ScienceSDS: A Novel Software Defined Security Framework for Large-scale Data-intensive Science[*]

Deepak Nadig Anantha[†]
Dept. of Computer Science and Engineering
University of Nebraska-Lincoln
Lincoln, Nebraska 68588
deepaknadig@cse.unl.edu

Byrav Ramamurthy
Dept. of Computer Science and Engineering
University of Nebraska-Lincoln
Lincoln, Nebraska 68588
byrav@cse.unl.edu

ABSTRACT

Experimental science workflows from projects such as Compact Muon Solenoid (CMS) [6] and Laser Interferometer Gravitational Wave Observatory (LIGO) [2] are characterized by data-intensive computational tasks over large datasets transferred over encrypted channels. The Science DMZ [7] approach to network design favors lossless packet forwarding through a separate isolated network over secure lossy forwarding through stateful packet processors (e.g. firewalls). We propose ScienceSDS, a novel software defined security framework for securely monitoring large-scale science datasets over a software defined networking and network functions virtualization (SDN/NFV) infrastructure.

CCS Concepts

•Networks → Network security; Network services; Network monitoring;

Keywords

Software Defined Security; Data-intensive Science; Service Function Chaining

1. INTRODUCTION

Experimental science workflows are often characterized by data-intensive tasks over large distributed networks, requiring significant computational and networking resources. Projects such as Compact Muon Solenoid (CMS) [6] and Laser Interferometer Gravitational Wave Observatory (LIGO) [2] are examples of scientific experiments that require data-intensive infrastructure. These experiments require high-speed data transfers between endpoints, often transferring large data files (upto peta-byte sized files) over high-bandwidth, high-latency networks. Examples of commonly used network protocols for large-volume data transfers over high-bandwidth, high-latency networks in cluster/grid environments include GridFTP [3] and XROOTD [8]. These protocols use multiple TCP sessions to maximize the throughput of data transfers. Multiple encrypted TCP sessions carry data simultaneously between endpoints. High-perform-ance computing (HPC) facilities need to have the appropriate cybersecurity measures in place to respond to security threats [9]; data-intensive science further increases the complexities of the associated security systems due to the large volume of data transferred between endpoints.

Security functions such as firewalls, deep packet inspection (DPI) units etc., are necessary to securely monitor/inspect the ingress traffic for potential threats. Although firewalls provide a transparent path for legitimate traffic, they are subject to scalability and performance constraints when used with fast transfers of large datasets from experimental science projects [7]. Stateful firewalls processing large data transfers introduce packet losses, thus preventing researchers from exploiting the high-bandwidth, high-latency network links. Further, with protocols such as GridFTP or XROOTD employing encrypted channels to transfer data, additional packet inspection may be necessary to classify a packet. Thus, attempting high-throughput transfers through such stateful network functions leads to network performance degradation.

In this work, we propose ScienceSDS, a novel software defined security framework for large-scale data intensive science over a SDN/NFV architecture. Unlike traditional stateful packet processors that subject all incoming traffic to processing, *our framework exploits application-layer and network-layer collaboration to optimize and enable selective packet processing*. Further, our framework enables the flexible composition of different security services using service function chaining (SFC). SFC is the ability to define an ordered list of network services (e.g. firewalls, accelerators, load balancers etc), which are then "stitched" together to create a service chain. The proposed framework facilitates complex security services creation and chaining for SDN managed experimental science data transfers. Incoming datasets that do not undergo any security checks due to the large volume of transfers between endpoints will be selectively routed through the SFC for monitoring and/or inspection. This guarantees secure data transfers between endpoints, while ensuring that the security services are dynamically composable and can be suitably adapted to changing traffic requirements.

[*] This material is based upon work supported by the National Science Foundation under Grant Numbers. ACI-1541442 and CNS-1345277.

