# Adaptive Network Management for Safety-Critical Systems

Cora Perner
XRC
Airbus
Munich, Germany
cora-lisa.perner@airbus.com

Holger Kinkelin and Georg Carle
Chair of Network Architecture and Services
Technical University of Munich
Garching near Munich, Germany
{kinkelin,carle}@net.in.tum.de

***Abstract*—Present networks within safety-critical systems rely on complex and inflexible network configurations. New technologies such as software-defined networking are more dynamic and offer more flexibility, but due care needs to be exercised to ensure that safety and security are not compromised by incorrect configurations. To this end, this paper proposes the use of pre-generated and optimized *configuration templates*. These provide alternate routes for traffic considering availability, resilience and timing constraints where network components fail due to attacks or faults.**

**To obtain these templates, two heuristics based on Dijkstra's algorithm and an optimization algorithm providing the maximum resilience were investigated. While the configurations obtained through optimization yield appropriate templates, the heuristics investigated are not suitable to obtain *configuration templates*, since they cannot fulfill all requirements.**

## I. Introduction

Networks currently used within safety-critical systems such as electricity grids or on-board aircraft networks rely on network switches to implement policies and routing strategies. What is more, the most critical paths are usually backed up by additional, hard wired paths that bypass the network completely. However, these networks are limited with respect to complexity and lack of scalability and are furthermore vendor-dependent [1]. Yet new strategies and technologies such as software-defined networking (SDN) offer an interesting prospect as they allow more flexible network management. SDN allows to separate the traffic handling from network management, where switches are mere forwarding agents and the traffic management is handled by at least one separate entity, the *controller*.

This additional flexibility could be used to adaptively react to changes in the network that are caused by faulty components as well as network security incidents. For example, if a network switch is targeted by an attack, the respective node can be isolated by rerouting traffic to avoid compromising further components. This adaptability could lead to the ability to tolerate more failures safely without the need to enter a degraded mode or postpone the need for a maintenance action. This would allow a continued, safe operation until a fault or vulnerability can be repaired, e.g. allowing an aircraft to complete its journey rather than making an emergency landing.

978-3-903176-15-7 

In addition to the network, hard-wired physical links exist between the most critical components. Consequently, if the network is sufficiently reliable, these links may safely be removed which allows to reduce weight – highly relevant in aircraft design – and complexity. Unfortunately, SDNs have been designed for non-critical applications which leads to a number of challenges for their usage in critical systems as the following needs to be ensured:

| | |
|---|---|
| Availability | The required information must be available at any given time. |
| Resilience | No single fault/failure may render the system inoperable. |
| Timing/Predictability | The required data must be available no later than a given time. |

Generally, when new network configurations are applied in SDN, routing information in the forwarding tables is changed by the controller. Contrary to most IT systems, the networks in safety-critical systems cannot stop transmission while changes are made to the forwarding tables. Hence, special care is needed during reconfiguration not to drop traffic on the paths unaffected by the failure. To this end, pre-generated and optimized *configuration templates* are introduced in this paper. These are a set of configurations that consider all the requirements listed above. The individual configurations differ by the resources used, thus providing resilience. They work as an extension to fail-safe states where only safe operation is guaranteed [2], but without the need of accepting performance degradation. When a fault has been detected, a different template not using the affected component can be applied on the network, thus allowing a continued safe operation with the new configuration. The main contribution of this paper is how to obtain said templates.

The structure of this paper is as follows: Section II provides an overview about related work, while Section III gives the mathematical formulation of requirements for *configuration templates*. The experimental setup and corresponding evaluation results are given and discussed in Section IV. A brief summary and outlook to future work in Section V conclude the paper.

## II. Related Work

There are a variety of heuristics that aim to dynamically reroute traffic during runtime. However, this approach is not

suitable for safety-critical systems that require performance guarantees and often deterministic behavior for certification. Hence, we limit our study to algorithms and implementations where the path computations are performed prior to deployment.

