2019 IEEE International Conference on Smart Computing (SMARTCOMP)

Magnalium: Highly Reliable SDC Networks using Multiple Control Plane Composition

Geng Li*, Akrit Mudvari*, Kerim Gokarslan*, Patrick Baker† Sastry Kompella‡,
Franck Le§, Kelvin M. Marcus¶, Jeremy Tucker†, Y. Richard Yang*, Paul Yu¶
*Yale University, USA, †UK Defence Science and Technology Laboratory, United Kingdom
‡U.S. Navel Research Lab, USA, §IBM T.J. Watson Research Center, USA, ¶U.S. Army Research Lab, USA

Abstract—**Existing software-defined SDx architectures highly depend on a centralized control plane and hence can face substantial reliability challenges in software-defined coalition (SDC) settings, in which the centralized control plane can be weakly connected to the data plane, or even disconnected from the data plane due to high dynamicity. On the contrary, distributed control planes (e.g., OLSRv2) provide autonomy but lose flexibility and global policy guarantees. In this paper, we present *Magnalium*, a novel system to achieve high reliability in SDC networks by composing multiple control planes in real-time. *Magnalium* introduces a novel, unified composition framework that uses a distributed verification to systematically generate forwarding rules in accordance with desired policy requirements. *Magnalium* also introduces several supporting components to address challenges in wireless environment and resource management. We conduct data-driven simulations, showing that *Magnalium* benefits from both centralized and distributed control planes and even reduces downtime by 65% over the most reliable individual control plane.**

I. INTRODUCTION

Software defined networking (SDN) allows a network to be effectively managed and controlled by a (logically) centralized controller with a global network view through the separation of control and data planes [10]. However, despite the benefits of the SDx architectures, there are serious problems in the centralized control planes as the network highly relies on the centralized controller [9]. When the controller becomes a bottleneck, the network can suffer from substantial performance and reliability degradation, in particular in software-defined coalition (SDC) networks with high mobility and dynamicity, lossy controller and disrupted, asynchronous communication channels [17]. For example, for the in-band control plane, the control messages may have a race condition with data messages, or the connection with the controller may get lost, resulting in unreliability and huge latency. Deployment of multiple controllers in SDN architectures for robustness is straightforward but also limited in response to network partition, as it is impossible to deploy a controller in every "potential" partitions.

On the contrary, traditional control planes use distributed protocols (e.g., OLSRv2 [4], DSDV [11]) to guarantee the autonomy and scalability in handling network dynamicity or partition. However, distributed control planes fail to provide global policy guarantees. For example, OLSRv2 computes routes by shortest path algorithm, which cannot ensure other desirable properties such as ACL, waypoints or route symmetry. Naively using distributed control planes in SDC limits the routing flexibility and may lead to serious security issues.

In this paper, we propose *Magnalium*, a novel system that composes multiple control planes in real-time. Gaining the benefits from both centralized and distributed control planes, *Magnalium* achieves high reliability in the context of SDC networks with lossy controllers, physical failures or even software bugs. The high level idea of *Magnalium* is easy to describe but realizing it has substantial challenges. 1) How can a device locally decide in real-time whether a control plane can be used? 2) How *Magnalium* works in wireless environment with features like broadcasting and high mobility? 3) Since *Magnalium* relies on running multiple control planes concurrently, how can *Magnalium* manage resource contention and usage among the various competing processes at each device?

To address the three challenges, *Magnalium* introduces a systematic, unified composition framework in each device with novel components: (1) a light-weight distributed event-driven verification module to verify forwarding information of each control plane dynamically, (2) a novel optimized verification messaging protocol to minimize broadcasting overhead, and a predictive mobility tracking model to improve system agility, and (3) an adaptive resource control mechanism to adjust resources to different processes.

We instantiate *Magnalium* system and evaluate it using experiments on large-scale simulations. We show that by composing multiple control planes in real time, *Magnalium* take the advantages of both centralized and distributed control planes. Specifically, the system experiment results demonstrate that in the event of a link failure, *Magnalium* reduces downtime by up to 65% compared to SDN, and by up to 77% when compared to distributed control planes.