[†] Corresponding Author

SDN-NFV Sec'17, March 22-24, 2017, Scottsdale, AZ, USA
© 2017 ACM. ISBN 978-1-4503-4908-6/17/03…$15.00
DOI: http://dx.doi.org/10.1145/3040992.3040999

The paper is structured as follows: Sections 2 presents the background and introduces important concepts such as Science DMZ architecture and service function chaining; Section 3 discusses the related work and provides the context for Science DMZ security; Section 4 presents the ScienceSDS, a security framework for data-intensive science and the traffic classification mechanisms using SNAG; Section 3 describes the SNAG approach to monitoring data flows; Section 4 describes the implemented architecture; Section 5 discusses the results from ScienceSDS and Sections 6 and 7 presents the conclusions and the future work.

2. BACKGROUND

2.1 Data-intensive Science

A Science DMZ [7] (de-militarized zone) architecture was proposed by ESNet. It is a structured network design which introduces a fast subnet at the network edge of each organization (e.g. university campuses, research labs etc), but without the associated performance limits generally arising from stateful packet processors. An example Science DMZ network design is illustrated in Figure 1. Two traffic paths are shown, with the Science DMZ traffic bypassing the campus network security measures such as a firewall. Thus, Science DMZ is designed to handle high-volume traffic typical of data-intensive science projects with optimizations for a moderate number of high-speed flows, and is typically deployed at the perimeter of the local network.

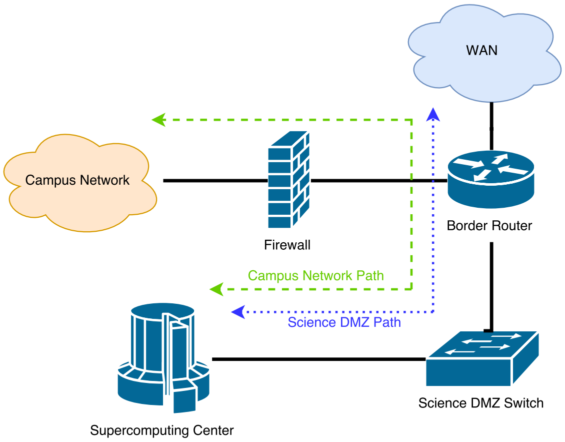


Figure 1: ScienceDMZ network architecture

The network design combines host security with router interface access control lists in a simple scalable fashion to protect the computing resources from intruders. The primary components of a Science DMZ architecture include: (i) Data Transfer Node (DTN) running parallel data transfer protocols such as GridFTP, (ii) perfSONAR for network performance monitoring, and (iii) high-performance routers and switches. Although a Science DMZ arguably provides adequate protection against intruders and attacks, it is not without its problems. Since it requires networks that are free from packet loss and middleboxes such as traffic shapers or stateful firewalls that reduce network performance, traffic monitoring/inspection is performed selectively thereby reducing the ability to detect attacks and intrusions. This reduces accountability for an institution and therefore necessitates additional cybersecurity measures for secure monitoring and transfer of data.

2.2 Virtualized Network Services

Recent advances in network functions virtualization (NFV) [13] and software defined networking (SDN) [20] allows agility and programmability in service-oriented networking. Service (or Network) functions can now be developed to address the need for effective provisioning and deployment, resulting in faster deployment cycles for new and innovative network services. Specialized service/network functions that were deployed as middleboxes (e.g. firewalls, IDS, NAT etc.) can now be defined as independent virtual network functions (VNFs). These VNFs can be logically chained together to form service chains (SCs) that represent a complete service overlay network (SON). Services can now be independently and programmatically composed for both well-established operations and new innovative user services. However unlike traditional service/network functions, virtualized service functions provide greater flexibility in management and composition while ensuring better scalability at a lower capital and operations expenditure (CAPEX/OPEX).

2.3 Service Function Chaining

Connected end-to-end network services can be construed as a set of independent service functions such as firewalls, intrusion detection systems (IDS), network address translator (NAT) boxes, or other application-specific functions. These service (or network) functions provide some value-added services to the traffic flows in the network.

