Original scientific paper

Evaluation of Reactive Service Function Path Discovery in Symmetrical Environment

Martins Mihaeljans, *Student Member, IEEE*, and Andris Skrastins, *Member*, *IEEE*

***Abstract* — In this paper we continue our study of path discovery process for Service Function Chaining (SFC) in Software Defined Network (SDN). By default, service function (SF) paths are established proactively - before data transmission takes place. We have argued that this constraint can be eliminated with the use of our proposed Reactive SF path discovery approach. Such SFs as network address translation (NAT) or stateful firewall (FW) are SF path's symmetry dependent requiring a visit of both ingress and egress flows. Thus, we evaluated SF path discovery processes in Mininet emulation network. Outcome of this study is a comparison of proactive and reactive SF path discovery processes for both asymmetrical and symmetrical SF paths. It shows that even in symmetrical environment reactive SF path discovery has a higher probability of successful SF path detection.**

***Keywords* — Network Service Header, Network function, Packet encapsulation, Symmetrical connection, Software Defined Network, Service Function Chaining.**

I. Introduction

End-to-End service delivery is a complicated task which can stress the forwarding plane and lead to unexpected errors, due requirement for data flow to cross multiple network functions (NF). Service Function Chaining (SFC) architecture [1] defines NFs as service functions (SF)s. Common examples of SFs are firewall (FW) and load balancer (LB). SFC is a method of data flow steering through ordered SF path derived from possibilities of Software Defined Networking (SDN) and Network Function Virtualization (NFV). SDN allows network operators to implement complicated network topologies by use of virtual NFs. This ability is a key benefit for operators to prefer SDN over legacy networks.

This paper is a continuation of our study of Reactive Service Function Path Discovery Approach in Software Defined Network [2]. We have argued that preset of a complete SF path policy before initial data flow arrival is an unnecessary requirement. This study focuses on previously untangled question of SF path symmetry's effect on path discovery process.

Outcome of this study is a comparison of proactive and reactive SF path discovery process for asymmetrical and symmetrical SF paths, indicating that even in symmetrical environment reactive SF path discovery has a higher probability of valid SF path discovery.

Paper received May 22, 2022; revised June 27, 2022; accepted June 28, 2022. Date of publication August 05, 2022. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Grozdan Petrović.

This paper is revised and expanded version of the paper presented at the 29th Telecommunications Forum TELFOR 2021 [2].

Martins Mihaeljans is with the Riga Technical University, Azenes str. 12 - 325, LV1010 Riga, Latvia,
(e-mail: martins.mihaeljans@edu.rtu.lv).

Andris Skrastins is with the Riga Technical University, Azenes str. 12- 325, LV1010 Riga, Latvia (e-mail: andris.skrastins@rtu.lv).

II. Related Works

H. Hantouti and N. Benamar [3] have argued that SF path symmetry has a great effect on network service delivery. In their study a reverse path calculation algorithm is introduced, yet SF path discovery is not considered. The study also suggests that having fully symmetrical SF paths is redundant and negatively affects the performance of Service Function Forwarders (SFF). Authors propose the use of partially symmetrical SF paths as a solution.

Authors of study paper [4] state that SF placement in virtual environment allows flexible SFC deployments. In such deployments resource cost between different SFs is unequal.

For proper resource cost computation, authors have proposed an algorithm that is based on the use of predefined candidate path sets. Although, service applicability and probability of valid service discovery has not been considered, research does outline the importance of distinguishing valuable SF paths.

Study [5] finds partial SF path symmetry to be an adequate choice for enabling network slicing technique in 5G networks. Unlike our emulation environment, SF symmetry requirement is detected by a controller and not the SFs themselves. Such configuration might require an extensive integration of each element into the proposed framework. Which is an obstruct that might lead to inability to use plug-and-play systems.

It's pointed out that symmetrical SF paths are mandatory for some SFs. Given examples are stateful FW and Intrusion Detection System (IDS). Such SFs require data flow to traverse them from both ingress and egress directions. An opposite are SFs concerned only with the incoming data flow such as spam checks and URL filters. Use of partially symmetrical SF path would result in latency reduction, greater policy flexibility and lowered deployment cost.

Study [6] describes the necessity of dynamic topology changes in SFC domain as a link failure can cause a complete service outage. Similarly, study [7] introduces a self-recovery scheme for SF failure scenarios. Authors of study discuss an objective of SFs hosting classification rules and participating in management of path policy.

