2016 IEEE 24th International Conference on Network Protocols (ICNP)
Poster Paper

ORSAP: Abstracting Routing State on Demand

Kai Gao†, Chen Gu*, Qiao Xiang‡, Xin Wang*, Y. Richard Yang‡, Jun Bi†
*Tongji University †Tsinghua University ‡Yale University

***Abstract*—Providing an interface for network applications to access network state, Software-Defined Networking (SDN) northbound API protocol is the foundation for the development of programmable networks with adaptive applications. However, with the growing network scale and applications' need for routing state at multi-domain level, feeding complete routing states to applications would jeopardize their scalability and network providers' privacy. Thus a good routing state abstraction is needed, which must be on-demand so that different applications can receive customized abstract state suiting their needs. Moreover, it must be minimal and equivalent, *i.e.*, containing all the necessary information for applications to make decisions as the complete state does with no redundancy. Current routing state abstractions are not on-demand, and adopt extreme aggregation approaches (*e.g.*, the *big switch*) to provide a minimal abstraction with the price of severe information loss. For instance, bottleneck links shared between flows are concealed, leading applications to make sub-optimal decisions. In this paper, we design *ORSAP*, the first on-demand routing state abstraction protocol, through which network applications can describe their demands while Internet service providers can provide the on-demand minimal equivalent routing state accordingly. ORSAP ensures applications' scalability, protects network providers' privacy, and significantly reduces the traffic to disseminate the information. Experiments show that with ORSAP and the abstraction engine we introduced in this paper, one can achieve a state abstraction ratio of up to 60% with an extremely low computation time even with large networks and complex application queries.**

I. Introduction

Providing applications with the network information including the routes and the attributes of the links on the routes is fundamental for developing adaptive network applications. Thus many SDN controllers, such as ONOS [1] and OpenDaylight [2], have provided the API for applications to access the global routing state.

However, feeding a *complete routing state* to applications causes severe privacy leaks for network service providers and brings significant challenges on the scalability of applications. And these drawbacks are magnified with the growing network size and applications' needs for routing state on the multi-domain level. It is important that an *abstract routing state* be provided, which should not only guarantee privacy and scalability but also reduce the load of information updates.

Despite these substantial benefits, designing such an routing state abstraction is a non-trivial task because a series of challenges need to be resolved: *1)* The abstraction must be *on-demand*. Different applications may have different requirements on the routing state. A good abstraction should allow applications to specify their needs and return customized abstract routing states. *2)* The abstraction must ensure that for any routing state query, there is *no information loss* in the returned abstraction. In other words, applications should be able to make the same decision based on the returned routing state as they do based on the complete routing state. *3)* The abstraction must be *scalable*. Returning a complete routing state for a given query is infeasible because the number of routes between a source-destination pair grows exponentially with the increase of the scales of network and attached properties. *4)* The abstraction must protect the *privacy* of network providers. Exposing the complete routing state to applications will cause massive information leaks, making the network more vulnerable.

Existing routing abstractions usually adopt two strategies. One is the *big switch* approach, in which a network is abstracted as a single node with all the details hidden from applications [3]–[5]. The other one is the *aggregation* approach, in which the network is divided into several groups and each group is aggregated as a single node [6], [7]. Though both approaches provide certain support for application scalability and network provider privacy, they are application-oblivious, and suffer from severe routing information loss, *e.g.*, bottleneck links shared between flows are concealed from applications, leading them to make sub-optimal decisions. Thus neither of them is the correct direction to an efficient on-demand routing state abstraction.

In this work, we explore the feasibility and benefits of providing such an on-demand routing state abstraction. In particular, we propose ORSAP, the first *On-demand Routing State Abstraction Protocol*. ORSAP allows network applications to specify their needs on routing state, and network administrators to provide network information based on the policies. We have also built a prototype providing the on-demand routing state abstraction service with ORSAP. A core component of the prototype system is a routing state abstraction engine that computes the abstract routing state with the complete network state, the set of paths computed from an application's query request and the network policies specified by administrators. The engine is capable of providing the *minimal equivalent* routing state in the sense that the result contains all the necessary information for the application to make decisions as the complete routing state does, and with no redundant information.