Service function chaining (SFC) is the definition, instantiation and deployment of an ordered set of service (or network) functions to build a composite service, and the subsequent routing of traffic through this ordered set [10]. SFC can be employed to override the default destination forwarding mechanism typically used in IP networks. Unlike network functions in traditional networks, where services are provided by a set of connected physical devices, SFC utilizes VNFs through a NFV architecture to provide a virtual environment for provisioning services. Therefore, SFC provides the ability to route flows through network paths other than the ones specified in the packet's destination routing table. For example, a service chain can force all traffic through a service function such as an IDS, even if the IDS may not be on the data-path between the source-destination pair. The ordering of multiple such (virtual) service functions can create a complex service chain, integrating different services into a logically-composable set of end-to-end functions. Each SFC consists of an ordered sequence of service (or network) functions, with each service function performing a specific task that aids in processing traffic between a given source-destination pair.

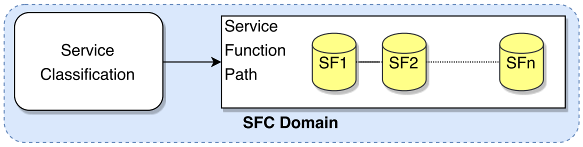
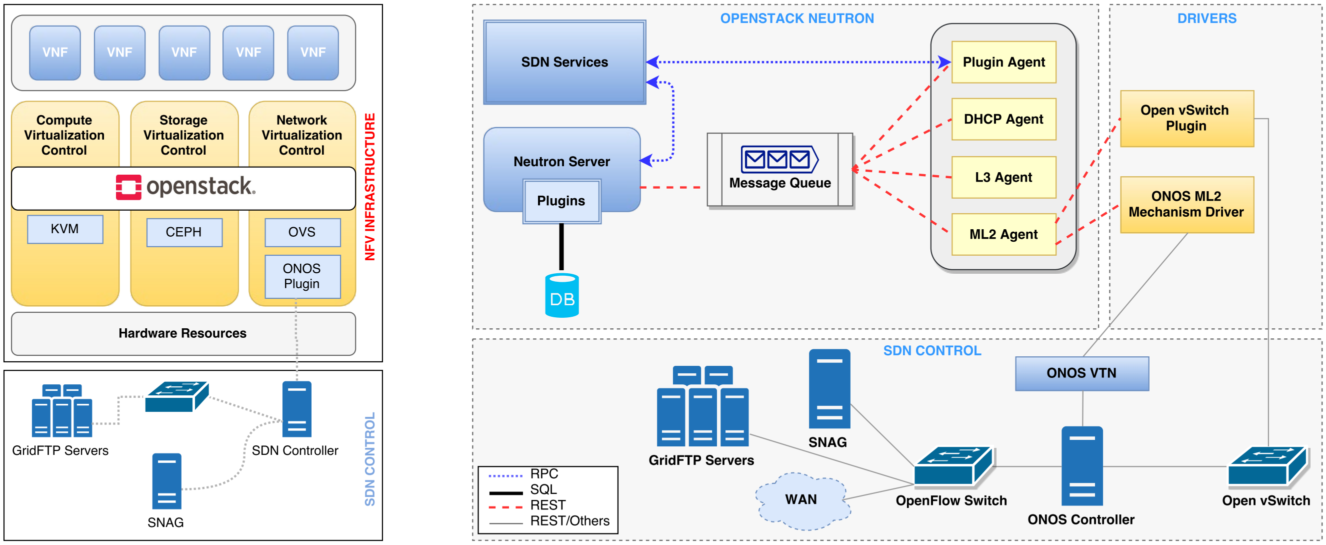


Figure 2: Example SFC Domain

An example SFC domain is shown in Figure 2. Each SFC domain consists of a service classifier and the actual service function path. One or more service function paths can exist



(a) High-level Overview

(b) Framework Components

Figure 3: A software defined security framework for data-intensive science.

in a typical setup, and each path represents a SFC composed of a set of service functions.

