A Reactive Security Framework for Operational Wind Parks Using Service Function Chaining

Konstantinos Fysarakis*, Nikolaos E. Petroulakis*‖, Andreas Roos†, Khawar Abbasi‡, Petra Vizarreta**,
George Petropoulos¶, Ermin Sakic§**, George Spanoudakis ‖, and Ioannis Askoxylakis*

*Foundation for Research and Technology-Hellas, Greece †Deutsche Telekom AG, Germany, ‡Intel, Ireland,
§Siemens AG, Germany, ¶Intracom SA Telecom Solutions, Greece, ‖City University of London, United Kingdom
**Technical University of Munich, Germany

***Abstract*—The innovative application of 5G core technologies, namely Software Defined Networking (SDN) and Network Function Virtualization (NFV), can help reduce capital and operational expenditures in industrial networks. Nevertheless, SDN expands the attack surface of the communication infrastructure, thus necessitating the introduction of additional security mechanisms. A wind park is a good example of an industrial application relying on a network with strict performance, security, and reliability requirements, and was chosen as a representative example of industrial systems. This work highlights the benefit of leveraging the flexibility of SDN/NFV-enabled networks to deploy enhanced, reactive security mechanisms for the protection of the industrial network, via the use of Service Function Chaining. Moreover, a proof of concept implementation of the reactive security framework for an industrial-grade wind park network is presented. The framework is equipped with SDN and SCADA honeypots, modelled on (and deployable to) an actual, operating wind park, allowing continuous monitoring of the industrial network and detailed analysis of potential attacks, thus isolating attackers and enabling the assessment of their level of sophistication.**

***Index Terms*—Software Defined Networks; SDN; Network Function Virtualization; NFV; Service Function Chaining; Reactive Security; Industrial networking; Wind Parks;**

I. Introduction

With anticipated exponential growth of connected devices, future networks require an open-solutions architecture, facilitated by standards and a strong ecosystem. Such devices need a simple interface to the connected network to request the kind of communication service characterized by guarantees about bandwidth, delay, jitter, packet loss or redundancy. In response, the network should grant the requested network resources automatically and program the intermediate networking devices based on device profile and privileges. A similar requirement also comes from business applications where application itself asks for particular network resources based on its needs. Software Defined Networking (SDN) and Network Function Virtualization (NFV), important parts of 5G networking provide promising combination leading to programmable connectivity, rapid service provisioning and service chaining and can thus help lower capital and operational expenditure costs (CAPEX/OPEX) in the control network infrastructure. Nevertheless, SDN and NFV expand the attack surface of the communication infrastructure, necessitating the introduction of additional security mechanisms. Industrial networks typically come with strict performance, security, and reliability requirements. Furthermore, by appropriately leveraging the flexibility of SDN/NFV-enabled networks in the context of the adopted security mechanisms, industrial infrastructures can not only match but also improve their security posture compared to the existing, traditional networking environments [1].

This paper showcases a representative use case of an industrial network by considering an industrial control network for wind park operations. The wind park control network has been chosen as a key industrial application as wind energy has now established itself as a mainstay of sustainable energy generation. Nevertheless, the flexibility of SDN networks means they can also help provide better security for industrial networks. Due to the controller's global view of the network and the ability to reprogram the data plane at real time, we can not only revisit old security concepts (e.g. Firewalls) but introduce new techniques as well (e.g. steering suspicious traffic to SDN/SCADA Honeypots, adopting moving target defense and other reactive techniques). The deployment of these enhanced security concepts is in line with the enhanced protection requirements of critical infrastructures, given that the old paradigm of perimeter defenses and trusted internal networks is obsolete, as recent attacks have demonstrated [2]. Thus, enhanced security services are more than "good practice", but a requirement, as evidenced, for example, by the recent update to North American Electric Reliability Corporation (NERC) Critical Infrastructure Protection standards, such as the measures detailed in the latest versions of CIP-007 (i.e. CIP-007-6 [3]), which dictate continuous network monitoring and deployment of network defenses to detect/block malicious activity within the Utilities' perimeter.

