Received: 8 May 2017 | Revised: 11 August 2017 | Accepted: 8 November 2017

DOI: 10.1002/ett.3269

SPECIAL ISSUE ARTICLE

WILEY

Reactive security for SDN/NFV-enabled industrial networks leveraging service function chaining

Nikolaos E. Petroulakis[1,2] | Konstantinos Fysarakis[1] | Ioannis Askoxylakis[1] | George Spanoudakis[2]

[1]Foundation for Research and Technology-Hellas, Heraklion, Greece

[2]City, University of London, London, UK

Correspondence
Nikolaos E. Petroulakis, Foundation for Research and Technology-Hellas, 700 13 Heraklion, Greece; or
City, University of London, London EC1V 0HB, UK.
Email: npetro@ics.forth.gr

Funding information
European Union's Horizon 2020 Research and Innovation programme, Grant/Award Number: 671648

Abstract

The innovative application of fifth-generation core technologies, ie, software-defined networking (SDN) and network function virtualization, 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. These major changes could not leave the industrial environment unaffected, with smart industrial deployments gradually becoming a reality, a trend that is often referred to as the Fourth Industrial Revolution or Industry 4.0. 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/network function virtualization–enabled networks to deploy enhanced reactive security mechanisms for the protection of the industrial network via the use of service function chaining. Moreover, the implementation of a proof-of-concept reactive security framework for an industrial-grade wind park network is presented, along with a performance evaluation of the proposed approach. The framework is equipped with SDN and supervisory control and data acquisition honeypots, modeled on and deployable to the 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. Moreover, the applicability of the proposed solutions is assessed in the context of the specific industrial application based on the analysis of the network characteristics and requirements of an actual operating wind park.

1 | INTRODUCTION

With the exponential growth of connected devices, networks will require an open-solutions architecture, facilitated by standards and a strong ecosystem. Heterogeneous devices will 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 the application itself asks for particular network resources based on its needs. Software-defined networking and network function virtualization (NFV), important parts of fifth-generation networking, provide a promising combination leading to programmable connectivity, rapid service provisioning, and service chaining, and can thus help lower capital and operational expenditure costs 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. SDN in the *Industry 4.0* arrives as a new concept to promote the computerization of the manufacturing part of the network that can help in communicating essential technologies such as the Internet of Things (IoT), communication machine-to-machine, and cyber-physical systems.[1] 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.[2]

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, SDN allows not only to revisit old security concepts (eg, firewalls) but also to introduce new techniques such as steering suspicious traffic to supervisory control and data acquisition (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.[3] Thus, enhanced security services are more than “good practice” but a requirement, as evidenced, eg, 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 (ie, CIP-007-6[4]), which dictate continuous network monitoring and deployment of network defenses to detect/block malicious activity within the utilities' perimeter.

Service function chaining (SFC) provides the ability to define an ordered list of network services.[5] The concept of SFC has shown promising results providing the ability to define an ordered list of a network services to create a service chain. These services 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 (DPI) 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.

Motivated by the aforementioned statements, we present a reactive security framework for next generation (and SDN/NFV in specific) fifth-generation-enabled industrial networks leveraged by SFC. Considering the energy production critical infrastructures, the framework features enhanced security functions such as SDN-aware DPI and SCADA honeypots modeled based upon an operational wind park. 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. The framework's Honeynet (consisting of both an active 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 (eg, from script kiddies to state actors). Building upon the concept presented in the work of Fysarakis et al,[6] this work highlights various use cases where the proposed mechanisms would be useful in the context of actual wind park deployments and associated business requirements. Moreover, a full implementation of the framework is presented, along with a performance evaluation, on a realistic testbed featuring various services and operational security service functions. The results of this evaluation are assessed in the context of an actual industrial network, highlighting the most viable options in the context of a real industrial application. The aforementioned assessment is based on a trace analysis conducted in an operating wind park's network (in Brande, Denmark), a representative use case of industrial networks studied in the context of the European project VirtuWind.[2,7]

The remainder of this paper is organized as follows. In Section 2, the background, motivation, and related work on SFC is presented, highlighting security-related aspects. In Section 3, a study on the various use cases and associated variations of the proposed scheme is presented. In Section 4, the reactive security framework and its key implementation elements (eg, security services and controller modules) are presented, whereas Section 5 details the performance evaluation results, along with their assessment in light of the actual network performance recorded in an actual operational wind park. Section 6 concludes this work with some discussion and pointers to future work.

2 | SERVICE FUNCTION CHAINING

2.1 | Background and motivation

In typical network deployments, the end-to-end traffic of various applications typically must go through several network services (eg, firewalls, load-balancers, and wide area network accelerators). It can also be referred to as service functions (or L4-L7 Services, or Network Functions, depending on the source/organization) that are placed along its path. This traditional networking concept and the associated service deployments have a number of constraints and inefficiencies[8] as follows.

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: 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, whereas 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, eg, using topology information.

Coarse traffic selection criteria: All traffic in a particular network segment typically has to traverse all the service functions along its path.

The aforementioned statements 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. The aforementioned 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 (eg, security) or service level agreements.

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.[5] An SDN-based SFC architecture such as the one defined by the open networking foundation[9] can extend this concept, exploiting the flexibility and advanced capabilities of software defined networks, to provide novel and comprehensive solutions for the aforementioned stated presented weaknesses of the legacy networks.

2.2 | Terms and definitions

In this subsection, the terms and their definitions, as used in this work, are mentioned. The definitions of SFC terms are described in the Internet Engineering Task Force (IETF), SFC architecture,[5] and SFC environment security requirements.[10] Key terms include the following.

Network service function: A function that is responsible for specific treatment of received packets.

Service function chaining: A service function chain defines an ordered set of abstract service functions and ordering constraints that must be applied to packets and/or frames and/or flows selected as a result of classification.

Service function forwarder: A service function forwarder (SFF) is responsible for forwarding traffic to one or more connected service functions according to information carried in the SFC encapsulation as well as handling traffic coming back from the service function (legacy or virtual).

Service function path: The service function path is a constrained specification of where packets assigned to a certain route must go. Any overlay or underlay technology can be used to create service paths (virtual local area network, equal-cost multipath, generic routing encapsulation, Virtual eXtensible Local Area Network, etc).

Service function classifier: An entity that classifies traffic flows for service chaining according to classification rules defined in an SFC policy table and to mark packets with the corresponding SF chain identifier. It can be on a data path or run as an application on top of a network controller.

SFC header: A header that is embedded into the flow packet by the SFC Classifier to facilitate the forwarding of flow packets along the service function chain path. This header also allows the transport of metadata to support various service chain related functionality.

2.3 | Related work

Several SFC-related research efforts can be identified in the literature. Nevertheless, a recent survey on the use of SFC[11] reveals a lack of work focusing on security-related applications, and this is a gap that the framework presented herein can cover. In terms of the key technological building blocks, network service header (NSH)[12] is an 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[13]; this NSH-based SFC approach is adopted in the framework presented herein.

StEERING[14] 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 but does not consider the security-based classification that forms the basis of the work presented here. Researchers have also introduced SIMPLE,[15] 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. More specifically, the European Telecommunications Standards Institute proposes a security management and monitoring specification in NFV that enable active and passive monitoring of the VNF and the SFC as provisioned in the NFV environment.[16] From this perspective, Mehraghdam et al[17] presented a formal model for specifying VNF chains and proposed a context-free language for denoting VNF compositions. Chain definitions in the work presented here are based on the structured format required by the testbed controller but a formal-based definition could be used if the corresponding module is appropriately extended, provided that the added complexity is justified by the application requirements. Blendin et al[18] exploited Linux namespaces to create isolated service instances per service chain, allowing one-to-one mapping of users to service instances; nevertheless, such an approach is not necessary in industrial environments, where, typically, the number of users is limited, and the management of multiple service instances can incur a significant administrative burden.