SF participation in path policy update process has been covered in our previous research study [2] as well. Unlike study [7] we do not encourage hosting classification rules at SFs as it might not only prolong policy update process but also lead to policy synchronization issues and requirement for SF hierarchy to eliminate concurring extensive path policy update requests.

III. Service Functions and Problem Statement

A. Open-source Virtual Switch and OpenFlow

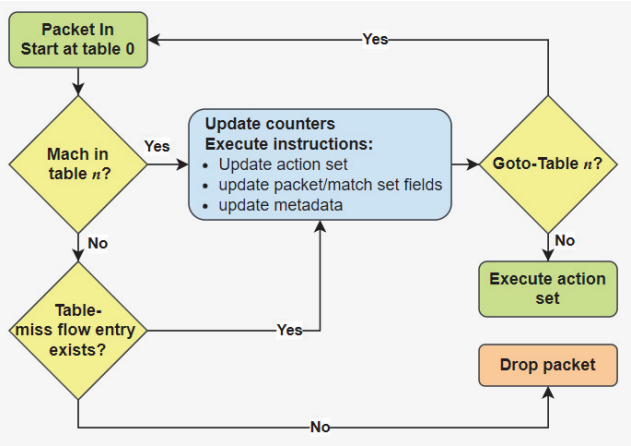

Fig. 1. Open-source virtual switch [8].

An open-source virtual switch (OVS) [8] enables packet processing in SFC domain. OVS working principle is shown in Fig. 1. On data flow arrival, OVS primarily determines if it has a match in existing path policy.

Miss-Flow entry is a multipurpose rule in path policy that is used only if no match against other rules is found. In flow table without Miss-Flow entry unmatched flows are dropped. In our emulations we used Miss-Flow entry to forward data flow through all SFs. Imitating an alternative method to discovery processes. Use of Miss-Flow entry does not require an additional encapsulation to be applied on data flows.

B. Service Function Chaining

SFC domain architecture shown in Fig. 2 implies no restrictions regarding path symmetry or forwarding direction. For example, SF1 can be crossed many times before data flow is forwarded to the destination.

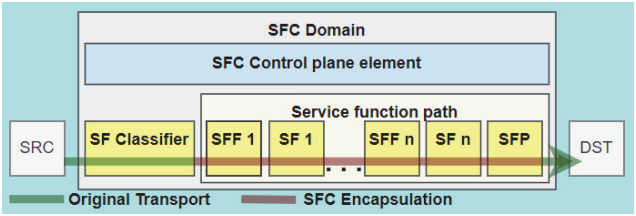

Fig. 2. Service Function Chaining domain.

SFC domain consists of classifier (CL) for adding SFC encapsulation on an incoming data flow and selecting a proper SF path. SF forwarder (SFF) for forwarding a data flow from one SFC element to another. Service function (SF). SF proxy (SFP) for SFC encapsulation removal when a data flow is destined to leave SFC domain. We have greatly explained SFC domain elements in our previous work [9] where we introduced two different SFC encapsulation application methods:

- **Initial-Partial encapsulation** – SF encapsulation that is applied at the time of initial classification and continued partially in SF path.
- **As Required encapsulation** – SF encapsulation that is applied only if SF path differs from underlying transport network topology.

C. Complicate Service Functions

Complicate SFs require for data flow to traverse them for both directions. A stateful FW is an example when an ACK won't be accepted if ACK SYN packet from a reverse direction wouldn't have been received prior. IDS is a SF that relies on the capability of comparison between received data flow and signature database for the detection of malicious activities.

An opposite are SFs requiring only for the incoming data flow to traverse them. Such SF is an Anti-spam check. It verifies the incoming e-mail and is not interested in two-way dialog between servers.

Three different SF path types shown in Fig. 3 are:

1. **Asymmetrical SF path** – Data flow crosses SFC domain only in one direction.
2. **Partially symmetrical SF path** – Data flow crosses SFC domain in both directions, but path taken in each direction can differ.
3. **Symmetrical SF path** – Data flow crosses SFC domain in both direction via the same path and returning data flow visits SFs in reverse order.

