SDN-enabled virtual data diode

Miguel Borges de Freitas[1], Luis Rosa[1], Tiago Cruz[1], and Paulo Simões[1]

Centre of Informatics and Systems, Department of Informatics Engineering,
University of Coimbra, Portugal
{miguelbf,lmrosa,tjcruz,psimoes}@dei.uc.pt

Abstract. The growing number of cyber-attacks targeting critical infrastructures, as well as the effort to ensure compliance with security standards (e.g. Common Criteria certifications), has pushed for Industrial Automation Control Systems to move away from the use of conventional firewalls in favor of hardware-enforced strict unidirectional gateways (data diodes). However, with the expected increase in the number of interconnected devices, the sole use of data diodes for network isolation may become financially impractical for some infrastructure operators.
This paper proposes an alternative, designed to leverage the benefits of Software Defined Networking (SDN) to virtualize the data diode. Besides presenting the proposed approach, a review of data diode products is also provided, along with an overview of multiple SDN-based strategies designed to emulate the same functionality. The proposed solution was evaluated by means of a prototype implementation built on top of a distributed SDN controller and designed for multi-tenant network environments. This prototype, which was developed with a focus in performance and availability quality attributes, is able to deploy a virtual data diode in the millisecond range while keeping the latency of the data plane to minimal values.

Keywords: Data Diode · Unidirectional gateways · Software Defined Networks · Industrial and Automation Control Systems.

1 Introduction

Industrial Automation and Control Systems (IACS) encompass a broad range of networks and systems used to monitor, manage and control cyber-physical processes in critical infrastructures, such as the power grid or water distribution facilities. The growing number of cyber-attacks against today's highly distributed IACS is raising awareness towards the need for in-depth cyber-security strategies, somehow leading to a shift back to these system's origins with the use of data diodes. When SCADA systems first appeared in the 1960's they were implemented as air-gapped islands restricted to the process control perimeter and specially isolated from corporative networks. Security was granted due to intrinsic isolation and the use of proprietary and poorly documented protocols [9].

In the 1990's, business requirements to increase productivity and performance, together and the massification of ICT technologies, broke with the previous isolated generation of IACS. Organizations began to adopt open TCP/IP

connections to link their process control and Enterprise Resource Planning (ERP) systems. Corporate management layers took advantage of this real-time data to manage plant inventories, control product quality and monitor specific process variables. It is estimated this network interconnection lead to 3-8% cost savings at large facilities [15] – however, it also brought a drastic increase on the inherent cyber-security risks, by contributing to expand the exposed IACS attack surface.

To mitigate unwanted accesses to the IACS network, middleboxes such as firewalls started to be implemented as digital barriers in the perimeter of both process and organizational networks, sometimes sitting behind a DMZ. Firewalls are often prone to configuration mistakes and are relatively accessible for exploit development by skilled individuals. In the long-term, firewalls are known to have considerable operating costs as firewall rules have to be continuously audited and maintained while firmware updates must be installed as soon as they are available [16]. The use of firewalls on IACS networks also contradicts some of their fundamental requirements: the need for real-time access to plant data, high availability and service continuity. As middleboxes, commercial off-the-shelf (COTS) firewalls introduce latency and jitter in the network, also introducing a point-of-failure (e.g., when subject to flooding attacks, throttling policies may cause service disruption).

Unlike firewalls, data diodes provide a physical mechanism for enforcing strict one-way communication between two networks. They are also known as *unidirectional gateways* since data can be securely transfered from an restricted access network (such as a process control network) to a less secure network (the corporate zone) with no chances of reverse communication. Data diodes are often built using fibre optics transceivers, through the removal of the transmitting component (TX) from one side of the communication and the respective receiver component (RX) from the opposite side [11]. This makes it physically impossible to compromise such devices to achieve reverse connectivity. Moreover, they usually do not contain firmware, requiring minimal or no configuration at all, or have minimal software supported by micro-kernels that can be formally verified [5].

Data diodes allow organizations to retrieve valuable data generated at the process level, while guaranteeing the trustworthiness and isolation of the critical infrastructure. They are the only devices receiving the Evaluation Assurance Level 7 (EAL7) grade in the Common Criteria security evaluation international standard. As a result, NIST recommends the adoption of data diodes [17].

Despite its advantages, from a security standpoint, data diode implementations come with high capital expenditure for organizations: it is estimated that for a typical large complex facility such costs can reach $250,000 while recurring support costs may ascend to values circa $50,000/year per data diode [19]. Furthermore, most data diode solutions are vendor-dependent, with the range of supported protocols strongly depending on the specific implementation – this means that many protocols on which some organizations rely upon may not be supported at all. Moreover, like any middlebox, data diodes need to be physically placed at a specific point in the network topology to be able to block network traffic, eventually requiring multiple deployments to secure dispersed network

segments. Considering such shortcomings, many organizations may not be willing to invest in devices that are not future-proof or lack flexibility, fearing they may become outdated by the time their break-even point is reached.