Motivated by the above, we present a Reactive Security Framework for next generation 5G (and SDN/NFV in specific) enabled industrial networks. More specifically, considering the energy production critical infrastructures, the framework features enhanced security functions, such as SCADA honeypots, are modelled based upon an operational wind park and ready to be deployed in one. The presented framework allows the continuous monitoring of the wind park industrial network, with provisions to reduce the impact of the security functions on the network's performance and to alleviate the burden of deploying and managing the security services themselves. Moreover, the framework's Honeynet (consisting of both active a SCADA-specific honeypot and a passive honeypot) facilitates the detailed analysis of potential attacks, isolating attackers and enabling the assessment of their level of sophistication (e.g. from script kiddies to state actors).

The remainder of this paper is organized as follows. In Section II, an overview of related work is presented. In Section III, the Service Function Chaining approach (background and motivation) is analysed. In Section IV, we preview the reactive security framework. Finally, Section V provides a discussion, conclusions and future steps of our work.

II. Related Work

Several SFC-related research efforts can be identified in the literature. Blendin et al. [4], exploit Linux namespaces to create isolated service instances per service chain, allowing one-to-one mapping of users to service instances. Network Service Headers (NSH) [5] is another approach that involves the introduction of SFC-specific 4-byte headers that include all the information needed (including associated metadata) to reach a policy decision with regard to what service chain the traffic should follow. As part of the relevant IETF efforts, the NSH approach has been extended to define a new service plane protocol (a dedicated service plane) for the creation of dynamic service chains [6]. StEERING [7] is an OpenFlow-based alternative that allows for per-subscriber and per-traffic type/application traffic routing to the various service functions, via simple policies propagated from a centralized control point. Researchers have also introduced SIMPLE [8], a policy enforcement layer that focuses on middleware-specific traffic steering and considers the inclusion of legacy service instances into the chain. It is based on monitoring and correlating packet headers before and after they traverse a specific service function, though this leads to a rather complex process (collecting packets for correlation, matching packets with high accuracy etc.). The chaining of Virtual Network Functions (VNFs) is another aspect examined in the literature, which considers the trend of virtualizing networks and network functions in modern networks. From this perspective, Megraghdam et al. [9] present a formal model for specifying VNF chains and propose a context-free language for denoting VNF compositions.

III. Service Function Chaining

A. Background and Motivation

In typical network deployments, the end-to-end traffic of various applications typically must go through several network services (e.g. firewalls, load-balancers, WAN accelerators). It can also be referred to as Service Functions (or L4-L7 Services, or Network Functions, depending on the source/organisation) that are placed along its path. This traditional networking concept and the associated service deployments have a number of constraints and inefficiencies [10], such as: topology constraints (network services are highly dependent on a specific network topology, which is hard to update); complex configuration and scaling-out (a consequence of topological dependencies, especially when trying to ensure consistent ordering of service functions and/or when symmetric traffic flows are needed; this complexity also hinders scaling out the infrastructure); constrained high availability (as alternative and/or redundant service functions must typically be placed on the same network location as the primary one); inconsistent or inelastic service chains (network administrators have no consistent way to impose and verify the ordering of individual service functions, other than using strict topologies - on the other hand, these topology constraints necessitate that traffic goes through a rigid set of services functions, often imposing unnecessary capacity and latency costs, while changes to this service chain can introduce a significant administrative burden); coarse policy enforcement (classification capabilities and the associated policy enforcements mechanisms are of coarse nature, e.g. using topology information); coarse traffic selection criteria (as all traffic in a particular network segment typically has to traverse all the service functions along its path). The above are exacerbated nowadays, with the ubiquitous use of virtual platforms, which necessitates the use of dynamic and flexible service environments. This is even more pronounced in service provider and/or cloud environments, with infrastructures spanning different domains and serving numerous tenants, each with their own requirements. Said tenants may share a subset of the providers' service functions, and may require dynamic changes to traffic and service function routing, to follow updates to their policies (e.g. security) or Service Level Agreements.

Service Function Chaining (SFC) aims to address these issues via a service-specific overlay that creates a service-oriented topology, on top of the existing network topology, thus providing service function interoperability [11]. An SDN-based SFC Architecture, such as the one defined by the Open Networking Foundation [12], can extend this concept, exploiting the flexibility and advanced capabilities of software defined networks, to provide novel and comprehensive solutions for the above-stated presented weaknesses of the legacy networks.