II. Protocol Design

The protocol is based on the Application-Layer Traffic Optimization [4] protocol, with the additional extensions:

Flow requests Applications must provide the interested flows as well as the desired attributes using the `FlowRequest` as demonstrated below. Each flow is specified by its header field

978-1-5090-3281-5/16/$31.00 ©2016 IEEE

values and is uniquely identified by its UUID. Applications can also provide additional application-specific constraints, which can be leveraged to further reduce the size of the abstract routing state.

```
object {
    FlowSpecMap flows;
    JSONString  attributes<0..*>;
    [JSONString  extra-constraints<0..*>;]
} FlowRequest;

object-map {
        UUID -> FlowSpec;
} FlowSpecMap;

object {
    IPv4Address source;
    IPv4Address destination;
        ...
} FlowSpec;
```

Routing state encoding Applications will receive the routing state in a format that is very similar to path vectors where a list of traversed links is provided for each flow. The representation is more compact to support links with multiple appearances and fine-grained attributes. Bottlenecks can also be identified where the paths for different flows have intersects.

The redundancy elimination algorithm One principle in designing ORSAP is to reduce the data size as much as possible while maintaining the *equivalence* condition. Our RE algorithm can identify and eliminate all redundant constraints and computes the *minimal equivalent* routing state, which contains the *minimal* number of links among all possible routing states that are equivalent to the complete routing state.

III. System Overview

Figure 1 gives an overview of our ORSAP system, which leverages the power of SDN and provides a general framework of computing on-demand abstract routing state. The system consists of five major components:

Path computation engine The path computation engine (PCE) finds the paths for the flows specified by the user request. It can be very efficient for SDN controllers with centralized routing algorithms.

Constraint compiling engine The constraint compiling engine (CCE) takes the policies from the network administrators and compiles them into *linear* constraints on the nodes/links.

NOS Adapter The NOS adapter provides a unified interface for the system to access the network information such as topologies and attributes for nodes/links.

Routing state assembler The routing state assembler combines the information collected by PCE, CCE and the NOS adaptor to construct the complete routing state.

The RE Abstraction engine The abstraction engine is the core of our framework. It uses the *redundancy elimination* algorithm to find the minimal equivalent routing state and can leverage parallel processing to reduce the execution time.

IV. Preliminary Evaluations

We evaluate our prototype system and demonstrate the preliminary results in Figure 2, where the compression ratio

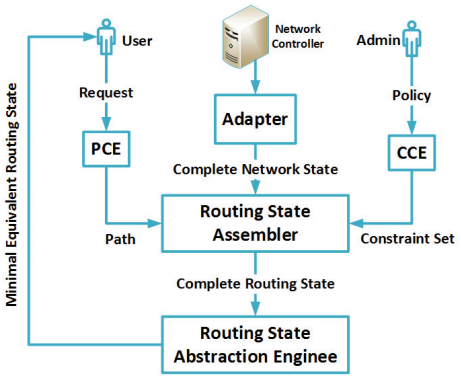
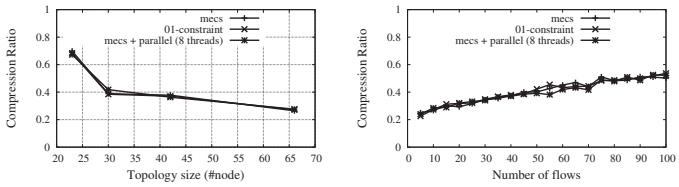


Fig. 1: Architecture of the ORSAP system.



(a) (b)

Fig. 2: Factors on the compression ratio.

denotes the portion of non-redundant links after the abstraction. As we can see from Figure 2(a), the compression ratio decreases as the topology size grows, which indicates that the ORSAP service has improved the scalability. Meanwhile, from Figure 2(b) we can infer that even the compression ratio grows with more flows, it can still reduce more than 40% of the complete routing state for flow requests of a moderate size.

V. Acknowledgement

This project is supported by the *National Science Foundation* (CNS-1018502, CC*IIE1440745), the *National Natural Science Foundation of China* (No.61472213, No.61502267) and the *Google Faculty Research Award*.

References

[1] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow *et al.*, "ONOS: towards an open, distributed SDN OS," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1–6.

[2] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Opendaylight: Towards a model-driven SDN controller architecture," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, 2014.

[3] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4), RFC 4271," 2006.

[4] R. Alimi, R. Penno, S. Previdi, A. Tian, Y.-S. Wang, and Y. R. Yang, "The ALTO protocol, RFC 7285," 2014.

[5] N. Kang, Z. Liu, J. Rexford, and D. Walker, "Optimizing the one big switch abstraction in software-defined networks," in *ACM CoNEXT'13*.

[6] T. Vu, A. Baid, H. Nguyen, and D. Raychaudhuri, "EIR: Edge-aware Interdomain Routing protocol for the future mobile Internet," *WINLAB, Rutgers University, Tech. Rep. WINLAB-TR-414*, 2013.

[7] W. C. Lee, "Topology aggregation for hierarchical routing in ATM networks," *ACM SIGCOMM Computer Communication Review*, vol. 25, no. 2, pp. 82–92, 1995.