To deal with the inherent limitations of existing solutions, we propose using Software Defined Networking (SDN) and Network Function Virtualization (NFV) to implement a cost-effective data diode. SDN aims at shifting the network equipment control plane functionality to a logically centralized entity – the network controller. In SDN, network switches are turned into "dumb" devices whose forwarding tables are updated by the network controller, using open protocols such as Openflow [13]. NFV provides a way to decouple network equipment functionality in several chained Virtual Network Functions (VNFs), which may be hosted in dispersed infrastructure points-of-presence.

SDN can be leveraged to implement innovative network security approaches: the network controller has a global view of the network topology graph, has real-time state awareness over all allowed network flows and can modify the network state by means of a proactive (preinitializing flow rules) or reactive (deciding upon packet arrival) approach. For such reasons, an SDN-based data diode could provide an alternative to both firewalls and conventional appliances. Note that to efficiently forward network packets, general purpose network switches contain forwarding tables called TCAMs (*ternary content-addressable memory*) which are able to perform an entire table lookup in just one clock cycle [18]. Hence, an SDN-enabled virtual data diode could effectively block traffic at Layer 2, avoiding the typical latency imposed by firewall middleboxes. Vendor lock-in, management complexity and deployment issues are also mitigated due to the use of open protocols, the existence of a single managing interface to control the overall network and the removal of placement restrictions imposed by hardware appliances. SDN also helps future proofing virtual data diode implementations: the data diode application can be easily adapted to support new protocols, and/or new network functions can be added to the network via NFV.

The remainder of this paper is organized as follows. Section 2 provides a review of the major COTS data diode products, together with an overview of the main challenges for protocol support in unidirectional communications. Section 3 explains how SDN can be leveraged to implement a functional data diode. Section 4 presents our proof-of-concept (PoC) prototype: a simple SDN data diode that is able to support the UDP protocol, implemented in a distributed network controller environment and geared towards performance and availability. Finally, Section 6 provides a wrap-up discussion and concludes the paper.

2 Data diodes for IACS security

Data diodes are devices that restrict the communication in a network connection so that data can only travel in a single direction, having borrowed their name from electronic diode semiconductors. Although different hardware implementations exist, supporting different physical layers (e.g. RS-232, USB, Ethernet), most make use of optical couplers to guarantee physical isolation. The trans-

mitter side of a data diode converts electrical signals to optical form using light emitting diodes (LEDs), while at the receiving end photo-transistors convert the optical data back to electrical form [8]. It is the physical air-gap in the optical-coupler that makes data diode devices so secure and appealing in the critical infrastructure context.

In IACS, data diodes are often deployed to isolate specific network domains or between corporate and process control networks, to support the unidirectional transfer of historian data, HMI screen replication, or for one-way telemetry (operational data, security events, alarms and syslog). Data diodes are commercially available in two different form factors: single-box solutions and PCI express cards [10]. The former category may also encompass single-box or split-device variations, in which a component is deployed at each side of the connection.

Despite the similarities in the key isolation mechanism, commercially available solutions differ significantly in terms of supported services and protocols. Data diodes have to make use of additional software components for each side of the unidirectional link to be able to support TCP/IP-based SCADA protocols, such as MODBUS/TCP, Ethernet/IP and DNP3. Such protocols were designed for bidirectional operation, relying on a three-way handshake and requiring continuous acknowledgments between communication peers. In [6] the TCP workflow in unidirectional links is explained along with the presentation of a design for a unidirectional gateway for IACS applications (Figure 1).

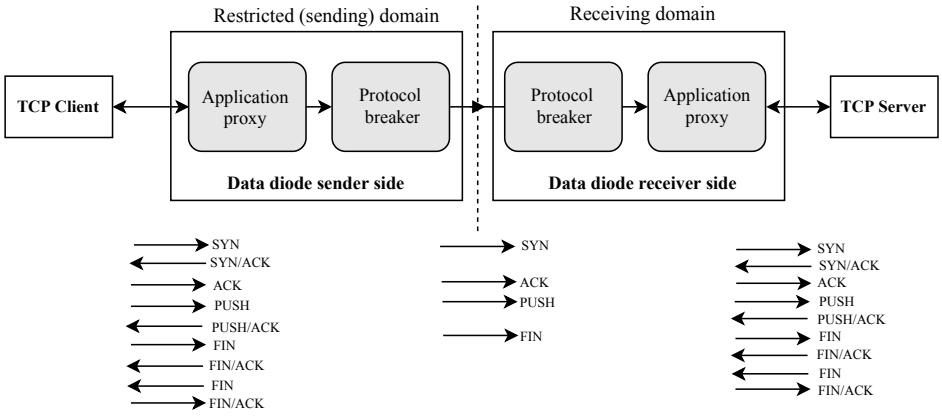

Fig. 1: TCP workflow in uni-directional gateways (adapted from [6]).

The architecture includes two different components at the edge of the unidirectional link: (i) an application proxy and (ii) a protocol breaker. The former is responsible for acting as a proxy for TCP connections. In the sender side of the data diode, the application proxy operates as a TCP server, automatically responding with SYN/ACK, PUSH/ACK and FIN/ACK to any SYN, PUSH or FIN packets sent by the TCP client. Any packet generated by the TCP client is

forwarded by the application proxy to the unidirectional link. On the receiving end, the application proxy simply emulates the TCP client forwarding any received packets. The protocol breaker component acts as a middleware for packet encapsulation for protocols that do not require acknowledgments (e.g. UDP). It can also be used to provide confidentiality within the unidirectional link or to apply forward error correction to the data transfer.