B. Security Service Chaining

Security services are a prime example of traditional network service functions that can benefit from the adoption of SFC, especially in the context of SDN networks. Indeed, security functions such as Access Control Lists (ACLs), Segment, Edge and Application Firewalls, Intrusion Detection and/or Intrusion Prevention systems (IDS/IPS) and Deep Packet Inspection (DPI) are some of the principal service functions considered by IETF when presenting SFC use cases pertaining to Data Centers [13] and Mobile Networks [14]. Said IETF studies consider several SFC use cases and highlight the numerous drawbacks of using traditional service provision methods when applying, among others, the security functions.

The security services themselves are typically been deployed as monolithic platforms (often hardware-based), installed at fixed locations inside and/or at the edge of trust domains, and being rigid and static, often lacking automatic reconfiguration and customization capabilities. This approach, combined with the typical networks' architectural restrictions mentioned above, increase operational complexity, prohibit dynamic updates and impose significant (and often unnecessary) performance overheads, as each network packet must be processed by a series of predefined service functions, even when these are redundant [15].

A typical example of an important, and also ubiquitous, security-related function is Deep Packet Inspection (DPI), whereby packet payloads are matched against a set of predefined patterns. DPI imposes a significant performance overhead, because of the pattern matching mechanisms that are at the core its operation, and thus largely unavoidable (motivating a wealth of research efforts focusing on improving their performance [16], [17]). Nevertheless, DPI, in one form or another, is part of many network (hardware or software) appliances and middleboxes; some examples can be seen in Table I. As Bremler-Barr et al. [18] have demonstrated, extracting the DPI functionality and providing it as a common service function to various applications (combining and matching DPI patterns from different sources) can result in significant performance gains; their benchmarks, involving a single Snort-based IDS service function, run in Mininet over OpenFlow to emulate an SDN deployment, compared to two separate traditional

TABLE I
EXAMPLES OF NETWORK APPLIANCES EMPLOYING DPI [18]

Appliance	Examples
Intrusion Detection System	Snort [3], Bro[4]
Antivirus/Anti-SPAM	ClamAV[5]
L7 Firewall	Linux L7-filter[6], ModSecurity[7]
L7 Load Balancer	F5[8] and A10[9]
Leakage/Data Loss - Prevention System	Checkpoint DLP[10]
Network Analyzer/Classifier	Qosmos[11]
Traffic Shaper/WAN optimization	Blue Coat PacketShapper[12]

instances of Snort, showed that the former (i.e. the single DPI service function) performed 67%-86% faster than the latter.

Leveraging the benefits of SDN-based SFC deployments involves reversing this trend for monolithic, ”all-in-one”, security services, which are now commonplace. This is an approach, brought forward in part because of the advancements in hardware performance, which meant that a single, relatively affordable, hardware platform had enough resources to accomplish multiple tasks simultaneously.

Instead, in the context of SFC, the focus is on breaking-up these complex services into dedicated service functions, each providing a single task. This shift is not dissimilar to the emergence of the Microservices[1] as described in [19], software architectural style (i.e. the Microservices Software Architecture, MSA), which moves developers away from the once-dominant paradigm of building entire applications as a monolith (again, leveraging the benefits of more capable hardware and mature, sophisticated programming tools), towards applications made up from a number smaller services (elastic, resilient, composable, minimal and complete[2]), each of them performing a single function (adopting the ”Do one thing and do it well” philosophy).

C. Security Service Functions

A list of key security mechanisms to be leveraged in a secure industrial infrastructure, and deployed as virtualized network service functions, appears below:

- Generic IDS/IPS is a service able to monitor traffic or system activities for suspicious activities or attack violations, also able to prevent malicious attacks if needed (in the case of IPS).
- SCADA IDS/IPS is a service able to monitor traffic or system activities for suspicious activities or attack violations, also able to prevent malicious attacks if needed (in the case of IPS).
- Honeynet - A set of functions (Honeypots), emulating a production network deployment, able to attract and detect attacks, acting as a decoy or dummy target.
- Firewall is a service or appliance running within a virtualised environment providing packet filtering. Legacy firewalls (e.g. actual hardware appliances) are also supported and can easily be integrated into the architecture.
- DPI is a function for advanced packet filtering (data and header) running at the application layer of OSI reference model. In DPI packet payloads are matched against a set of predefined patterns.
- Network Virtualisation - The use of Virtual eXtensible Local Area Network (VXLAN[13]), a VLAN-like encapsulation technique to encapsulate MAC-based OSI layer 2 Ethernet frames within layer 4 UDP packets, brings the scalability and isolation benefits needed in virtualised computing environments.
- Access Control Lists, used at the entry of the wind park domain to route traffic to the appropriate isolated virtual networks and the corresponding security service functions.