II. OVERVIEW

Magnalium is a general, distributed system deployed at each network device that aims to ensure the network robustness by dynamically composing with multiple control planes. The key components of *Magnalium* are shown in Fig. 1.

1) Control plane as black boxes: In *Magnalium*, each device is allowed to run a set of control plane (CP) instances $\mathcal{CP} = \{CP_1, ..., CP_k\}$ in parallel. A control plane instance may be either centralized (e.g., SDN), or distributed (e.g.,

978-1-7281-1689-1/19/$31.00 ©2019 IEEE
DOI 10.1109/SMARTCOMP.2019.00035

IEEE computer society

Authorized licensed use limited to: Yale University. Downloaded on December 15,2023 at 14:34:57 UTC from IEEE Xplore. Restrictions apply.

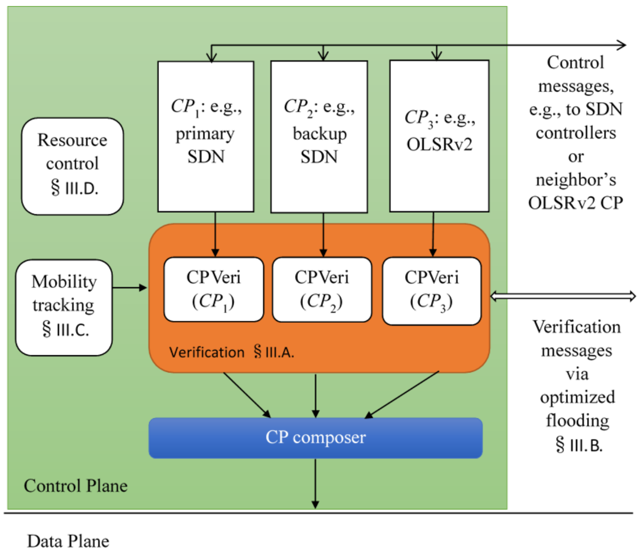

Control messages, e.g., to SDN controllers or neighbor's OLSRv2 CP
Resource control § III.D.
CP_1: e.g., primary SDN
CP_2: e.g., backup SDN
CP_3: e.g., OLSRv2
Mobility tracking § III.C.
CPVeri (CP_1)
CPVeri (CP_2)
CPVeri (CP_3)
Verification § III.A.
Verification messages via optimized flooding § III.B.
CP composer
Control Plane
Data Plane

Fig. 1: *Magnalium* architecture and roadmap.

OLSRv2). For example, the device shown in Fig. 1 is running three control planes instances, with CP_1 being a primary SDN control plane and receiving OpenFlow messages from a stable SDN controller, CP_2 also being a SDN control plane but receiving OpenFlow messages from a backup SDN controller, and CP_3 being a traditional link-state routing protocol such as OLSRv2. By design, we treat every control plane instance as a black-box. This design decision allows *Magnalium* to use any of the existing implementation (either open source, or commercial) for each of the control plane instance. *Magnalium* does not require any modification to any of them. Instead, the outcome, e.g., forwarding information base (FIB), of each control plane instance is simply used as a unified abstraction.

2) Verification in a distributed manner: Each CP is associated with a light-weight event-driven verification module (*CPVeri*), which upon detecting any change in a FIB or local forwarding state (e.g., port/link failures), verifies the correctness requirement for that CP in real time. The correctness requirement consists of certain desired properties and constraints of a subspace of packets specified by the administrators. For example, the primary SDN controller (CP_1) may be responsible for enforcing waypoint traversal for a set of flows but may have a software bug causing incorrect Openflow rules, and result in a blackhole. The goal of *CPVeri* is to detect when the CP correctness requirement is no longer satisfied. The verification can not completely rely on a centralized entity, since in the event of network partitions, the centralized entity may not be reachable. Although offloading by a centralized entity is allowed, the verification module must be able to run locally, with information exchange with neighbors. Details of the verification modules are described in Section III-A.

