Inter-domain Routing Scheme based on Secure Multi-Party Computation over SDN Architecture

Xiaoling Ni Hui Zhao
School of Computer Science and Communication Engineering
Jiangsu University

优胜奖

Abstract

The Internet including enterprise-level networks, data center networks, and wide area networks can be divided into many autonomous domains. Autonomous domains can be managed by different administrative units or organizations. Therefore, when calculating routes between domains, autonomous domains that do not trust each other will have security and privacy considerations. In order to solve the problem of cross-domain privacy protection and optimize routing strategies, we propose an inter-domain routing method based on secure multi-party computation under the SDN architecture. Our method effectively protects the private information in each domain and accelerates the convergence of routing. We elaborated on the preliminary exploration of this idea, and the evaluation showed that our solution only brings millisecond latency.

Method Design

1. Threat Model and Design Goal

Generally speaking, adversaries will compromise nodes(such as routers or switches) in the domain and disrupt the routing process within the domain by identifying active routers or switches and other network forwarding devices. In order to protect the privacy in the domain, we need to prevent the adversary from knowing the route. In the SDN domain, the rules of whether routers or switch nodes forward and how to forward are all made by the SDN controller. Therefore, the privacy mentioned of our method means that the behavior of how the router or switch forwards data packets formulated by the SDN controller should be protected. On the other hand, when the sender and the receiver are not in the same domain, that is, when the data transmission between the two needs to span multiple domains, the relevant information in the SDN domain is protected while ensuring that the routing path between the sender and the receiver is optimal to save network resources.

In our design, like BGP, in each SDN domain, the SDN controller calculates the path to each target IP prefix according to the global network view it owns, and sorts the path information according to the priority of different attributes , As one of the routing strategies owned by the SDN controller, in addition, the SDN controller can link the domain to other neighbors, that is, through a neighbor, whether the current domain can successfully forward the data packet to the destination nodes will also be used as one of its routing strategies. In SMPC, these routing strategies are all expressed as the private input x_i of the SDN domain.

2. Basic Definition

We define the participating roles in the method as follows:

Definition 1: The original participant, acting as an input provider, is generally embodied as an autonomous domain. In our method, we use an SDN network architecture, so tasks such as computing routing in the SDN domain are undertaken by SDN controller, It has all the routing information in the domain, so we define the original participant as the SDN controller, which is represented as $\{C_1, C_2, \dots, C_n\}$, which can also represent the corresponding domain.

Definition 2: The computing participants are a group of computing servers that perform SMPC, expressed as $\{S_1, S_2, \dots, S_k\}$. When multiple rounds of communication and calculations are required between domains, and each group of participants may be required to exchange messages in each round, this is obviously impractical in the case of large n, and the cost is extremely high. Therefore, we refer to the model in [1] and outsource computing tasks to computing server clusters. These computing servers can be selected from all SDN controllers and selected randomly. This is to save network resources while preventing the computing servers from being compromised. Each random selection can ensure the freshness of the computing servers.

Each SDN domain controller divides its own routing strategy x_i into k shares, and passes the results to k computing server clusters as input to their computing programs. The computing server cluster performs SMPC based on these inputs to calculate the output. After the calculation is completed, the calculation results of these computing servers each have a part of each SDN domain route, and these results are sent to the controllers of each domain, and the controller learns how to reach the destination by combining the k shares of routing information.

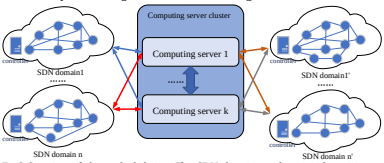

Fig.3.Overview of the method design. The SDN domain, as the original participant, interacts with k computing servers to obtain BGP routes about their respective domains. Among them, the computing servers run secure multi-party computation.

3. Design Details

The scheme structure is as shown in Fig.3, the computing server cluster is composed of k independent and non-colluding computing parties $\{S_1, S_2, \dots, S_k\}$ running the SMPC protocol. As the original participant, each SDN domain will encrypt its routing strategy, divide it into k shares, and notify the computing server. Each computing server in turn sends a part of its output to the original participant after SMPC is completed, and the original participant uses the collected multiple outputs to restore its routing information to the destination domain.