2.4 | 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 (ACL), segment, edge and application firewalls, intrusion detection and/or intrusion prevention systems (IDS/IPS), and DPI are some of the principal service functions considered by IETF when presenting SFC use cases pertaining to data centers[19] and mobile networks.[20] The aforementioned 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 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 aforementioned, 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.[21]

A typical example of an important, and also ubiquitous, security-related function is 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 of its operation, and thus largely unavoidable (motivating a wealth of research efforts focusing on improving their performance[22,23]). Nevertheless, DPI, in one form or another, is part of many network (hardware or software) appliances and middleboxes; some examples can be seen in the work of Bremler-Barr et al.[24] As Bremler-Barr et al[24] has 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 with 2 separate traditional instances of Snort, showed that the former (ie, 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,[25] as described in the work of Thönes,[26] software architectural style (ie, the microservices software architecture), 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), toward applications made up from a number smaller services (elastic, resilient, composable, minimal, and complete[27]), each of them performing a single function (adopting the "Do one thing and do it well" philosophy).

Some key security mechanisms to be leveraged in a secure industrial infrastructure and deployed as virtualized network service functions appear as follows.

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 is formed by a set of functions (honeypots), emulating a production network deployment, able to attract and detect attacks, and acting as a decoy or dummy target.

Firewall is a service or appliance running within a virtualized environment providing packet filtering. Legacy firewalls (eg, 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 open systems interconnection reference model. In DPI, packet payloads are matched against a set of predefined patterns.

Network virtualization, via the use of Virtual eXtensible Local Area Network[28], a VLAN-like encapsulation technique to encapsulate MAC-based OSI layer 2 Ethernet frames within layer 4 user datagram protocol (UDP) packets, brings the scalability and isolation benefits needed in virtualized computing environments.

ACLs are used at the entry of the wind park domain to route traffic to the appropriate isolated virtual networks and the corresponding security service functions.

Packet inspectors detect malformed packets or malicious activity (internet protocol flow information export and denial-of-service).

Secure communication protocols with packet encapsulation services (eg, internet protocol security)

Other than the ones aforementioned, other application-specific service functions could be included in a real deployment, such as hypertext transfer protocol (HTTP) header enrichment functions, transmission control optimizers (TCP) optimizers, resource signaling, etc.

3 | USE CASES OF SECURITY SERVICE FUNCTION CHAINING

Depending on the focused aspect, which is of relevance for each deployment of the proposed service function chaining-enabled framework, some mainly security-focused use cases flavors (subuse cases) are identified. Each of them is described in the following subsections in more detail.

3.1 | Per-tenant-type classification

One of the main project objectives in industrial networks is faster service provisioning. The time to provision the service is foreseen to be reduced from several days to several minutes. The concept of SFC has already shown promising results in enabling the faster time-to-market for the new services in the domain of telecom operators. This also implies the potential to reduce capital expenditure and operating expenses, especially for a short-lived service. In the context of next-generation industrial networks, one of the promised services is the possibility to instantiate virtual tenant networks (VTN) on demand. The purpose of virtual tenant networks is to setup virtual networks that contains a set of

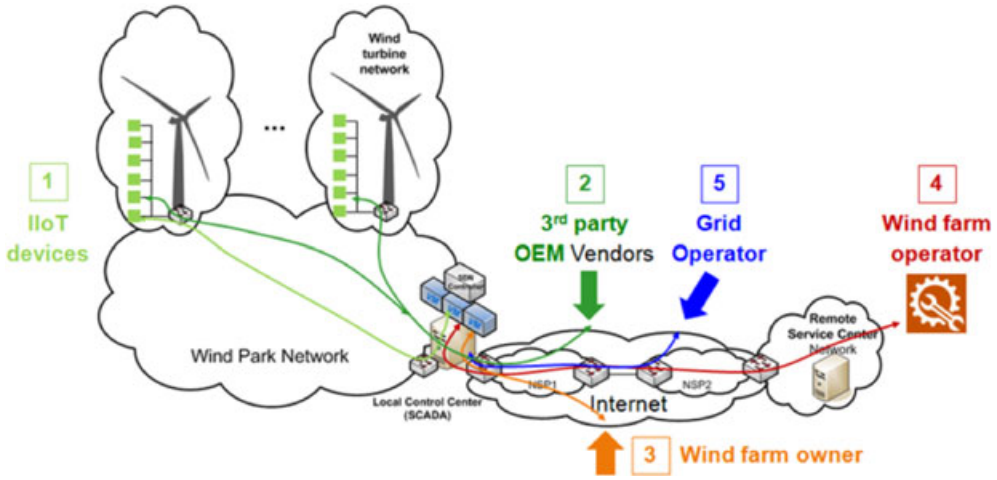


FIGURE 1 Tenants in wind park. IIoT, Industrial Internet of Things; NSP, network services platform; OEM, original equipment manufacturer; SCADA, supervisory control and data acquisition

functionalities on the same physical hardware platform and network architecture. One objective is that different VTNs are not influencing each other. From industrial networks perspective the tenants are related to different stakeholders in a wind park, as shown in Figure 1. The stakeholders may include: wind farm operator, transmission system grid operator, original equipment manufacturer, IoT device vendors, wind farm owner, and SCADA application.

Each of the stakeholders shown in Figure 1 can have different requirements and constraints for setting up a dedicated VTN for their purposes. The different requirements and constraints can lead to different VTN flavors when talking about the VTN setup and network configuration. SFC could be exploited to instantiate VTN, but beyond that aspect, each tenant should be able to add functions on demand in their network and control the traffic that is allowed within their own VTN. In the aforementioned context, 2 different flavors of tenant-based chain classification can be envisioned presented as follows.

3.1.1 | Security services on-demand tenant use case

As an example, let us assume that equipment vendor (eg, Siemens) requests a VTN to inspect the turbines of wind park operator (eg, E.ON). Siemens would request a slice of the operator's network allowing it to access (only) the turbine configuration and log files with certain quality of service (eg, high availability). Additionally, Siemens must be able to create the list of the engineers and the technicians who are authorized to do a troubleshooting on turbines of a given wind park. Siemens and/or E.ON would also like to make sure that members of VTN are performing only the authorized tasks (upgrade, traces, etc.). Hence, tenant-specific security components should be added to a security function chain of the wind park. Example of tenant-specific functions are ACL and DPI. The other security functions, like firewall and IDS, might be still shared with other VTNs in the wind park.

In another related scenario, the flexibility of function chaining could allow tenants to change the deployed security mechanisms dynamically. Thus, eg, using tenant-based classification, Tenant 2 could only be using a firewall protection, whereas Tenant 1 could be using a firewall and an IDS appliance. During operation, the preferences of Tenant 2 could be updated (Tenant 2') to also necessitate the presence of a honeypot or Honeynet, triggering the corresponding update to his/her function chain. This is depicted in Figure 2A, which presents an example of such a setup with the following chain definitions.