The output from *CPVeri* for a given control plane CP_i is (HS_i^T, HS_i^F), where HS_i^T is the set of packets for which CP_i provides correctness, and HS_i^F is the complement, i.e., $HS_i^T \cap HS_i^F = \varnothing$. A naive way to verification is to verify only forwarding rules, then the HS_i^T will be the union of the matches of all valid rules for CP_i. However, this method is too coarse-grained and provides less availability. For example, a rule matching $dstIP = D$ violates the requirement of blocking $dstPort = 22$ traffic. Simply putting $dstIP = D$ in HS_i^F can lead to false negative, i.e., blocking all traffic including $dstPort! = 22$. The correct verification result should only include $dstIP = D \& dstPort = 22$.

Match set	CP assignment
HS_1^T	CP1
$HS_1^F \cap HS_2^T$	CP2
$HS_1^F \cap HS_2^F \cap HS_3^T$	CP3

TABLE I: CP assignment table.

Implementing *Magnalium* on topologies with certain features, i.e., highly mobile and wireless environment in the SDC networks, presents certain key challenges. In *Magnalium*, we implement multiple strategies that lets us tackle such problems and optimize the performance of a multi-control plane system. **Optimized flooding.** We propose a novel approach for disseminating information, such as updates in *Magnalium*'s policies, quickly and while spending very little bandwidth. The key idea is to broadcast the message along paths that only go through the routers that need a particular policy update. For instance, if a packet is dropped due to a link failure and one of the routers in the path realizes this failure, it will send out the *Magnalium* verification message. We discuss this idea in a greater detail in Section III-B. All the routers that are informed about the failure using this message will update their verification results. **Mobility tracking.** *Magnalium* in a wireless environment uses mobility tracking to keep the routing policies updated, which helps us minimize packet losses and increase response time quickly. This approach is especially beneficial in highly mobility scenarios such as the tactical edge. Mobility tracking is discussed in Section III-C.

3) Real-time CP composer: Based on the output of *CPVeri*, we aim to provide a selection mechanism to determine a correct, usable control plane for each incoming packet in real time. Composing control planes independently for different packet sets allows for greater flexibility in the usage of more preferred control planes. To perform this composition, *Magnalium* introduces the CP composer, which takes as an input the (HS_i^T, HS_i^F) results from *CPVeri* and outputs a CP assignment table. The CP assignment table is the most intuitive, flexible and compact specification of CP composition, which maps sets of packets to the best usable control plane. Table I shows CP assignments for a three-control plane network based on the (HS_i^T, HS_i^F) results.

4) Adaptive resource control: Magnalium consists of a large number of running processes (e.g., control plane instances, verification modules, CP composer) that compete for shared resources (e.g., CPU, memory, bandwidth). Resource control, therefore, plays a critical role. Without resource control, a control plane process may use all of the device's resources, resulting in resource starvation for other processes, and ultimately causing poor overall performance. To address

Authorized licensed use limited to: Yale University. Downloaded on December 15,2023 at 14:34:57 UTC from IEEE Xplore. Restrictions apply.

this issue, we develop an adaptive resource control mechanism to adjust and allocate resources to different processes. Intuitively, when several of the preferred control plane instances already achieve the desired objectives and performance, the resources allocated to other control plane instances (e.g., CPU and control channel bandwidth) can be reduced. The details of the resource control will be described in Section III-D.

III. Technical Specifications

In this section, we present the technical details of *Magnalium*, including the distributed verification approach , the optimized messaging protocol, the mobility tracking model and the adaptive resource control scheme.

A. Verification

In *Magnalium*, correctness verification is carried out using the *CPVeri* algorithm, which allows us to create a global routing policy that is not constantly reliant on an individual control plane. While the control planes do play a role in determining the possible routes, *Magnalium* itself actively decides how the packets are forwarded. This allows every router, regardless of the availability of specific control plane, to utilize a globally defined, predetermined, rule to send the packets through a specific path. Each specific path is defined as an equivalence class entry, and every participant router for this path will store the information on equivalence class entry in its local equivalence class (LEC) table. Local equivalence class, as compared to equivalence class, only maintains the information for the downstream section of the path and does not track which intermediate routers were involved. When the packet is received by a router, *CPVeri* discerns the appropriate LEC entry corresponding to a unique downstream path using header space analysis, and this LEC entry is used for forwarding the packet.