In principle, there are two strategies to obtain path computations: heuristics e.g. [3]–[5] and optimal solutions e.g. [6], [7]. The authors of [3] compute backup paths in advance to ensure network connectivity against failures of individual nodes or links. To this end, shortest paths are calculated from the adjacency matrix of the failed component. However, the authors did not intend to create node and/or link disjoint paths. This is one of the key requirements to allow resilience, as it ensures that there is no single cause of failure that can effect both primary and backup path(s). Furthermore, [5] compare different shortest path and greedy algorithms as well as iterative versions. The authors then use deterministic network calculus to prove that constraints are not violated. However, while no requirements are violated by this approach, separating optimization and requirements might result in sub-optimal routing.

In [4], Yen's algorithm [8] is used to pre-compute $k$ shortest paths. These paths are then used as input to an online, multi-objective performance optimization. However, the authors advocate for a separation of network policies and performance characteristics. Yet in safety-critical systems, network policies that guarantee safety and performance criteria affect each other.

In addition to those heuristics, there are also several different optimization goals that have been investigated. Maximizing rule sharing between flows to different endpoints has been proposed in [6]. However, traffic requirements are not taken into account, and some constraints of critical traffic may be violated with this approach. In addition, the capacity of the ternary content addressable memory (TCAM) is especially relevant for large networks.

Beyond that, some works have investigated using pre-defined configurations to address security incidents. The dynamic use of pre-defined responses to address security incidents has been studied previously e.g. [9], [10]. However, the focus of these works is on selecting the most suitable response rather than obtaining candidate responses that satisfy given requirements. Policy-based configurations and interactions between various resilience mechanisms on a high level of abstraction are described in [11]. However, consider mechanisms within the network such as firewalls are considered rather than network configurations on the data plane. As traffic requirements are not taken into account, they are not suitable for critical systems. Finally, requirements of safety-critical traffic to solve a minimum cost optimization problem are considered in [7]. However, this work only establishes two node-disjoint paths, thus only providing one back-up path in case of failure or attack. Moreover, safety considerations take precedence over cost in critical systems.

Calculating traffic routes prior to deployment is necessary for critical systems to ensure traffic demands are satisfied. Yet to the best of our knowledge, previous work does not provide a mechanism to obtain a set of configurations that consider availability, timing, and resilience. Such configurations would allow to dynamically react to faults and compromises in security without negative effects on safety-critical traffic.

## III. Problem statement

This paper considers networks as a graph $G = (V, E)$, where the network switches are represented as members of the node set $V$ and the links as directed edges $e(i, j) \in E$ from source $i$ to destination $j$. Every SDN switch has a throughput limit of $\max c_v$ and a limit of forwarding table entries $\max r_v$ due to the limited amount of TCAM memory [6]. Additionally, every link has a throughput limit of $\max c_{e(i,j)}$.

This network needs to satisfy a set of traffic demands $D$ e.g. a brake pedal input signal to the hydraulic brakes. Here, each demand $d \in D$ originates from a source node $a \in V$ and ends at a destination node $b \in V$. In addition, it has a bandwidth requirement $\mathrm{bw}$, a maximum delay during transport through the network $\max \Delta t$, and a resilience requirement that defines the number of node- and link-disjoint paths $k$. This formulation is necessary as one network may support systems of different criticality and hence different resilient requirements, such as braking signals on the one hand and cabin light control on the other. Consequently, each traffic demand $d$ consists of five elements: $d = (a_d, b_d, \mathrm{bw}_d, \max \Delta t_d, k_d)$.

The $r$th- disjoint path of the respective traffic demands is noted as a node sequence in $P_{r,d} = \{e(a, m), \ldots, e(n, b)\}$. The binary flow variable $x[r, d, i, j] \in X$ indicates whether the demand is routed through a specific link:

$$x[r, d, i, j] = \begin{cases} 1 & (i, j) \in P_{r,d} \\ 0 & (i, j) \notin P_{r,d} \end{cases} \tag{1}$$

for $r \in R_d = \{1, \ldots, k_d\}$ of the resilient paths, since multipath routing of one demand is not considered.

