection.

Nadeem *et al.* proposed a dashboard to facilitate AG exploration and make the graphs more responsive and interactive [4]. This dashboard offers a recommender matrix which shows the different attack stages (ASes) and their urgency for a given dataset of alerts. The attack stages are the different types of performed attacks, e.g. *Network DoS*. This matrix can assist Security Analysts in finding which ASes in the attack graphs they should inspect. The current metric for the matrix is based on node prevalence and severity.

The current urgency metric disregards properties of the graphs generated by SAGE and is not very granular/customisable for Security Analysts. An example of this is Figure 1, where even though the ASes *Network DoS* and *Data Exfiltration* have similar urgency, one is noticeably more urgent based on the number of paths crossing it. The goal of this research is to propose an alternative metric for tackling these issues and to compare it to the current implementation. We propose PICA (Paths, Integrity, Confidentiality and Availability), inspired by the CIA triad (Confidentiality, Integrity and Availability) [5]. It consists of a path factor and CIA scores based on the action-intent framework [6] and severity scores used in SAGE.

The goal of PICA is to be a more intuitive metric that allows for more customisation by the Security Analyst, as they can have varying preferences when it comes to urgency [7][8]. One company might put importance on e.g. availability of their service over integrity, while other companies concentrate more on integrity. An example could be a game company hosting multiplayer servers, they would like their downtime to be as low as possible and might put less importance on the integrity of the server data. PICA is supposed to be integrated into the dashboard, to facilitate a more personalised and granular graph analysis.

In order to compare PICA and the baseline, the main research question is asked: *"How does PICA compare to alert frequency and severity as a prioritisation metric for alert-driven attack graphs by SAGE?"* is answered in the paper. Our main contribution is a new prioritisation metric for alert-driven attack graphs created by SAGE which:

1. Incorporates graph properties
2. Allows for customisation
3. Works on node and attack-stage level

In section 2, related work is discussed. Section 3 defines the baseline and the proposed metric. Section 4 gives an outline of the used methodology for the different research questions. Section 5 provides an overview of the used experimental set-up. The results are provided in section 6. Section 7 discusses limitations and future work. An overview of the reproducibility and risks associated with this research is provided in 8. Finally, a conclusion is given in section 9. Section 10 finished with acknowledgements.

2 Related Work

SAGE is an explainable, deterministic algorithm that can derive AGs from intrusion detection alerts without *a priori* expert/network knowledge. It uses a suffix-based probabilistic deterministic finite automaton learnt by FlexFringe [9]. AGs are extracted for each combination of victim-objective, where an objective is a high-severity attack stage. It manages to compress over 330,000 alerts into 93 different AGs, ready for

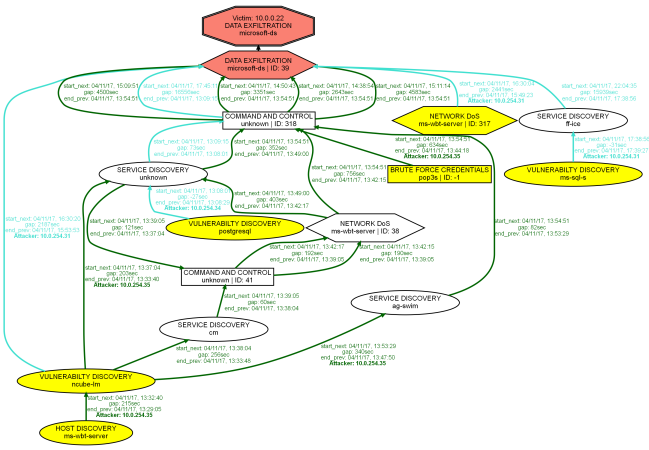


Figure 1: Hypothetical attack graph for victim 10.0.0.22 - *Data Exfiltration*. *Network DoS* and *Data Exfiltration* have the same prevalence and severity, leading to the same score in the current urgency metric, even though *Network DoS* accounts for only three paths and *Data Exfiltration* accounts for eight. This shows the importance of including paths in the urgency analysis.

analysis [3]. The AGs created by SAGE all follow a similar structure of which an example can be seen in Figure 2.

The generated AGs are still too high in complexity and quantity, as was discovered in preliminary interviews with Security Analysts [4]. To facilitate AG analysis, a dashboard was proposed by Nadeem *et al.* [4]. This dashboard provides an interactive way to analyse the graphs, while also offering filters such as victim, attacker and objective, and a recommender matrix for the attack stages. The matrix provides an overview of the urgency of each AS and can be used to show the paths leading to these stages. This urgency score is calculated based on prevalence and severity and will be used as the baseline in this paper.

The current urgency metric has two main flaws. Firstly, it has a relatively low granularity, as it does not take into account the effect the AS has on the affected system. SOCs have different preferences on what is important [7][8], which is difficult to adjust for in this metric. A game company with multiplayer servers will put more importance on availability than on integrity, while a cloud service might value integrity over availability.

Another apparent issue is that it does not incorporate graph properties. It looks at the most present/severe attack stages, but the presence of more nodes of a specific AS does not guarantee many paths towards the objective. The number of paths depicts how often an attack stage has been visited, while the number of nodes only shows the minimum number of paths. An example can be seen in Figure 1, where *Network DoS* and *Data Exfiltration* both have two nodes, even though *Network DoS* only accounts for three of the paths in this graph and *Data Exfiltration* for eight. In the baseline, these two attack stages would get the same urgency score, as they have identical prevalence and severity, while one is evidently more urgent than the other.

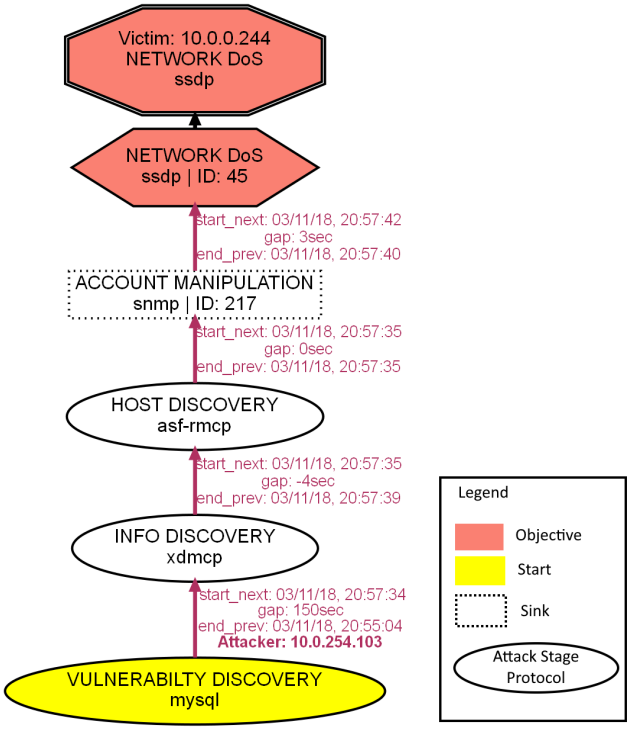


Figure 2: Attack graph For CPTC-2018 victim 10.0.0.244 - *Network DoS*. Octagonal nodes are of high severity, rectangular nodes of medium severity and oval nodes of low severity. Yellow nodes form the start of an attack path and red nodes the final objective of a path. Dotted nodes are sinks which occur too infrequently for learning [3]. Attack stages are written at the top of the node with the used protocol and ID on the next line, if applicable. Edges show the start time of the next Attack Stage and the end of the previous one. Paths start at a yellow node and end at a high-severity objective which points to the victim.