Chain 1 - Tenant 1: Firewall -> IDS -> Output
Chain 2 - Tenant 2: Firewall -> Output
Chain 3 - Tenant 2' (after update): Firewall -> Honeypot/Honeynet -> Output

3.1.2 | Industrial Internet of Things tenant use case

Considering the Industrial Internet of Things (IIoT) use case of wind parks deployment and in the context of Tenant-based classification, SFC could also be exploited at the application level in order to provide dynamic real-time access to the required data of the IIoT sensors. Therefore, depending on the tenant's requirements/agreement, etc, each of them get

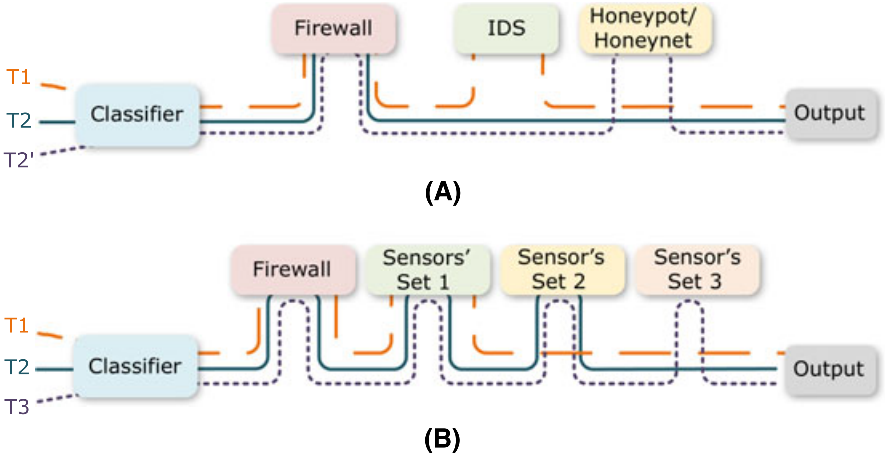


FIGURE 2 Service function chaining: per-tenant classification example. A, Security functions on demand; B, Industrial Internet of Things sensing data. IDS, intrusion detection system

presented with a different subset of all parameters monitored by IIoT sensors, even though all tenants will reach the same resource (eg, web interface on backend monitoring server). This could be achieved by a simple HTTP header enrichment service running on each of the service functions, with each of these services adding the corresponding subset of sensed data into the final web pages that the tenants will see on their web browsers. An example of the aforementioned concept is depicted in Figure 2B, whereby the following chains are defined.

Chain 1 - IIoT Tenant 1: Sensors' Set 1 -> Output
Chain 2 - IIoT Tenant 2: Sensors' Set 1 -> Sensors' Set 2 -> Output
Chain 3 - IIoT Tenant 3: Sensors' Set 1 -> Sensors' Set 2 -> Sensors' Set 3 -> Output

3.2 | Per-application-type classification

This variation of the SFC use cases classifies traffic based on the originating application. Thus, after a stage of DPI, the application is identified and the corresponding chain is assigned. An example of chains tailored to specific applications could include forwarding SCADA traffic to a SCADA-specific IDS and a generic IDS for other traffic, thus limiting the delay imposed on the SCADA traffic by the IDS (as it depends on the number of rules/patterns in the IDS's database, which could be significantly lower in the case of an IDS, which only has SCADA-specific rules installed). Another example could be having video surveillance traffic go through a firewall and a rate limiter to lower the transmission rate, respecting quality of service (QoS) requirements. Thus, potential chains in this case (as depicted in Figure 3) could include the following.

Chain 1 - Unknown application: Firewall -> DPI -> Output
Chain 2 - Video surveillance: Firewall -> Rate Limiter -> Output
Chain 3 - Alarm monitoring: Output
Chain 4 - SCADA: Firewall -> SCADA-IDS -> Output
Chain 5 - Other application: Firewall -> IDS -> Output

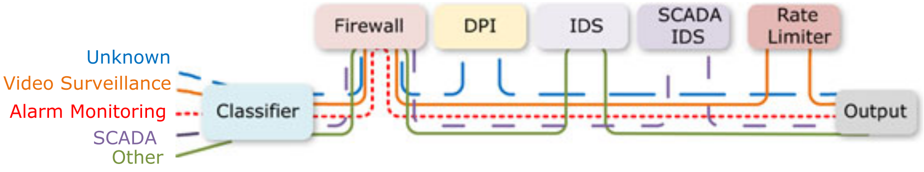


FIGURE 3 Service function chaining: per-application classification example. DPI, deep packet inspection; IDS, intrusion detection system; SCADA, supervisory control and data acquisition

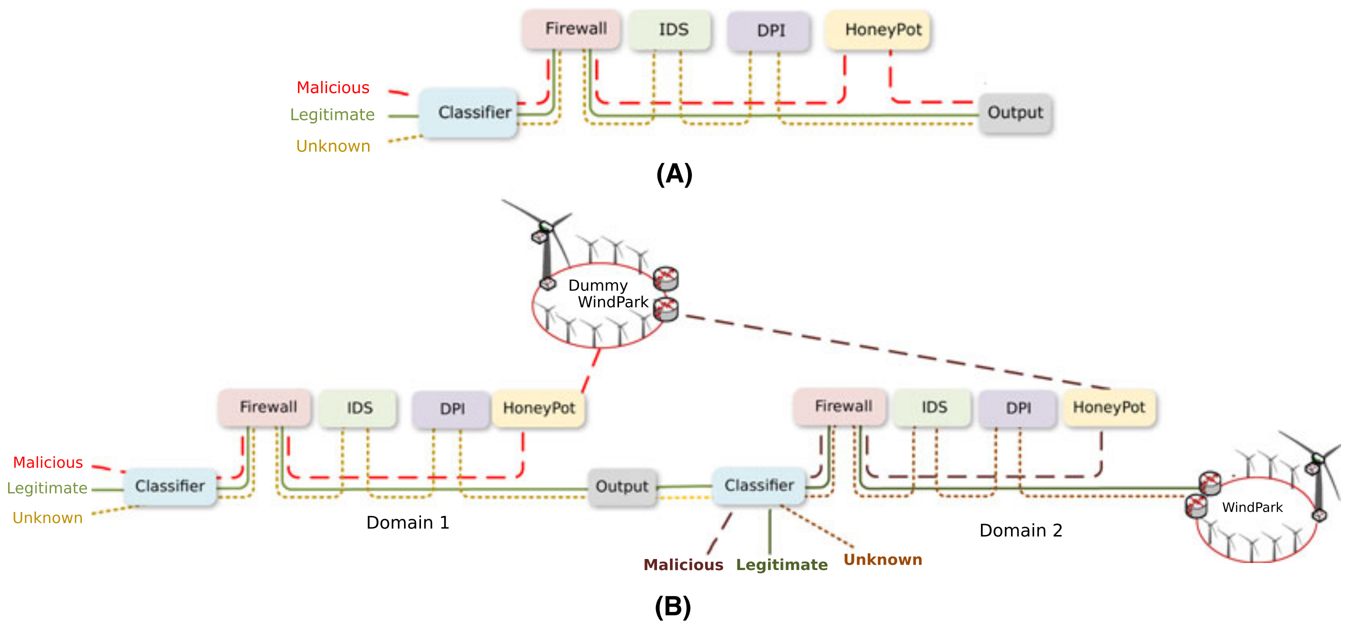


FIGURE 4 Service function chaining: per-traffic-type classification example. A, Intradomain; B, Interdomain. DPI, deep packet inspection; IDS, intrusion detection system

3.3 | Per-traffic-type classification

This use case includes a security SFC-based enhancement for both intradomain and interdomain deployments, with the ability to forward traffic based on its security classification (eg, 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 (eg, exploiting machine learning). As an example, let us assume that a data packet enters the intradomain wind park deployment. Based on its classification (from the aforementioned categories listed), the traffic will be directed to 1 of 3 different paths, as depicted in Figure 4A. The aim for this process is to route unknown/suspicious traffic via the intrusion detection and DPI service functions, in order to classify it (as either legitimate or malicious), thus allowing us to forward it to the wind park or the honeypot accordingly. Thus, malicious traffic can be isolated in the honeypot, allowing us to track the attacker, identify its purpose, and keep him occupied.