Flow classification mechanisms are necessary to route the flows to specific service chains. The flow classifier is a logical entity that can be used to reclassify traffic at any point on the service chain. Traffic classification is achieved by inserting SFC-specific headers such as service chain headers [21] or network service headers [18], to the packets/frames of the traffic flows. These headers serve as identifiers that facilitate packet forwarding along the service function chain. The following section outlines the open problems associated with service deployments in existing networks.

3. RELATED WORK

The Next Generation Service Overlay Network (NGSON) [1] project is an IEEE sponsored effort to standardize an open service ecosystem, where diverse service providers collaborate to satisfy user requirements through service composition and delivery across heterogeneous provider networks. The framework organizes application services offered by various networks into logically separate service- and transport-related functions over an overlay network. NGSON can be used to offer services that are context-aware and tightly integrated [12]. However, divergent views on network/service provisioning make the adoption of such a framework nontrivial. Although, benefits of faster roll-outs of network services exist, the lack of flexibility in network management and interoperability concerns result in slow and expensive changes to the network. Specialized security service functions are currently provided using highly specialized middleboxes resulting in unwieldy static chains of network/service appliances without specific consideration for end-point requirements [15]. Our proposed framework uses a combined SDN/NFV approach to address the above problems while deploying security service functions over an existing network.

While SDN capabilities can be used to setup data-paths for traffic to traverse specific service chains, different data processing requirements can be handled by routing traffic through dynamically established service chains. The work in [14] presents a service-oriented SDN controller that allows provisioning programmable data-paths through a set of dynamically created virtual service sequences. However, the network use is sub-optimal as the traffic traverses through a general set of middleboxes, and the context-awareness is limited to a common set of service classes with no large-scale evaluations. Other work include service specific instances that use proactive and application-driven network configurations [11], where a session- based model installs flow entries proactively based on network services before connecting to SDNs. The work in [17] proposes the use of SDN to improve the management of middleboxes while limiting the number of flow entries of the switches. [22] also leverages SDN and OpenFlow architecture to propose a scalable framework for dynamically routing of traffic through any sequence of middleboxes based on the use of multiple tables and metadata. However, both proposals do not address a comprehensive design of service chaining within the framework of service overlay networks while incorporating the dynamic definition of chains and the context-aware delivery of application service data. NFShunt [16] presents an extension to the Science DMZ design to use a firewall that combines Linux *netfilter* and OpenFlow switching. NFShunt allows selective bypass-switching policy and expresses it as an *iptables* firewall rule-set. Unlike [16], our framework selectively filters trusted flows through application- and network-layer collaboration and therefore only reroutes non-science traffic.

4. SECURITY FRAMEWORK

ScienceSDS shown in Figure 3 for data-intensive science relies on accurate and reliable classification of science datasets for selective classification and rerouting of flows to a virtualized security services infrastructure. Traditional networks use specialized security functions (or middleboxes) that are physical devices modeled as a transparent '*bumps-in-the-wire*'. Multiple middleboxes can exist in the data transfer path creating a service infrastructure that is both unwieldy and that have a 'strict' ordering of functionality.

To solve the problems associated with static service chaining in traditional networks that use a 'strict' ordering of network resources, the security services in our framework are modeled as virtual network functions (VNFs). The VNF lifecycle management is handled by a cloud-based NFV system[1]. This provides greater flexibility in the creation, modification, placement and deletion of the network services dynamically to suit traffic requirements. These VNFs are then combined together to form service function chains (SFCs). This system is now used to provide security functions to enable software-defined security for securely managing and monitoring incoming traffic. SDN control using ONOS[2] is employed to reroute the classified traffic to traverse appropriate service chains by programming the ingress OpenFlow switches. Thus our framework adopts a security-as-a-service approach to network security, where independent security service functions can be 'chained' together to form an integrated network security service.