Other than the ones employed above, other Service Functions could be included in a real deployment, such as HTTP header enrichment functions, TCP optimisers, packet inspectors (IPFiX, DDoS), IPSec, Resource Signalling, etc.

IV. THE REACTIVE SECURITY FRAMEWORK - IMPLEMENTATION APPROACH

Considering the above, this work focuses on providing a security framework to protect critical industrial infrastructures, considering the wind park as a characteristic example, also studying the more complex multi-tenant use case (i.e. a service provider serving multiple tenants; and its evolution, whereby multiple virtual tenant networks have to be established) and the chaining of vital security functions. This work follows closely the standardization efforts of IETF, and the SFC Working Group[14] in specific, building on top of the work of the Open Networking Foundation and the associated OpenDaylight Controller modules, adopting and extending their features. Moreover, special care is given to the security of the SFC mechanisms, e.g. by guaranteeing the integrity of SFC-related data added to the packets for identifying the service functions chains, and by ensuring that no sensitive SFC data (and the associated metadata), crosses different SFC domains, or legacy networks, unprotected.

One of the goals of this effort is to provide a secure industrial networking infrastructure, via the associated security mechanisms, such as network monitoring and intrusion detection for industrial SDN networks. To achieve this objective, the security framework presented herein includes network monitoring and intrusion detection for identification of attacks and run-time network adaptation for attack response and mitigation mechanisms. By leveraging security network functions such as Firewalls, IDS, DPI, Honeypots and Honeynets, the framework can create a number of service function chains, to forward traffic based on the type of traffic or running application. The aim of this Service Chaining is to overcome constraints and inefficiencies, as mentioned previously. This can be used to fulfil the target of providing security profiles per application

[1] http://martinfowler.com/articles/microservices.html

[2] http://www.nirmata.com/2015/02/microservices-five-architectural-constraints/

[3] http://www.snort.org

[4] http://bro-ids.org

[5] http://www.clamav.net/

[6] https://sourceforge.net/projects/l7-filter/

[7] https://www.modsecurity.org/

[8] https://f5.com/products/modules/local-traffic-manager

[9] http://www.a10networks.com/products/axseries-aflex_advanced_scripting.php

[10] http://www.checkpoint.com/products/dlp-software-blade/

[11] http://www.qosmos.com/products/technology-overview/

[12] https://www.bluecoat.com/products-and-solutions/wan-optimization-packetshaper

[13] https://tools.ietf.org/html/rfc7348

[14] https://datatracker.ietf.org/wg/sfc/charter/

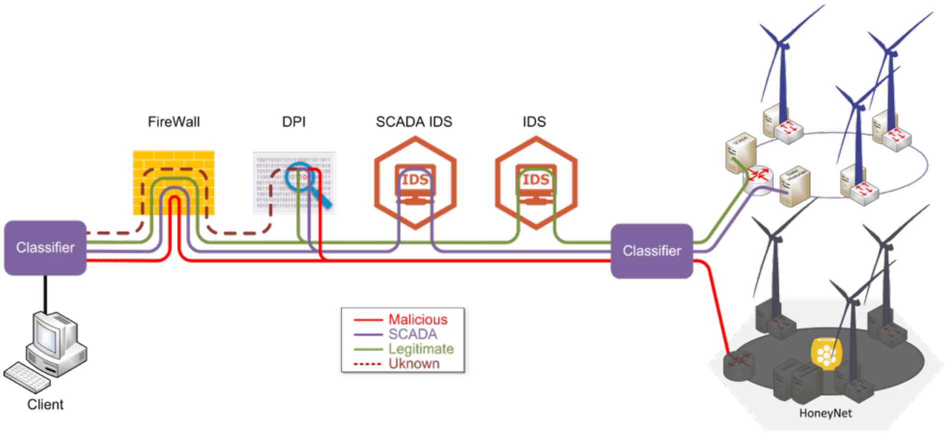


Fig. 1. Reactive Security Framework The Service Chains

classification based on the originating application, or per tenant classification serving multiple virtual tenant networks with the chaining of vital security functions or, alternatively, per traffic classification, for both intra- and inter- domain deployments, following predefined Service Function Paths for each traffic type.