CPVeri is capable of detecting network events such as forwarding rule update and link failure. When such an event is detected locally, an LEC update message is created, which is a tuple of the form (n, HS_{aff}, P). Here, n is the device where the event is detected, HS_{aff} is the entry which is affected by the change, and P is the new path to be taken by packets which have been following this LEC entry until now. Since each HS_{aff} corresponds to a certain property, all the routers that store and use this property need to be informed about the update. New routers that now participate in the updated path have to adopt this LEC entry into their corresponding tables, and the routers that do not participate need to remove it.

Link sensitive updating. If a router senses that a packet it recently forwarded has been dropped or that a packet using a certain path is expected to be dropped in a near future (which is decided through mobility tracking discussed in Section III-C), *CPVeri* initiates a protocol that allows *Magnalium* to update the LEC table and immediately forward the packet through another, functional channel. In a wireless environment, link metric values are used to keep track of the quality of communication channel between any two routers. The metric itself may be calculated using measurements such as SNR (signal to noise ratio), transmission time, and link bandwidth [3], [6]. These values can be used to inform the routers that a link could be failing and it would be prudent to switch to an alternative route. The mechanism for forwarding the update message in a wireless environment is discussed in Section III-B. In *Magnalium*, every time the value of the link metric falls below a predefined threshold, we immediately trigger the process that updates equivalence class table. In this situation, since MANET protocols including OLSRv2 are expected to keep their FIB updated [5], [7], we can derive new equivalence class entry using the forwarding rule of such protocols. Since we know that there are first-hop neighbors that use the same equivalence class entry, the notification of failure should to be relayed to those routers quickly, along with the information on the updated route. In order to transmit the message of this link failure, to all the participant routers for this path, we introduce the idea of *Magnalium* verification message, which is explained in Section III-B. When a router is notified about the path update by the verification method, the router updates its LEC table to include the updated routes.

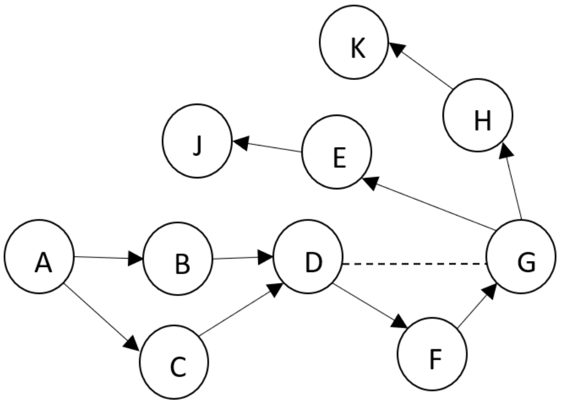
B. Optimized Flooding with Magnalium *Verification Message*

Magnalium verification messaging borrows some ideas from optimized broadcasting and multipath relay [5] employed by the OLSR protocol. This verification message will be broadcasted every time a router makes changes to a certain entry in the equivalence class table locally, and needs to convey the information about this change to all other participant routers who use this particular entry.While broadcasting can ensure that all the routers will get the messages, it will unnecessarily consume the bandwidth of the entire topology, even though the LEC update might only be warranted for a very small section of the topology. Rather than conveying the message to all the routers, selective broadcasting is employed such that only the routers that utilize this particular equivalence class entry (the one that just got updated) will participate. All other routers will immediately drop the received message.

If a router senses that the link between two routers is down, *Magnalium* initiates a search for all the LEC entries that involve this particular link. For each such entry, *Magnalium* will find an alternative route and once the route is determined, prepare a message containing the old LEC entry as well as the new one. Then this message is sent to all the one-hop neighbors. Only the neighbors that participate in old or new paths will update their LEC tables and broadcast the packets. The rest will drop the packet to avoid unnecessary congestion. This way, all the participant get the message and the bandwidth usage is minimized.