For the interdomain use case, ie, Figure 4B, the procedure is similar to the intradomain scenario. However, a more sophisticated honeypot deployment, such as a Honeynet, can be used as an emulated wind park, having similar services and functions as the original wind park. Moreover, in this case, having acquired the needed tag (as malicious or legitimate) in other parts of the larger wind park deployment, the traffic can avoid going through the same procedures (ie, service functions) again, better highlighting the benefits of SFC in terms of potential performance gains. 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. The (attached to the SFF) classifier forwards the packets through one of the predefined function chains. In more details, based on the classification of each packet, the traffic can be classified as legitimate, unknown (suspicious), or malicious. Thus, 3 different chains are defined as follows.

Chain 1 - Legitimate (known) traffic: Firewall -> Output
Chain 2 - Suspicious traffic: Firewall -> IDS -> DPI
Chain 3 - Malicious traffic: Honeypot/Honeynet

4 | REACTIVE SECURITY FRAMEWORK IMPLEMENTATION

4.1 | Overview

Motivated by the aforementioned statement, 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 multitenant use case (ie, 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[29] in specific, building on top of the work of the Open Networking Foundation and the associated OpenDaylight (ODL) controller modules, adopting and extending their features. Moreover, special care is given to the security of the SFC mechanisms, eg, 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 aforementioned. This can be used to fulfill the target of providing security profiles per-application 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 intradomain and interdomain deployments, following predefined service function paths for each traffic type.

In contrast to the proactive deployment of specific security mechanisms 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 aforementioned network, eg, 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.

4.2 | Deployed security service functions

The framework includes a number of different service functions as detailed below.

4.2.1 | Intrusion detection system and supervisory control and data acquisition intrusion detection system

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[30] 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 (eg, for more sophisticated pattern matching). Moreover, a SCADA-specific instance of Snort[31] 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).

4.2.2 | 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, and 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[32] and SCADA-specific honeypots[33] 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[34]) are also 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.

4.2.3 | 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[35]) 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 virtual machine (VM) series from Palo Alto[36]). 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.

4.2.4 | Deep packet inspection

In the proposed framework's proof-of-concept implementation, nDPI[37] 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. Since the default nDPI did not meet the framework's requirements, some changes were made to support the SFC applications. In order to have an up-to-date view of the SFC status, the required information (eg information about the various chains or information on return traffic) was fetched from the controller via constantly-running scripts.

In terms of packet processing, nDPI listens for pcap packets generated by the IP stack of the kernel. In this implementation, since the packets were forwarded from SFF, a listener was implemented on TCP port 6633 to handle incoming packets from that port; those packets were complete ethernet (ETH) packets that also had NSH headers. First, the NSH header was extracted to check if the packet was already processed (in which case, the packet chain ID would be that of a reverse chain). If it was already processed, it was simply forwarded to the SFF. If the packet has not been processed before, the developed application encapsulates the ETH packet at a pcap packet and then sends it to the *nDPI engine* for processing. As soon as a response is received from the nDPI engine, the appropriate chain IDs and the appropriate next hops from rendered service chains are fetched from the controller, and then, based on the received information, the NSH headers are generated and the ETH packet is encapsulated appropriately before being forwarded to the SFF. The ETH generation is based on the response of the *nDPI engine* and the appropriate service function chains in order to support dynamic packet flow change. The response of the *nDPI engine* is based on a set of rules that the engine has compiled to classify traffic types and can be extended by writing additional rules (for instance, TCP and UDP ports 502 associated with Modbus traffic can be defined as being malicious, if such traffic is not expected in the specific part of the network). The response from the *nDPI engine* classification of each packet is either the protocol/application/framework ID that the aforementioned rules define or *UNKNOWN* if it could not be determined.

4.3 | Service chain classification

The per-traffic type classification, as described in Section 3.3, is adopted for the reactive security framework, integrating all the security service functions detailed in Section 4.2 via the following implemented service function chains as follows.

Chain 1 - Unknown traffic: Traffic that is of unknown type (ie, cannot be classified based on simple ACL rules that the Classifier has) is routed to the *Firewall* and then to the *DPI* (nDPI) where it is analyzed, classified, and its headers are updated appropriately, then being assigned to the appropriate Chain (Chains 2 to 4).

Chain 2 - SCADA traffic: SCADA traffic is routed through the *Firewall* and then through the *SCADA-IDS*, which features only SCADA rules to minimize the performance impact before being forwarded to its intended destination.

Chain 3 - Legitimate traffic: Other (non-SCADA) legitimate traffic is routed through the *Firewall* and then through the generic *IDS* before being forwarded to the intended destination (in this case, the data historian).

Chain 4 - Malicious traffic: Traffic tagged as malicious (eg, nmap port scanning), either by the Classifier or by the DPI functionality, is routed through the Firewall and then to the appropriate part of the honeynet; the latter can either be the SCADA honeypot (if its original target was a SCADA system), the generic honeypot (if its original target was an SDN device or a production system such as the data historian), or the passive EWIS honeypot (if the original target was some unused address, indicating malicious probing/footprinting of the network).

Important parts of the implementation of this functionality are the Classifier and the DPI service function. When there is no previous acquired knowledge about the packet's classification (ie, no tag on the packet header), the Classifier will assign the packet to the unknown chain (*Chain 1*), 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. The nDPI 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. However, even if this chain will protect the 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, *Chain 2* or *Chain 3*), 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 *Chain 4*, to isolate and investigate the attack.

4.4 | Implemented SDN controller modules

To implement the aforementioned functionality, other than the security service functions themselves (eg, IDS and honeypots) that need to be installed and setup appropriately, certain purpose-built modules as well as enhancements to existing SDN controller modules are needed.

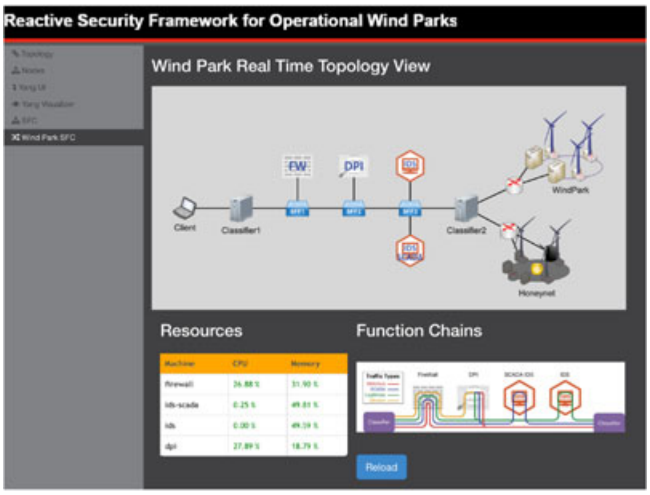
4.4.1 | SFC manager

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. Each SFC configuration includes a set of service nodes, a set of service functions, a set of SFFs, a set of service chains, a set of service paths, and a set of configurations for classifiers (ACL/NSH). The SFC manager aggregates this information, combines it, and sends service chains in commands to the SDN controller (ODL[38] and SFC-ODL[39] are used). 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.