4.1 Traffic Classification using SNAG

GridFTP transfers create multiple parallel connections to maximize TCP throughput during data transfers. Since the TCP streams are encrypted, the classification of data transfers is dependent on authenticated user information and destination data storage server locations, both of which are application layer data. Due to the encrypted nature of the data transfer streams, traditional approaches to networking monitoring/classification fail, requiring deep packet inspection and other security implements.

Our SDN-managed Network Architecture for GridFTP transfers (SNAG) proposed in [4], and shown in Figure 4, enables the transfer of GridFTP files using SDN-based network management. It focuses on exposing an Application Program Interface (API) to obtain the application-layer information from the trusted GridFTP process. The GridFTP server is extended to interact securely with the SDN controller and automatically forwards application layer metadata, including source/destination address/port pairs, session and file transfer information. This information is exploited by the network layer, allowing it to accurately track all currently active flows using the application metadata. However, SNAG is only responsible for traffic classification and monitoring of CMS traffic, and therefore *does not* provide any security services for traffic handling.

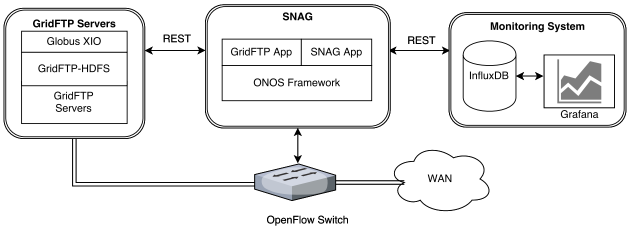


Figure 4: SNAG Components [4]

4.2 Virtualized Security Infrastructure

Service function chains (SFCs) contain independent security functions (i.e. VNFs) such as firewalls, DPI units and IDS boxes etc. We use SDN control of both the ingress OpenFlow switches (See §4.3), as well as the Open vSwitch[3] (OVS) instances created by the NFV infrastructure for routing traffic to different service functions. In this work, we use the Open Network Operating System (ONOS) [5] SDN framework to intelligently steer traffic to the appropriate service functions based on the traffic classification information obtained from SNAG.

In order to manage different virtualized service functions (i.e. VNFs in this case), we use OpenStack [19], an open source cloud computing platform. OpenStack provides the ability to create and manage different service functions. We use an ONOS Virtual Tenant Network (VTN) application to interface to the OpenStack system. OpenStack provides RESTful APIs that can be used to communicate SFC information to the SDN controller. The interaction is enabled using two OpenStack plugins: `networking-sfc`[4] and `networking-onos`[5].

4.3 Network Topology

Our framework is setup to work with a U.S. CMS Tier-2 site capable of handling high-volume transfers to Fermilab. The site stores approximately 2PB of data from projects such as CMS and LIGO, and uses GridFTP protocol for bulk batch transfer jobs and the XROOTD for interactive jobs. The network topology at Holland Computing Center[6] (HCC) is as shown in Figure 5. A Brocade MLXe is used as the data center's border router connecting to the WAN at 100Gbps. Two OpenFlow capable switches, a Dell S6000 40GbE switch, and an Edge-Core AS4600-54T switch connect to the border router as shown in the figure. The first switch hosts the production GridFTP and XROOTd servers, while the second hosts a testbed environment with a GridFTP server removed from the production pool for testing purposes.

The framework is deployed on the testbed environment through the HCC's internal cloud. The components integrated into the framework include:

- OpenStack cloud for VNF lifecycle management, Neutron servers and Module Layer 2 (ML2) plugins.
- ONOS SDN controllers for intelligent flow management and the associated ONOS SNAG application.
- Components required for interaction with the GridFTP servers including GridFTP-HDFS (for communicating with the storage systems), and the GridFTP-XIO callout module (to expose a RESTful API to the SDN controller)