### A. Constraints

*1) General networking constraints:* Like all network optimization problems, several general constraints need to be satisfied. To begin with, at every node $v \in V$, the sum of the bandwidth used by the resilient paths $r \in R$, all demands $d \in D$, all outgoing edges $e(v, j) \in E$ and all incoming edges $e(i, v) \in E$ needs to be less or equal to the maximum available capacity $c_{\max}$ of the switch:

$$\forall v \in V : \sum_{r \in R} \sum_{d \in D} \sum_{e(v,j) \in E} x[r, d, v, j] \cdot \mathrm{bw}_d + \sum_{r \in R} \sum_{d \in D} \sum_{e(i,v) \in E} x[r, d, i, v] \cdot \mathrm{bw}_d \leq \max c_v.$$

While it is possible that source and destination switch are identical, rerouting cannot mitigate a failure in such circumstances and is hence not considered. Likewise the bandwidth used by the flows on all links $e(i, j) \in E$ needs to be below the maximum capacity $\max c_{e(i,j)}$ of the link between the nodes $i \in V$ and $j \in V$

$$\forall e(i, j) \in E : \sum_{r \in R} \sum_{d \in D} x[r, d, i, j] \cdot d_{\mathrm{bw}} \leq \max c_{e(i,j)}.$$

Since best-effort strategies are unsuitable for safety-critical systems, all demands have to be placed $k$-times to satisfy the availability requirement $k_d$ to tolerate $k-1$ failures in the path of the demand

$$\forall d \in D : \sum_{r \in R} \sum_{e(a_d,j) \in E} x[r, d, a_d, j] = k_d$$

at the respective source node $a \in V$ and all requests at the destination node $b \in V$ to ensure resilience against link failures and satisfy the requirements at the source $a_d$ and the destination node $b_d$

$$\forall d \in D : \sum_{r \in R} \sum_{e(i,b_d) \in E} x[r, d, i, b_d] = k_d.$$

Additionally, flow continuation is required for all resilient paths and for all demands at every node:

$$\forall r \in R, \forall d \in D, \forall v \in V : \sum_{e(j, v \neq b_d) \in E} x[r, d, j, v] = \sum_{e(v \neq a_d, j) \in E} x[r, d, v, j].$$

*2) Specific constraints:* In addition to these general constraints, a number of additional constraints arise from the application to safety-critical systems. Firstly, in order to prevent a single cause of failure affecting several resilient paths and ensure resilience, they must not use the same switches except the origin and destination between the resilient paths:

$$\forall d \in D, v \in V : \sum_{r \in R} \sum_{(v,j) \in E} x[r, d, v, j] \leq \begin{cases} +k_d & \text{if } v = a_d \\ -k_d & \text{if } v = b_d \\ 1 & \text{otherwise} \end{cases}$$

To address failures of origin or destination switches, other techniques such as demand replication may be used. These are not explicitly addressed as constraints since these are included in the demand set if required. Likewise, at most one path for each demand may use any specific link:

$$\forall d \in D, \forall e(i,j) \in E : \sum_{r \in R} x[r, d, i, j] + \sum_{r \in R} x[r, d, j, i] \leq 1$$

Beyond that, the size of the forwarding table at each switch needs to adhere to the memory limit imposed by the hardware:

$$\forall v \in V : \sum_{r \in R} \sum_{d \in D} \sum_{e(v,j) \in E} x[r, d, n, j] \leq \max r_v$$

Since only forwarding rules are saved in the TCAM, only the flows leaving the switch need to be considered. Additionally, the maximum latency i.e. the sum of the respective delay due to propagation $t_{pp}$, transmission $t_t$, processing $t_{pr}$ and queuing $t_q$ delay for each demand must not be exceeded to satisfy timing and predictability requirements:

$$\forall r \in R, \forall d \in D : \sum_{r \in R} \sum_{e(i,j) \in E} (t_{pp(e)} + t_{t(e))}) + \sum_{s \in S} (t_{pr(s)} + t_{q(e)}) \leq \max \Delta t_d$$