2.1 Data Diode Products

There are diverse commercial data diode solutions in the market, depending on the specific use case and protocol support. Table 1 provides a summary of the three most notorious products in the context of IACS. Next, we provide a brief review of those products, based on publicly available documentation.

Table 1: IACS commercial data diodes.

Company	**Owl CyberDefense**	**Fox-IT**	**Waterfall**
Form Factors	1U rack mount, DIN rail, PCIe cards	1U rack mount	Modular designs: gateway pairs (1U), single box(1U), DIN rail
Bandwidth	10Gbps	1.25Gbps	1Gbps
IACS Applications	Rockwell, OSIsoft PI, Schneider Electric	OSIsoft PI	OSIsoft, GE, Schneider Electric, Siemens, Emerson, Areva, Honeywell, AspenTech, Scientech, Rockwell
IACS Protocols	Modbus, OPC	Modbus, DNP3, OPC, ICCP	OPC DA/HDA (backfill)/UA, A&E, DNP3, ICCP, Siemens S7, Modbus, Modbus Plus, IEC 60870-5-104, IEC 61850
CC Certification	EAL4	EAL7+	EAL4+

Owl CyberDefense provides the DualDiode^{TM} technology as part of the company cross domain solution portfolio. Owl's data diodes make use of a hardened Linux kernel, providing optical separation and a protocol breaker that converts all packets to non-routable Asynchronous Transfer Mode (ATM) cells, also supporting data transfers up to 10Gbps [14]. Protocol support includes TCP/IP connections, UDP, Modbus and the OPC family, as well as historian solutions from Rockwell Automation, Schneider Electric and OSIsoft. Latest revisions of DualDiode^{TM} Network Interface cards received CC EAL4 certification.

Fox-IT's DataDiode^{TM}, is compliant with the highest level of CC certification: EAL7+ [3]. It implements full protocol break capabilities and uses a single

optical fiber strand, together with custom optoelectronics designed for one-way operation. Being a firmware-less device, it has no configuration or local state, relying on proxy servers deployed on each side of the connection. These proxies implement several techniques for error detection and increased reliability, using metadata for lost packet detection (supported by proxy-level logs, for manual retransmission), forward error correction codes and heartbeat mechanisms [4]. In government editions, the device includes an anti-tampering mechanism [3]. The Fox-IT data diode is able to achieve 1.25Gbps in the link layer, although the actual speed is lower due to the proxy servers. It claims to support Modbus, DNP3, OPC and ICCP protocols along with file transfers, SMTP, CIFS, UDP and NTP [7]. The OSIsoft PI Historian is also supported.

Waterfall Security Solutions provides data diode appliances in multiple form-factors, including split-pair, single-box and DIN rail versions, based on a modular combination of hardware and software[22]. Such unidirectional gateways include a TX-only module (containing a fiber-optic laser), a fiber optic cable, an RX module (optical receiver), together with host modules that gather data from industrial servers and emulate different protocols and industrial devices. The latter are provided either as standalone physical modules or virtual machines. Popular industrial applications/historians are supported (e.g. Osisoft PI System, GE iHistorian, Schneider-Electric Instep eDNA), as well as a long list of industrial protocols (e.g. Modbus, DNP3, OPC, Modbus Plus [20]). Devices support up to 1Gbps data transfers and are certified EAL4+. The company recently announced a reversible hardware-enforced unidirectional gateway whose direction can be controlled by software, using a schedule or exception-based trigger mechanism [21].

3 Leveraging SDN to virtualize the data diode

The OpenFlow protocol is a Layer 3 network protocol that gives access to the forwarding plane of a network switch over the network. It enables network controllers to determine the path of network packets across the switch fabric. The protocol works on top of TCP/IP although the communication between the controller and the switch can also be configured to make use of the Transport Layer Security (TLS) protocol. The protocol works in a *match-action* manner: when a packet arrives at a switch port, the switch starts by performing a table lookup in the first flow table to match the packet headers against the set of flow rules installed in the switch. If a match is found, the switch applies the instruction set configured in the flow rule. In case of a table miss, the corresponding packet action depends on the table configuration: the packet can be forwarded to the controller for further processing (using *Packet-In* messages), can be moved further on the flow table pipeline, can have header fields re-written or can simply be dropped [13]. The match fields in an OpenFlow flow table comprise fields ranging from Layer1 to Layer4 (Table 2) permitting a fine-grained control over the packet identification and ultimate destination.

Table 2: The OpenFlow flow table match fields [13].

Match Field	Description
IN_PORT	Ingress Port (physical or a switch defined logical port)
ETH_DST	Ethernet destination MAC address
ETH_SRC	Ethernet source MAC address
ETH_TYPE	Ethertype of the packet payload
IPv4_SRC	Source IP address
IPv4_DST	Destination IP address
IPv6_SRC	Source IP address (IPv6 format)
IPv6_DST	Destination IP address (IPv6 format)
TCP_SRC	TCP source port
TCP_DST	TCP destination port
UDP_SRC	UDP source port
UDP_DST	UDP destination port

Taking into account the workflow of a packet reaching an OpenFlow enabled switch, we identify three different approaches for an SDN-based virtual data diode: proactive; reactive; and NFV-assisted.

