Concepts for Reliable Communication in a Software-defined Network Architecture

Ferdinand von Tüllenburg and Thomas Pfeiffenberger

Salzburg Research ForschungsgmbH
Jakob Haringerstr. 5/3, 5020 Salzburg, Austria
{fredinand.tuellenburg,thomas.pfeiffenberger}@salzburgresearch.at
http://www.salzburgresearch.at

Abstract. Not available services or service interruption could have different impact to our social life. Emails or messages which are not delivered in a proper time-frame could lead to omit a meeting or a discussion with colleagues. Interconnected CPS in different domains, like autonomous driving, smart grids, Industry 4.0, needs a guaranteed and safe delivery of information.
Nowadays distributed application in critical infrastructures such as transportation (e.g. air traffic management, train control, traffic management), financial services, or electricity systems, are often implemented in dedicated network infrastructures not using the public Internet. This leads to high expenditures (CAPEX and OPEX) for the companies to maintain these separated and dedicated telecommunication infrastructure.
Our approach in this work is to verify concepts to share the Internet, as a telecommunication infrastructure for critical and non-critical applications. This reduces the effort to implement and to manage different communication architectures. The present work develops and evaluates methods and procedures that enable high reliable communication between two endpoints over several shared telecommunication networks for future critical and uncritical applications. Our approach shows that it is possible to use the public Internet for future communication requirements in a converged network. Further innovations include the integration of novel network technologies, such as software-defined networks (SDN), programming protocol independent packet processors (P4), and self-adaptive and autonomous network management functions.

Keywords: Critical infrastructure; High Reliable Communication; Software defined Networking; Network Function Visualisation; P4; self adaptive network management; NFV orchestration

1 Introduction

Recent studies predict a tremendous increase of interconnected Cyber-Physical-Systems (CPS) for the near future [1] [2]. It is expected that huge numbers of sensors, actors and computer systems will be interconnected in order to provide new

applications in many areas. These developments give raise to questions regarding the management of the underlying complex networks, which consists of multiple management domains, contain a large number of interconnected end systems and, operate new applications which have individual requirements for their communication quality. Main reasons for the challenges in network management are proprietary control protocols and vendor and device specific configuration interfaces for switches, routers and middle boxes such as firewall, intrusion detection system (IDS), load balancers, etc. [3]. A particular challenge when configuring the underlying communication networks is given, when end-to-end connections extend over several separate and dedicated communication networks. To maintain and manage such proprietary systems, expert knowledge is necessary for multitude of devices and applications in the network. This problem intensifies, if networks grow from local area networks (LAN) to distributed wide area networks (WAN) in central managed critical system.

In this paper, we focus especially on applications within critical infrastructures such as transportation, electricity system or financial services, which we expect (and actually noticed) to also experience the predicted developments. Applications within critical infrastructures in particular desire high-dependable communication networks. Adapting the generic definition of Avizienis at al. [4], dependability of a computer network can be defined as its capability to provide the intended data communication service (regarding functional specification) at any random time an application needs to transmit or receive data. However, this is a fairly broad definition and in order to derive a concept of dependability for communication networks used by critical infrastructures, concrete requirements need to be formulated which must be met to achieve dependable communication. In this context, Avizienis named the means to achieve dependability fault forecasting, fault prevention, fault tolerance, and fault removal.

This paper presents concepts that enable dependable communication services between two end points across multiple network domains including application-specific traffic treatment for critical and uncritical applications. In order to achieve dependable communication four reliability methods are presented, based on different redundancy-levels. For the cross-domain end-to-end connectivity a new management architecture allowing for application-specific network configuration is introduced. The provided concepts are based on novel networking technologies such as software-defined networking (SDN), programming protocol-independent packet processors (P4) [5], and self-adaptive and autonomous network management functions.

The paper is structured like follows: Section 2 gives a state-of-the-art overview of technologies for dependable communication. Section 3 gives a brief overview of SDN and describes the opportunities and challenges SDN has related to dependable communication. Section 4 presents the developed reliability methods and the proposed end-to-end communication architecture.