TABLE I
NUMBER OF VLANS FROM SOURCE SRC TO DESTINATION DST [12]

| src/dst | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | | 71 | 78 | | | | | 34 |
| 2 | 72 | | | 77 | | | | 34 |
| 3 | 90 | | | 212 | 35 | | 42 | 52 |
| 4 | | 97 | 134 | | | 37 | 35 | 48 |
| 5 | | | 80 | | | 72 | 64 | |
| 6 | | | | 82 | 61 | | 52 | |
| 7 | | | 52 | 47 | 59 | 67 | | |
| 8 | 51 | 45 | 43 | 52 | | | | |

### B. Optimization goal

Finally, the objective function for maximum resilience can be formulated as a linear optimization problem. As no links can be shared between resilient flows, the number of resilient flows can be at most the number of outgoing links from the source switch so $k_d \leq |e_{a_d,j}|$, and thus the optimization goal can be formulated to maximize the number of resilient paths:

$$\text{maximize} \sum_{d \in D} k_d.$$

## IV. NUMERICAL RESULTS AND DISCUSSION

### A. Experimental setup

The Avionics Full DupleX Switched Ethernet (AFDX) network used in this study follows the description in [12]. It is based on the on-board network of a modern transport category aeroplane able to seat several hundred passengers. The main network properties are summarized in Tables I and II. From these values, the maximum usable bandwidth per flow is defined as

$$\forall d \in D : \text{bw}_d = \frac{\text{Frame length}}{\text{BAG}}, \tag{2}$$

where the bandwidth allocation gap (BAG) defines the time between two successive frames, dividing the frame length by this value allows to normalize the bandwidth over time. The frame length indicates the size of the data to be transmitted. The number of virtual LANs (VLANs) does not match between Tables I and II due to multicast. From those tables, probability density functions have been calculated. Random samples following these distributions have been used to create 10 demand sets for a varying number of demands $N$. For the experiments described in this paper, $N$ has been increased from 98 to 1868 in regular intervals to investigate the effect of network load. With respect to maximum bandwidth, it has been assumed that links and switches can use up to 1 Gigabyte. In addition to the optimization problem described in the previous section, paths were calculated using two heuristics. The first uses the implementation of Dijkstra's algorithm provided by the *networkx*-package from Python to calculate two link-disjoint paths. The second heuristic calculates the paths in relation to the timing requirements, starting with the most time critical. For this, all simple paths (i.e. without loops) are calculated and the two with the smallest latency selected. The complete algorithm for the earliest-deadline-first (EDF) heuristic is provided by Algorithm 1.

TABLE II
BANDWIDTH ALLOCATION GAP (BAG) AND AFDX FRAME LENGTHS [12]

| BAG ms | Number of VLANs | Frame length (bytes) | Number of VLANs |
|---|---|---|---|
| 2 | 20 | 0-150 | 561 |
| 4 | 40 | 151-300 | 202 |
| 8 | 78 | 301-600 | 114 |
| 16 | 142 | 601-900 | 57 |
| 32 | 229 | 901-1200 | 12 |
| 64 | 220 | 1201-1500 | 35 |
| 1280 | 255 | >1500 | 3 |

Algorithm 1. Earliest-deadline-first (EDF)

```
# sort demands by timing requirement
sorted_demands=sort_demands_by_timing()
# iterate over all demands
for d in sorted_demands:
  # obtain all loop-free paths between
  # source and destination node:
  get_all_simple_paths(source, destination)
  # calculate the resulting latency of all
  # paths based on the link usage:
  calculate_latency(link_usage)
  # order the paths by ascending latency:
  ordered_paths=sort_by_latency()
  # make lowest latency path default path
  # and next lowest backup path:
  default=ordered_paths[0]
  backup=ordered_paths[1]
  # add selected paths to link usage:
  update_link_usage(default, backup)
```

### B. Memory Usage

The optimization problems were solved using the commercial Gurobi solver, as the upper limit of the number of constraints of the open-source `glpk` solver only allows to consider up to 1120 demands. Figure 1 shows the maximum amount of RAM that was required to solve for maximum resilience, measured using `GNU Time`. While the smallest problems require very little memory, the full demand set uses nearly 120 GB of RAM, thus necessitating high-performance hardware. For mobile safety-critical systems requiring network functionality such as aircraft, such hardware will not be available during operation. Hence, it is necessary that the required calculations are performed in advance and the results provided in the *configuration templates* introduced in this paper.