3.1 Proactive data diode

A proactive data diode is an SDN unidirectional gateway implementation that takes advantage of OpenFlow's proactive flow rule instantiation It is the simplest and most limited implementation since it can only support applications that rely on the UDP protocol. Considering two networks with different degrees of classification (cf. Figure 3), the network controller installs (in advance) two rules in the restricted (sending) domain uplink switch (cf. Table 3). One of the rules instructs the switch to drop any packets entering the switch and originating at the switch port that is connected to the receiving network uplink switch. The other rule forwards any packets entering the remaining switch ports to the receiving domain uplink switch port.

Further limitations can be applied in the second flow rule to limit the devices from the receiving domain network that are allowed to unidirectionally transfer data, using the IN_PORT, ETH_SRC and IPv4_SRC/IPv6_SRC match fields. For the proactive data diode to support TCP applications, the sending machine has to encapsulate the packet into an UDP packet. Alternatively, flow rules can be installed on the switch to set the UDP source and destination ports (and replace the TCP source and destination fields) to any TCP packets entering the switch. In both cases, in order to support the TCP protocol, additional software is required in the receiving machine to disassemble the received packets into usable data. For this type of virtual data diode, only the uplink switch in the restricted network domain needs to support OpenFlow. The remaining sections of both networks may still rely on traditional network architectures.

3.2 Reactive data diode

Instead of installing rules in the up-link network switch, the reactive data diode instructs the switch to forward any received packet headers to the network controller for further processing (Table 4). The network controller can check if the received packet comes from the receiving domain (by looking up the input port) and simply instruct the switch to drop the packet. Similarly, it can instruct the switch to forward the packet if it originates from the restricted (sending) network domain.

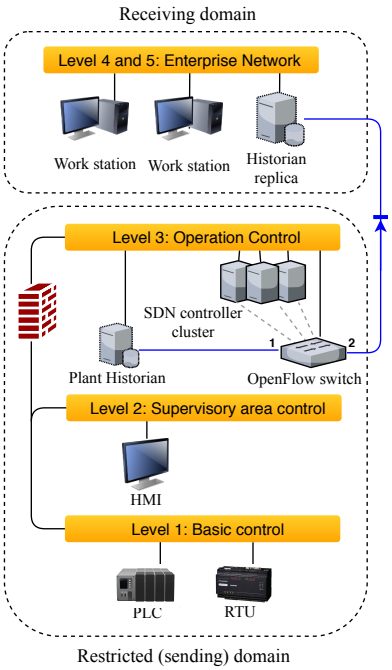

Fig. 2: SDN-enabled virtual data diode.

Table 3: Proactive data diode flow table.

Table	Match Fields	Action
0	in_port=1	output:2
0	in_port=2	drop

Table 4: Reactive data diode flow table.

Table	Match Fields	Action
0	in_port=1	output:controller
0	in_port=2	output:controller

Using a reactive approach, the network controller has greater flexibility since it can add support to the TCP protocol. It can behave as *an application proxy* for TCP connections implementing a workflow similar to the one in Figure 1. TCP acknowledge packets can be faked by the controller and outputted via a switch port to the host establishing the connection. Hence, the TCP protocol can be supported while still only allowing unidirectional communications as long as an application proxy is able to perform the same workflow in the low-priority network. Furthermore, there are some cases in which bi-directional communication between both networks is required or should be temporarily enabled (e.g. an application that relies on TCP for initial connection establishment). The network controller can be programmed in such a way that bi-directional communication is enabled in certain situations. Thus, it is possible to emulate the behavior of the Waterfall's reversible data diode. Despite the provided flexibility, this approach

introduces latency in network flows (due to additional packet processing) and the network controller is vulnerable to flooding attacks. Packets originating in the receiving network domain will not be forwarded to the hosts on the restricted domain without the permission of the controller. Nevertheless, hosts on the receiving domain are able to flood the OpenFlow switch with packets destined to the restricted network. Those packets are ultimately redirected to the controller, causing a denial of service which disrupts the unidirectional communication that is expected to happen in the reverse direction.

3.3 NFV-assisted data diode

This approach requires SDN support at the edge of the restricted (sending) network, as well as a virtualization infrastructure containing a virtual OpenFlow switch (e.g. OpenvSwitch). It represents a combined approach where the processing step is supported by virtualized hosts close to the uplink OpenFlow switch and directly accessible to the SDN network. Network traffic originating in the restricted domain with the receiving network as destination is offloaded by the first OpenFlow switch to a dedicated virtual host. This virtual host can either be a virtual machine or an application container with two virtual Ethernet interfaces: one for receiving network packets and another for the output of packets. TCP emulation is performed within the virtual host by automatically generating acknowledgment packets for the three-way handshake and subsequent TCP transfers. Packets that are meant to be sent to the low priority network are chained from the input virtual interface to the output virtual interface (e.g. using IPtables).