2 State of the Art for Reliable Communication

Today several technologies are available to support user requirements on a requested communication quality. Communication availability can be mapped on delay, jitter, duplicated and lost packets.

If the link is down, packets are dropped. The transport layer of the communication stack can store packets for a retransmit or send the packet over a different link.

If the quality of the communication has inappropriate values or the service level agreement (SLA) is not complied with packet drops, delay or jitter requirements for a safe performance of an application could not be guaranteed. Several methods to manage the Quality of Service (QoS) have been implemented to guarantee the SLA, such as IntServ [6] or DiffServ [7]. Also some overlay network technologies were introduced to support QoS within big network implementations like MPLS. These methods often used for Wide area networks (WAN). In Local area networks (LAN) different layer 2 and layer 3 protocols where implemented to connect switches and nodes redundant. STP/RSTP supports a mechanism to detect loops in layer 2 switching architectures. This enables to have more physical connection at the same time using only one connection at layer 2. If the active connection fails and a timer expire the system tries to find a different layer 1/2 connection to the destination. It activates the second connection and establishes a layer 2 connection. These systems are reasonable for a disconnection time in several seconds to small values of seconds. Redundancy protocols duplicate the packets and sent the packets on two or more disjunct communication paths towards the destination. Proprietary network devices receive the duplicated packets and forward only one packet to the receiving node. This duplication approach avoids packet losses through link failures up to the amount of disjunct paths.

The presented methods are often not usable on a multi domain communication infrastructure to guarantee high reliability in a shared environment.

3 Software-defined Networks for Reliable Communication

Software-defined networking (SDN) describes an approach for programmable computer networks with the aim to support increasing dynamics in future networks together with simplified management and maintainability. Increasing dynamics in networks is a consequence of always shorter innovation cycles in networked computing environments (new technologies, new protocols, etc.) coming along with the advent of new applications (particular from the area of the Internet of Things).

The main approach of SDN is abstraction, softwarisation and centralisation of lower-level network functionality within rigorouly separated control and forwarding plane. While the forwarding plane is responsible for handling packets at network devices (forwarding, dropping, packet modification), the control plane's responsibility lies in maintaining the network's state and configuring the forwarding service of a network. Thus, the control plane logically centralizes network

control functionality and is aware of all controllable devices in the forwarding plane. The control plane is implemented by a high redundant distributed SDN controller application. The general SDN concept is depticted in Fig. 2.

The network state consists of topology information including detailled device information and existing links between CPS as well as traffic information including forwarding rules at network nodes, flows between network nodes and possibly bandwidth demands of flows.

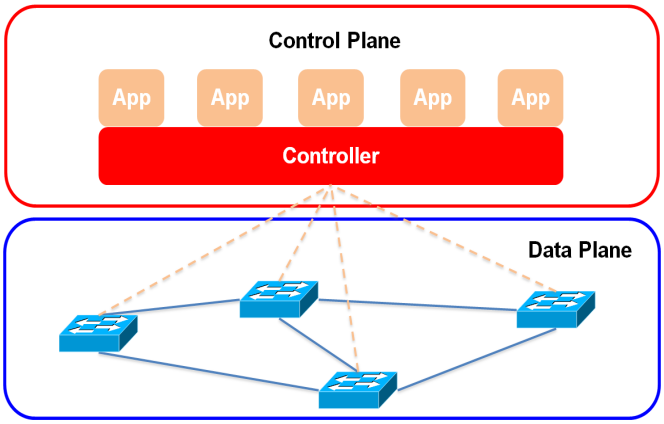

Fig. 1. SDN applications running on a central controller determine overall network behavior by deploying forwarding rules in simple network switches

Accompanied with the separation of control plane and forwarding plane, also a standardized and vendor independent configuration interface for forwarding plane devices has been introduced. The aim of this, commonly referred to as southbound interface (SBI), was to massively simplify the configuration of the forwarding behaviour within the network by making it unnecessary to configure each forwarding device using vendor specific tools and knowledge.