3 Problem Definition

This section aims at defining the metrics used. Section 3.1 defines the baseline, while PICA is defined in section 3.2.

3.1 Baseline

The current metric used for prioritising attack stages in the dashboard is based on prevalence and severity. It can be defined as follows:

- $Prevalence(AS) = \frac{count(node \in AS)}{count(node \in graph)}$
- $Severity(AS) = \begin{cases} 0.25 \text{ if low severity} \\ 0.5 \text{ if medium severity} \\ 1 \text{ if high severity} \end{cases}$
- $Urgency(AS) = prevalence(AS) \cdot severity(AS)$

To compare the baseline to PICA, the final score is normalised using linear scaling, as it can magnify the effect of outliers but preserves the relative order and distance in the data. The values used for severity in the experiment can be found in appendix C.1.

3.2 PICA

PICA is inspired by the CIA triad [5] and the graph properties of the AGs. It allows analysts to have weights for the different parts of the CIA triad, facilitating customisation based on their preferences. On top of that, it incorporates the number of paths going through a node, detecting important junctions used by attackers. PICA is based upon the assumption that all known paths are modelled in the AG, meaning the number of incoming edges equals the number of paths crossing an edge. We define PICA as follows:

- C, I, and A are values for Confidentiality, Integrity and Availability defined in C.2
- Each CIA value has a weight (respectively, w_1, w_2, w_3) defined by the Security Analyst
- $Urgency(node) = \frac{in\_deg(node)}{argmax_{nodes}(in\_deg)} \cdot \frac{w_1 \cdot C + w_2 \cdot I + w_3 \cdot A}{w_1 + w_2 + w_3}$
- $Urgency(AS, X) = avg(Urgency(node))$
  $\forall\, node \in AS$ and $Urgency(node) \in$ top X% for AS

The nodes used by PICA are those in the AG generated by SAGE, with the exception of "victim" nodes. They are excluded as PICA focuses on the paths taken by the attacker, and they are not part of the actual path but rather an artificial -root node made to combine the AGs based on victim-objective combinations. The final score generated by PICA is normalised on a node level using linear scaling such that it is comparable with the baseline.

The C, I, and A values are similar to the severity used in the baseline. If an AS affects a certain part of the CIA triad, its respective value for this part of the triad is 0.25 for low severity, 0.5 for medium severity and 1 for high severity. If the AS does not affect a part of the triad, its respective value is 0.1 to prevent substantially decreasing the weighted average.

In RQ_1, different percentages for $Urgency(AS, X)$ will be experimented with, in order to find an optimal value.

4 Methodology

In order to answer the main research question *"How does PICA compare to alert frequency and severity as a prioritisation metric for alert-driven attack graphs by SAGE?"*, the following sub-questions are investigated:

1. How does PICA affect the (number of) urgent attack stages for different averaging percentages?
2. How are PICA's urgent nodes positioned in the attack graphs?
3. What are the effects of changing the CIA weights in PICA?

RQ_1 will be answered by measuring the distribution of low-, medium-, and high-urgency attack stages generated by the metrics (for PICA with varying percentages for $Urgency(AS, X)$), and a qualitative analysis of the types of attack stages. This analysis is required to compare the behaviour of both methods for the different attack stages. It is suspected that PICA will detect the important junctions and deem them of higher urgency, meaning that the type of attack stage can greatly vary as junctions are not more likely to be of a specific type. The baseline will pick the most present/severe stages in the graph. The question is answered when differences and similarities of the types of urgent ASes discovered by PICA and the baseline become clear and are explained.

RQ_2 is measured by distance to the closest root (i.e., victim node) and distance to the closest starting point. Measuring this is essential to show where in the graph the urgent nodes are positioned, relative to the root and starting nodes. As junctions can be present at any point in the graph, we expect PICA will have the urgent nodes scattered through the graph, meaning a higher variance in distance to the root/starting point. For decreasing urgency, an overall increase in distance from the root is still expected as all paths end in objective nodes, which are close to the root node. This question will be answered when the node positions of different levels of urgency become clear and are explained.

For RQ_3, the first question will be repeated for varying weights of the CIA triad, with the same averaging percentage as discovered in RQ_1. This analysis is required to verify whether the weights for the CIA triad have the desired influence on the urgency score. The hypothesis is that the dominant weight of the CIA triad will give higher priority to related types of attacks, while important junctions will still substantially affect urgency. The question is answered when PICA is compared for different weights of the CIA triad and the impact of the weights is shown.

5 Experimental Setup

SAGE To generate the attack graphs, SAGE was used. For consistency, the algorithm was used with the same parameters as the original paper, an alert-filtering window of 1s and an alert-aggregating window of 150s [3]. The final implementation and used version of SAGE can be found in appendix E.

Dataset Two datasets were used as input for SAGE; CPTC-2017 and CPTC-2018, which are based on the National Collegiate Penetration Testing Competition where multiple red teams try to compromise an enterprise cyber-infrastructure [10]. The comparisons were made using the resulting AGs from SAGE. Table 1 displays the properties of their output when ran through SAGE. More statistics of the data, specifically the node count per AS, can be found in appendix A in Table 3.

Dataset	AGs	Nodes	Attack stages
CPTC-2017	108	331	19
CPTC-2018	75	247	20

Table 1: Properties of the CPTC-2017 and CPTC-2018 output from SAGE. The table displays the number of attack graphs generated, the total number of distinct nodes, and the number of distinct attack stages.

Networkx To analyse the graphs, (Python) code was written to merge all sub-graphs into one large graph, containing all nodes, using networkx. Networkx can read in the dot files produced by SAGE and perform actions on the corresponding graph data structure, e.g. joining graphs.

Metric		Number of Urgent Attack Stages (Low <= 0.25 < Medium <= 0.5 < High)					
		CPTC-2017			CPTC-2018		
	Top X%	High	Med	Low	High	Med	Low
PICA	1	8	4	7	10	2	8
	5	8	4	7	10	2	8
	10	6	6	7	5	5	10
	15	**4**	**8**	**7**	**5**	**5**	**10**
	25	3	2	14	2	8	10
	33	2	4	13	1	9	10
	50	0	5	14	0	4	16
	75	0	3	16	0	2	18
	100	0	1	18	0	0	20
Baseline		2	1	16	4	2	14

Table 2: Number of attack stages per urgency level for CPTC-2017 and CPTC-2018 in both the baseline and PICA when averaging over different percentages. When increasing the percentage of PICA, the number of high/medium urgency attack stages decreases as more nodes with a lower in-degree are included in the calculation. Overall, the baseline is more skewed towards low urgency while PICA is more evenly distributed. The row marked in bold was used for further comparison against the baseline.