A. *Service Functions in the Framework*

The reactive security framework includes a number of different service functions as detailed in the subsections below.

1) IDS and SCADA IDS: The framework's security mechanisms include continuous network monitoring and intrusion detection for identification of attacks and run-time network adaptation for attack response and mitigation mechanisms. More specifically, IDS instances of Snort[15] are deployed, with scripts to ensure that the most up-to-date rules are constantly active. A database for event monitoring is present, while provisions are made to allow for future extensions to transmit relevant information to a security backend (e.g. for more sophisticated pattern matching). Moreover, a SCADA-specific instance of Snort[16] is also deployed, where SCADA traffic will be routed. This limits the delay imposed on the SCADA traffic by the IDS functionality (a delay that significantly depends on the number of rules/patterns in the IDS's database, which will be significantly lower in the case of the IDS which only has SCADA-specific rules installed).

2) Honeynet: Network-based honeypots have been widely used to detect attacks and malware. A honeypot is a decoy deployment that can fool attackers into thinking they are hitting a real network whereas in the same time it is used to collect information about the attacker and attack method. A Honeynet is a set of functions, emulating a production network deployment, able to attract and detect attacks, acting as a decoy or dummy target. In the protected wind park network, a Honeynet is deployed, consisting of Honeypots emulating SDN and other network elements, as well as Honeypots emulating the operational systems of the wind park, and more specifically elements such as the SCADA systems and the data historian. Simple Honeypots[17] and SCADA-specific Honeypots[18] are deployed to emulate the exact network and SCADA system setup present in the SDN-enabled wind park. Moreover, passive Honeypots (Early Warning Intrusion Detection Systems, EWIS, in specific [20]) are also be part of the Honeynet, acting as a network telescope on the production part of the industrial network, to monitor all activity in normally unused parts of the network. Such activity is a good indicator of malicious entities operating on the network (such as an attacker probing/foot-printing the network), thus providing early warning of incoming attacks.

3) Firewall: A software or hardware firewall instance is also deployed on the wind park's network to implement network perimeter security. This is a software firewall (instance of pfsense[19]), but a hardware (legacy) firewall appliance already present in the industrial network could also be used, or even a virtualized commercial firewall appliance (such as the VM-Series from Palo Alto[20]). The type of firewall, as well as its placement, is irrelevant in the context of the reactive security framework employed to protect the industrial network, as the service plane view of the framework focuses on the type of service and not the underlying technology that is used to offer this service, allowing for the use of any type of firewall, and for its placement in any place on an SDN network deployment.

4) DPI: In DPI packet payloads are matched against a set of predefined patterns. DPI imposes a significant performance overhead, but nevertheless, in one form or another, is part of many network (hardware or software) appliances and middleboxes. As Bremler-Barr et al. [18] have demonstrated, extracting the DPI functionality and providing it as a common service function to various applications (combining and matching DPI patterns from different sources) can result in significant performance gains. In the proposed framework's proof-of-concept, nDPI[21] is employed to implement the DPI function, monitor incoming traffic, and assign it to the (sub-)set of security service functions intended for the corresponding traffic type.

[15] http://www.snort.org

[16] http://blog.snort.org/2012/01/snort-292-scada-preprocessors.html

[17] Honeyd (simple honeypot), https://github.com/sk4ld/gridpot

[18] Adapting Honeyd for SCADA emulation, https://sourceforge.net/projects/scadahoneynet/, http://scadahoneynet.sourceforge.netcite/

[19] https://pfsense.org/

[20] https://www.paloaltonetworks.com/products/secure-the-network/virtualized-next-generation-firewall/vm-series

[21] http://www.ntop.org/products/deep-packet-inspection/ndpi/

B. Traffic Classification and Function Chaining

The per-traffic type classification deployment example of the Reactive Security framework is detailed below. In this instance, a security SFC-based enhancement, for both intra- and inter- domain deployments, features the ability to forward traffic based on its security classification (e.g. unknown/malicious/legitimate), following predefined Service Function Paths for each traffic type. This type of classification opens up various possibilities for the integration of advanced malicious traffic detection techniques (e.g. exploiting machine learning). As an example, let us assume that a data packet enters the intra-domain win park deployment. Based on its classification by the DPI, the traffic will be directed to one of three different paths as depicted in the following figure. A core part of this use case is the classifier. The classifier is responsible for classifying and forwarding packets based on predefined rules, exploiting pattern matching and tags found on the packet headers, and forwards the packets through one of the predefined function chains. The classification of unknown traffic is based on the nDPI library detailed above. The DPI entity disassembles the traffic packets, assesses their content and decides on their traffic type. Then, the packet is repackaged, assigning the appropriate headers to allow for its routing through the corresponding service chain.