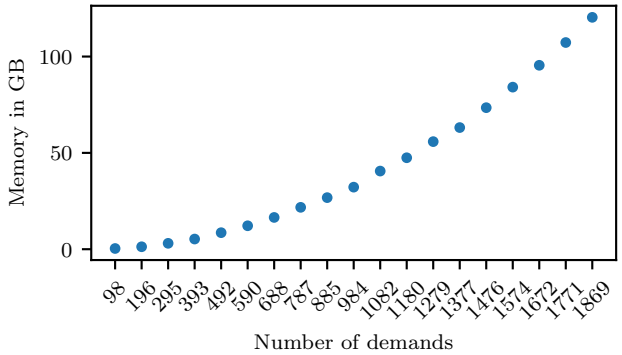


Fig. 1. RAM required to solve maximum resilience optimization problem

### C. Number of Disjoint Paths

In order to provide the highest availability, the number of resilient paths is of particular interest. Surprisingly, the number of resilient paths does not change with the demand set investigated, as depicted in Figure 2. As shown in Figures 3 and 6, the network operates well below capacity limits. Hence, the topology has a more significant impact on resilience than the size of the demand set. While the original data set already includes some redundancy, the paths thus obtained could be used simultaneously without violating constraints.

However, this does not imply that the network would only tolerate between one and four failures. Rather, this work investigated the number of node- and link-disjoint paths available. While this constraint results in fewer configuration templates, it prevents several configurations to be unusable in the event of failure because of shared resources. It is a major certification requirement to prevent any single cause resulting in system failure. The fact that there are no demands with $k = 2$ in the full demand set is not significant, since there are only between 5 and 10 demands with that resilience in the other problems.

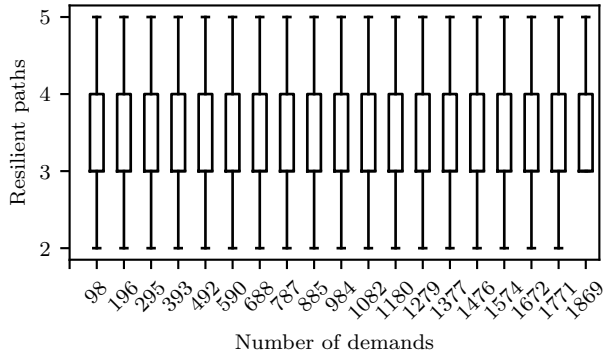


Fig. 2. Surprisingly, the number of resilient paths obtained through maximum resilience optimization does not vary with increasing the demand set.

### D. Link Usage

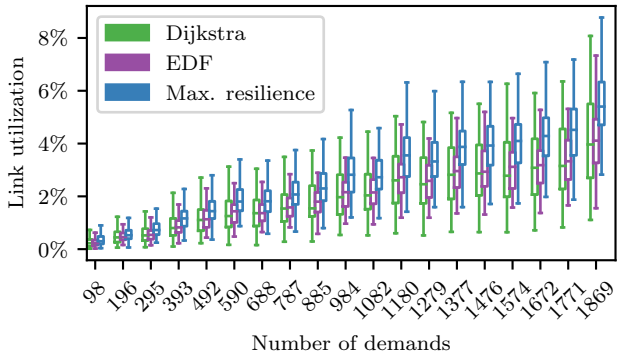


Fig. 3. Relative link usage with varying number of demands

To assess resource usage and identify the ability of the network to adapt to future needs with different demand sets,

the bandwidth used at each link has been investigated. The resulting relative link usage is depicted in Figure 3. Here, several observations can be made. Firstly, even for the full demand set with the maximum resilience the link utilization does not exceed 10% of the available capacity. While one might consider the network to be over-dimensioned, safety-critical systems and hence also the networks on board may be in operation for 20 years and beyond. Since future traffic demands may be uncertain during the design phase, it can be beneficial to provide larger capacities, as modifications during operation are difficult, expensive and may require additional/new certification. In addition, the most critical traffic must not experience congestion, hence larger network capacities are also advantageous here.