6 Results

This section describes and discusses the results. Section 6.1 investigates the results for RQ_1. RQ_2 Is discussed in section 6.2 and RQ_3 in section 6.3.

6.1 Attack Stage Distribution

First, a quantitative analysis of the results for PICA and the basline is described in section 6.1. Furthermore, section 6.1 goes more in-depth by doing a qualitative analysis of the types of ASes targeted by the baseline and PICA.

Quantitative Analysis

When looking at the number of low-, medium-, and high-urgency AS for different top percentages in PICA, similar patterns arise for CPTC-2017 and CPTC-2018. It shows that for an increasing percentage the urgency of most ASes decreases, while a lower percentage leads to relatively many high-urgency AS, as can be seen in Table 2. When looking at the discovered highly-urgent AS for PICA, the lower percentages (1, 5 and 10), contain most of the discovery-type ASes, which are less interesting to a Security Analyst, as some of them typically have a relatively high in-degree. To have the right balance between the number of high-/medium-/low-urgency AS, PICA will be used with the average of the top 15% of urgent nodes for further comparison with the baseline.

When comparing PICA's (15%) distribution to the baseline's, it shows that PICA finds twice the number of highly-urgent AS, eight times as many medium-urgency AS and less than half the number of low-urgency AS for CPTC-2017. For the 2018 dataset, PICA finds nearly the same number of highly-urgent AS, more than twice the number of medium-urgency AS, and roughly the same number of low-urgency AS. Overall, PICA is more evenly distributed over the different urgency levels than the baseline, where most AS are either of high urgency or low urgency. This is a positive development, as it allows Security Analysts to have a more spread-out view of the urgency levels.

A small quantitative factor contributing to these differences is the severity score. In the baseline, each AS has a severity of either 0.25, 0.5 or 1. In PICA, depending on which parts of the system it affects, it can vary as the average of the scores for confidentiality, integrity and availability, which are also 0.25, 0.5 or 1, are used. The exact scores can be found in appendix C.1 and C.2. Since the CIA triad and the severity scores are correlated, the difference only has a minor impact. Further reasoning for the change in the number and types of results will be given in the qualitative analysis.

Qualitative Analysis

First, the differences within PICA itself for different averaging percentages can be attributed to the inclusion/exclusion of nodes with a lower in-degree, leading to a different overall score. In CPTC-2017 and CPTC-2018, there are discovery nodes with a very high in-degree compared to other nodes, e.g. *Host Discovery* in CPTC-2017 with an in-degree 29 compared to a max in-degree of 37. This leads to a higher urgency in PICA's lower percentages. Increasing the percentage leads to adding discovery nodes with a lower in-degree, moving them to the medium/low urgency range. An example of this is the *Host Discovery* AS going from high urgency to low urgency when switching from 10% to 15%. The rest of this section will focus on the comparison of the baseline and PICA with 15% as it has shown to be a good balance between not having the discovery nodes and retaining the highly-urgent AS.

CPTC-2017 The urgency scores for the baseline and PICA can be found in Figure 3. When comparing these heatmaps, it becomes apparent that PICA makes different discoveries:

- Baseline → PICA: *AS*
- High → medium: *Data Delivery*
- Medium → high: *Network DoS*
- Low → high: *Vulnerability Discovery*, *Arbitrary Code Execution*
- Low → medium: *Command and Control*, *Host Discovery*, *Account Manipulation*, *Info Discovery*, *Service Discovery*, *Surfing*, *Remote Service Exploit*

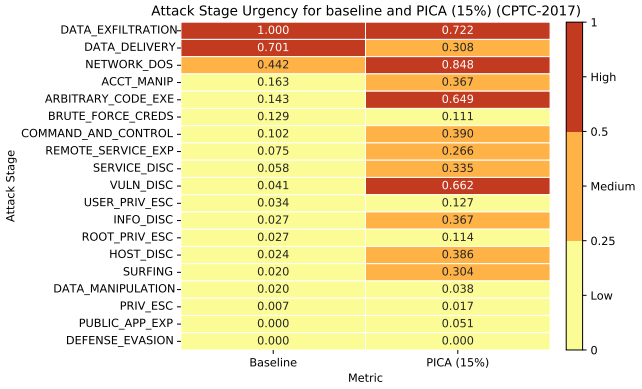


Figure 3: Attack stage urgency heatmap for baseline and PICA (15%) for CPTC-2017. PICA discovers more high- and medium-urgency attack stages compared to the baseline, while still not discovering the exact same high-urgency attack stages. PICA is more evenly distributed throughout the different urgency levels.

CPTC-2018 The heatmap for CPTC-2018 can be found in Figure 4 for both the baseline and PICA. The following changes in urgency levels can be found:

- Baseline → PICA: *AS*
- High → medium: *Resource Hijacking*, *Data Delivery*
- Medium → high: *Network DoS*, *Root Privilege Escalation*
- Low → high: *Info Discovery*
- Low → medium: *Arbitrary Code Execution*, *Vulnerability Discovery*, *Account Manipulation*

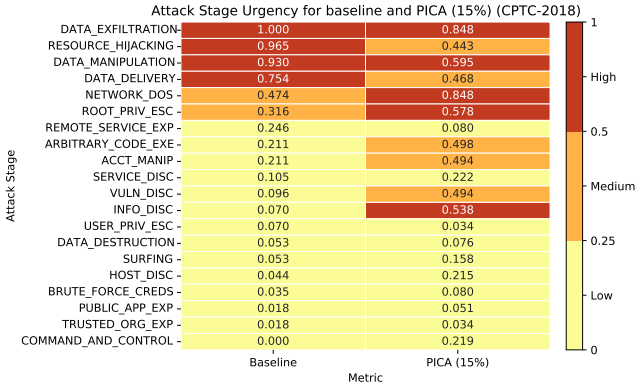


Figure 4: Attack stage urgency heatmap for baseline and PICA (15%) for CPTC-2018. PICA discovers more high- and medium-urgency attack stages compared to the baseline, while still not discovering exactly the same attack stages. PICA is more balanced throughout the different urgency levels.

When comparing the differences for both CPTC-2017 and CPTC-2018, similar patterns show. First of all, discovery nodes gain a higher urgency score because of their high in-degrees. The increase of urgency in discovery ASes is an undesired side-effect, as these happen often and do not severely affect a system. Although it typically marks the start of an attack, they are used to find vulnerabilities which means that fixing discovery nodes will only prevent new attackers from finding the same exploits while the current attackers already know what to target. In order to stop the actual attack, nodes further down the chain have to be tackled. An example of this can be seen for *Info Discovery* in Figure 5, where the nodes selected by PICA have a high in-degree, leading to a higher urgency score than if prevalence were to be used. In this example, info discovery has 10 nodes out of 247 in the graph, leading to a low prevalence, while the selected nodes have in-degrees 37 and 21, where 37 is the max in-degree out of all nodes, leading to a high urgency score. The inverse also holds, as certain ASes lose urgency because of a low in-degree in their nodes. An example is *Data Delivery* in CPTC-2017, going from high to medium urgency. This change can be attributed to the fact that there are many *Data Delivery* nodes (high prevalence) but the majority of these nodes have a relatively low in-degree as can be seen from the used nodes in Figure 6. *Data Delivery* has 52 nodes out of 331 (max is 74). For the used nodes the in-degrees are 8,6 and 5, while the max in-degree is 37, meaning it is relatively low. Other changes in urgency can also be attributed to these patterns.