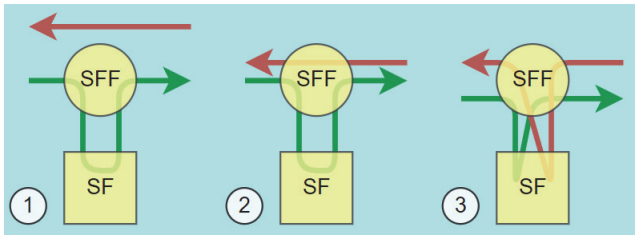

Fig. 3. SF path types.

Asymmetrical environment – emulation network setup where SFC domain ingress data flow is steered trough SF path but returning data flow traverses topology without entering SFC domain. A compliant type is **Asymmetrical SF path**.

Symmetrical environment – emulation network setup where SFC domain is traversed for both ingress and egress directions. Path types are **Partially symmetrical SF path** and (Fully) **Symmetrical SF path**.

D. Control plane for Service Function Chaining

Control interfaces shown in Fig. 4 and Fig. 7 enable complete SF path synchronization among all SFC elements. With the use of control interfaces SFC data plane components share information related to SF activity and availability with the controller and between each other. While Fig. 4 depicts the general purpose of each interface, Fig. 7 shows usage in our emulation network topology.

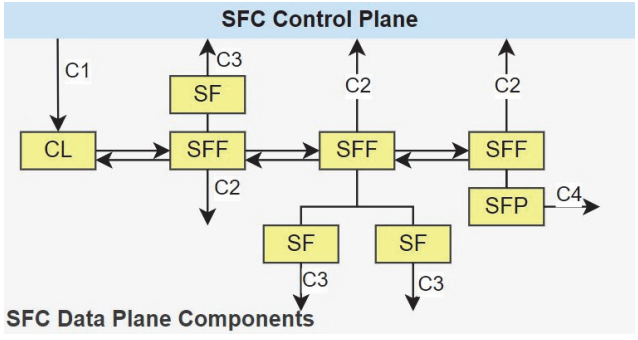

Fig. 4. SFC Control interfaces [10].

SFC Control plane components and requirements draft [10] define SFC classification rule as an entry in SF classifier's policy for binding an incoming data flow to an appropriate path. These entries fill up SF path's policy table that reflects data flow forwarding policy enforced by SFF. Interfaces from Fig. 4 can be described as follows:

- C1 – CL classification rule management
- C2 – SFF data flow forwarding management
- C3 – SF interaction interface
- C4 –SF Proxy instruction interface

E. Problem statement

A proactive SF path discovery method prohibits the use of dynamic SF paths. This constraint derives from the objective where in a default case (described by SFC architecture) SF encapsulation is added to an arriving data flow only if a previously defined classification policy exists.

In Fig. 5 a proactive SF path discovery method is shown. At data flow arrival incoming packets are matched against OVS flow table. When a match is found a forwarding decision is made and processing continues at SF path's policy table where an encapsulation is added.

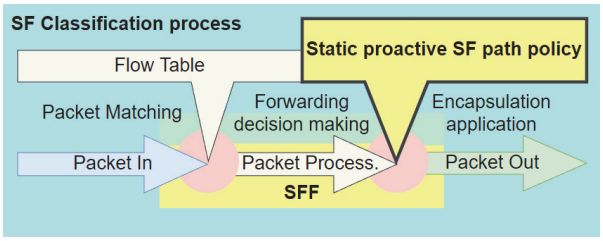

Fig. 5. Proactive SF path discovery.

A manually created proactive SF path policy is unable to make a data flow reclassification. Thus, dynamic path discovery cannot be achieved. Such a constraint does negatively affect a possibility to detect a valid SF path for an incoming data flow. Returning data flow can only find a valid SF path if required policy has been previously setup.

IV. Reactive Service Function Path Discovery

In our previous study Reactive Service Function Path Discovery Approach in Software Defined Network [2] we proposed **Reactive SF path discovery method** (shown in Fig. 6) as an alternative to proactive discovery. Our method gains an additional subtask for reverse path mapping for the use of (Fully) symmetrical SF paths. Partially symmetrical SF paths are discovered via a reactive SF path discovery process for returning data flow.

As shown in Fig. 7, reactive SF path discovery relies on SF's ability to inform the controller about success or failure in service application by use of control interface C3. The information also consists of received packet which SF had not been able to process. A controller does the packet process. Afterwards, it updates SFF path policy tables through interface C2 and SF classifier through interface C1. A controller also updates SF proxy for proper SF encapsulation removal at the egress through a control interface C4. SF path generation automation is achieved via the use of counting available SFs and reclassifications made for a received data flow, and mapping NSH headers.