The separation of the control and forwarding plane also reduces the complexity of network nodes towards simple forwarding devices that are reduced to pattern matchers with the task to match incoming packets against forwarding rules and execute actions specified in these rules. The forwarding rules are created at the networks control plane making in unneccessary for forwarding devices to implement complex protocols - except of Ethernet and common wireless technologies (such as wireless LAN), which are seen as foundation of packet-switched data communication nowadays.

The following sections enlight the relationship between SDN and reliable communication in terms of opportunities and challenges.

3.1 Opportunities of SDN for Reliability

In light of the dependability concept, separation of control and forwarding plane, less complex forwarding nodes in conjunction with their abstracted configuration can be seen as a matter of fault prevention by bringing a system design in place, where several single components (network applications, standardized interface between control and data plane, forwarding devices) only need to be well developed once and can be reused several times. Of course, much effort must be put into the development of these single components in order to achieve dependable solutions.

Global view on and central programmability of the forwarding plane opens opportunities for network abstraction, which can be utilized as a means for reliable communication. SDN allows network abstraction in beyond the seven OSI layers of the Internet [8] opening opportunities to application-specific traffic treatment. This is done by introducing network slices and overlay networks for particular applications consisting only of a subset of forwarding devices, links, or even available bandwidth of a link. In doing so, each application in the network can be provided with application-specific characteristics such as network functions (Firewall, DHCP) or traffic constraints (delay, bandwidth). An example where this capability can be used for the purpose of reliable communication is, when slicing is used for the isolation of critical from non-critical traffic in the same network in order to minimize disturbing influences induced by non-critical traffic onto critical traffic.

Furthermore, the centrally maintained global network state can be utilized for verification and validation purposes under consideration of an entire network in scope of the control plane. Verification techniques run at the control plane and use a formalized model derived from the network state to examine effects of network modification onto the global network state. For instance, the side-effects (consistency of rules, safety of network configuration) a particular modification would have on the network state is computed beforehand a flow modification is actually executed on the data plane. Recent research activities tackled the concerns about flow verification in various approaches. For example, model checkers can be used in order to check if flows are correct and include no blackholes or loops, traffic isolation, or forwarding rule consistency as done by Kang et. al.[9]. Another approach with a strong focus on real-time network state checking, fulfilling similar goals is presented with VeriFlow [10] and one another mainly aimed at maintaining security invariants is Flover[11].

In addidtion, the centralized control plane can also be actively used for validation purposes. For example, validation applications using the northbound API of the control plane can be used for active performance measurements. Such tests allow to examine the forwarding behaviour that critical traffic experiences within the network. The results of these measurements can be checked against the traffic specification and in cases of requirement violations, suitable countermeasures such as traffic rerouting can be initiated. Furthermore, fault detection and fault removal techniques employed in legacy networks frequently rely on global net-

work information gathered by monitoring devices. In software-defined networks, these information can be more easily obtained from the central control plane.

Reliable Communication by OpenFlow OpenFlow [12] is the most prominent implementation of a southbound interface consisting of a standardized model for functionalities of SDN-enabled forwarding plane devices. This model is an important component for network abstraction in SDN networks. Furthermore, OpenFlow also provides a communication protocol between control plane and data plane allowing controllers and controller applications to modify the forwarding behavior of SDN devices.

The main capability OpenFlow provides, is flow matching based on specifically defined header fields and defining corresponding actions to matched packets (such as forwarding to specific port or dropping a packet). With fast-failover, metering and queueing, OpenFlow provides additional capabilities particular important for achieving reliable communication.

OpenFlow's fast-failover concepts allows to define a list of ports on which a packet may be sent out in order to reach its final destination. The switch decides for each forwarding action which of these ports is used based on the liveliness of the corresponding link.