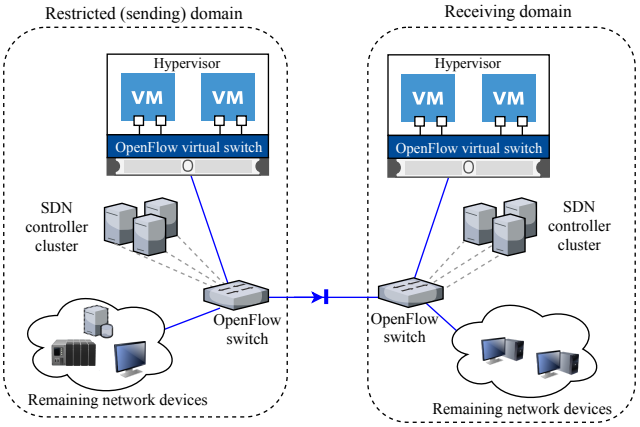

Fig. 3: NFV assisted SDN virtual data diode.

Flow rules are proactively installed by the network controller in the uplink switch to: (i) drop any packets coming from the receiving network; (ii) forward any packets from the virtualization host (output port) to the switch port connected to the receiving network; (iii) forward any other packets to the virtualiza-

tion host input port. In the receiving network domain, a TCP emulation proxy host should also exist and a similar approach can be applied. This data diode implementation avoids flooding attacks against the control plane while keeping the flexibility of the reactive design approach. Protocol support can easily be added to the virtual application proxy. The global network topology available at the controller can be used to automatically find the path (sequence of ports) leading to the virtual host. Moreover, if a layer of orchestration is added to the controller, it can continuously monitor the state of the virtual host and request the creation of a new one in case of failure (adjusting the flow rules to respect the new virtual ports).

4 Proof-of-Concept Virtual Data Diode Prototype

In the context of IACS, availability, performance and the need for real-time operation are the key design system attributes. As such, we developed our PoC virtual data diode using distributed SDN controllers, so that the control plane itself does not represent a single point of failure in the overall system operation. Distributed controllers are multi-node architectures where each OpenFlow switch maintains an active connection to one of the controller nodes (the master node) but is configured to use redundant connections to other nodes (slaves), in the case of master node failures. Although many network controller projects exist, only a small minority is distributed [12]. Among those, we selected the Open Network Operating System (ONOS) because it matches well into the critical infrastructure use-cases: high throughput (up to 1 M requests/second), low latency (10-100 ms event processing) and high availability (99.99% service availability)[2]. Our PoC virtual data diode uses a proactive approach regarding flow rule instantiation. Flow rules are installed from a dashboard containing the global topology graph of the network. Using this approach, the OpenFlow switches are still able to virtualize a data diode even in the case of an hypothetical full control plane failure. To increase performance, the data diode does not rely on any controller external interfaces. It was implemented directly in the application (using its OSGi services), extending its external interfaces (REST, command-line and websockets). Figure 4 presents the PoC architecture.

By default there is no connectivity between hosts in the SDN network. The *Proxy ARP* application (ONOS-bundled) proactively installs rules in the switch fabric to forward any ARP packets to the controller so the topology graph and host location can be computed. The developed *Network Manager* application relies on intent-based networking to provide connectivity between a set of hosts in the network. Intents are ONOS high-level abstractions (protocol independent) that allow applications to define generic connectivity policies that are translated internally to flow rules. For each host pair, the host-to-host intent results into two installed rules (with fixed priority) using the *in_port*, *eth_src* and *eth_dst* as match-fields and outputting to a port leading to a path to the host location. ONOS monitors the network state and any installed intents: if a network switch is unavailable and a redundant path between hosts exists, a new set of flow rules

is generated and installed, keeping the intent active. By ensuring selected host connectivity, the *Network Manager* application creates logical subsections in the overall topology graph, providing the basis for multi-tenancy.

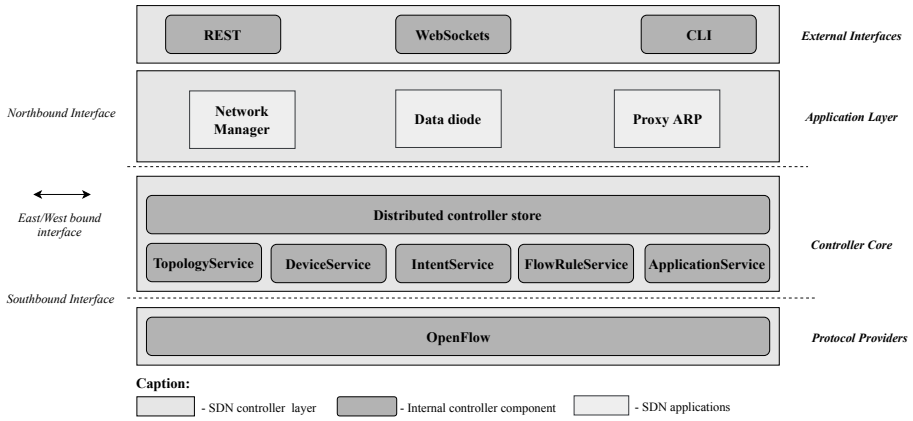

Fig. 4: Architecture of the Virtual Data Diode PoC.