While it can be reasoned that not being able to detect many nodes with a low in-degree allows for targeted attacks, as this can be exploited to stay under the radar, PICA bases the path factor on the max in-degree of the nodes, meaning that if attackers start avoiding visiting the same node many times, the overall in-degrees will decrease and PICA will still be able to detect the high-urgency attack stages. The same does not go for the baseline, which cannot alter for the high in-degree nodes, as if the prevalence is equal for all nodes, then their severity will be equal depending on their severity level. An alternative approach for this trade-off is discussed in future work in section 7.

6.2 Node Properties

Distance to Closest Root Node When looking at the distance to the closest root node, similar patterns show for CPTC-2017 (Figure 7 and CPTC-2018 (appendix, D.1 Figure 12). When urgency increases, the distance to the root node increases on average. This aligns with the hypothesis which stated that urgent nodes appear later in the path of an attack, i.e., closer to the root node of the AG. Since all paths in SAGE's AGs end in high-severity objectives, it can be reasoned that these objectives have many paths visiting them in each AG. This means that nodes higher in the graph i.e., closer to the root node, will have an increased in-degree. As Figure 7 shows, the high-urgency ASes have some outliers which are further from the root as junctions can also be found further away. Overall, a decreasing trend can be seen for decreasing urgency.

Distance to Closest Start Node When investigating the distance to the closest start node, unexpected behaviour shows in both CPTC-2017 (Figure 8) and CPTC-2018 (appendix D.1, Figure 11). The distances follow the same trend as the root node; when the urgency decreases, so does the distance to the closest starting node. Even though this is

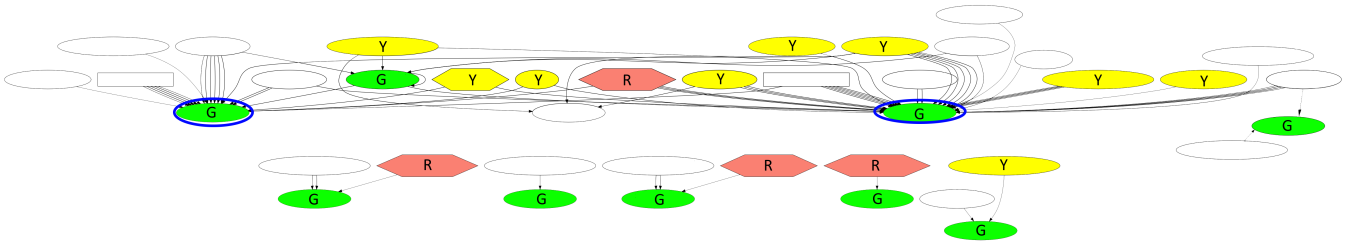


Figure 5: Complete *Info Discovery* sub-graph of CPTC-2018. *Info Discovery* nodes are green (G). Nodes selected by PICA are circled in blue, yellow (Y) nodes are starting nodes, and white nodes are different attack stages. The high in-degree of the selected nodes leads to a high urgency score for *Info Discovery* in PICA, while the low prevalence results in a low urgency score for the baseline.

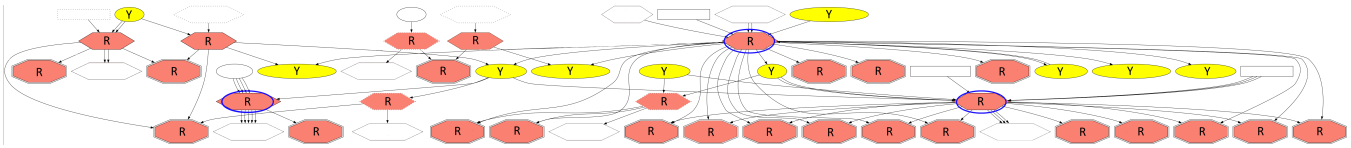


Figure 6: Part of the *Data Delivery* sub-graph of CPTC-2017. Red (R) nodes are *Data Delivery*, yellow (Y) nodes are starting nodes, and white nodes are different attack stages. Double-edged nodes are victim nodes. Nodes circled in blue are selected for the *Data Delivery* urgency calculation of PICA. The selected nodes have a relatively low in-degree, leading to a lower urgency for PICA compared to the baseline, as there are many (victim) nodes (i.e., high prevalence).

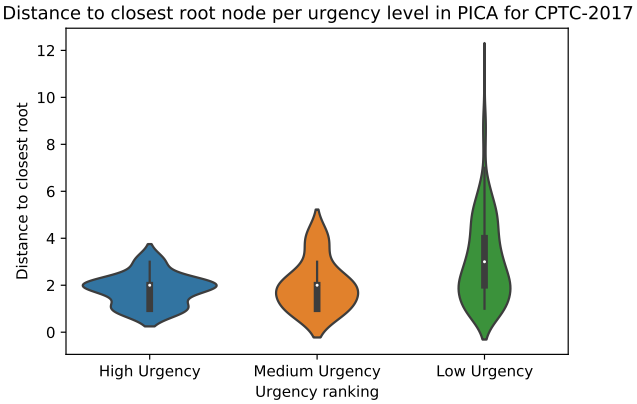


Figure 7: Violin plot showing the distance to the closest root node per urgency level of PICA on CPTC-2017. Distance to the closest root node increases when urgency decreases.

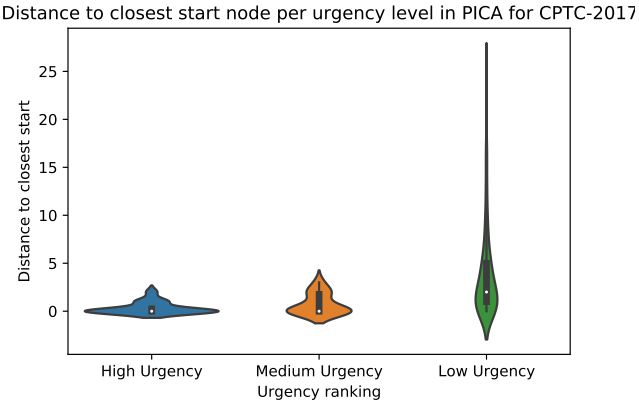


Figure 8: Violin plot showing the distance to the closest starting node per urgency level of PICA on CPTC-2017. Distance to the closest start node increases when urgency decreases.

counter-intuitive and does not align with the original hypothesis, qualitative analysis shows it holds. Figure 9 shows that often high-severity nodes are starting nodes and that these can be close to the root of the AG. When inspecting the sub-AGs, this does not immediately show, as high-severity objectives are coloured red in their respective AGs. This means that there is a loss of information when generating the sub-graphs, as a final objective which is also a starting node is more interesting than a normal final objective.