The SDN controller provides an abstraction view of the network topology. This allows the SFC manager to focus on the chaining itself and not on the internal topology of the network controller. This means that the SFC manager manages forwarding rules and flow filters on external ports. This significantly simplifies the configuration. However, the SDN controller does the necessary transformations to put the paths (sequence of service instances where the packet traverses) and filters (associate user based on his profile to its respective service chain) in the forwarding devices (OF-enabled). In particular, the job of the SFC manager is to register external ports of the SDN transport network (which is being used for SFC) and to declare and associate service instances to those external ports. In the wind park case, such service instances may include vFirewall, IDS, DPI, and honeypot. These services may be composed of one or multiple instances. These may be the physical appliances or virtual machines running in network function virtualization infrastructure. At the management and control planes, the SFC manager and the SFC-enabled SDN controllers are responsible for administrating the services chains, ie, for translating the operator's/tenant's/application's requirements into service chains. At the data plane, Classifiers are responsible for assigning traffic to the appropriate service chain (based on various criteria such as its maliciousness or the tenant that it belongs to, assuming tenant identities have already been validated by authentication/authorization components) and service forwarders and proxies (where needed) are responsible for steering traffic accordingly to realize aforementioned service chains. The data plane entities are responsible for steering traffic accordingly to realize the service chains. The Classifier assigns traffic to its intended service chain (based on predefined criteria), and the SFFs steer traffic to the various service function nodes. If the service function nodes are not OpenFlow-speaking or SFC-aware, or are in different domains, SFC proxies are needed.

4.4.2 | 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 5. 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



Reactive Security Framework for Operational Wind Parks

Wind Park Real Time Topology View

Resources

Machine	CPU	Memory
firewall	26.88 %	31.90 %
ids-scada	0.25 %	49.81 %
ids	0.00 %	49.19 %
dpi	27.89 %	18.79 %

Function Chains

FIGURE 5 Graphical user interface for real-time monitoring of the operation of the reactive security framework on the ODL controller. DPI, deep packet inspection; FW, Firewall; IDS, intrusion detection system; SCADA, supervisory control and data acquisition; SFF, service function forwarder

intended system at the wind park. Malicious traffic (eg, 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, where a modification in the header of the packer can forward the traffic to the respective active chain (legitimate, SCADA, or malicious).

To preview the topology of the network, Node.js library[40] was used to present network topology at real time. Suitable representational state transfer interfaces were implemented to import network components such as switches such as forwarders and classifiers, security functions (firewall, DPI, IDS, and SCADA-IDS), and end-hosts such as wind parks, scada, honeypots, etc. Moreover, the condition of service functions with respect to CPU and memory utilization of the various security service functions (ie, the VMs running aforementioned service functions) is also imported automatically by the use of implemented representational state transfer interfaces presented also in real time on a separate table.

4.5 | Architecture sketch: module placement

In today's wind parks, security-related functionalities and SCADA applications are running on dedicated locations in the network architecture. Nowadays, these locations have to be decided in the network planning phase and are very difficult to change during the lifetime of a wind park. The approach proposed in this work enables future scenarios to add or delete functionalities or applications in the wind park network during runtime. In order to enhance this flexibility in a network, the principles of NFV management and orchestration (MANO) can be combined with the framework presented here. The expanded architecture of the framework can be aligned to the approach described in the European Telecommunications Standards Institute Group Specification NFV 002,[41] as depicted in Figure 6. This enhances the proposed framework with flexible deployment and instantiation of new VNFs and the automated preparation of service functions chains and will be explored in future work.

5 | PERFORMANCE EVALUATION

To evaluate the performance of the reactive framework, an experimental testbed was developed. Furthermore, the per-traffic classification was evaluated.

5.1 | Testbed setup

A testbed featuring multiple VMs was developed and deployed on a Proxmox virtualization environment,[42] an open source virtualization management software, which run on a server system (2 Intel Xeon E5-2630 v2 6-core/12-thread CPUs at 2.6 GHz with 32-GB RAM). The following VMs were required in order to implement the described scenario; 3 different types of virtual instances were created (in parentheses, the resources dedicated to each VM).

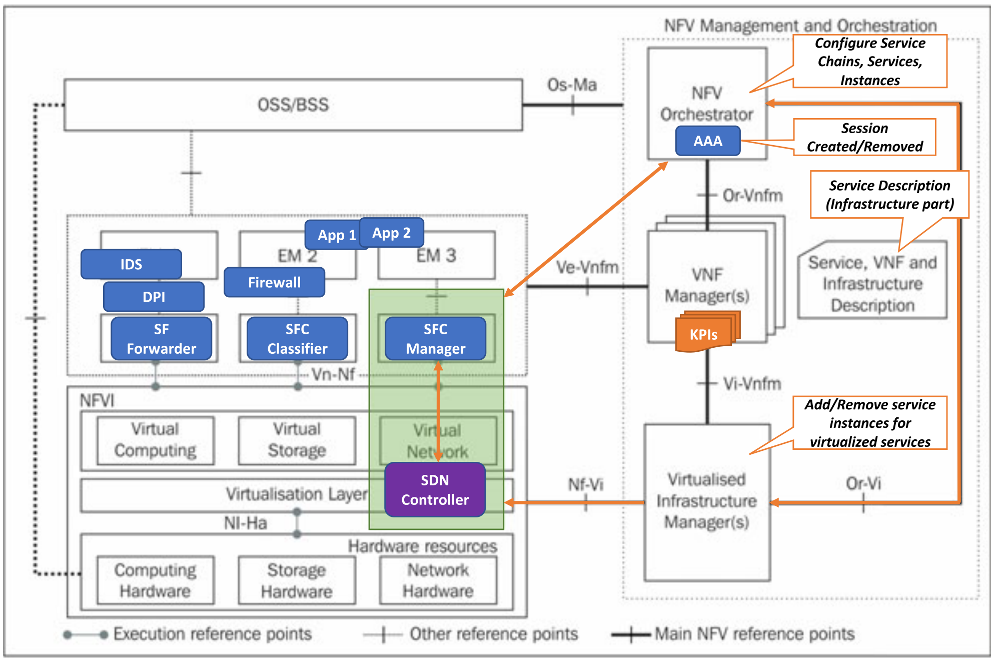


FIGURE 6 Expanded, network function virtualization (NFV) orchestrator–managed and European Telecommunications Standards Institute–aligned framework architecture. BSS, business support system; DPI, deep packet intrusion; EM, emerging market; IDS, intrusion detection system; KPIs, key performance indicators; NFVI, NFV interface; OSS; operational support system; SDN, software-defined networking; SF, service function; SFC, SF chaining; VNF, virtual network function

1 ODL controller instance: (Boron release (4 CPU cores, 4-GB RAM)), **5 Open vSwitch**[43] **(v2.59) instances:** (2 Classifiers (4 CPU cores, 1-GB RAM), 3 SFFs (4 CPU cores, 1-GB RAM)), **4 security service functions:** (1 firewall instance (4 CPU cores, 2-GB RAM), 1 DPI, ie, the custom nDPI-based implementation (4 CPU cores, 4-GB RAM), 1 Snort-based IDS with all generic rules (4 CPU cores, 1-GB RAM), 1 Snort-based SCADA-IDS with SCADA-only rules (4 CPU cores, 1-GB RAM), **4 End-hosts:** (1 emulated data historian (4 CPU cores, 1-GB RAM), 1 emulated SCADA system (4 CPU cores, 1-GB RAM), 1 passive EWIS honeypot (4 CPU cores, 1-GB RAM), 1 SCADA honeypot (4 CPU cores, 1-GB RAM)).