In Fig. 2, consider an LEC entry for forwarding a packet from A to H. These packets have to go through DG initially. However, the link could break and, in such cases, an alternative route (i.e., the entry ABDFGH) will have to be selected. This means updating LEC entry in all the participant routers, which includes the participants of the new and the old routes. Let us assume that D is the first participant to realize the link is broken. It is able to figure out using OLSR's FIB that

Authorized licensed use limited to: Yale University. Downloaded on December 15,2023 at 14:34:57 UTC from IEEE Xplore. Restrictions apply.



LEC entries for A:		
	Match	Path
LEC1	*DstIP =H, DstPort =22*	*ABDFGH*
LEC2	*DstIP =H, DstPort! =22*	*ACDFGH*

Fig. 2: Arrangement of routers where a failure of link DG triggers an update process. This leads to each participants (A, B, D, F, G and H) learning about the new path, LEC2, through *Magnalium* verification messaging.

an alternative route (DFG) is available. After D updates its LEC entry, it needs to inform all the participants that the updated route has to be taken from now on. We employ selective broadcasting method to inform all the participants about the update without unnecessarily consuming bandwidth of the entire topology. When D broadcasts this message, B and C will update their corresponding tables and re-broadcast the message. A will also update its LEC entry, but it will not forward the message as it knows that it is the source. Similarly, H will update its its LEC entry but will not forward the message to K as it is the destination. Knowing that it has no role to play for this LEC entry, E will immediately drop the packet. Hence, links such as EJ will not be disturbed by the *Magnalium* verification message.

Jitters. *Magnalium* verification messaging employs Jitters to ensure that packets do not collide with one another during broadcast. For instance, both B and C might be broadcasting the message for A at the same time, leading to a collision. Since the updates are expected to take place at the same time, randomized time stamps for broadcasting minimizes collision between the packets carrying the same message.

C. Mobility Tracking

To further improve the robustness of *Magnalium*, we can employ predictive tracking of the mobile routers. By keeping track of deteriorations in the link metric values, we are able to update routing tables when it appears that a link is going to be broken. Some suggested methods for tracking routers in a wireless scenario include modified Kalman filtering [18], autoregressive models [19], and radio interferometry [8]. In vehicular ad-hoc networks, velocity has been used as a tracking mechanism for mobility tracking [14]. Aforementioned techniques can be specific to certain kinds of mobility environment, such as vehicular ad-hoc networks and unmanned aerial vehicles, and attempt to optimize communication in those specific environment. In our design, we take into consideration all possible situations and put emphasis on quick and non-intensive computation the within local *Magnalium* itself. In this paper, we track the mobility of all one-hop neighbors of a router by observing the change in link metric values.

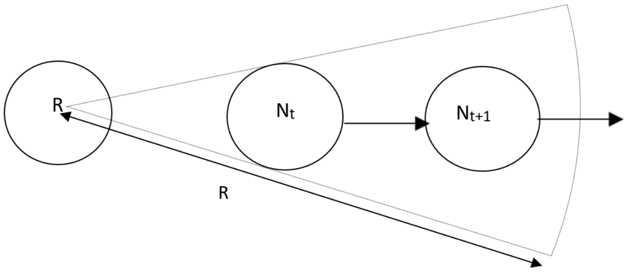


Fig. 3: Demonstration of receding 1-hop neighbor in a mobile environment. Router N is expected to stop being capable of communicating with R as it moves away and the link metric value between the routers is expected to fall below the required threshold m.

Consider a router R and its neighbor N as shown in Fig. 3. As the router N moves away from R, we can expect the link metric value to deteriorate, but the transfer of packets will continue right until the link metric value falls below the acceptable threshold, m. We can predict weather or not a packet is expected to be dropped based on current trajectory as long as the link metric value is deteriorating and m is known. So, if we observe the change in value at t and $t + 1$ to be such that we can expect the value to fall below m at $t + 2$, we preemptively initiate LEC entry update algorithm to avoid potential data loss. This way, when the router moves beyond the threshold, alternative routes will be used to deliver the packets that previously used RN as a link. This will also give us the time to let other participants of the particular LEC entry be informed about the update in routing policy.