The *Data Diode* application then uses the information stored by the *Network Manager*. When a deployment is requested, given a topology edge link and the network name, the application finds the connection point (host-switch/port) in the graph and requests the *Network Manager* the list of hosts belonging to that network. The application then installs one rule per network host-pair in the edge switch (identical to one of the rules installed by the *Network Manager*) with the action field set to *Drop*. Those flow rules have a higher priority field than the rules defined by the *Network Manager* application superseding them. A workflow similar to the one depicted in Table 3 was not followed in the implemented prototype, in order to avoid binding physical ports to data diode deployments and preserve multitenancy support. Additionally, the *Data Diode* implements a monitor that asynchronously receives any events produced by the network (*Network Manager* application) – its purpose is to install new rules to enforce the diode behavior for each new host.

5 Evaluation

This section discusses the experimental evaluation of our PoC virtual data diode.

5.1 Experimental Testbed

Figure 5 ilustrates the testbed and network topology used for the validation of the virtual data diode prototype. It consists of a single OpenFlow switch controller by a three node ONOS cluster. The OpenFlow switch was running OpenvSwitch (CentOS 7) in a COTS server (Dell Poweredge R210), with six available gigabit Ethernet interfaces. The server was configured with Intel DPDK

for increased network performance (bypassing the Linux Kernel and promoting direct memory access using hugepages and the VFIO universal IO driver). The switch configured to use the three controller plane nodes, connects to the master node via an off-band management network not accessible to the hosts in the SDN network. The three controller nodes were CentOS 7 virtual machines, each with 4GB of RAM. The network hosts are composed by an Environmental Monitoring Unit (EMU) and two Modbus TCP agents. The EMU is an arduino-based board with built-in Ethernet ASIC (Freetonics EthertTen), containing a DTH11 sensor and an electromechanical relay. The temperature, humidity and relay state values are kept updated in three holding registers, and made available in the SDN network via the Modbus TCP protocol.

The Modbus TX and RX hosts are virtual machines with gigabit ethernet configured in passthrough mode. Their role is to emulate the behavior application proxies and protocol breakers found in commercial data diodes (cf. Figure 1). The TX agent queries the EMU holding registries, serializes the data into the pickle format and sends it through the UDP protocol to the RX agent. This RX agent behaves as the EMU device on the other side of the network. It desserializes the received data, updates the internal registries and exposes a Modbus TCP server. A virtual data diode was deployed (from the SDN controller) in the edge link connecting the switch to the RX agent. Thus, the connection between both agents was considered unidirectional (TX $\rightarrow$ RX only).

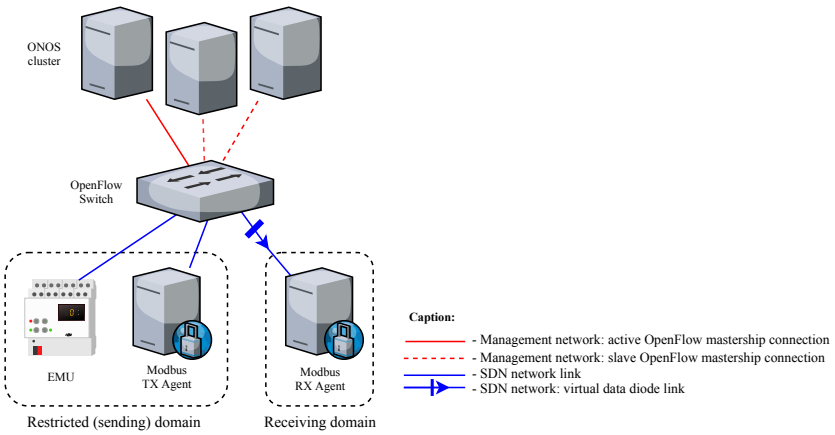

Fig. 5: Experimental testbed.

5.2 Validation and Lessons Learned

The functional validation of the virtual data diode was achieved recurring to the Netcat tool: the RX agent was configured as an UDP server while the TX agent acted as a client and vice-versa. We confirmed that in the former case packets were able to flow while in the last no communication occurred. The non-functional validation focused on assessing the prototype performance. Experiments focused on three aspects:

(a) the effect of the data layer on the latency of Modbus TCP readings;
(b) the overall network performance of the data plane;
(c) and the deployment latency of the virtual data diode.

For (a) we designed a test consisting of an increasing number of sequential reads of ten EMU holding registries. For the TX agent we removed the ability to process and packetize the obtained data and measured the time immediately before and after each query. The measured times should be taken as the base values for reading latency. For the RX readings, the time was recorded right after data has been desserialized and updated in the agent context. Furthermore, a counter was increased upon receiving a reading from the TX agent. Total test duration was computed using the temporal instant before the first query by the TX agent as starting time. Both machines were synchronized via NTP before performing the test and each test was repeated five times. Table 5 summarizes the obtained latencies (and percentage of failed readings). Confidence intervals were calculated using a t-student distribution with a 95% confidence interval.