6.3 Effects of Changing CIA Weights

One final improvement offered by PICA compared to the baseline is the granularity in the severity. The CIA triad offers to prioritise different types of ASes. Figure 10 depicts the changes in urgency score when varying the weights in the weighted CIA average for CPTC-2017 (for CPTC-2018, see appendix D.2). To limit the scope of this research question, the weights were only tested with varying values of 1 and 2 to show their impact and patterns.

The heatmap in Figure 10a is run on the implementation of PICA used in the previous research questions. During analysis, it showed that when changing weights, ASes which were expected to be positively impacted (i.e., they affect the weight that has been increased) showed a decrease in urgency. This is an artefact of normalisation, as the highest-ranked attack stage positively affected by this weight increased the max score, leading to a larger normalisation for the less urgent attack stages. Even though their absolute score increased, the heavier normalisation causes an overall decrease. Since urgency is a metric that is relative to other attack stages, this

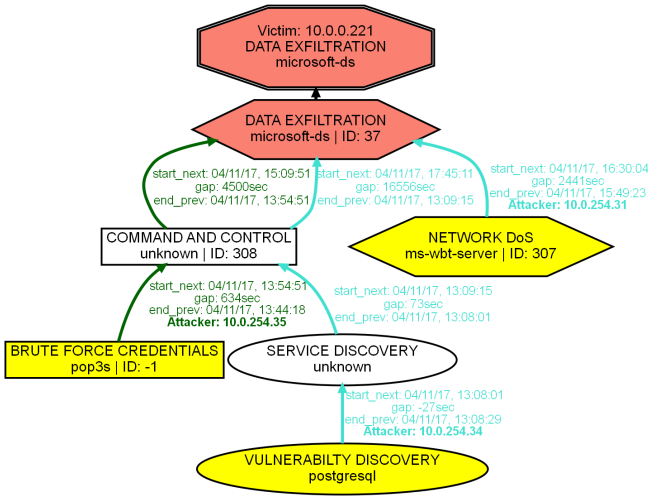
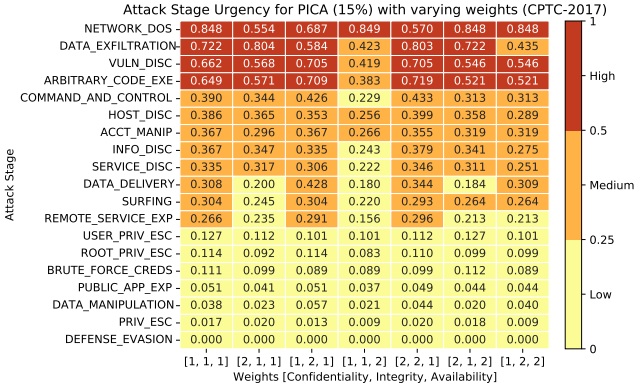


Figure 9: Attack graph for victim 10.0.0.221 - *Data Exfiltration*. *Network DoS* is both a starting node in this graph and a final objective in a different graph. This means that, for *Network DoS*, the distance to the closest starting node will be 0. It also means that information is lost in the *Network DoS* graph, as the node is shown as an objective instead of a starting node.

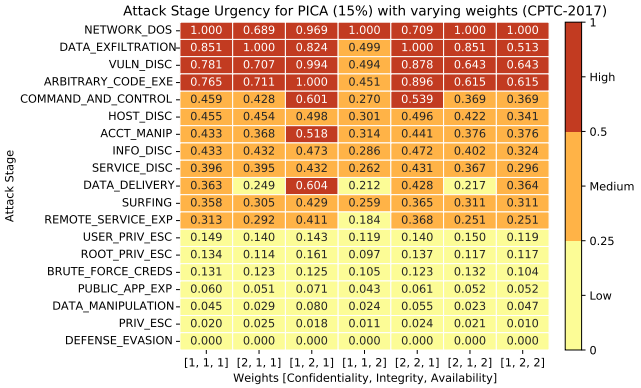
behaviour, although maybe undesired, is logical. The effect of normalisation posed the question of whether or not it is logical to normalise the scores on a node level (current implementation) rather than an attack stage level. Therefore, both versions were used in this section. For the CPTC-2018 heatmap in appendix D.2, more drastic changes can be seen when changing the normalisation. This is due to the medium-urgency scores being close to the border of becoming highly urgent and the normalisation pushing them past this border. Section 7 proposes further work to investigate this issue.

Since PICA consists of 50% of the weighted average of the CIA triad, a logical development should be that increasing a weight positively affects the attack stages included in this part of the triad, and negatively affects the attack stages that are not. Although this pattern emerges in both Figure 10a and 10b, the urgency scores are overall higher when normalising on an AS level. For equal weights, although scores vary, both implementations of PICA find the same attack stages for the different urgency levels. It is when adjusting weights that differences become evident.

Columns 2-4 of Figure 10 show the impact of changing the weights one at a time. When prioritising e.g. confidentiality ([2, 1, 1), a decrease in urgency can be noticed for ASes which do not affect confidentiality, e.g. *Network DoS*, *Data Delivery* and *Surfing*. An increase can be seen for *Data Exfiltration* which does affect confidentiality. An artefact of score normalisation can be seen when closely investigating *Remote Service Exploit* in the same column. It affects confidentiality, meaning its score should theoretically increase, but because the maximum urgency score of the nodes increases, the normalisation causes it to decrease. When changing the weights of integrity and availability, similar behaviour shows. Since most of the ASes do not affect availability, many of them change to low urgency, while for integrity, the inverse holds. A similar pattern can be noticed in the attack-stage-normalisation in Figure 10b, although with elevated scores leading to more urgent attack stages.



(a) Urgency scores for normalisation at a node level.



(b) Urgency scores for normalisation at an attack stage level.

Figure 10: Urgency for varying weights of PICA (15%) on CPTC-2017 when normalising the urgency score at a node level (top) and an attack stage level (bottom). It shows that for equal weights they discover the same urgent attack stages, but for varying weights, they differ and attack-stage-level normalisation finds more medium/high-urgency attack stages. The more urgent attack stages which are positively affected by changed weights, show an increased urgency score, while lower-urgency attack stages which were also positively affected, show a decreased score due to normalisation.

The remaining three columns of Figure 10 focus on the changes when two weights are altered simultaneously. It appears that when two weights change, the normalisation phenomenon described in the previous paragraph does not occur as frequently or as impacting as it does when altering a single weight, e.g. *Remote Service Exploit* in column 6 shows an increase as it affects both Confidentiality and Integrity. Since it is a lower-scoring attack stage and the score showed an increase instead of a decrease, it shows that normalisation was less substantial. This claim holds true for both implementations, as Figure 10b displays the same pattern.

7 Limitations and Future Work

A trade-off that can be seen in the results, is that an attack stage with many nodes containing a single path is considered less urgent in PICA as they have a low in-degree. This can lead to an AS that is executed many times, but that can have a lower urgency score than an AS that has few nodes but a slightly higher in-degree in its most-urgent nodes. This is intentional, however, as it is often the case that removing the node from this more urgent AS leads to breaking more paths with less investigation, thus being worth investigating first. A targeted attack can be done where scripts attack many nodes a single time to reach an objective, in order to decrease the in-degree, meaning PICA will not discover this. If attackers start avoiding high in-degrees, the maximum in-degree of the graph will change and PICA's urgency scores will alter accordingly. The inverse cannot be said about the baseline, where it does not have a way to distinguish ASes with equal prevalence and severity.