In a wireless environment, *Magnalium* attempts to optimize the delivery of packets by employing mobility-aware forwarding . For instance, based on the mobility parameters, special rules may be employed to ensure that routers with high mobility are provided with reliable communication channels. If a router is moving very fast and consequently sees its communication links change rapidly, *Magnalium* will use a different routing mechanism to ensure that packets are still delivered.

D. Resource Control

Magnalium consists of a multiple processes (e.g., *CPVeri* processes for each control plane) competing for different resources (e.g., CPU, memory and I/O) within a device, and resource allocation among these processes is therefore critical. For example, if a *CPVeri* process of an control plane consumes

Authorized licensed use limited to: Yale University. Downloaded on December 15,2023 at 14:34:57 UTC from IEEE Xplore. Restrictions apply.

all of a device's execution resources, other processes would starve and not be able to progress.

In particular, when an event happens (e.g., link failure), and multiple *CPVeri* processes are launched, and competing for resources, how should the resources be allocated? Intuitively, important processes should be assigned more resources. However, how should importance be defined in this specific context? In addition, uncertainty can further exacerbate the problem. For example, if more resources are assigned to a verification process which after running for a period of time returns False, would such resources have been better assigned to a different process?

To address this problem, we propose an *adaptive resource allocation* that takes into account both importance and uncertainty so that *Magnalium* can operate properly, and efficiently.

Importance. We define importance of a process based on the following observations. First, a *CPVeri* process runs when there is an event causing a CP to change its FIB. Each *CPVeri* is associated with a CP, therefore, we consider the global preference order between the CPs specified by the administrator as a factor of importance. Second, the size of LEC (i.e., how many packets are belong to the LEC) for which *CPVeri* process is launched is proportional to the importance of that *CPVeri* process. Finally, the need for a *CPVeri* process to verify an LEC becomes less important if the LEC is verified by a verification process of another CP having higher preference order.

Uncertainty. A *CPVeri* process verification result can only be known after executing it. We therefore introduce the probability of a *CPVeri* that shows the likelihood of that process to verify its LEC. A *CPVeri* process with a higher probability is preferred.

Magnalium uses the aforementioned importance and uncertainty factors as the utility functions of a *CPVeri* process; and we then simply allocate the resources to the different verification processes by maximizing the sum of their utilities subject to the total amount of resources available. We assign fixed amount of resources to the CP process and the CP composer. The CP composer controls all communication between the control planes and the data plane as shown in Fig. 1. Moreover, our evaluation shows that CP processes consume very limited amount of resources compared to *CPVeri*. In contrast, the *CPVeri* processes have shorter lifetimes, and are assigned dynamic amounts of resources determined by the adaptive resource allocation. A *CPVeri* process starts when a CP detects a change in a FIB. The *CPVeri* process goal consists in verifying a correctness requirement for a LEC. The adaptive resource allocation focuses on determining the resources allocated to each of the *CPVeri* processes, based on their utility function values.

Algorithm. *Magnalium* requires to run adaptive resource allocation in every time slot to respond real-time resource demands. Thus, solving the maximization problem using integer programming is not possible due to its high time complexity. We introduce two intuition of adaptive resource allocation: First, we pick the CP with highest utility / resource demand ratio for each LEC. Second, if a verification processes of CP with higher ordering already verified a LEC, it is unnecessary to run lower ordered verification processes for that LEC. We implement a greedy algorithm to allocation problem based on the two main ideas of the adaptive resource allocation described above. The algorithm consists of two parts. First, it calculates the utility value for each verification process *CPVeri*. Second, it allocates available resources to the first verification processes in waiting for resources for each LEC using 0-1 integer knapsack algorithm. The items of knapsack have resource demand as ”weights” and utility / resource demand ratios as ”values”. The capacity of the knapsack is the available resources.