Secondly, while the mean link utilization is somewhat lower for the heuristics, it should be noted that these only provide $k = 2$, while the optimization problem provides a mean resilience of $\mu = 3.53$ over all demand sets.

Thirdly, Dijkstra's algorithm results in the smallest mean latency, but also has the largest variance of the algorithms investigated. With the EDF heuristic, the mean latency increases slightly, while the optimization yields the mean link utilization. This clearly shows the effect of the latency constraint reducing the link usage of individual links if critical demands are routed across. Consequently, this results in longer paths for less critical demands. Since the EDF heuristic only considers timing and resilience constraints, the effect here is less pronounced than for the optimization.

### E. TCAM Usage

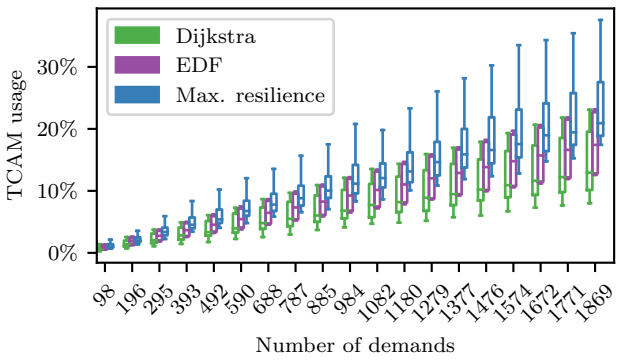


Fig. 4. Relative TCAM usage

The relative usage of the TCAM has been investigated due to its effect on power consumption [13]. This is a significant issue for mobile networks such as those on-board aircraft due to heat dissipation. It has been assumed that the memory can hold up to 8000 rules. Figure 4 shows the relative usage of the TCAM memory with varying size of the demand set. Unsurprisingly, the optimization providing maximum resilience uses the most memory here. In order not to violate timing constraints of the most critical demands, the less critical demands may not use certain links, thus leading to an increase in maximum resource usage.

Notwithstanding, considering that the maximum TCAM usage is 37.56% for the full demand set and there is at most one switch per problem that uses over 30% TCAM, the optimization still provides satisfactory results. With respect to heuristics, Dijkstra's algorithm favors shortest paths, hence fewer forwarding rules need to be placed on the switches.The consideration of latency by the EDF heuristic, on the other hand, leads to longer paths and hence to more memory being used. While the maximum resilience optimization uses the most memory, the higher number of demands that are being placed more rules need to be distributed across the switches. In theory, it would be possible to mitigate this by sharing rules between different demands (cf. [6]). However, such an approach would be less beneficial for critical systems, since different levels of criticality need to be addressed, thus reducing the potential benefit.