With queueing, OpenFlow (available since OpenFlow version 1.0) supports simple QoS mechanisms where flows can be mapped to queues, which in turn has been attached to ports. The queues are used to enforce a specific forwarding behavoir for packets sent via that port (e. g. minimum data rate). Metering support of OpenFlow (available since OpenFlow version 1.3) can also be used to implement QoS capabilities such as rate limiting. In contrast to queues, meters are not mapped on ports but attached directly to flow entries. The meter then controls the (aggregate) rate of the flows it is attached to [12]. Using metering and queuing allows further opportunities for network slicing and traffic separation by assigning bandwidth portions to distinct applications. With this approach overallocation of bandwidth on single links can be avoided and combined with gloabal network information available at the control plane also bandwidth control of end-to-end paths will be possible.

For certain applications static paths can be configured through the network. These paths can be established along, e. g., a chain of selected devices, or devices which provide special capabilities for traffic - such as additional middle boxes (Hardware Firewalls) which are not directly controllable by the SDN control plane. Such static paths can be configured in a way which forbids the control plane to alter them. One scenario for applying static path would be to establish flows for non-critical Internet traffic throughout the network in order to avoid influences on critical traffic.

Dependability by P4 Programming Protocol-Independent Packet Processors (P4) [5] is an upcoming technology improving packet filtering and matching at the incoming communication interface. While packet filtering and matching with OpenFlow is limited to particular specified header types up to IP layer and

the used hardware, P4 allows maximum flexibility to filter and match future protocols for interconnecting CPS.

P4 defines a configuration language usable to define arbitrary header definitions which can be downloaded to the P4 enabled switch hardware. This enables P4 switches to process any arbitrary application layer protocol. This is of importance for CPS applications, as they frequently use special protocols such as MQTT [13] or SPDY/HTTP2 [14] which are more suitable for low energy devices (sensors, actuators). With P4 in place, networks can be flexibly adapted to the deployment of new CPS applications in areas such as IoT, Industry 4.0, etc. P4 definitions can be compiled against many different types of execution machines such as general-purpose CPUs, FPGAs, SoCs, network processors, and ASICs. Different vendors like Intel, Cavium, Pica8, metaswitch or small start-up companies like barefoot are developing hardware supporting P4. However, native P4 switches are currently not widely available. The P4 specification and language itself, is developed and maintained by the P4.org consortium, which ensures that future P4 developments are open to the public.

3.2 Challenges of SDN for Dependability

While decoupling the control plane from the data plane brings various opportunities for enhancing the dependability of SDN networks, the new concepts in place also raise several questions. The first concern lies in the conceptually centralized control of the network where two challenges arise. First, the logically centralized SDN controllers and their applications are prone to become single-points-of failure. Even if distributed controller architectures are used, it could happen that faults occuring in controller or application implementations affect the forwarding behaviour in an unintended and uncontrollable way. Second, the controllability of the network is highly dependend on a working connection between control plane and data plane. In common SDN architecuters, this connection is often established through a dedicated management network connecting forwarding devices to potentially multiple SDN controllers. Operating a dedicated infrastructure for management communication makes the setup of SDN networks more complex. The main problem however is, that individual switches become uncontrollable if the management network fails and neither reconfiguration nor monitoring of the forwarding behaviour is possible. Even if the multi-controller support of OpenFlow (since version 1.3) is utilized where each switch maintains connections to multiple controllers. In cases where the management network as whole becomes unreachable for a switch, none of the controllers can be reached.

Another concern lies in the limited capabilities of data plane devices regarding packet processing and monitoring. Although SDN switches are extremely fast in packet filtering and forwarding and also support simple metering and traffic engineering functionalities, they are commonly not capable of more complex operations such as high performance packet processing. While this is mainly due to the design of SDN, which centralizes network complexity at the control plane and keeps the data plane as simple as possible to achieve high performance

and standardized configuration, in context of dependability, however, enhanced packet processing capabilities could have benefits.

In a common OpenFlow based SDN, fault removal and some fault tolerance functionalities can only be executed by applications running on the SDN controller or dedicated middleboxes. This requires connectivity and costs at least one RTT between switch and the corresponding application. Thus it is not suitable for fault-tolerance techniques, which require deep packet inspection or packet processing. In such cases, offloading control functions to the forwarding devices can reduce the reaction time for fault removal processes, which in turn can be vital for critical traffic flows.