PICA treats the connected nodes in the AGs as sequential. In reality, nodes can be executed in parallel, e.g., through scripting, meaning that SAGE possibly learns a non-existing path. This leads to PICA relying on wrong paths and possibly providing incorrect urgency scores as a consequence. When using this metric, this is a limitation that should be considered.

PICA calculates the in-degrees based on the complete AG generated when merging all sub-AGs from SAGE. SAGE merges all low-severity nodes where the context, as defined in the SAGE paper [3], does not make a difference and removes their IDs in the process. An artefact of this can be found when merging the sub-graphs, as these nodes might have different contexts between different sub-AGs, but they are merged into a single node in the complete AG, as they have the same name/ID. For this reason, the in-degrees of these nodes, typically discovery-type nodes, are inflated. Future work can be proposed for dealing with these types of nodes. Multiple approaches can be taken: splitting these nodes and keeping their IDs or giving the merged nodes a new ID.

Whilst performing the research, some implementation bugs were found in SAGE's code. For reproducibility purposes, the used version of SAGE is included in the GitHub repository in appendix E containing the code to run the experiments.

Lastly, some improvements can be suggested for PICA. To tackle the noticed trade-off with the baseline, PICA can incorporate the number of nodes and their corresponding in-degree as a weight. Another weight that could be used for nodes is their distance from the victim. To overcome the normalisation issue discovered when changing weights, different normalisation metrics can be experimented with in different parts of the urgency calculation, e.g. in the AS urgency calculation instead of in the node urgency calculation. On top of that, the level where normalisation occurred should be investigated further.

8 Responsible Research

This section describes how this research adheres to responsible research practices. First, it talks about the reproducibility of the work. Second, it talks about how the five principles of the 2018 *Netherlands Code of Conduct for Research Integrity* were followed [11]. Lastly, risk mitigation is proposed for discovered shortcomings.

Reproducibility The algorithm used to generate the AGs, together with PICA and the baseline metric, are fully deterministic. The used datasets, CPTC-2017 and CPTC-2018, are publicly available. This means that all conducted experiments are 100% reproducible as the source code for both SAGE, the baseline and PICA are publicly available (see appendix E). The setup of the experiments is documented in this paper and the code for PICA is documented for ease of use, further facilitating reproducibility.

Principles for Good Research Section 6 and 7 describe all discovered shortcomings of PICA, in order to be *honest* about findings and not hide any important details. All executed experiments were designed with the scientific method in mind, striving for *scrupulousness*. To provide *transparency*, all used datasets and algorithms are mentioned and described and the used algorithms have a focus on explainability and interpretability. To remain *independent*, this research has not been influenced by outside factors of a commercial or political nature and has been objectively carried out to ensure impartiality. Finally, intermediate results were discussed with peers and supervisors to avoid operation in isolation and to ensure *responsability*. By adhering to the above-mentioned principles, our research aims for scientific integrity.

Risk Mitigation The use of the proposed metric carries several risks. As mentioned in the discussion, it is possible for PICA to miss high-urgency attack stages that have many nodes but with low in-degrees. This can have disastrous consequences, as disregarded dangerous Cybersecurity attacks can pose major issues such as privacy breaches, data leaks and downtime of important services. The paper has discussed both strengths and weaknesses of PICA in order to inform its users of these risks, but the final responsibility for mitigating these risks lies on the user's side. One straightforward mitigation could be to run multiple metrics in order to prevent bias towards a single urgency type.

Another risk, not mentioned in the original paper introducing SAGE, is that SAGE models the alerts generated by the IDS, which means that any attack that does not generate an Intrusion Detection Alert is not modelled. As a result of this, Security Analysts should be cautious when using this tool. IDS Should be configured correctly and alternative analysis/detection systems should also be employed to prevent specific attacks from bypassing an IDS.

9 Conclusion

Although the AGs generated by SAGE remain too high in quantity and complexity to analyse, a good prioritisation metric is a step in the right direction. Prioritising attack stages allows Security Analysts to investigate them in an organised and ranked manner. PICA makes this prioritisation more customisable by using the CIA triad and more intuitive by deciding prevalence based on the number of paths rather than the number of nodes. PICA does what it is designed to do: discover attack stages with high-in-degree nodes and the option to prioritise a specific category of attack.

The main findings were:

- PICA with the average of the 15% most-urgent nodes maintained a good balance between reducing the urgency of discovery-type ASes while still retaining the important ASes. Overall, the distribution of ASes over the different urgency levels for PICA was more even compared to the baseline.
- PICA discovered different attack stages, resulting in a trade-off. PICA fails at finding ASes that contain many nodes with a low in-degree, while the baseline will discover these. The inverse also holds, where the baseline does not discover fewer nodes with a high in-degree, while PICA does. Future work was proposed in order to mitigate this issue.
- For different urgency levels, the distance to the closest root node follows a logical trend, where it increases for decreasing urgency. For the distance to the closest starting node, however, the same pattern was discovered which seems illogical at first. It became apparent that the cause of this pattern is that multiple high-severity nodes (i.e., end-objectives) are also starting nodes, which gives them a distance of 1 to the root. This discovery showed that when SAGE creates sub-AGs, there is a loss of information as it is not apparent for some objectives that they can also be starting nodes.
- When changing the weights of the CIA triad, PICA showed a positive bias towards the larger weights, as expected. An artefact of the linear normalisation showed, where scores that intuitively should have increased instead decreased. This was attributed to the maximum urgency score increasing leading to a larger normalisation.

Although trade-offs were discovered, PICA delivered on its promises to find attack stages containing urgent nodes which have a high number of paths crossing them. Future work was proposed to further improve PICA's flexibility and resistance, but the overall goal was accomplished as it has proven to be an improvement compared to the current implementation.

10 Acknowledgements

I would like to thank Dr. Ir. Sicco Verwer and PhD Candidate Azqa Nadeem, with whom I had the pleasure of collaborating. They have provided me with the necessary feedback and instructions in order to keep this research on the right track. I would also like to mention my peers who have provided valuable insights: *Jegor Zelenjak*, *Ioan-Cristi Oprea*, *Alexandru Dumitriu* and *Vlad Constantinescu*.

References

[1] Wajih Ul Hassan, Shengjian Guo, Ding Li, Zhengzhang Chen, Kangkook Jee, Zhichun Li, and Adam Bates. NoDoze: Combatting Threat Alert Fatigue with Automated Provenance Triage. In *Proceedings 2019 Network and Distributed System Security Symposium*, San Diego, CA, 2019. Internet Society.

[2] S. Jha, O. Sheyner, and J. Wing. Two formal analyses of attack graphs. In *Proceedings 15th IEEE Computer Security Foundations Workshop. CSFW-15*, pages 49–63, Cape Breton, NS, Canada, 2002. IEEE Comput. Soc.