### F. Requirement Violations with Heuristics

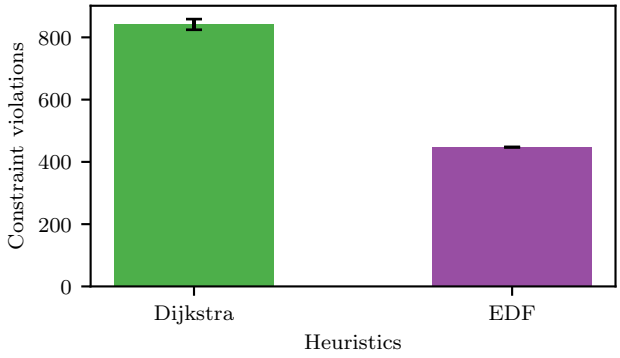


Fig. 5. Violations of requirements using alternate strategies

To evaluate the results of the optimization algorithm, two heuristics have been considered to investigate whether safe configurations could also be obtained using fast and simple algorithms, potentially allowing to find alternative configurations during operation. However, Figure 5 shows the number of constraint violations for the full demand set ($n$=1869) that occur if heuristics are used to calculate *configuration templates*. While the Earliest-Deadline-First heuristic performs significantly better than Dijkstra's algorithm ($\mu$=447.3 vs. $\mu$=841.5), these violations are clearly unacceptable for safety-critical traffic. In addition, this number of violations occurs for $k = 2$, while the optimization is able to provide $k \geq 3$ without violating requirements. For the AFDX network, all the violations that occurred were for timing requirements, since the network operates well below capacity. However, for different topologies or demand sets, violations of infrastructure constraints may occur even if a bounded delay is not the main concern.

### G. Latency

As mentioned in the introduction, timing and predictability are key features of safety-critical systems, hence the distribution of resulting latencies has been investigated. Figure 6 shows

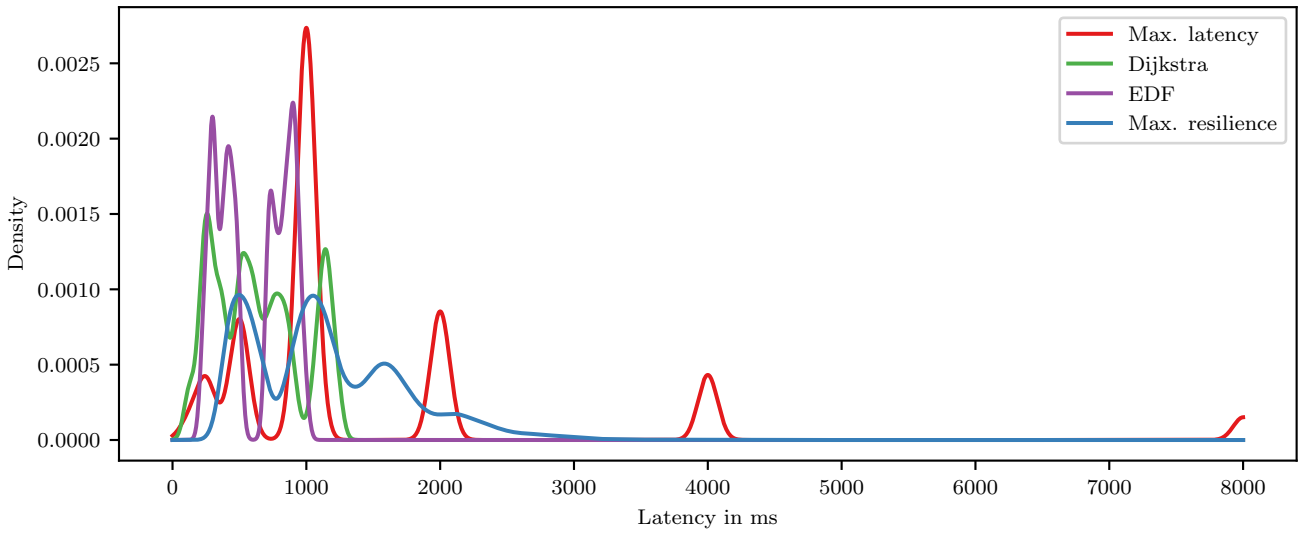


Fig. 6. Latency distributions for $|d|$=1869

the latency distributions of the delay bound(Max. latency), and the delay distributions achieved by the maximum resilience objective function as well as by Dijkstra and EDF heuristics. It should be noted that the latter two only provide two resilient paths. While Dijkstra's algorithm generally results in a lower mean latency as well as a smaller variance, the most critical timing constraints are being violated, as discussed in the previous paragraph. The EDF heuristic has smaller variance in the latency and the density has two main peaks rather than Dijkstra's four. As the smallest possible latency is selected for each demand, the paths are more evenly routed across the network. However, as less critical demands are also being placed on the most advantageous path for them, they may still violate existing constraints of more critical ones.