Applications scenarios would be the implementation of control functions, where software components or software agents are running at forwarding devices which would allow the implementation of advanced fault-removal techniques by distributed monitoring applications such as IDS/IPS or network validation applications. Other examples are per-flow encryption or enhanced metering functionalitiy including QoS surveillance of individual flows. Until now, however, it is not clear to what extent control functionality should be off loaded to the data plane devices while preserving simplicity of the southbound interface and not to overstress the devices capabilities.

One first approach towards offloading control functionalities to forwarding devices is proposed with OpenState [15]. An approach to enhance metering functionality and dynamic network reconfiguration using software agents at forwarding devices is the sFlow network monitoring solution [16]. The sFlow solution, however, is closed source and only supported by some OpenFlow switches.

4 SDN based Reliable end-to-end Communication

One of the outcomes of the work done in the projects OFSEGrid and OPOSSUM were four concepts for improving communication reliability, which are especially designed to be operated in software-defined networks.

4.1 Reliability Methods

The first developed concept is called "Managed Connectivity" (MC) and provides automatic switch-over capabilities in case network or link failures occur in a network. Similar to what is known from the rapid spanning tree protocol (RSTP), this concepts operates reactively: As soon as a lost link between two physical SDN nodes has been detected, all controlled traffic flows are automatically redirected over other paths as shown in [17]. The switch-over times in the area of several milliseconds originate mainly from the link failure detection. Link failure detection in this concept is based on the Link Layer Discovery Protocol (LLDP), which periodically sends link discovery packets throughout the network. In terms of fault tolerance, this concepts provide a means to overcome link failures in short time, although, connectivity interruptions are not completely ruled out.

The second concept denoted as "Fast Failover" (FF) provides a prepared alternate forwarding path for particular flows at a given SDN node. Beforehand of a flow installation, at each particular hop between the communication endpoints a secondary path to forward packets is computed and installed. OpenFlow protocol specifiy the us of Fast Failover Groups. In contrast to the Managed Connectivity concept, link failure detection time is significantly reduced, first by using information provided by the switch firmware instead using LLDP and, second, by avoiding reactively computing new forwarding paths. The switch-over time of this approach lays within portions of milliseconds and depends on the link down dedection of the hardware. Due to this, packet loss is reduced to a minimum. The approach has some similarities to the Fast Reroute functionality of RSVP-TE.

The third concept "Mutual Interference Avoidance" (MIA) creates disjoint paths for distinct traffic flows in order to avoid mutual interference. This concept is separating the traffic from applications exposing dynamic behavoir in terms of bandwidth, sending intervals, etc., from more critical applications which would extraordinarily suffer from link congestion potentially leading to packet loss or out of time delivery. Another use case would be the separation of paths at particular applications-specific communication stages. For instance, traffic flows related to initializing a relation between client and server part of an application (e. g., TCP handshake procedure, client registering, etc.) might use another path throughout a network as data transfers during operational use of the application. Regarding to communication links, such a procedure could provide protection against flooding or denial of service attacks. Mutual Interference Avoidance is considered as proactive approach, as flow specification and pathcomputation happens before flows are actually installed.

Finally, the Controlled Duplication and Duplicate Removal (CDUP) concept uses disjoint paths throughout (parts of) a network to send network packets belongig to the same flow in parallel towards it's destination. At some predefined node packets are sent out on multiple interfaces at the same time (fork point). Each of the packets is following a separate path throughout the network and the paths are disjunct to each other. Another node of the network acts as conjunction point, where the separated paths ends up. At this point an application is removing duplicate packets in a way that only the first of the duplicated packets are forwarded in direction of the destination.