[3] Azqa Nadeem, Sicco Verwer, and Shanchieh Jay Yang. Sage: Intrusion alert-driven attack graph extractor. In *Symposium on Visualization for Cyber Security (Vizec)*. IEEE, 2021.

[4] Azqa Nadeem, Sonia Leal Dıaz, and Sicco Verwer. Critical Path Exploration Dashboard for Alert-driven Attack Graphs. *https://vizsec.org/files/2022/vizsec_p4_abstract.pdf*.

[5] Spyridon Samonas and David Coss. The cia strikes back: Redefining confidentiality, integrity and availability in security. *Journal of Information System Security*, 10(3), 2014.

[6] Stephen Moskal and Shanchieh Jay Yang. Cyberattack action-intent-framework for mapping intrusion observables, 2020.

[7] Sharman Lichtenstein. Factors in the selection of a risk assessment method. *Information Management*, 1996.

[8] Ankit Shah, Rajesh Ganesan, Sushil Jajodia, and Hasan Cam. A Two-Step Approach to Optimal Selection of Alerts for Investigation in a CSOC. *IEEE Transactions on Information Forensics and Security*, 14(7):1857–1870, July 2019.

[9] Sicco Verwer and Christian A. Hammerschmidt. flexfringe: A passive automaton learning package. In *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 638–642, 2017.

[10] Nuthan Munaiah, Justin Pelletier, Shau-Hsuan Su, S Jay Yang, and Andrew Meneely. A cybersecurity dataset derived from the national collegiate penetration testing competition. In *HICSS Symposium on Cybersecurity Big Data Analytics*, 2019.

[11] KNAW, NFU, NWO, TO2-Federatie, Vereniging Hogescholen, and VSNU. Nederlandse gedragscode wetenschappelijke integriteit, 2018.

A Statistics CPTC-2017 & CPTC-2018

Attack Stage	Node Count	
	CPTC-2017	CPTC-2018
Account Manipulation	25	13
Arbitrary Code Execution	22	13
Brute Force Credentials	20	3
Command and Control	16	1
Data Delivery	52	22
Data Destruction	N/A	2
Data Exfiltration	74	29
Data Manipulation	2	27
Defense Evasion	1	N/A
Host Discovery	9	7
Info Discovery	10	10
Network DoS	33	14
Privilege Escalation	2	N/A
Public Application Exploit	1	2
Remote Service Exploit	12	15
Resource Hijacking	N/A	28
Root Privilege Escalation	5	19
Service Discovery	19	14
Surfing	8	8
Trusted Organisation Exploitation	N/A	2
User Privilege Escalation	6	5
Vulnerability Discovery	14	13

Table 3: Node count for all attack stages present in CPTC-2017 and CPTC-2018.

B SAGE Implementation Bug

While working with the SAGE algorithm *Jegor Zelenjak* and I discovered a bug in the creation of the AGs. This section of the appendix aims to describe the bug and propose testing to help discover similar bugs.

B.1 Bug Description

In the creation of the AGs, all attacks belonging targeting a victim-objective tuple were added to the corresponding AG. In the code, a loop over all attacks was done to filter out those that belong to the victim. This was done by comparing the victim IP and the attack string (”attacker IP -¿ victim IP) using the ”in” operator in Python. This resulted in attacks targeting e.g. ”10.0.0.25” to be added to the victim ”10.0.0.2”, inflating the number of paths and AGs. Fixing this bug reduced the number of AGs drastically, for CPTC-2017 from 167 to 108 and for CPTC-2018 from 76 to 75.

B.2 Proposed Testing Technique

A way to detect this in the testing stage would be to augment the information saved by SAGE such that all edges in a dot file have an ID that identifies which path they belong to. This way, paths from different dot files can be compared with each other. If two AGs contain a path with the same ID, it is clear that this path has been duplicated as it cannot belong to two different objectives, since a path goes from a starting point to a specific objective.

C Severity Scores

C.1 Baseline Scores

	Attack Stage	Severity Score
Low Severity	Target Identification	0.25
	Surfing	0.25
	Social Engineering	0.25
	Host Discovery	0.25
	Service Discovery	0.25
	Vulnerability Discovery	0.25
	Info Discovery	0.25
Medium Severity	User Privilege Escalation	0.5
	Root Privilege Escalation	0.5
	Network Sniffing	0.5
	Brute Force Credentials	0.5
	Account Manipulation	0.5
	Trusted Organisation Exploitation	0.5
	Public Application Exploitation	0.5
	Remote Service Exploitation	0.5
	Spearphishing	0.5
	Service Specific	0.5
	Defense Evasion	0.5
	Command and Control	0.5
	Lateral Movement	0.5
	Arbitrary Code Execution	0.5
	Privilege Escalation	0.5
Medium Severity	Endpoint DoS	1
	Network DoS	1
	Service Stop	1
	Resource Hijacking	1
	Data Destruction	1
	Content Wipe	1
	Data Encryption	1
	Defacement	1
	Data Manipulation	1
	Data Exfiltration	1
	Data Delivery	1
	Phishing	1

Table 4: Severity Scores for the baseline urgency metric.

C.2 PICA Scores

	Attack Stage	Confidentiality Score	Integrity Score	Availability Score	Average
Low Severity	Target Identification	0.1	0.1	0.1	0.1
	Surfing	0.1	0.1	0.1	0.1
	Social Engineering	0.25	0.1	0.1	0.15
	Host Discovery	0.25	0.1	0.1	0.15
	Service Discovery	0.25	0.1	0.1	0.15
	Vulnerability Discovery	0.25	0.25	0.1	0.2
	Info Discovery	0.25	0.1	0.1	0.15
Medium Severity	User Privilege Escalation	0.5	0.1	0.5	0.37
	Root Privilege Escalation	0.5	0.5	0.5	0.5
	Network Sniffing	0.5	0.1	0.1	0.23
	Brute Force Credentials	0.5	0.1	0.5	0.37
	Account Manipulation	0.5	0.5	0.5	0.5
	Trusted Organisation Exploitation	0.5	0.5	0.1	0.37
	Public Application Exploitation	0.5	0.5	0.5	0.5
	Remote Service Exploitation	0.5	0.5	0.1	0.37
	Spearphishing	0.5	0.1	0.1	0.23
	Service Specific	0.5	0.5	0.5	0.5
	Defense Evasion	0.1	0.1	0.1	0.1
	Command and Control	0.5	0.5	0.1	0.37
	Lateral Movement	0.5	0.5	0.1	0.37
	Arbitrary Code Execution	0.5	0.5	0.1	0.37
	Privilege Escalation	0.5	0.1	0.1	0.23
Medium Severity	Endpoint DoS	0.1	0.1	1	0.4
	Network DoS	0.1	0.1	1	0.4
	Service Stop	0.1	0.1	1	0.4
	Resource Hijacking	0.1	0.1	1	0.4
	Data Destruction	0.1	1	1	0.7
	Content Wipe	0.1	1	1	0.7
	Data Encryption	0.1	1	1	0.7
	Defacement	1	1	0.1	0.7
	Data Manipulation	0.1	1	0.1	0.4
	Data Exfiltration	1	0.1	0.1	0.4
	Data Delivery	0.1	1	0.1	0.4
	Phishing	1	0.1	0.1	0.4

Table 5: Severity Scores for PICA including the unweighted average.