4.4 Implementation and Design

Our framework integrates SDN control with NFV to provide a unified security framework for data-intensive science. The framework is illustrated in Figure 3. A high-level overview of the framework is shown in Figure 3a. It can be seen that OpenStack is used for providing VNF resources, management and orchestration. Networking between various VNFs is provided through OpenStack Neutron[7]. Neutron is used

[1] http://www.openstack.org

[2] http://onosproject.org

[3] http://openvswitch.org/

[4] https://github.com/openstack/networking-sfc/

[5] https://github.com/openstack/networking-onos/

[6] http://hcc.unl.edu/

[7] https://github.com/openstack/neutron

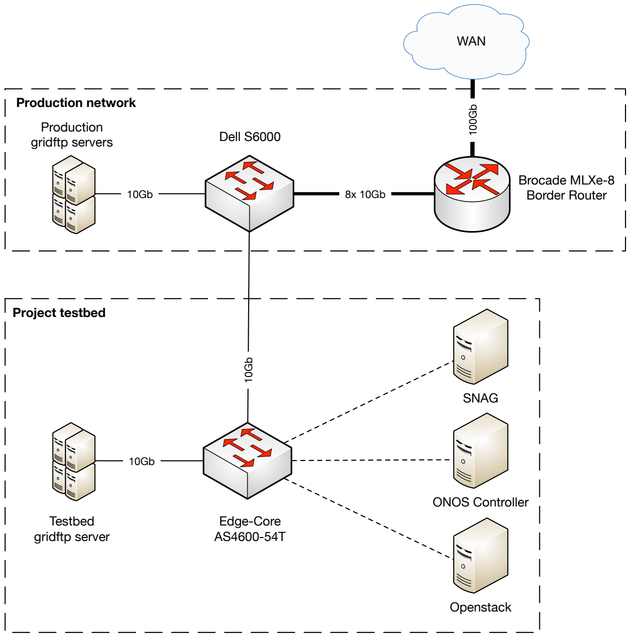


Figure 5: Network Topology

for managing the internal virtual networks within OpenStack and interfaces with the ONOS SDN controller using an ONOS Plugin. The ONOS SDN controller is responsible for configuring the data paths traversed by the science datasets, and through its integration with the SNAG system provides accurate information about all current ongoing GridFTP transfers.

Figure 3b shows the interaction between the NFV infrastructure, SDN control and the GridFTP servers. The Neutron server provides a plugin architecture for enabling the OpenStack to interact with different layer 2 technologies through the ML2 agent service. ONOS plugs-in to the neutron ML2 agent via a driver that provides an interface to the external ONOS controller through the ONOS Virtual Tenant Network (VTN) application. The SDN controller can now interface with the NFV infrastructure to obtain information about all available service chains and flow classifiers. This information is useful to steer the traffic to appropriate service chains for stateful packet processing. On the other side, the SDN controller interfaces to the ONOS SNAG application and uses the traffic classification information to selectively reroute non-science traffic to the determined security service for monitoring and inspection.

5. EXPERIMENTS

5.1 Setup

ScienceSDS was tested in the project testbed shown in Figure 5. The testbed hosts three servers namely i) An OpenStack server to provide the NFV infrastructure and provides a platform for hosting the security service functions, ii) An ONOS SDN controller responsible for interfacing to both the NVF infrastructure and the traffic classification systems, and iii) A server hosting SNAG components to expose application-layer metadata from the testbed GridFTP server. Multiple service chains were created with the corresponding service functions (VNFs), with each chain steering traffic from one to six service functions. Every service function in the experiments host a generic Linux *iptables* firewall populated with basic rules. Each VNF was configured to create unique SFC elements per service fuction, with fully independent port pairs and port groups. Two flow classifiers were assigned to each SFC, with the first flow classifier used to steer TCP traffic and the second for UDP traffic.