Duplication detection can either be based on frame analysis, where the contents of the frame are inspected and all packets with equal contents are considered as duplicates and not forwarded - this approach needs additional computing capabilities at the conjunction node and must be extended if heartbeat protocols are used. Another possibility would be to add markers in the packet payloads as duplicate identifiers. These identifiers are added at the fork point and get removed at the conjunction point, thus, computional power is required at both. This approach is a proactive approach requiring configuration of conjunction and fork points beforhand a flow is installed. A SDN-based proof-of-concept has been developed as a vendor independent replacement for Parallel Redundancy Proto-

col / High availability seamless redundancy approaches. The last two concepts do not expose recovery times as they use concurrency.

In order to build a computer networking system supporting applications demanding reliable communication, multiple of these concepts can be combined within a particular network and applied even to a particular flow.

4.2 End-to-end Communication Architecture

The architecture is targeted to provide a comprehensive solution for reliable end-to-end communication in a multi-domain network environment. A main question in this aspect is how such a management solution could be designed, at least capable of:

End-to-end path computation in correspondence to the required reliability parameters of each particular application, measures for improving deployment and delivery times for emerging applications, and support for potentially autonomous and self-adaptive network reconfigurations in order to deal with dynamic changes within one or multiple domains.

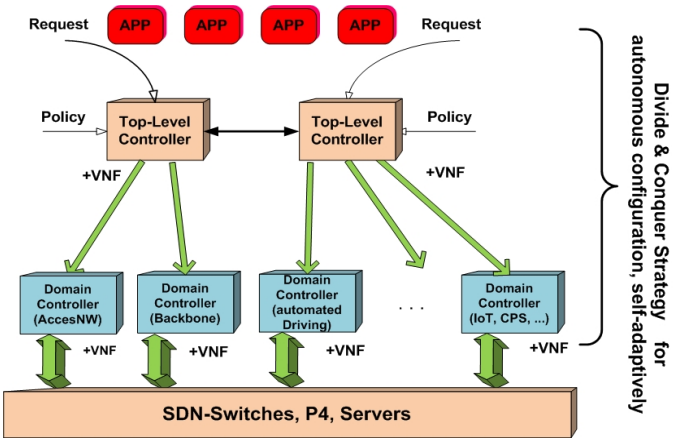

Fig. 2. Multi-domain network control architecture

Fig.2 depicts a general draft of such an architecture using a hierarchical approach to cover several domains underneath a top-level controller infrastructure. This approach allows applications to request a communication services at a top-level controller. The top-level controller has interfaces towards underlying control domains and can forward the requests according to the application requirements. The hierarchy, in general, follows a divide-and-conquer strategy for network configuration. Reliable end-to-end connectivity requires application or flow specific control capabilities finally at each particular device at the networks data plane well integrated into a cross-domain management architecture. With

SDN-enabled hardware a first step in this direction has already been taken, but with regard to reliable communication, SDN capabilities will likely not suffice to meet the requirements of future networks. For instance, the most prominent SDN configuration protocol specification OpenFlow is only capable of matching traffic flows based on packet header fields. For reliable application specific end-to-end communication, however, more fine-grained matching capabilities are required. Also more advanced computing capabilities would be beneficial to support advanced packet processing (such as deep packet inspection or packet manipulation) directly at device level. Novel network technologies, such as P4, are seen as promising candidates, but the technology itself has not been proven its operational fitness and it is currently open how practicable this P4 is and how it should be integrated in to a network control and management system.

The architecture should support self-adaption capabilities and autonomy. These are important features of future carrier networks with their ever rapidly increasing number of connected CPS and applications. Novel application have particular performance requirements regarding traffic forwarding within the network. This finally requires an application-specific configuration of the entire network. As manual configuration is infeasible raise is given to new approaches of autonomy in network control. One major goal of self-adaptively and autonomy mechanisms in network control will be providing automatisms to optimize network control corresponding to performance requirements of individual CPS and particular applications.

5 Future Work and Conclusion

The main focus of the concept is the development of a network control solution to enable reliable end-to-end connectivity across multiple network domains. The developed concept should focus to support quick deployment of critical applications in a shared communication infrastructure with strong guarantees on reliability. The main steps in future work are as follows:

Development of methodologies for reliable end-to-end communication in cross-domain environments:
The main concept of reliable communication is redundancy and the implementation of reliable end-to-end communication methods. This is based on the concepts of reactive and proactive path redundancy and packet redundancy, which has been already developed in [18]. These concepts, however, need to be adapted to the cross-domain environment.