	SDN	*Magnalium*	OLSR
Average Downtime	27.34 ms	9.65 ms	42.53 ms

TABLE II: Average downtime for different CPs

IV. Evaluations

In this section, we evaluate the effectiveness of *Magnalium* in recovering from link failures.

Environment. We run the experiment in a virtualized environment on a Rocketfuel topology [2] (AS 1755). The routers runs as a separate Docker containers, with multiple CPs: an OLSR, and an SDN. Our evaluation shows the benefits of using multi CPs over an individual CP, and so, the evaluation could be replicated for a wider range of distributed protocols.

Methodology. We randomly select a pair of nodes and generate UDP traffic between them by running iperf [1].For the SDN CPs, we use precomputed paths. We then randomly select links to fail on one of the paths used by the UDP traffic. For the SDN CPs to recover from the failures, we implement a reactive approach. That is, when a failure happens, the device that detects it sends a control message to the controller to recalculate alternative forwarding paths and update the FIB of the affected devices accordingly [12]. For each run, we measure the downtime defined as the amount of time between the moment the destination stops receiving packets because of the failure to the moment the receiver starts receiving packets again. We obtain an average value from multiple runs.

Results. Our evaluation shows that while the average downtime of the SDN CP is 27.34 ms, the average downtime of OLSR is 42.53 ms, and *Magnalium* is even lower than both of them with 9.65 ms. *Magnalium* can reduce the downtime of SDN by 65%, and that of OLSR by 77%. The gain comes from two aspects. First, *Magnalium* has the fastest recovery time compared to individual CPs. That is, in some runs, the SDN CP recovers faster, whereas in other runs, the OLSR CP recovers faster; and in every case, *Magnalium* switches to the one that recovers the fastest. Second, *Magnalium* provides diversity between CPs. That is, if a failure does not affect all CPs, *Magnalium* can switch to a non-failing CP without any throughput loss.

***Magnalium* on EMANE.** Once *Magnalium* is implemented using OLSR, the next step is to implement *Magnalium* on EMANE (Extendable Mobile Ad-hoc Network Emulator). EMANE is a software that allows the users to test network

Authorized licensed use limited to: Yale University. Downloaded on December 15,2023 at 14:34:57 UTC from IEEE Xplore. Restrictions apply.

architectures in a realistic wireless environment. [13] So implementing *Magnalium* on EMANE will allow us to verify that this system works reliably and efficiently when subjected to conditions prevalent in the tactical settings. For implementation, we run *Magnalium*, along with the desired applications such as SDNs, OLSR and EMANE in a virtual environment that is capable of simulating the wireless tactical environment. In this setting, we will be able to ensure that *Magnalium* provides a robust, real-time verification mechanism. The tight integration of *Magnalium* and EMANE is our future work.

V. Related Work

Systems composed of layers of multiple control planes have also been proposed, such as [15], [16]. In these systems, in the event of the failure of a control plane, the composition allows for usage of a backup control plane to ensure reliability. Tilmans, et al. [15] designed an architecture, "IGP-as-a-Backup to SDN", in which each device contains a local software agent that runs a link-state IGP as a backup to traditional SDN routes. However, their model does not allow for any policy guarantees aside from traditional SDN, and also restricts the distributed protocol to only link-state protocols. Vissicchio, et al. [16] provides a new classification scheme of routing protocols based on their interactions with the Routing Information Base (RIB) and Forwarding Information Base (FIB) of network devices and provide sufficient conditions to prevent routing loops and black holes. However, their analysis does not provide guidance or mechanisms on guaranteeing other desirable properties such as waypoints or route symmetry. In contrast, *Magnalium* utilizes a unified distributed verification approach associated with each control plane to support a broad range of correctness properties for each packet in the network.

VI. Conclusion

Achieving high reliability is essential for SDC networks in the event of software errors or hardware failures. *Magnalium* offers a novel multiple control plane composition framework in SDC networks to achieve high reliability in real-time. *Magnalium* utilizes a distributed verification to generate forwarding rules based on the policy requirements while ensuring the correctness. *Magnalium* also introduces novel components to address challenges like high mobility in wireless environments and resource allocation for the competing processes at each device. Our simulations show that *Magnalium* reduces downtime by up to 65% compared to SDN, and by up to 77% compared to OLSR.