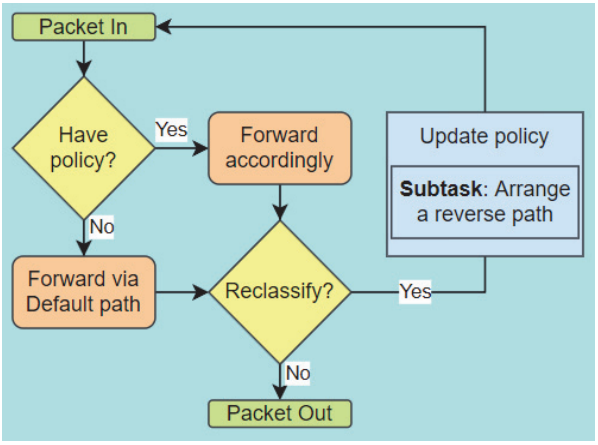

Fig. 6. Reactive SF Path discovery.

Work principle consists of four steps (see Fig. 8):

1. Initial SF path detection – All previously unknown data flows are steered through a default SF path.
2. Updating SF path policy – If SF indicates that service has not been applied and SF path is faulty then SF paths policy update is enforced among all SFC domain elements.
3. Reclassifying data flow – SF path policy update results in data flow reclassification that enables subsequential SF detection.
4. Subsequential SF path detection (an equal process to initial SF path detection only exploited on previous detection failure) – steering traffic via a newly created SF path.

Steps 2 to 4 are optional if an arriving data flow has had a previously created SF path policy at the time of arrival, otherwise they are essential in the detection of SF path with a valid SF application.

This discovery process can be repeated as many times as required for a valid SF path to be found. Or it can be disrupted in cases where time to leave (TTL) value of Network Service Header (NSH) [11] encapsulation is reached, or cycle disruption occurs due to unpredictable matters connected to communication specifics like session timeout, low delay tolerance and other.

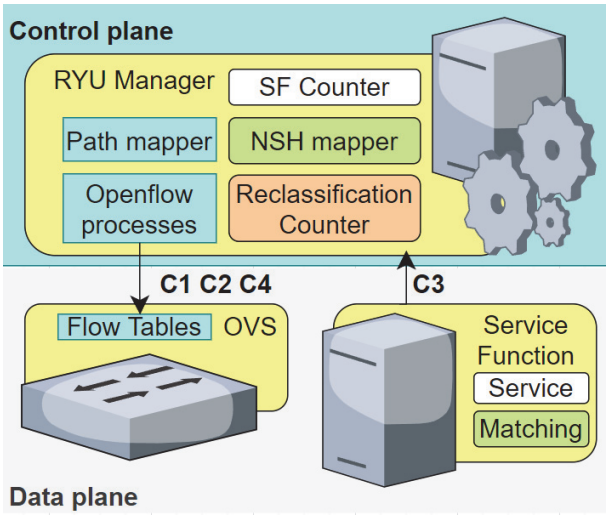

Fig. 7. Control interface usage.

V. Network Emulation Setup

SFC domain topology, emulated in Mininet network emulator is shown in Fig. 7. It consists of:

- 3 OVS switches serving as SFFs.
- 2 host serving as a data flow source and destination.
- 1 to 5 hosts (depending on emulation requirements) serving as SFs.
- Ryu controller [12] serving as a SFC control element.

SFFs connected to a data flow source and destination for ingress traffic and vice versa for egress traffic served also as SF classifiers and SF proxies. All SFs were connected to SFF at the middle of the topology (OVS with only SFF function in Fig. 8). We did emulations with up to 5 SFs in topology. Each emulation differed from one another with the count of SFs. SF path's length stayed 1 SF long throughout all emulations. Packet processing in SFs was made by the use of SCAPY packet crafting tool [13] and its contributory library that contains a NSH protocol.

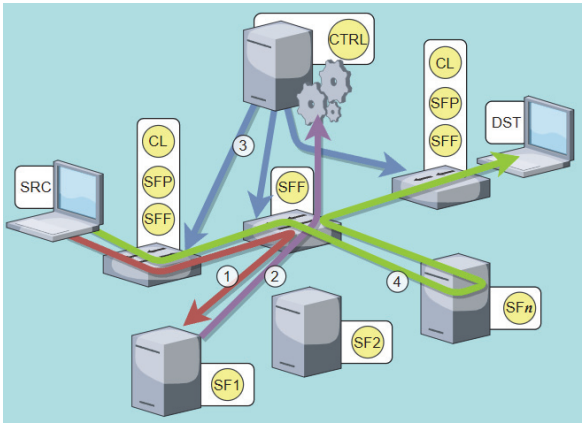

Fig. 8. Network emulation topology.