Adoption of the Application Based Network Operations (ABNO) [19] concept providing a management solution for reliable cross-domain communication between two or more endpoints:
One main concept of ABNO is an interface allowing applications directly to interact with the network. Via this interface applications can request end-to-end connections or provide detailed information regarding their communication requirements with respect to traffic characteristics such as endpoints, bandwidth, sending interval, peak rate, burst rate, etc. The information, in turn, can be used

by the ABNO system for traffic optimization processes including networks within multiple-domains. A significant contribution to this goal is also the development of a comprehensive, ABNO based, network control architecture supporting the concepts of reliable end-to-end communication needed for emerging critical applications across multiple network domains. Although ABNO hasnt been widely adopted for network management solutions so far, some results of the STRAUSS project [20] can potentially provide first inputs here.

The Integration of emerging networking technologies SDN and P4 supports global configuration capabilities and increase configuration flexibility:
The software-defined networking paradigm has been first adopted by data centres to increase the flexibility and performance of data centre networks and to reduce cost and vendor dependencies. It is currently recognizable that the concepts also get adopted in other areas of data communication such as company connection strategies with the SD-WAN paradigm and even within carrier networks, software-defined networking gets more and more prominent. The [21] project, validates a SDN integration into a carrier network. In contrast to SDN, P4 introduces three major advancements:

1. Switches are not anymore tied to a particular set of protocols;
2. The way, how switches processes packets can be reconfigured after deployment in the field;
3. Even more hardware abstraction compared to SDN-based solutions;

Thus, the introduction of P4 leads to more flexibility regarding network control and can potentially accelerate deployment of new applications and protocols within a network, which is important for increased dynamic in future networks.

Integration of network control mechanisms to enable autonomous network configuration and self-adaptive behaviour:
It is a difficult task to ensure reliable end-to-end connectivity over a potentially longer period for a particular (critical) application. During the runtime of such an application, the network is experiencing continuous change: The number of communicating applications my change over time or even the communication characteristics of present applications change. This is exactly the point addressed by mechanisms of self-adaptive or autonomous network control. A (self-)monitoring system steadily checks if policies of the network or even of particular reliable end-to-end connections are met and a re-configuration system comes into action as soon as policies are getting violated. Thus, self-adaptive / autonomous network control also requires a metering infrastructure collecting relevant data for reliable communication. Furthermore, machine-learning (ML) mechanisms will also be considered to improve self-adaptive behaviour of the network control system. ML, for instance, can provide a means for prediction of network utilization based on historical data and occurring traffic patterns. In case high utilization will be expected in some parts of the network, affected traffic flows of reliable end-to-end connections may be rerouted precautionary. Another use case of ML might be automatic traffic classification, which can also be used for prediction purposes. Furthermore, ML can provide a decision basis

on which reliable communication methods will be selected for particular flows or particular network domains.

Provision of a prototypical implementation including all components necessary to show case and evaluate the developed solution:
For the evaluation of the developed overall system, a proof-of-concept implementation will be deployed in an extended testbed consisting of a lab environment and a connected optical-fiber network in production operation. The test scenarios will be focused on the implemented cross-domain network control system and especially in its performance to maintain reliable end-to-end connections in a dynamic network. The extended testbed environment with its connection to the production optical-fiber network connecting CPS and real end users (private homes and companies) provides the necessary dynamic, which allows first evaluations of the systems behavior under real-world conditions.

Acknowledgments. The work described in this paper was part of the project Open Flow Secure Grid (OFSE-Grid) and SDN OpenFlow-based communication system for multi-energy domains (OPOSSUM) which was funded by the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT).