Acknowledgment

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] Iperf, a tool for active measurements of the maximum achievable bandwidth on ip networks. https://iperf.fr/.

[2] Rocketfuel: An isp topology mapping engine. http://www.cs.washington.edu/research/networking/rocketfuel.

[3] R. Aguero, J. A. Galache, and L. Munoz. Using snr to improve multi-hop routing. In *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, pages 1–5, April 2009.

[4] T. Clausen, C. Dearlove, P. Jacquet, and U Herberg. The optimized link state routing protocol version 2, 2014.

[5] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr), 2003.

[6] Saumitra M. Das, Himabindu Pucha, Konstantina Papagiannaki, and Y. Charlie Hu. Studying wireless routing link metric dynamics. In *Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference, IMC 2007, San Diego, California, USA, October 24-26, 2007*, pages 327–332, 2007.

[7] Y. Huang, S. N. Bhatti, and D. Parker. Tuning olsr. In *2006 IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5, Sep. 2006.

[8] Branislav Kusy, Janos Sallai, Gyorgy Balogh, Akos Ledeczi, Vladimir Protopopescu, Johnny Tolliver, Frank DeNap, and Morey Parang. Radio interferometric tracking of mobile wireless nodes. In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, MobiSys '07, pages 139–151, New York, NY, USA, 2007. ACM.

[9] G. Li, Y. Qian, C. Zhao, Y. R. Yang, and T. Yang. Ddp: Distributed network updates in sdn. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 1468–1473, July 2018.

[10] Geng Li, Qiao Xiang, Christopher Dearlove, and Y. Richard Yang. A self-organizing sdn architecture for mobile tactical edge networks. https://dais-ita.org/sites/default/files/S_028-paper.pdf.

[11] Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, SIGCOMM '94, pages 234–244, New York, NY, USA, 1994. ACM.

[12] Sachin Sharma, Dimitri Staessens, Didier Colle, Mario Pickavet, and Piet Demeester. A demonstration of fast failure recovery in software defined networking. In *International Conference on Testbeds and Research Infrastructures*, pages 411–414. Springer, 2012.

[13] Eric Schreiber Steven M. Galgano, Kaushik B. Patel. EMANE user manual 0.8.1. https://downloads.pf.itd.nrl.navy.mil/docs/emane/emane.pdf, 2013.

[14] T. Taleb, E. Sakhaee, A. Jamalipour, K. Hashimoto, N. Kato, and Y. Nemoto. A stable routing protocol to support its services in vanet networks. *IEEE Transactions on Vehicular Technology*, 56(6):3337–3347, Nov 2007.

[15] Olivier Tilmans and Stefano Vissicchio. Igp-as-a-backup for robust sdn networks. In *Network and Service Management (CNSM), 2014 10th International Conference on*, pages 127–135. IEEE, 2014.

[16] Stefano Vissicchio, Luca Cittadini, Olivier Bonaventure, Geoffrey G Xie, and Laurent Vanbever. On the co-existence of distributed and centralized routing control-planes. In *Computer Communications (INFOCOM), 2015 IEEE Conference on*, pages 469–477. IEEE, 2015.

[17] Qiao Xiang, Franck Le, Yeon-Sup Lim, Vinod Mishra, Christopher Williams, Y Richard Yang, and Hongwei Zhang. Opensdc: A novel, generic datapath for software defined coalitions. pages 1–6, 10 2018.

[18] Z. R. Zaidi and B. L. Mark. A mobility tracking model for wireless ad hoc networks. In *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, volume 3, pages 1790–1795 vol.3, March 2003.

[19] Z. R. Zaidi and B. L. Mark. Mobility tracking based on autoregressive models. *IEEE Transactions on Mobile Computing*, 10(1):32–43, Jan 2011.

Authorized licensed use limited to: Yale University. Downloaded on December 15,2023 at 14:34:57 UTC from IEEE Xplore. Restrictions apply.