For a data flow to be steered through a desired path other than the one taken by the direction of underlying transport topology an additional packet encapsulation is required. We used Network Service Header (NSH) protocol as SF encapsulation in our emulation setup as it is complying with requirements implied by both SFC architecture and SFC control plane components and requirements draft.

Ryu SDN controller (CTRL in Fig. 8) enabled dynamic SF path creation with the use of gathering network related information from all SFC elements. This capability is the key benefit of network automation that relaxes the necessity of manual configuration from a network administrator.

However, in reactive SF path discovery a single proactive SF path rule entry is required. It is the default SF path that all previously unknown data flows will take at the initial SF path detection. It is configured manually.

VI. Results

SF path symmetry is a fundamental SF path attribute. It affects all SF path parameters. The direct effect of SF path symmetry on a valid SF path discovery is revealed by evaluation of such parameters as the probability of valid SF path discovery and SF matching ratio.

As SF path's symmetry is not a performance dependent subject (in the context of this study) we solely focused on distinguishing the impact of symmetry constraint on SF path detection. We categorized this section according to measured values which are:

- **Probability of valid SF path discovery** - a likeliness of discovering a valid SF path out of all possible SF paths available.
- **SF matching ratio** – indication of the unequalness between all possible valid SF path detections and required reclassification count to distinguish them.
- **Elapsed relative time** – a time interval required for SF path discovery process.
- **SFC generated overhead** – a cumulative value of overhead gained by the use of SFC encapsulation.
- **SF path discovery successfulness** – a scale between the probability of valid SF path discovery and SF matching ratio.
- **SF path discovery expenses** – a scale between elapsed relative time and SFC generated overhead.

A. Probability of valid SF path discovery

The probability of valid SF path discovery is shown in Fig. 9. Initial SF path detection is equal for all discovery methods as reclassification cannot occur on a first occasion (step 1 in Fig. 8) in a single conditioned path policy. In asymmetrical environment all SF path discovery methods at initial SF detection have a probability of 0.5 as a single decision can only lead to either a valid or a faulty SF path.

In symmetrical environment all discovery methods at initial SF path detection gain a lower value of 0.33 as a path for returning data flow must be considered as well.

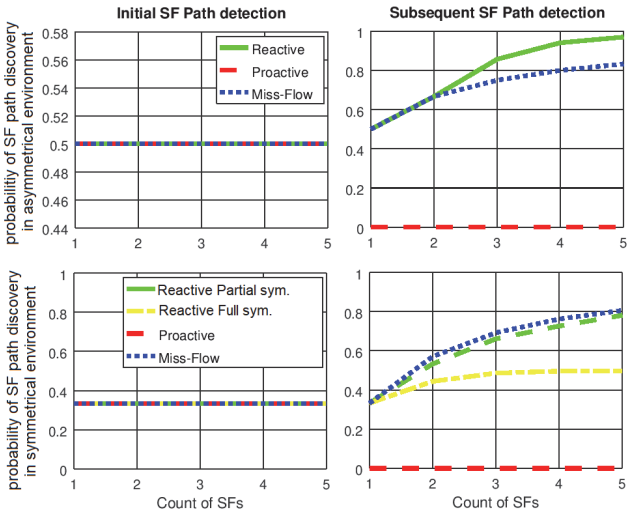

Fig. 9. Probability of valid SF path discovery.

In asymmetrical environment at subsequent SF path detection both Miss-Flow and reactive discovery shows a gradual increase of their capability of finding a successful SF path, but proactive discovery falls to a value of 0 as it is incapable of data flow reclassification.

In symmetrical environment at subsequential SF path detection a similar pattern to asymmetrical environment is encountered for all discovery processes. Reactive discovery with full symmetry climbs half the growth in value of partial symmetry, because returning path cannot do reclassification due constraint that it must be equal with a forwarding path taken by an incoming data flow.