In more detail, based on the classification of each packet, the traffic can be classified as unknown (original state when entering the SFC plane), SCADA (legitimate SCADA traffic), other (any other, non-SCADA, legitimate traffic) or Malicious. Thus, four different chains are defined as follows (Figure 1):

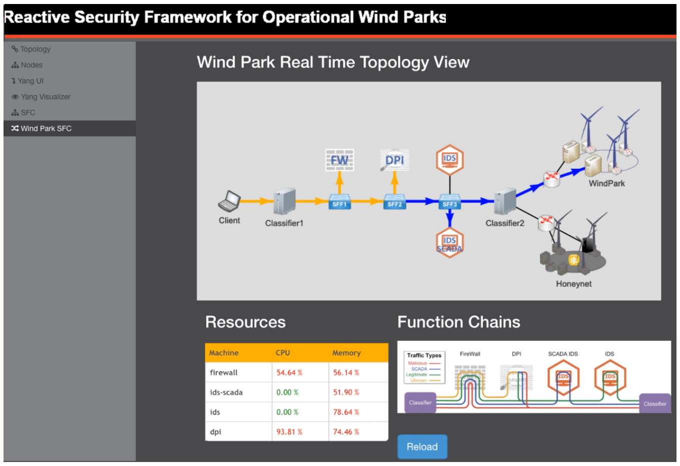
- SFC1 - **Unknown:** Firewall - DPI
- SFC2 - **SCADA:** Firewall - SCADA IDS
- SFC3 - **Other:** Firewall- IDS
- SFC4 - **Malicious:** Firewall- Honeypot/Honeynet

When there is no previous acquired knowledge about the packet's classification (i.e. no tag on the packet header), the classifier will assign the packet to the Unknown chain (SFC1), aiming to detect any malicious activity, assess its impact, and attach the associated tag, to help form the system's response and enhance the attack mitigation effectiveness. However, even if this chain will protect SDN network from malicious attacks, the procedure will add delay to the transmission. Thus, in the case of packets already carrying a tag classifying it as legitimate, it will only be forwarded to the firewall (via the associated chain, SFC2 or SFC3), providing faster packet transmission. Finally, in case of a malicious type of packets, the classifier will forward the packets to the honeypot (or honeynet, depending on the deployment), via SFC4, to investigate the type of attack and the purpose of the attacker.

C. SDN Controller Modules

To implement the above functionality, other than the security service functions themselves (e.g. IDS, Honeypots) that need to be installed and setup appropriately, certain purpose-built modules as well as enhancements to existing SDN controller modules are needed; in this case for the OpenDaylight (ODL) controller.

1) SFC Manager: In more detail, the SFC Manager controller module exposes a number of interfaces that various components can use to provide and receive information about service chains that need to be built, which tenants want to use them, which destinations are being accessed, what applications the traffic pertains to and about the service instances of the network functions. The SFC Manager aggregates this information, combines it, and sends service chains in commands to the SDN Controller (we use OpenDaylight[22] and SFC-ODL[23]). The SDN controller, in turn, programs the underlying forwarding elements that do the actual packet forwarding. In essence the SDN Controller is converting commands from the high-level SFC language to the low-level flow filters of expressed in the OpenFlow semantics.