Table 5: Latency effect of the data layer on Modbus TCP readings.

Modbus Agent	**Number of Queries**	**Time (s)**	**Failed Reads (%)**
TX	1	0.067 ± 0.139	-
	10	9.889 ± 0.640	-
	100	111.045 ± 0.331	-
	500	566.654 ± 0.558	-
RX	1	0.654 ± 0.344	0
	10	10.185 ± 0.777	0
	100	111.820 ± 0.897	0
	500	567.679 ± 0.549	0.360 ± 0.444

It is possible to conclude that even though the added latencies show a cumulative effect with respect to the number of readings (almost defining a linear trend) the latency increase is almost negligible. For 500 EMU readings, the additional processing by the agents and the subsequent network transfer only delays the overall reading time by 1 second. It is also possible to see that, as the number of queries increases, we start noticing a minimal amount of readings not reaching the RX agent – although being reported as sent by the TX agent. This can be explained by the no-guarantee nature of the UDP protocol. While this problem could be mitigated by adding error correction mechanisms to the unidirectional data packets or sending the same packet multiple times, in experiment (b) we analysed the effect of the sender/receiver buffer size on packet loss. This experiment also measured the maximum bandwidth of the data diode link.

For asssessing (b), iPerf was used to limit the TX sender bandwidth at values ranging from 10 Mbps to the maximum theoretical value of the link (1 Gbps) while changing the sender buffer size (100-6000 KB). The virtual data diode was disabled during this test, since Iperf requires an initial TCP connection. Measurements show that the buffer size plays a significant role on the packet

loss, since it affects the total number of packets that can be sent in a single transfer (cf. Figure 6).

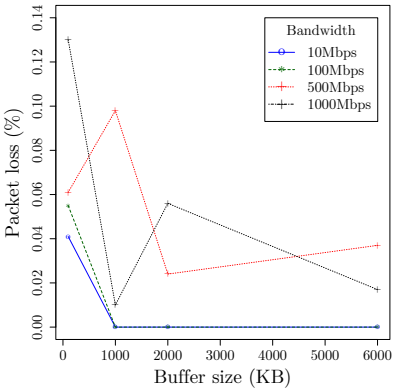

Fig. 6: Percentage of lost packets vs. bandwidth and write buffer size.

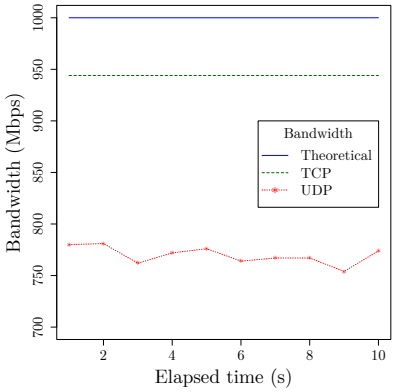

Fig. 7: UDP bandwidth vs. TCP and Theoretical bandwidth.

If the bandwidth is known beforehand, both agents can be optimized for minimal packet loss. This is important in IACS scenarios, since SCADA traffic patterns tend to be predictable, with stable network topologies [1]. Regarding the stress test on the data diode link, we started by performing a TCP test. The bandwidth achieved by TCP is expected to be higher than the actual bandwidth of the UDP transfer since it optimizes the transfer window size during the transfer. We took the measured value (944 Mbps) as the reference for the actual bandwidth. The maximum bandwidth using UDP was 769.7 ± 7.4 Mbps (cf. Figure 7), a value in line with some commercial switches, despite our software-based testbed.

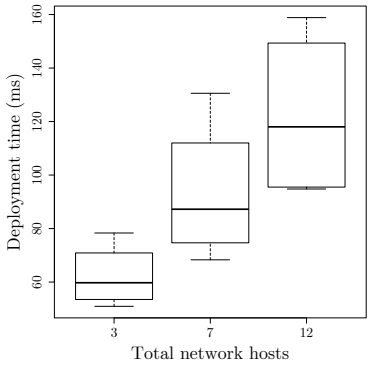

Fig. 8: Virtual data diode deployment times *vs* number of network hosts.

Table 6: Virtual data diode deployment times depending *vs* number of network hosts

Network hosts	**Deployment time (ms)**
3	62.188 ± 14.810
7	93.330 ± 33.345
12	122.418 ± 39.790

For the (c) experiment, we measured how the deployment times of the virtual data diode varied, accordingly to the number of network hosts. Hosts were

"faked" by changing the MAC address and the IP address of one of the machines, followed by the generation of ARP packets. Upon detection in the network controller, those fake hosts were added to the previously created network. A controller command-line command was introduced in the *Data diode* application to deploy and remove the virtual-data diode in a loop, while collecting the elapsed time. Table 6 and Figure 8 present the results. Although the deployment times increase with the number of network hosts, it is in the millisecond range. A small value considering that for n hosts, n-*1* flow rules have to be installed and the datastore has to be consistently synchronized between all the controller nodes.