5.2 Results and Discussion

The objective of our experiments is to evaluate the performance costs associated with the use of virtual infrastructure for stateful packet processing. Each experiment was run 10 times with the same configuration settings. We evaluate the performance of ScienceSDS by looking at both the time required to setup a SFC for varying number of security service functions in the path, and also the delay cost of steering the traffic through SFCs of different sizes. Figure 6 shows the service chain setup times with one to six security service functions in the SFC path. The setup times vary between 10 and 30 seconds and increase monotonically when the number of service functions in the SFC increases. Although the service chain setup times increase with an increasing number of service functions, this is a one-time cost and does not affect ingress traffic unless these chains are setup very frequently for different traffic flows. Figure 7 shows the delays associated with traversing the SFC. From the graph it can be seen that the average delays for traversing the SFC varies between 1ms to about 5ms. Thus, traversing the SFC with virtualized security service functions does not incur significant packet processing delays.

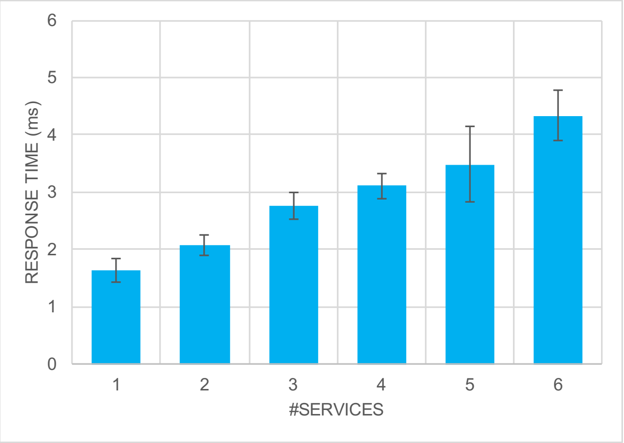


Figure 6: Performance of service chain setups.

6. CONCLUSIONS

ScienceSDS demonstrates an SDN/NFV approach to securing large-scale science datasets without sacrificing management flexibility and topology independence. By combining a virtualized security infrastructure with traffic classification based on application- and network-layer collaboration, stateful security processing can be performed on the traffic without incurring significant performance overheads. Also, by subjecting incoming traffic to prior classification, security-related processing tasks can be limited to

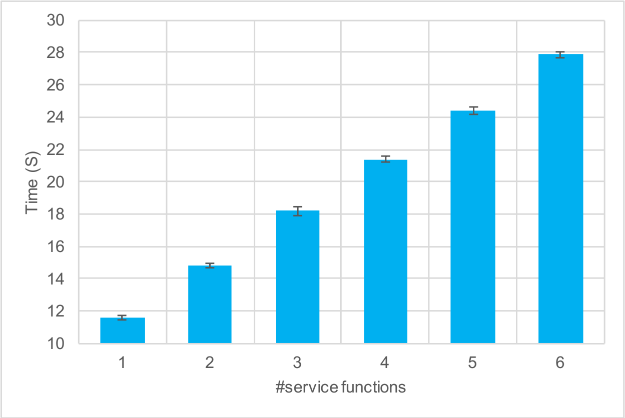


Figure 7: Response times for SFC traversals.

non-science traffic, thereby reducing the load on the security functions. ScienceSDS also ensures flexibility in security service composition through the use of appropriate VNFs, thus ensuring that the security framework is responsive and adaptable to both traffic changes and security attacks.

7. FUTURE WORK

Future work will focus on the performance of the framework when heterogeneous security functions are incorporated into the SFC. Scalability studies by processing traffic from a larger percentage of GridFTP servers are necessary to ensure at-scale performance of the framework. We will focus on tighter integration with other protocols such as XROOTD, and data-plane optimizations for the virtual infrastructure.

8. REFERENCES

[1] IEEE Draft Standard for a Next Generation Service Overlay Network. *IEEE P1903/D2, July 2011*, pages 1–148, Aug 2011.

[2] B. P. Abbott, R. Abbott, R. Adhikari, et al. LIGO: the Laser Interferometer Gravitational-Wave Observatory. *Reports on Progress in Physics*, 72(7):076901, 2009.