5.2 | Evaluation methodology and results

To give a proof of concept of the implemented framework and testbed, a 2-step approach is followed. The first step contains the import of suitable templates (service functions, SFFs, service function classifiers, service function chains, and access control lists) in the controller in JavaScript Object Notation formats. These templates include all the required information of the testbed, as presented in the previous subsection. Furthermore, the implemented GUI module depicts real-time network traffic monitoring interfaces and functions and service function resource monitoring interfaces and functions custom of the imported data, a screenshot of which can be seen in Figure 5. Changes in path for different traffic types are depicted on the GUI, with different colors to differentiate between the active chains at each instance in time; the various options that can be active (depicting real-time traffic flows and their associated chains) appear in Figure 7. Moreover, in Table 1, the results of the conducted experiments are presented. Apart from the delay between end-hosts, following the respective service function chain, the delays between end-hosts: 1) when there is no function in the middle, and 2) when all the security functions are used, are also presented. Although, the results of the experiments are related to the location and the distance of the VMs, (in this case, all are located on the same server) the correlation between the number of functions and the delays is obvious and is useful to evaluate the results of the real traces as are presented in the next subsection.

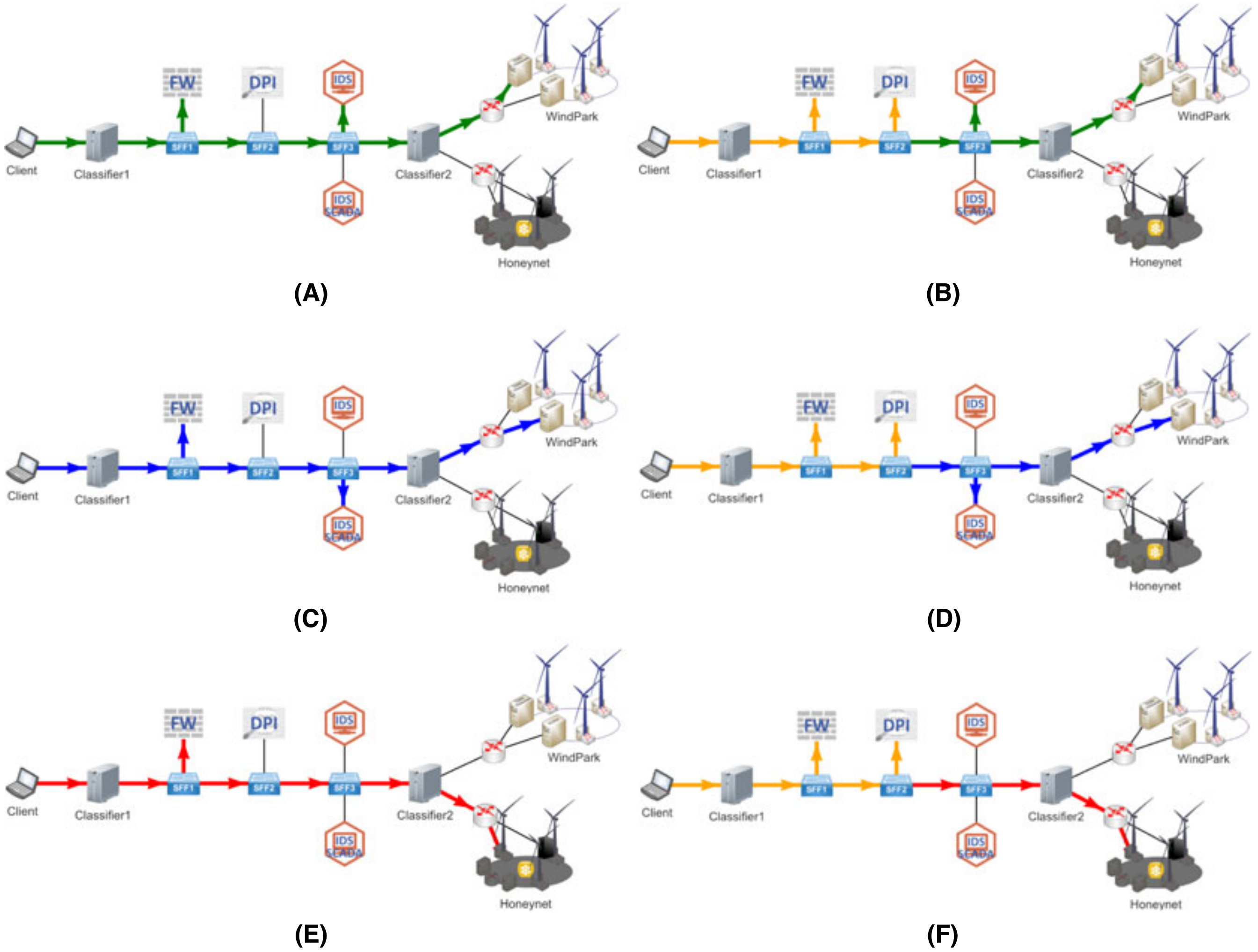


FIGURE 7 Real-time traffic flows as depicted on the controller's graphical user interface. Activated service chains are color coded. A, Legitimate traffic; B, Legitimate traffic classified by deep packet intrusion (DPI); C, supervisory control and data acquisition (SCADA) traffic; D, SCADA traffic classified by DPI; E, Malicious traffic; F, Malicious traffic classified by DPI. FW, firewall; IDS, intrusion detection system; SCADA, supervisory control and data acquisition; SFF, service function forwarder

TABLE 1 Experimental results

Type	DPI	Firewall	IDS	SCADA-IDS	End-Hosts	No. of Functions	Delay
No function					Anywhere	0	0, 45 ms
Chain 1 - legitimate		O	O		Historian	2	66, 57 ms
Chain 2 - SCADA		O		O	SCADA	2	45, 83 ms
Chain 3 - malicious		O			Honeypot	1	13, 53 ms
Chain 4 - unknown -> legitimate	O	O	O		Historian	3	169, 74 ms
Chain 4 - unknown -> SCADA	O	O		O	SCADA	3	138, 92 ms
Chain 4 - unknown -> malicious	O	O			Honeypot	2	116, 72 ms
All functions	O	O	O	O	Anywhere	4	191, 24 ms

5.3 | Analysis of results - an operational wind park's network

In this section, network traces captured from an operational wind park (in Brande, Denmark) are analyzed to highlight the specificities of industrial traffic in the context of this application domain[44] and assess the presented framework's performance in this context.

The subject wind park consists of 4 wind turbines connected in a redundant star topology. These turbines themselves consist of 2 switches in series, 1 at the bottom, the other at the top. Connected to these switches are numerous measurements systems, sensors, and actuators, which communicate with a SCADA server also connected to the star topology (while these connections are currently over wired networks, they are expected to be replaced with wireless links, tailored to industrial environments,[45,46] in the future). A router then ensures the connection between the central switch and the Internet. The park control system mainly consists of 2 parts: the wind farm SCADA system and the wind farm grid control system. While the SCADA server is responsible for the reporting, supervision, acquisition, and storage of data from the turbines, the control system server is responsible for controlling the power output of the different wind turbines and to adapt it to the requirements received from the grid operator. Traces were gathered from 3 different locations, detailed as follows: traffic to/from the grid control system server, traffic to/from the SCADA server, and traffic flowing through the intradomain router. The purpose and requirements of the flows recorded have been analyzed thanks to the input of the network engineers maintaining the subject wind park.