B. SF matching ratio

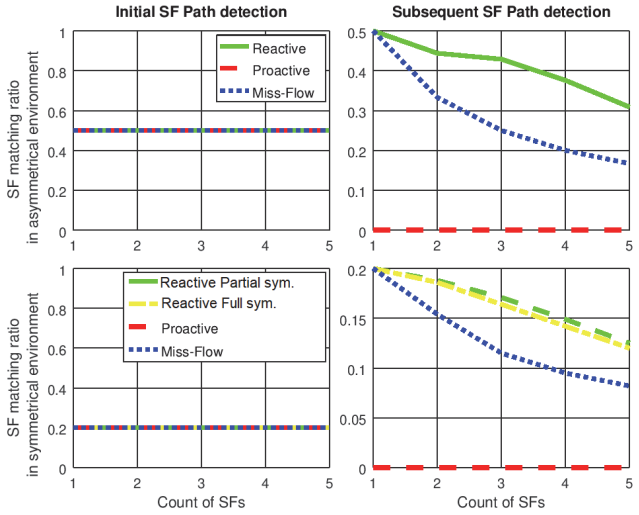

Fig. 10. SF matching ratio.

For matching ratio (Fig. 10) in both asymmetrical and symmetrical environments at initial detection all discovery processes held equal values (0.5 in asymmetrical env. and 0.2 in symmetrical env.). At subsequent SF path detection reactive discovery processes gain a higher ratio as they require for additional reclassification in comparison to Miss-Flow. Proactive SF path discovery falls to a value equal to 0 as it cannot do reclassification.

C. Elapsed relative time

Fig. 11 shows the time required for service application both spent while searching for an SF path and time required when the SF path is found.

While static methods as proactive discovery and Miss-Flow do not change at either of time objectives, reactive discovery processes do gain an advantage over Miss-Flow entry after a valid SF path is detected both in asymmetrical and symmetrical environments.

Relative time values are in the range from 10ms to 100ms. The use of relative time values enables the calculation of methods effectiveness in other setups.

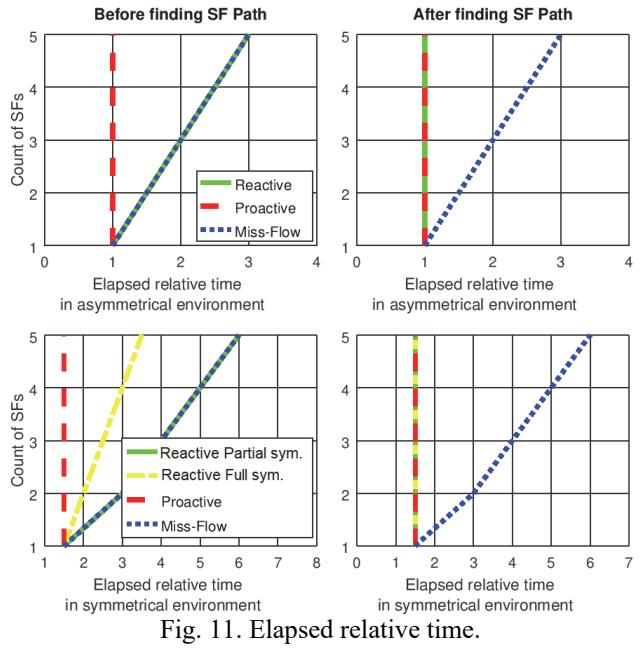

Fig. 11. Elapsed relative time.

D. SFC generated overhead

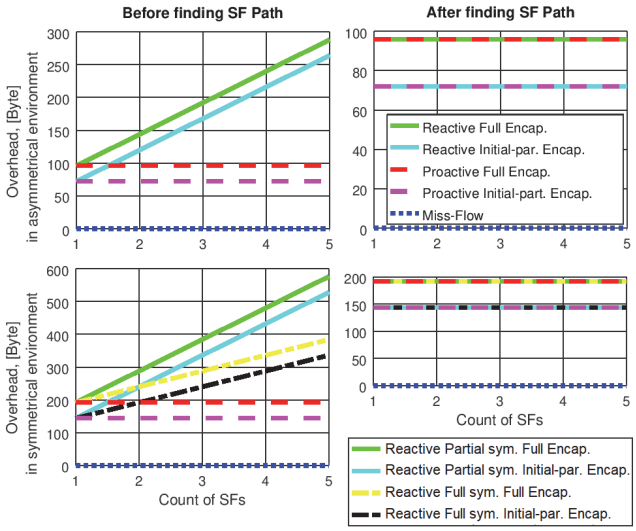

Fig. 12. SFC generated overhead.