[3] W. Allcock, J. Bresnahan, R. Kettimuthu, and M. Link. The Globus Striped GridFTP Framework and Server. In *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference*, pages 54–54, Nov 2005.

[4] D. N. Anantha, Z. Zhang, B. Ramamurthy, et al. SNAG: SDN-managed Network Architecture for GridFTP Transfers. In *Proceedings of the Third Workshop on Innovating the Network for Data-Intensive Science, INDIS '16*, Salt Lake City, Utah, November 2016.

[5] P. Berde, M. Gerola, J. Hart, et al. ONOS: Towards an Open, Distributed SDN OS. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, pages 1–6, New York, NY, USA, 2014. ACM.

[6] S. Chatrchyan et al. The CMS experiment at the CERN LHC. *JINST*, 3:S08004, 2008.

[7] E. Dart, L. Rotman, B. Tierney, et al. The Science DMZ: A network design pattern for data-intensive science. In *2013 SC - International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pages 1–10, Nov 2013.

[8] A. Dorigo, P. Elmer, F. Furano, and A. Hanushevsky. XROOTD/TXNetFile: A Highly Scalable Architecture for Data Access in the ROOT Environment. In *Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics*, TELE-INFO'05, pages 46:1–46:6, Stevens Point, Wisconsin, USA, 2005. World Scientific and Engineering Academy and Society (WSEAS).

[9] I. Gorton, P. Greenfield, A. Szalay, and R. Williams. Data-Intensive Computing in the 21st Century. *Computer*, 41(4):30–32, April 2008.

[10] J. M. Halpern and C. Pignataro. Service Function Chaining (SFC) Architecture. RFC 7665, Oct. 2015.

[11] E. Kissel, G. Fernandes, M. Jaffee, et al. Driving Software Defined Networks with XSP. In *2012 IEEE International Conference on Communications (ICC)*, pages 6616–6621, June 2012.

[12] S. I. Lee and S. G. Kang. NGSON: features, state of the art, and realization. *IEEE Communications Magazine*, 50(1):54–61, January 2012.

[13] Y. Li and M. Chen. Software-Defined Network Function Virtualization: A Survey. *IEEE Access*, 3:2542–2553, 2015.

[14] B. Martini, F. Paganelli, A. A. Mohammed, et al. SDN controller for Context-aware Data Delivery in Dynamic Service Chaining. In *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*, pages 1–5, April 2015.

[15] A. M. Medhat, T. Taleb, A. Elmangoush, et al. Service Function Chaining in Next Generation Networks: State of the Art and Research Challenges. *IEEE Communications Magazine*, PP(99):2–9, October 2016.

[16] S. Miteff and S. Hazelhurst. NFShunt: A Linux firewall with OpenFlow-enabled hardware bypass. In *2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pages 100–106, Nov 2015.

[17] Z. Qazi, C.-C. Tu, R. Miao, L. Chiang, V. Sekar, and M. Yu. Practical and Incremental Convergence between SDN and Middleboxes. *Open Network Summit, Santa Clara, CA*, 2013.

[18] P. Quinn and U. Elzur. Network Service Header. IETF Internet Draft, Sept. 2016.

[19] T. Rosado and J. Bernardino. An overview of Openstack Architecture. In *Proceedings of the 18th International Database Engineering & Applications Symposium*, IDEAS '14, pages 366–367, New York, NY, USA, 2014. ACM.

[20] W. Xia, Y. Wen, C. H. Foh, et al. A Survey on Software-Defined Networking. *IEEE Communications Surveys Tutorials*, 17(1):27–51, First quarter 2015.

[21] C. Zhang, L. Fourie, R. Parker, and M. Zarny. Service Chain Header. IETF Internet Draft, Dec. 2014.

[22] Y. Zhang, N. Beheshti, L. Beliveau, et al. StEERING: A software-defined networking for inline service chaining. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*, pages 1–10, Oct 2013.