5.3.1 | Traffic to/from the grid control system server

The data to and from the grid control system server was gathered during approximately 715 seconds. Flows to and from the grid control server are mostly instantaneous, especially for TCP, around 1300 instantaneous flows are observed. Most of these TCP flows correspond to exchanges for the verification of wind characteristics. These flows send a very low amount of data (3 frames of 62 bytes for the data, 3 frames of 60 bytes for the acknowledgements). Hence, though numerous, they consume a very little amount of data rate. These flows have an end-to-end latency requirement of 500 ms. Of interest are the flows of longer duration. Both for TCP and UDP, these connections send data at a constant rate. However, it is observed that these connections also consume a very low amount of data rate. Indeed, the most consuming connection has an average rate of 529 kb/s, which is very low in comparison with the available 1 Gb/s links. While the TCP flows correspond to database operations (for logging and scheduling) and have an end-to-end latency requirement of 100 ms, the long duration UDP flows correspond to the regulation communications between the grid control server and the turbines. These UDP flows are the most critical and have a strict low end-to-end latency requirement of 10 ms and averages rates of 496 kb/s, 80 kb/s, 40 kb/s, and 192 b/s depending on the flows.

5.3.2 | Traffic to/from the SCADA server

The data to and from the SCADA server was gathered during approximately 1000 seconds. Whereas the traces contain way much connections than for the grid control server (around 20.0000), most of them are best-effort interdomain access flows. The data rate usage is also very low and most of the connections only send a very small amount of data. More specifically, the most consuming TCP connection consumes 363 kb/s, whereas the UDP connections are really negligible with 8 kb/s for the most consuming one. The UDP connections mostly consist of network time protocol (NTP) and dynamic host configuration protocol exchanges, as the SCADA server hosts both an NTP and a dynamic host configuration protocol server. Though small (average rate in the order of hundreds of b/s), these numerous connections are considered critical and have end-to-end latency requirements of 10 ms and 100 ms, respectively. A substantial amount (around 2000) of single-packet simple network management protocol (SNMP) exchanges with end-to-end latency requirements of 500 ms is also recorded. For their part, the TCP connections mostly consist of online data exchange between the SCADA server and the wind turbines. Though critical, these flows only have end-to-end requirements of 100 ms, 250 ms, and 500 ms depending on the specific service. Lots of instantaneous single-packet UDP exchanges between the SCADA server and the turbines can also be observed with a more stringent end-to-end delay requirement of 10 ms.

5.3.3 | Traffic flowing through the intradomain router

The data flowing through the intradomain router was gathered during approximately 1500 seconds. While the data rate consumption observed in the grid control and SCADA traces was low, the intradomain router trace shows even fewer data transmissions. Most of the TCP connections are instantaneous. These correspond to best-effort interdomain database accesses. The only long duration TCP connection with real-time requirements corresponds to an interdomain database access with an end-to-end real-time requirement of 100 ms. Some short TCP connections can also be observed, which correspond to regulatory exchanges with real-time requirements of 250 ms or 500 ms. Some observed UDP connections with a (nearly) constant rate correspond to critical NTP connections, which have a stringent end-to-end latency requirement of 10 ms. Around 400 seconds and 600 seconds, bursts are observed for several connections. These correspond to SNMP exchanges between the devices in the network and an external network management system. These exchanges are not critical and correspond to scheduled jobs for visual display and reporting.

5.3.4 | Discussion

The analysis of the traces has shown that industrial networks have very low data rate requirements. While a lot of short bursty flows exist, only a few of them have QoS requirements. Though these requirements can reach hundreds of milliseconds, a couple of critical flows have stringent low latency requirements of the order of tens of milliseconds. It was also observed that the traces contain a lot of best-effort traffic, also bursty (short duration) for most of them. Even though the average data rate is not a critical factor, this shows that proper QoS management is needed to guarantee that the real-time requirements of critical flows are met even during the short periods when bursts of best-effort database or SNMP exchanges are also transmitted on the network.

Considering the performance of the framework, as evaluated, and in light of the above findings on the actual wind park, several conclusions can be made. An important observation for the results is that the proposed reactive framework is applicable to operational wind parks, as the additional delays are affordable for most critical services (which all have latency requirements of 100 ms, 250 ms, or 500 ms). Furthermore, although the emulated experiments of service SFC provisioning were conducted between VMs on a same server, thus minimizing network delays, the multiple gigabit interconnections present in a wind park all feature low latencies (through over-provisioning), and are therefore expected to introduce minimal delays. Finally, the evaluation of the proposed framework in an actual wind park validates the feasibility of the approach, providing the necessary sophisticated, dynamic, and continuous security monitoring required in industrial networks nowadays.

6 | CONCLUSIONS AND FUTURE WORK

This work presented an approach to achieve reactive security for SDN/NFV-enabled industrial networks based on the use of SFC to dynamically chain various security functions, classify traffic, and steer traffic accordingly. The proof-of-concept application of this approach led to the development of a reactive security framework modeled 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 (eg, from script kiddies to state actors). 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 minimized as the traffic only has to go through 1 DPI instance and the same can be said for the IDS/IPS functionality, as, eg, SCADA traffic only has to go through a faster-performing SCADA-specific IDS instance. The performance evaluation of the framework's implementation validates the feasibility of the approach, also considering the actual performance and requirements of an operating wind park.

As for the future work, improvements will be investigated in both the security service functions as well as the implementation of the DPI functionality (essential for traffic-type classification) to minimize the performance impact of the framework and enable its use in more time-critical industrial applications. The framework will be enhanced via the use of an NFV MANO software stack, which, via the definition of service templates, will be responsible for the boot up of the necessary VMs using a virtual infrastructure management software (eg, OpenStack). The MANO can be used to program the ODL controller, passing the necessary information to the SFC manager, also enabling accurate monitoring of the service functions' resources (eg, triggering the instantiation of additional VMs when a function is overloaded). Moreover, the automated reactiveness of the framework will be enhanced with the integration of SDN security patterns[47,48] 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 and heterogeneous platforms,[49] even across domains.[50] Finally, the performance of the individual components and the framework as a whole will be evaluated in detail to assess the impact of the proposed mechanisms in the context of the industrial domain and its associated intricacies. For this purpose, a testbed is already being setup on the operating wind park in Brande, Denmark where the trace analysis was conducted; this testbed will form the basis for the real-time evaluation of the framework's performance as well as its behavior under different attack scenarios.

ACKNOWLEDGEMENTS

This work has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant 671648 (VirtuWind). The authors would like to thank their colleagues from project VirtuWind for their valuable feedback on the framework's design as well as the network engineers maintaining the subject wind park for their input in defining the application requirements and analyzing the network traces.