D Results for CPTC-2018

D.1 Distance to Root and Starting Node for CPTC-2018 (RQ2)

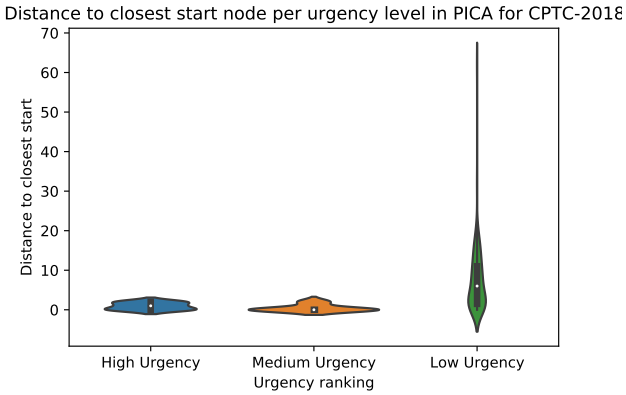


Figure 11: Violin plot showing the distance to the closest starting node per urgency level of PICA on CPTC-2018. The distance increases as the urgency decreases.

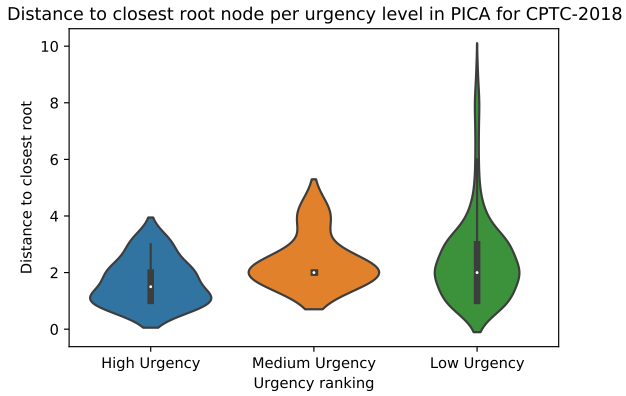
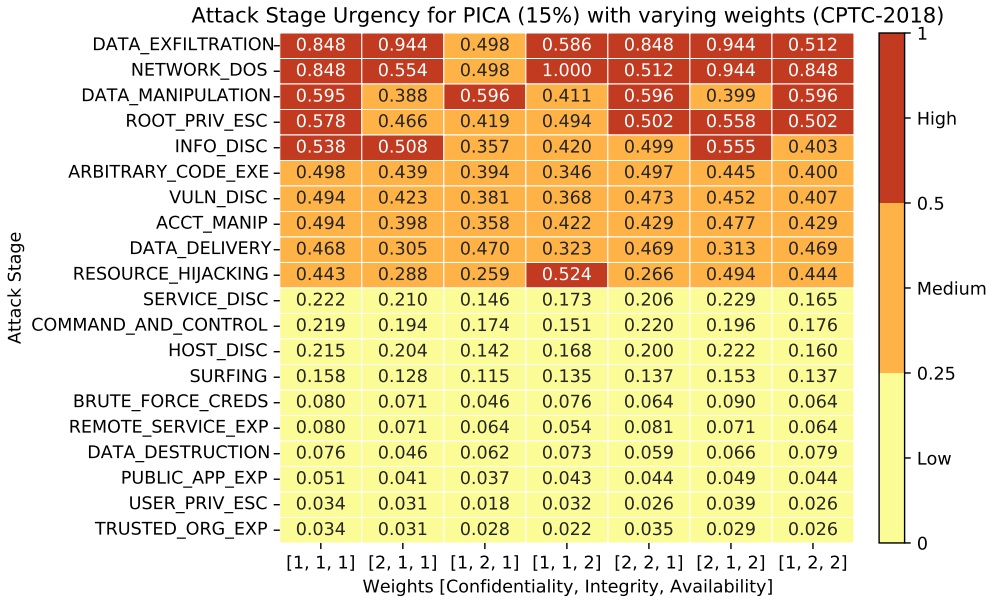
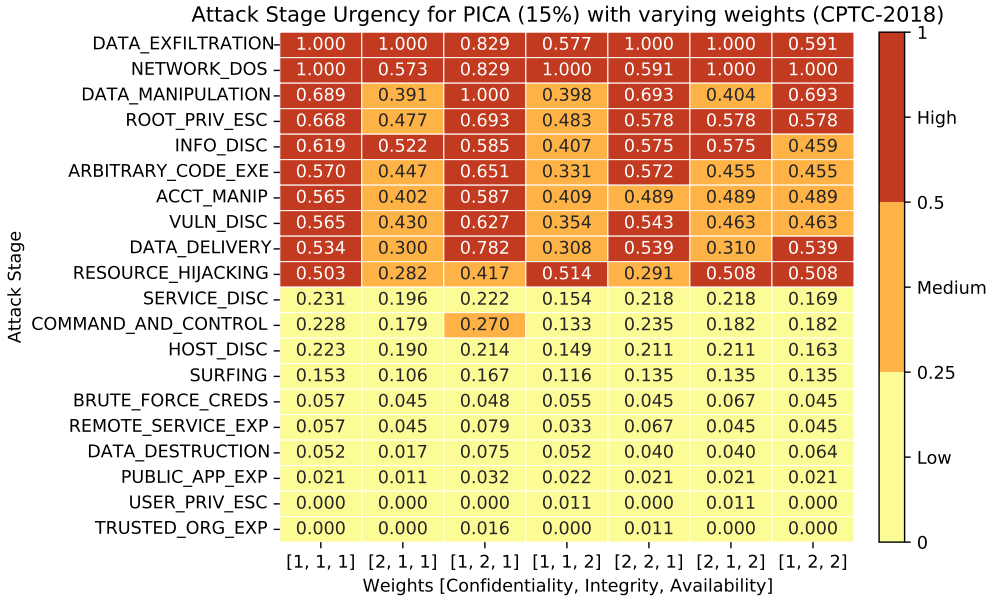


Figure 12: Violin plot showing the distance to the closest root node per urgency level of PICA on CPTC-2018. The distance increases as the urgency decreases.

D.2 Effect of Changing PICA Weights (RQ3)



(a) Urgency scores for normalisation at a node level.



(b) Urgency scores for normalisation at an attack stage level.

Figure 13: Urgency for varying weights of PICA (15%) on CPTC-2018 when normalising the urgency score at a node level (top) and an attack stage level (bottom). It shows that for equal weights the attack stage normalisation tips the medium-urgency attack stages to become of high urgency, but for varying weights. A similar pattern shows when varying weights. The more urgent attack stages which are positively affected by changed weights, show an increased urgency score, while lower-urgency attack stages which were also positively affected, show a decreased score due to normalisation.

E Implementation of Metrics and SAGE

The used code to run the experiments can be found in the following GitHub repository: https://github.com/smzvandenbroec/SAGE.

The official repository of SAGE can be found here: https://github.com/tudelft-cda-lab/SAGE