References

1. A. Nordrum, "Popular internet of things forecast of 50 billion devices by 2020 is outdated," *IEEE Spectrum*, 08 2016. [Online]. Available: http://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated
2. I. Gartner, "Gartner says 8.4 billion connected "things" will be in use in 2017, up 31 percent from 2016," *Gartner Newsroom*, 02 2017. [Online]. Available: http://www.gartner.com/newsroom/id/3598917
3. N. Feamster, J. Rexford, and E. Zegura, "The road to SDN," *Queue*, vol. 11, no. 12, p. 20, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2560327
4. A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, Jan. 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1335465
5. P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: programming protocol-independent packet processors," *Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014. [Online]. Available: http://doi.acm.org/10.1145/2656877.2656890
6. R. T. Braden, D. D. D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," RFC 1633, Jun. 1994. [Online]. Available: https://rfc-editor.org/rfc/rfc1633.txt
7. F. Baker, J. Babiarz, and K. H. Chan, "Configuration Guidelines for DiffServ Service Classes," RFC 4594, Aug. 2006. [Online]. Available: https://rfc-editor.org/rfc/rfc4594.txt
8. ISO/IEC, *Information technology - Open Systems Interconnection - Basic Referece Model: The Basic Model*, ISO/IEC Std., 1996.

9. M. Kang, E.-Y. Kang, D.-Y. Hwang, B.-J. Kim, K.-H. Nam, M.-K. Shin, and J.-Y. Choi, "Formal Modeling and Verification of SDN-OpenFlow," in *IEEE Sixth International Conference on Software Testing, Verification and Validation 2013*. IEEE, Mar. 2013, pp. 481–482. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6569764
10. A. Khurshid, X. Zou, W. Zhou, M. Caesar, and P. B. Godfrey, "Veriflow: Verifying network-wide invariants in real time," in *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, 2013, pp. 15–27. [Online]. Available: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/khurshid
11. S. Son, S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "Model checking invariant security properties in OpenFlow," in *Communications (ICC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1974–1979.
12. Open Networking Foundation, "OpenFlow Switch Specification 1.5.1," Apr. 2015, technical Specification TS-012.
13. *MQTT Version 3.1.1*, OASIS Std., 10 2014. [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf
14. M. Belshe, R. Peon, and M. Thomson, "Hypertext Transfer Protocol Version 2 (HTTP/2)," RFC 7540, May 2015. [Online]. Available: https://rfc-editor.org/rfc/rfc7540.txt
15. G. Bianchi, M. Bonola, A. Capone, C. Cascone, and S. Pontarelli, "Towards wire-speed platform-agnostic control of OpenFlow switches," *arXiv preprint arXiv:1409.0242*, 2014. [Online]. Available: http://arxiv.org/abs/1409.0242
16. "sFlow.com." [Online]. Available: http://sflow.com/
17. F. von Tüllenburg and T. Pfeiffenberger, "Layer-2 failure recovery methods in critical communication networks," in *12th ICNS International Conference on Networking and Services*, F. von Tüllenburg and T. Pfeiffenberger, Eds., June 2016.
18. T. Pfeiffenberger, J. L. Du, P. Bittercourt, and A. A., "Reliable and flexible communications for power systems: Fault-tolerant multicast with sdn/openflow," in *7th IFIP Soft Computing Methods for the Design, Deployment, and Reliability of Networks and Network Applications*, July 2015, pp. 1–6.
19. D. King and A. Farrel, "A PCE-Based Architecture for Application-Based Network Operations," RFC 7491, Mar. 2015. [Online]. Available: https://rfc-editor.org/rfc/rfc7491.txt
20. V. López, T. Tsuritani, N. Yoshikane, I. Morita, R. Muñoz, R. Vilalta, R. Casellas, and R. Martínez, "End-to-end SDN orchestration in optical multi-technology and multi-domain scenarios," in *11th International Conference on IP + Optical Network (iPOP2015)*, Apr. 2015.
21. (2015, June) Open flow based system for multi energy domain. Salzburg Research ForschungsgmbH. Accessed: 2017-01-24. [Online]. Available: https://www.salzburgresearch.at/en/projekt/opossum-2/