SFC generated overhead in bytes is shown in Fig. 12. The division between objectives of searching and having a valid SF path is complimentary to the one describing time domain in Fig. 11.

While static methods as proactive SF path discovery and Miss-Flow entry does not change between objectives, the reactive SF path discovery in all cases does limit its overhead once a valid SF path is found.

Difference between Full SFC encapsulation application method and our (in earlier study [10] proposed) Initial-partial SFC encapsulation method is overhead truncation only by 20 bytes constantly throughout emulations. An explanation to such consistency hides in the simplicity of our network emulation topology. In all emulations we were able to remove encapsulation from only a single link.

As the use of Miss-Flow entry requires no additional packet encapsulation it stays at the value of 0 bytes. It might seem advantageous, but the drawback hides in Fig. 11. Where others are either still or improve their score, Miss-Flow entry has the longest time spent on discovery.

E. SF path discovery successfulness

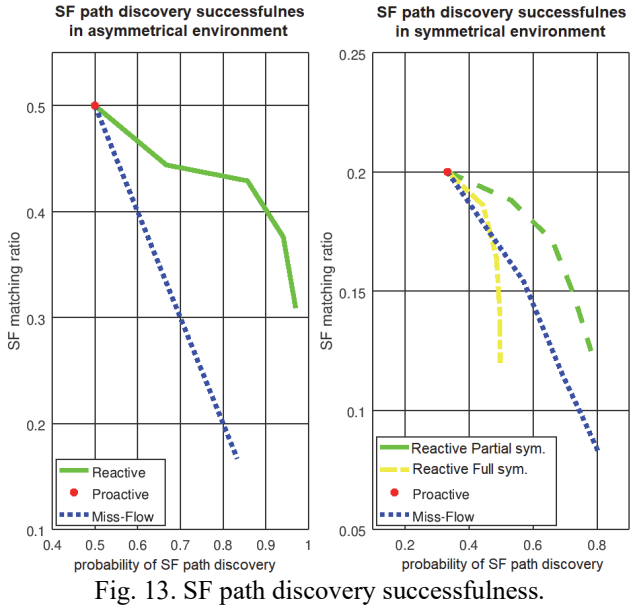

Fig. 13. SF path discovery successfulness.

Comparison in Fig. 13 places SF matching ratio and the probability of valid SF discovery at scales. As probability is closer to the value of 1 as more likely for a successful SF path detection to occur. It is the other way around for SF matching ratio as it is closer to the value of 0 as less reclassifications are likely to occur before a successful SF path detection is made.

Comparison indicates that in asymmetrical as well as symmetrical environments it is more likely to find a valid SF path either by doing a data flow reclassification (if SF path turns out to be faulty and trying a subsequential SF path detection) or by steering a data flow through all SFs in the topology no matter if the service that these SFs provide gets applied or not. The least desirable option is proactive discovery as it stays static for both comparison scales (probability and matching ratio) regardless of the environment.

F. SF path discovery expenses

At comparison in Fig. 14 overhead and elapsed relative time are at scales. Expenses mentioned do relate only to the discovery process itself (while a valid path has not been found). Spending less time in path detection as well as adding the least overhead are considered as the best outcome for the discovery processes.

In asymmetrical environment staying at a value below 100 bytes for overhead and a value of 1 for elapsed relative time, also in symmetrical environment staying at a value below 200 bytes for overhead and a value of 1.6 for elapsed relative time proactive SF path discovery gains an upper hand in both scales thanks to its static nature.

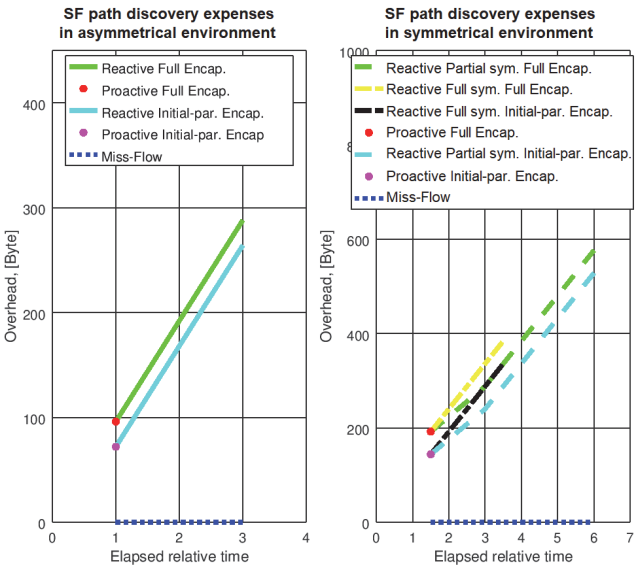

Fig. 14. SF path discovery expenses.