6 Conclusion

Current trends, such as Industry 4.0 and Internet of Things are evolving industrial control networks towards ubiquity, moving away from the traditional monolithic and self-contained infrastructure paradigm, in favor of highly distributed and interconnected architectures. In this perspective, the use of data diodes provides a convenient way to isolate mission-critical network domains, while still allowing for relevant information (i.e., telemetry) to be accessed from the outside. However, as the number of interconnected devices increases, the costs of multiple physical data diodes may become impractical for organizations.

To deal with the inherent limitations of traditional implementations, we proposed the virtual data diode concept, which leverages the benefits of SDN and NFV. This concept was demonstrated and evaluated by means of a proof-of-concept prototype, designed with performance and availability in mind. The use of proactive flow rule instantiation removes the complete dependency on the control plane, allowing the virtual data diode to use the available switch bandwidth. The use of a distributed controller provides reliability and continuous operation in case of controller node failures. Prototype evaluation measurements recorded virtual data diode deployment latencies in the millisecond range, with minimal latency in the link layer. Even stressing the switch to its full rate capacity (with much higher values than the ones typically found in IACS), packet loss in the link was minimal. While not providing the same security levels of physical data diodes (it is a software implementation), the virtualized version still compares favorably with diode alternatives, such as firewalls, while maintaining functional equivalence to its physical counterpart.

Acknowledgements

This work was partially funded by the ATENA H2020 Project (H2020-DS-2015-1 Project 700581) and Mobiwise P2020 SAICTPAC/0011/2015 Project.

References

1. Barbosa, R.: Anomaly detection in SCADA systems : a network based approach. Ph.D. thesis, University of Twente (2014), doi:10.3990/1.9789036536455

2. Berde, P., Gerola, M., et al.: ONOS: towards an open, distributed SDN OS. Proceedings of the third workshop on Hot topics in software defined networking - HotSDN '14 pp. 1–6 (2014), doi:10.1145/2620728.2620744
3. FoxIT: Fox DataDiode Data Sheet (2018), https://www.fox-it.com/datadiode/downloads/
4. FoxIT: Fox IT FAQ. Online (2018), https://www.fox-it.com/datadiode/faq/
5. Genua: Data Diode cyber-diode. Brochure (2018), https://www.genua.de/fileadmin/download/produkte/cyber-diode-flyer-en.pdf
6. Heo, Y., et al.: A design of unidirectional security gateway for enforcement reliability and security of transmission data in industrial control systems. In: Int. Conf. on Advanced Communication Technology (2016), doi: 10.1109/ICACT.2016.7423372
7. Jeon, B.S., Na, J.C.: A study of cyber security policy in industrial control system using data diodes. In: 18th International Conference on Advanced Communication Technology (ICACT). p. 1 (jan 2016), doi:10.1109/ICACT.2016.7423373
8. Jones, D.W.: RS-232 Data Diode - Tutorial and reference manual. Tech. rep., United States (2006)
9. Mckay, M.: Best practices in automation security (2012), doi:10.1109/CITCON.2012.6215678
10. Mraz, R.: Data Diode Cybersecurity Implementation Protects SCADA Network and Facilitates Transfer of Operations Information to Business Users. Presentation (2016)
11. Okhravi, H., Sheldon, F.T.: Data Diodes in Support of Trustworthy Cyber Infrastructure. In: Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research. pp. 23:1—-23:4. CSIIRW '10, ACM, New York, NY, USA (2010), doi:10.1145/1852666.1852692
12. Oktian, Y.E., et al.: Distributed SDN controller system: a survey on design choice. Comp. Networks **121**, 100–111 (2017), doi:10.1016/j.comnet.2017.04.038
13. Open NF: OpenFlow Switch Specification Version 1.5.1 (Protocol v. 0x06) (2015), https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf
14. Owl Cyberdefense: Learn about data diodes. Online (2018)
15. Peterson, D.G.: Air Gaps Dead, Network Isolation Making a Comeback. Online, http://www.digitalbond.com/blog/2011/07/19/air-gaps-dead-network-isolation-making-a-comeback/
16. Scott, A.: Tactical Data Diodes in Industrial Automation and Control Systems. Tech. rep., United States (2015)
17. Stouffer, K.A., et al.: NIST SP 800-82 rev2. Guide to Industrial Control Systems (ICS) Security: SCADA Systems, DCS, and Other Control System Configurations Such As Programmable Logic Controllers (PLC). Tech. rep., USA (2015)
18. Sun, Y., Liu, H., Kim, M.S.: Using TCAM efficiently for IP route lookup. In: 2011 IEEE Consumer Communications and Networking Conference, CCNC'2011. pp. 816–817 (2011), doi:10.1109/CCNC.2011.5766609
19. Waterfall Security: Unidirectional Security Gateways vs. Firewalls: Comparing Costs. Tech. rep., Israel (2012)
20. Waterfall Security: Unidirectional Security Gateways (2018), https://static.waterfall-security.com/Unidirectional-Security-Gateway-Brochure.pdf
21. Waterfall Security: Waterfall FLIP (2018), https://waterfall-security.com/wp-content/uploads/Waterfall-FLIP-Brochure.pdf
22. Waterfall Security: Waterfall WF-500 product datasheet. Product Datasheet (2018), https://waterfall-security.com/wp-content/uploads/WF-500-Data-Sheet.pdf