ORCID

Nikolaos E. Petroulakis http://orcid.org/0000-0002-3489-7763

REFERENCES

1. da Silva MJ, Lins T, Oliveira RAR. Software-defined networking for industry 4.0. Paper presented at: The 20th Advanced International Conference on Telecommunications; 2016; Valencia, Spain.
2. Petroulakis N, Mahmoodi T, Kulkarni V, et al. Virtuwind: virtual and programmable industrial network prototype deployed in operational wind park. Paper presented at: 25th European Conference on Networks and Communications; June 27-30 2016; Athens, Greece.
3. Cyber-attack against Ukrainian Critical Infrastructure. Alert (IR-ALERT-H-16-056-01); 2015.
4. NERC Standard CIP. 007-6-Cyber Security-Systems Security Management; 2013.
5. IETF RFC 7665. Service Function chaining (SFC) Architecture; 2015.
6. Fysarakis K, Petroulakis NE, Roos A, et al. A reactive security framework for operational wind parks using service function chaining. Paper presented at: IEEE Symposium on Computers and Communications (ISCC); 2017; Heraklion, Greece.
7. Mahmoodi T, Kulkarni V, Kellerer W, et al. Virtuwind: virtual and programmable industrial network prototype deployed in operational wind park. *Trans Emerging Tel Tech*. 2016;27(9):1281-1288.
8. Quinn P, Nadeau T. Problem statement for service function chaining. RFC 7498; 2015.
9. L4-L7 Service Function Chaining Solution Architecture. Open Networking Foundation; 2015.
10. SFC environment Security requirements. https://tools.ietf.org/html/draft-mglt-sfc/securityenvironment-req-01
11. Bhamare D, Jain R, Samaka M, Erbad A. A survey on service function chaining. *J Netw Comput Appl*. 2016;75:138-155.
12. Quinn P, Guichard J. Service function chaining: creating a service plane via network service headers. *Computer*. 2014;47(11):38-44.
13. Quinn P, Elzur U. Network service header. Network Working Group, IETF Draft; 2016.
14. Zhang Y, Beheshti N, Beliveau L, et al. StEERING: a software-defined networking for inline service chaining. Paper presented at: 21st IEEE International Conference on Network Protocols (ICNP); 2013; Göettingen, Germany.
15. Qazi ZA, Tu C, Chiang L, Miao R, Sekar V, Yu M. SIMPLE-fying middlebox policy enforcement using SDN. Paper presented at: ACM SIGCOMM 2013 Conference on SIGCOMM; 2013; New York, NY, USA.
16. GS NFV-SEC 013 ETSI. Network Functions Virtualisation (NFV) Release 3; Security; Security Management and Monitoring Specification; 2017.
17. Mehraghdam S, Keller M, Karl H. Specifying and placing chains of virtual network functions. Paper presented at: IEEE 3rd International Conference on Cloud Networking (CloudNet); 2014; Luxembourg, Luxembourg.
18. Blendin J, Ruckert J, Leymann N, Schyguda G, Hausheer D. Position paper: software-defined network service chaining. Paper presented at: 3rd European Workshop on Software Defined Networks (EWSDN); 2014; London, UK.
19. Kumar S, Tufail M, Majee S, Captari C, Homma S. Service function chaining use cases in data centers. *IETF SFC WG*. 2016.
20. Haeffner W, Napper J, Stiemerling M, Lopez D, Uttaro J. Service function chaining use cases in mobile networks. *Internet Engineering Task Force*. 2015.
21. John W, Pentikousis K, Agapiou G, et al. Research directions in network service chaining. Paper presented at: IEEE SDN for Future Networks and Services (SDN4FNS); 2013; Trento, Italy.
22. Liao H, Richard Lin C, Lin Y, Tung K. Intrusion detection system: a comprehensive review. *J Netw Comput Appl*. 2013;36(1):16-24.
23. Vokorokos L, Ennert M, Cajkovský M, Radušovský J. A survey of parallel intrusion detection on graphical processors. *Open Comput Sci*. 2014;4(4):222-230.
24. Bremler-Barr A, Harchol Y, Hay D, Koral Y. Deep packet inspection as a service. Paper presented at: Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies; 2014; Sydney, Australia.
25. Microservices: a definition of this new architectural term. http://martinfowler.com/articles/microservices.html
26. Thönes J. Microservices. *IEEE Softw*. 2015;32(1):116.
27. Microservices Five architectural constraints. http://www.nirmata.com/2015/02/microservices-five-architectural\discretionary{-}{}{}constraints/
28. IETF. Virtual eXtensible Local Area Network (VXLAN): A framework for overlaying virtualized layer 2 networks over layer 3 networks [Technical Report]. RFC 7348; 2014.
29. Service function chaining (sfc) working group. https://datatracker.ietf.org/wg/sfc/charter/

30. Snort. http://blog.snort.org/2012/01/snort-292-scada-preprocessors.html
31. Snort 2.9.2: SCADA Preprocessors. http://www.snort.org
32. Honeyd. https://github.com/sk4ld/gridpot
33. SCADA HoneyNet Project. http://scadahoneynet.sourceforge.net
34. Chatziadam P, Askoxylakis IG, Fragkiadakis A. A network telescope for early warning intrusion detection. Paper presented at: International Conference on Human Aspects of Information Security, Privacy, and Trust; 2014; Heraklion, Greece.
35. Open Source Security. https://pfsense.org/
36. VM-Series: Next-Generation Security for Private and Public Clouds. https://www.paloaltonetworks.com/
37. nDPI: Open and Extensible LGPLv3 Deep Packet Inspection Library. http://www.ntop.org/products/deep-packet-inspection/ndpi/
38. OpenDaylight: open source SDN platform. https://www.opendaylight.org
39. ODL wiki: Service Function Chaining. https://wiki.opendaylight.org/view/Service_Function_Chaining:Main
40. Nodejs library. http://www.nodejs.org
41. ETSI Group Specification NFV 002. Network Functions Virtualisation (NFV); Architectural Framework; 2014. http://www.etsi.org/deliver/etsi_gs/nfv
42. Proxmox Virtual Environment. http://www.proxmnox.com
43. Open vSwitch. http://www.openvswitch.org
44. Project VirtuWind. Deliverable D3.2: detailed intra-domain SDN & NFV architecture; 2017.
45. Martinez B, Vilajosana X, Kim IL, et al. I3Mote: an open development platform for the intelligent industrial internet. *Sensors*. 2017;17(5):986.
46. Dujovne D, Watteyne T, Vilajosana X, Thubert P. 6TiSCH: deterministic IP-enabled industrial internet (of things). *IEEE Commun Mag*. 2014;52(12):36-41.
47. Petroulakis NE, Spanoudakis G, Askoxylakis IG. Patterns for the design of secure and dependable software defined networks. *Comput Netw*. 2016;109:39-49.
48. Petroulakis NE, Spanoudakis G, Askoxylakis I, Miaoudakis A, Traganitis A. A pattern-based approach for designing reliable cyber-physical systems. Paper presented at: IEEE Global Communications Conference (GLOBECOM); 2015; San Diego, CA.
49. Fysarakis K, Hatzivasilis G, Askoxylakis I, Manifavas C. RT-SPDM: real-time security, privacy and dependability management of heterogeneous systems. Paper presented at: International Conference on Human Aspects of Information Security, Privacy, and Trust; 2015; Los Angeles, CA.
50. Fysarakis K, Soultatos O, Papaefstathiou I, Askoxylakis I. XSACd - cross-domain resource sharing & access control for smart environments. *Futur Gener Comput Syst*. 2016. https://doi.org/10.1016/j.future.2016.05.023

How to cite this article: Petroulakis NE, Fysarakis K, Askoxylakis I, Spanoudakis G. Reactive security for SDN/NFV-enabled industrial networks leveraging service function chaining. *Trans Emerging Tel Tech*. 2018;29:e3269. https://doi.org/10.1002/ett.3269