Miss-Flow entry is the second desirable method as during discovery process it gains no overhead. Thus, its only expenses are in time domain where in asymmetrical environment its elapsed relative time grows up to a value of 3 while in symmetrical environment it doubles this value.

Reactive SF path discovery does gain the most overhead (up to 600 bytes) while a valid SF path has not been detected. Afterwards, however, once the valid SF path is found the reactive SF path discovery has equal expenses to those of proactive SF path discovery.

VII. Conclusion

This paper is a continuation of our research of Reactive SF path discovery Approach in Software Defined Network [2] proposing reactive SF path discovery as an alternative to proactive SF path discovery. While proactive SF path discovery relies on the use of a manually configured predefined SF path policy, reactive SF path discovery leverages data flow reclassification for SF path generation reactively at the arrival of a previously unknown data flow.

This study focuses on SF path symmetry's effect on path discovery as it is a fundamental attribute of each path. In addition to proactive and reactive SF path discovery methods we also emulated the use of Miss-Flow entry. Reactive SF path discovery is the only method of those that we emulated which can leverage network automation via a SDN controller.

Results indicate that proactive SF path discovery has a static nature in asymmetrical as well as symmetrical environments, thus it does not support a subsequential SF path detection in single conditioned path policy.

Usage of partial symmetry is advisable in cases where a high probability of valid SF path discovery is required and additional expenses for discovery process can be tolerated.

Usage of full symmetry is advisable in cases where SFs require visibility of both ingress and egress traffic, while a lower probability of valid SF path detection is available.

In overall, reactive SF path discovery gains a higher probability of valid SF path discovery than proactive discovery in asymmetrical and symmetrical environments.

References

[1] J. Halpern and C. Pignataro, "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015.

[2] M. Mihaeljans and A. Skrastins, "Reactive Service Function Path Discovery Approach in Software Defined Network," *2021 29th Telecommunications Forum (TELFOR)*, 2021, pp. 1-4.

[3] H. Hantouti and N. Benamar, "Partially Symmetric Service Function Chains," *2019 2nd IEEE Middle East and North Africa COMMunications Conference (MENACOMM)*, 2019, pp. 1-6.

[4] Y. Zhang and X. Cao, H. Yu and G. Sun, "Service Function Chain Deployment Based on Candidate Paths", *6th International Conference on UK-China Emerging Technologies*, 2021.

[5] H. Hantouti, N. Benamar, M. Bagaa and T. Taleb, "Symmetry-Aware SFC Framework for 5G Networks," *IEEE Network*, vol. 35, no. 5, pp. 234-241, September/October 2021.

[6] M. Polverini, J. Gal´an-Jim´enezy, F. G. Lavaccaz, A. Cianfrani, V. Eramo, "Dynamic In-Network Classification for Service Function Chaining ready SDN networks", *10th International Conference on the Network of the Future*, 2019.

[7] S. Lee and M. Shin, "A self-recovery scheme for service function chaining," *2015 International Conference on Information and Communication Technology Convergence (ICTC)*, 2015, pp. 108-112.

[8] Open Networking Foundation, "OpenFlow Switch Specification Version 1.5.1," 2015.

[9] M. Mihaeljans and A. Skrastins, "Network Topology-aware Service Function Chaining in Software Defined Network," *2020 28th Telecommunications Forum (TELFOR)*, 2020, pp. 1-4.

[10] M. Boucadair, "Service Function Chaining (SFC) Control Plane Components & Requirements," draft-ietf-sfc-control-plane-08 (Informational), October 2016.

[11] P. Quinn, U. Elzur and C. Pignataro, "Network Service Header (NSH)," IETF RFC 8300, 2018.

[12] Nippon Telegraph and Telephone Corporation, Ryu, [Online] Available:https://ryu.readthedocs.io/en/latest/writing_ryu_app.html, last viewed May 2022.

[13] P. Biondi, Scapy, [online] Available: https://scapy.readthedocs.io/, last viewed May 2022.