Resources

Machine	CPU	Memory
firewall	54.64 %	56.14 %
ids-scada	0.00 %	51.90 %
ids	0.00 %	78.64 %
dpi	93.81 %	74.46 %

Fig. 2. Graphical User Interface for real-time monitoring of the operation of the Reactive Security Framework on the ODL Controller

2) Graphical User Interface: To assess and manage the proof of concept implementation of the Reactive Security Framework, a Graphical User Interface (GUI) was developed, as an additional module on the ODL SDN Controller. The GUI displays instantiated VMs/Service Chains and traffic paths, based on the chains seen in Figure 1. Based on this classification, SCADA traffic goes to the SCADA IDS and then to its intended SCADA system at the wind park. HTTP traffic goes to normal IDS and then to its intended system at the wind park. Malicious traffic (e.g. nmap port scan) is detected and goes to Honeypot/Honeynet instead of its intended target wind park system. Finally, unknown traffic is routed to DPI for classification. Based on the DPI classification results, a modification in the header of the packer, can forward the traffic to the respective active chain (legitimate, SCADA or malicious). Changes in path for different traffic types are depicted on the GUI. Moreover, the resources (i.e. CPU and memory load) of the various security service functions, is presented in real-time on a separate table. The above are depicted in Figure 2.

V. Discussion, Conclusions and Future Work

This work presented a reactive security framework modelled on (and deployable to) an actual, operating Wind park, allowing continuous monitoring of the industrial network and detailed analysis of potential attacks, thus isolating attackers and enabling the assessment of their level of sophistication (e.g. from script kiddies to state actors).

In contrast to the proactive deployment of specific security mechanisms (such as the ones detailed in the Section IV),

[22] https://www.opendaylight.org

[23] https://wiki.opendaylight.org/view/Service_Function_Chaining:Main

that are setup and deployed before an attack takes place (typically at the network's design phase), the reactive mechanisms employed here are able to react in real time to changes in the network as well as the traffic traversing said network, e.g. to automatically mitigate attacks, block malicious entities, route them to specific, dummy network components to allow for enhanced monitoring of their actions or even trigger the deployment of new security functions to help alleviate the effects of an ongoing attack. By leveraging the flexibility of SDN-based deployments and the concept of SFC, a service-specific overlay creates a service-oriented topology, on top of the existing network topology, thus providing service function interoperability. Service Function Chaining provides the ability to define an ordered list of network services. The framework's Service Functions (SFs) include the security functions proactively deployed (as detailed in the previous subsection); whether the underlying network and the service functions are virtualised or not, is irrelevant from the perspective of the SFC. These service are then ”stitched” together in the network to create a service chain, allowing us to route unknown/suspicious traffic via the Intrusion Detection and Deep Packet Inspection Service Functions, to classify it (as either legitimate or malicious), allowing us to forward it to the wind park or the honeypot, accordingly. With this mechanism, malicious traffic can be isolated in the honeypot, allowing us to track the attacker, identify her purpose and keep her occupied. The honeypot itself is modelled after the actual operating wind park, fully emulating both the network (SDN-based) elements as well as the industrial application-related devices (e.g. SCADA systems), by combining the appropriate honeypot/honeynet security tools, as detailed above. Using this scheme, the Honeynet's effectiveness is enhanced, taking advantage of the SDN capabilities of dynamic network reconfigurations and traffic forwarding, and this is something that is exploited in the context of VirtuWind's reactive security framework, to reroute malicious traffic to Honeypots/Honeynets instances. Moreover, the deployment of this reactive security framework not only enhances the industrial network's security, but also decreases the performance impact of the security functions. The DPI's performance impact is minimised as the traffic only has to go through one DPI instance, and the same can be said for the IDS/IPS functionality, as e.g. the SCADA traffic only has to go through a faster-performing, SCADA-specific IDS instance.

As future work, we intend to enhance the framework via the use of an open source NFV Management and Orchestration (MANO) software stack, which, via the definition of the service templates at the MANO, will be responsible for the boot-up of the necessary VMs using a Virtual Infrastructure Management (VIM) software (e.g. OpenStack). In turn, the MANO will be used to program the ODL Controller accordingly, passing the necessary information to the SFC Manager. This will also enable a more accurate monitoring of the Service Functions' resources, also allowing the instantiation of additional VMs when, e.g., one of the existing functions is overloaded. Moreover, the automated reactiveness of the framework will be enhanced with the integration of SDN security patterns [21] on the ODL controller via the development of an associated model and the introduction of an adaptive access control mechanism that will enable the policy-based management of multiple controllers, across domains [22]. Finally, the performance of the whole framework will be evaluated in detail, in a number of application scenarios, to assess the impact of the proposed mechanisms in the context of the industrial domain and its intricacies in terms of QoS requirements.

Acknowledgement

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 671648 (VirtuWind).