## V. Summary & Conclusion

This paper has shown that using heuristics to calculate *configuration templates* that deal with network failures caused by faults and security incidents results in requirements of safety-critical traffic to be violated. On the other hand, optimizing for maximum resilience yields suitable *configuration templates*. Since optimization requires too many resources to be computed during operation, *configuration templates* need to be calculated in advance to provide alternative routes in failure cases.

Future work needs to investigate how to apply these *configuration templates* safely on the network without interrupting ongoing transmissions. In order to do that, mechanisms need to be established to translate configuration templates into running configurations.

## Acknowledgments

This work was partially funded by the German Federal Ministry of Education and Research (BMBF) under Grant Nr. 16KIS0537K (DecADe).

## References

[1] M. Bouet, K. Phemius, and J. Leguay, "Distributed SDN for mission-critical networks," in *IEEE Military Commun. Conf.*, Oct 2014, pp. 942–948.

[2] A. Avizienis, J. . Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Trans. Depend. Sec. Comput.*, vol. 1, no. 1, pp. 11–33, Jan 2004.

[3] N. L. M. van Adrichem, F. Iqbal, and F. A. Kuipers, "Computing backup forwarding rules in software-defined networks," in *2016 IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov 2016, pp. 179–185.

[4] S. Prabhu, M. Dong, T. Meng, P. B. Godfrey, and M. Caesar, "Let me rephrase that: Transparent optimization in SDNs," in *Proc. Symp. SDN Res.*, ser. SOSR '17. New York, NY, USA: ACM, 2017, pp. 41–47. [Online]. Available: http://doi.acm.org/10.1145/3050220.3050226

[5] B. Cattelan and S. Bondorf, "Iterative design space exploration for networks requiring performance guarantees," in *36th IEEE/AIAA Digit. Avionics Syst. Conf. (DASC)*, Sept 2017, pp. 1–10.

[6] P. G. Kannan, M. C. Chan, R. Ma, and E.-C. Chang, "Raptor: Scalable rule placement over multiple path in software defined networks," in *16th Int. IFIP TC6 Netw. Conf.*, June 2017.

[7] C. Perner, "Network optimization for safety-critical systems using software-defined networks," in *2018 Archit. Comp. Syst. (ARCS)*, M. Berekovic, R. Buchty, H. Hamann, D. Koch, and T. Pionteck, Eds. Cham: Springer International Publishing, 2018, pp. 127–138.

[8] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quart. Appl. Math.*, vol. 27, pp. 526–530, 1969/1970.

[9] N. Herold, M. Wachs, S.-A. Posselt, and G. Carle, "An optimal metric-aware response selection strategy for intrusion response systems," in *8th Int. Symp. Foundations Practice Sec. (FPS)*. Springer International Publishing, 2016.

[10] G. Gonzalez-Granadillo, E. Alvarez, A. Motzek, M. Merialdo, J. Garcia-Alfaro, and H. Debar, "Towards an automated and dynamic risk management response system," in *Sec. IT Sys.*, B. B. Brumley and J. Röning, Eds. Cham: Springer International Publishing, 2016, pp. 37–53.

[11] P. Smith, A. Schaeffer-Filho, D. Hutchison, and A. Mauthe, "Management patterns: SDN-enabled network resilience management," in *IEEE Netw. Oper. and Manage. Symp. (NOMS)*, May 2014, pp. 1–9.

[12] H. Charara, J. L. Scharbarg, J. Ermont, and C. Fraboul, "Methods for bounding end-to-end delays on an AFDX network," in *18th Euromicro Conf. Real-Time Syst. (ECRTS'06)*, 2006.

[13] S. Zhang, F. Ivancic, C. Lumezanu, Y. Yuan, A. Gupta, and S. Malik, "An adaptable rule placement for software-defined networks," in *44th Annu. IEEE/IFIP Int. Conf. Depend. Syst. Netw.*, June 2014, pp. 88–99.