In order to achieve efficient SMPC calculation, we model the routing strategy[1] in each domain as follows according to the basic definition of the routing strategy in section 2:

$N = \{C_1, C_2, \dots, C_n\}$represents a series of SDN domains, $R = \{r_1, r_2, \dots, r_m\}$ is the reachable route in each domain, and we define the egress strategy as an $n * m$ matrix G. If $G_{i,j}$ is 1, it means that the route r_j can reach C_i, otherwise, $G_{i,j}$ is 0, where $1 \le i \le n$, $1 \le j \le m$. The controller of each SDN domain prioritizes the reachable routes in its domain according to certain attributes such as hop count and bandwidth. When the egress strategy is the same, the route with higher priority should be considered. Our scheme takes $k = 2$, that is, two computing servers as an example. For the module design of this part, we refer to the design ideas in [2] like Fig.2. The difference is that we define that an SDN controller undertake the main body of the computing task, and the computing server can also be selected from all SDN controllers.

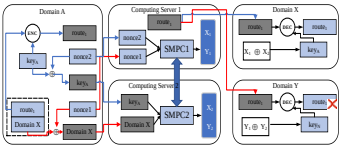

Fig.2. The interaction process between the domain and the computing server cluster. After passing through Export-All component, without destroying the privacy of domain A, only domains that can obtain the correct key can obtain the corresponding routing information.

Evaluation

In this section we analyze the performance about latency of the proposed scheme. The hardware with Intel (R) Core (TM) i7-9700 @ 3.00GHz CPU, 16GB RAM consists of VMware Workstation. A prototype system on three servers with Linux operation systems needs to be implemented. We installed Mininet 2.3.0d6, Openvswitch 2.5.0, and Ryu 4.34 on one of the servers to simulate a SDN environment. The network consists of 16 hosts, which are interconnected using a fat tree of twenty 4-port switches. The other two servers act as computing servers to interact with the domain controller. The computing server is composed of a set of work processes coordinated through a single processing program flow. We implemented SMPC components shown in Fig.3 using the ABY framework[3]. ABY provides low-level primitives for building Boolean circuits that are evaluated efficiently with the GMW protocol.

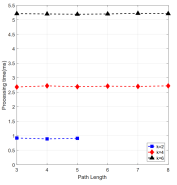

Fig.3(a) Processing time of controllers based on different network size.

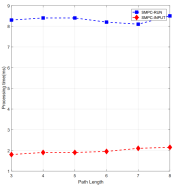

Fig.3(b) Processing time of SMPC module based on SDN architecture.

Fig.3 the processing time of the controller designed in this paper and the calculation time of SMPC. Fig.3(a). Processing time of controllers based on different network size. The processing time of the controller changes with the network scale. When k=4, the average processing time of the SDN controller is 2.70 ms. Fig.3(b). Processing time of SMPC module based on SDN architecture. The time of creating the input values for the SMPC processing is far less than the time required to run the SMPC computation.

Conclusion

We propose an inter-domain routing optimization method based on secure multi-party computation over the SDN architecture. The solution uses the global view feature of the SDN controller to schedule routes and make inter-domain routing decisions, and is based on the GMW protocol to achieve intra-domain privacy protection. The results show that while ensuring the privacy in the domain, the latency caused by the design of the scheme is limited.

Reference

[1] D Gupta, A Segal, A Panda, et al. A new approach to interdomain routing based on secure multi-party computation[C]//Proceedings of the 11th ACM Workshop on Hot Topics in Networks.

[2] M Chiesa, D Demmler, M Canini, M Schapira and T Schneider, SIXPACK: securing internet exchange points against curious onlookers[C]//Proceedings of the 13th International Conference on emerging Networking Experiments and Technologies. 2017: 120-133.

[3] D Demmler, T Schneider and M Zohner, ABY-a framework for efficient mixed-protocol secure two-party computation[C]//The Network and Distributed System Security Symposium , 2015.

Contact information

Contact person：Xiaoling Ni
Phone number：18852856507
Mailbox：18852856507@163.com