References

[1] N Petroulakis, Toktam Mahmoodi, Vivek Kulkarni, Andreas Roos, Petra Vizarreta, Khawar Abbasik, Xavier Vilajosana, Spiros Spirou, Anton Matsiuk, Ermin Sakic, et al. Virtuwind: Virtual and programmable industrial network prototype deployed in operational wind park, 2016.

[2] Cyber-Attack Against Ukrainian Critical Infrastructure. *ICS-CERT, Alert (IR-ALERT-H-16-056-01)*, 2015.

[3] NERC Standard CIP. 007-6-Cyber Security-Systems Security Management. 2013.

[4] Jeremias Blendin, Julius Ruckert, Nicolai Leymann, Georg Schyguda, and David Hausheer. Position paper: Software-defined network service chaining. In *Proceedings - 2014 3rd European Workshop on Software-Defined Networks, EWSDN 2014*, pages 109–114, 2014.

[5] Paul Quinn and Jim Guichard. Service function chaining: Creating a service plane via network service headers. *Computer*, 47(11):38–44, 2014.

[6] P. Quinn and U. Elzur. Network Service Header, 2016.

[7] Ying Zhang, Neda Beheshti, Ludovic Beliveau, Geoffrey Lefebvre, Ravi Manghirmalani, Ramesh Mishra, Ritun Patneyt, Meral Shirazipour, Ramesh Subrahmaniam, Catherine Truchan, and Mallik Tatipamula. StEERING: A software-defined networking for inline service chaining. In *Proceedings - International Conference on Network Protocols, ICNP*, 2013.

[8] Zafar Ayyub Qazi, Cheng-Chun Tu, Luis Chiang, Rui Miao, Vyas Sekar, and Minlan Yu. SIMPLE-fying middlebox policy enforcement using SDN. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM - SIGCOMM '13*, page 27, New York, New York, USA, 2013. ACM Press.

[9] Sevil Mehraghdam, Matthias Keller, and Holger Karl. Specifying and placing chains of virtual network functions. In *2014 IEEE 3rd International Conference on Cloud Networking, CloudNet 2014*, pages 7–13, 2014.

[10] Paul Quinn and Tom Nadeau. Problem Statement for Service Function Chaining, apr 2015.

[11] Service Function Chaining (SFC) Architecture. 2015.

[12] L4-L7 Service Function Chaining Solution Architecture. *Open Networking Foundation*, pages 1–36, 2015.

[13] S. Kumar, M. Tufail, S. Majee, C Captari, and S. Homma. Service Function Chaining Use Cases in Data Centers. 2016.

[14] W. Haeffner, J. Napper, M. Stiemerling, D. Lopez, and J. Uttaro. Service Function Chaining Use Cases in Mobile Networks. 2015.

[15] Wolfgang John, Konstantinos Pentikousis, George Agapiou, Eduardo Jacob, Mario Kind, Antonio Manzalini, Fulvio Risso, Dimitri Staessens, Rebecca Steinert, and Catalin Meirosu. Research Directions in Network Service Chaining. In *IEEE SDN for Future Networks and Services (SDN4FNS)*, pages 1–7. IEEE, nov 2013.

[16] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[17] Liberios Vokorokos, Michal Ennert, Marek >Čajkovský, and Ján Radušovský. A Survey of parallel intrusion detection on graphical processors. *Open Computer Science*, 4(4), jan 2014.

[18] A Bremler-Barr, Y Harchol, D Hay, and Y Koral. Deep Packet inspection as a service. In *10th ACM International Conference on Emerging Networking Experiments and Technologies, CoNEXT 2014*, pages 271–282, 2014.

[19] Johannes Thönes. Microservices. *IEEE Software*, 32(1), 2015.

[20] Panos Chatziadam, Ioannis G. Askoxylakis, and Alexandros Fragkiadakis. A network telescope for early warning intrusion detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8533 LNCS, pages 11–22, 2014.

[21] Nikolaos E Petroulakis, George Spanoudakis, and Ioannis G Askoxylakis. Patterns for the design of secure and dependable software defined networks. *Elsevier Computer Networks*, 109:39–49, 2016.

[22] Konstantinos Fysarakis, Othonas Soultatos, Charalampos Manifavas, Ioannis Papaefstathiou, and Ioannis Askoxylakis. Xsacdcross-domain resource sharing and access control for smart environments. *Future Generation Computer Systems*, pages –, 2016.