Pacific Northwest
NATIONAL LABORATORY

PNNL-32412

Software-defined Networking for Energy Delivery Systems (SDN4EDS): An Architectural Blueprint – Summary Final Report

Cybersecurity of Energy Delivery Systems (CEDS) Research and Development

December 2021

SR Mix
CE Eyre
CA Bonebrake
D Gammel
K Phan
R Shiplet
CJ Glatter
O Green
MD Hadley
LH Chang
KW Thornhill
RH Smith
J Patel
J Gin
N Dossaji
S Patel
SV Singh
SC Tollbom
CA Goranson
A Chavez
B Radosevich
JR Hamlet
WM Stout
C Powell

U.S. DEPARTMENT OF ENERGY

Prepared for the U.S. Department of Energy
under Contract DE-OE0000190

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

PNNL-32412

Software-defined Networking for Energy Delivery Systems (SDN4EDS): An Architectural Blueprint – Summary Final Report

Cybersecurity of Energy Delivery Systems (CEDS) Research and Development

December 2021

SR Mix[1]
CE Eyre[1]
CA Bonebrake[1]
D Gammel[2]
K Phan[3]
R Shiplet[3]
CJ Glatter[3]
O Green[4]
MD Hadley[1]
LH Chang[1]
KW Thornhill[1]
RH Smith[2]
J Patel[3]
J Gin[3]
N Dossaji[3]
S Patel[5]
SV Singh[1]
SC Tollbom[1]
CA Goranson[1]
A Chavez[3]
B Radosevich[3]
JR Hamlet[3]
WM Stout[3]
C Powell[5]

[1] Pacific Northwest National Laboratory
[2] Schweitzer Engineering Laboratories, Inc.
[3] Sandia National Laboratories
[4] Spectrum Solutions, Inc.
[5] National Renewable Energy Laboratory

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Revision History

Revision	Date	Deliverable (Reason for Change)	Release #
1	02/15/2018	Deliverable 1.1: Initial draft	PNNL-27303
2	05/15/2018	Deliverable 1.2: Added SDN4EDS Tabletop Red Team Assessment; added Tabletop Red Team Assessment Summary	PNNL-27520
3	09/15/2018	Deliverable 1.3: Added Acknowledgements page Updated Decision Process Updated operational use cases Updated Reference Architecture section	PNNL-27901
Final	11/31/2021	Consolidate all interim reports into final Blueprint Architecture Report Provide additional guidance based on lessons learned from the project	PNNL-32368
Summary	12/31/2021	Condensed Final Report into shorter summary report	PNNL-32412

Summary

The initial version of this report provides a reference for suppliers and energy companies of all sizes to deploy networks based on software-defined networking technology (SDN) to improve reliability, reduce cybersecurity attack surface, and facilitate mitigation of adversarial behavior. It is a living document and will progress over the life cycle of the Software-Defined Networking for Energy Delivery Systems (SDN4EDS) project.

Version 2 of this report provides information on the Red Team tabletop assessment performed against the initial reference architecture.

Version 3 of this report updates the reference architecture with lessons learned from the Red Team tabletop assessment, as well as provides additional details for the use cases. It also provides information on the decision process that could be used by an organization when considering deploying SDN in their environment.

The final version of this report consolidates all the interim reports generated by the project into a final report. It also draws from PNNL's experience in deploying SDN to make recommendations on how SDN could be deployed in a utility environment, and provides rationale for those decisions allowing individual utilities to make risk-based and knowledge-based decisions on how to best deploy SDN in their own environment

This summary report provides a higher-level overview of the project reports. Readers interested in additional detail, including results of the Red Team assessments and the final configuration, are encouraged to read the full final report.

Acknowledgements

The authors acknowledge the contributions and support provided by their project partners from the following organizations:

- AECOM
- California Independent System Operator
- Dispersive Technologies
- Juniper Networks, Inc.
- National Renewable Energy Laboratory
- Sandia National Laboratory
- Schweitzer Engineering Laboratories, Inc.
- Southern California Edison

Acronyms and Abbreviations

API	application programming interface
ARP	Address Resolution Protocol
BA	Binary Armor
BCA	BES Cyber Asset (a NERC term)
BCS	BES Cyber System (a NERC term)
BES	Bulk electric System
BUM	broadcast, unicast, multicast
CEDS	Cybersecurity for Energy Delivery Systems
CIP	Critical Infrastructure Protection
CIDR	classless interdomain routing
CPU	central processing unit
DDoS	distributed denial of service
DHCP	dynamic host configuration protocol
DNP3	distributed network protocol version 3 (IEEE Standard 1815)
DOE	U.S. Department of Energy
DoS	denial of service
EA	engineering access
EACMS	Electronic Access Control or Monitoring System (a NERC term)
EDS	energy delivery system
ELK	Elasticsearch, Logstash, and Kibana
ESP	Electronic Security Perimeter (a NERC term)
FAT	factory acceptance test
FTP	file transfer protocol
GOOSE	IEC 61850 generic object-oriented substation events
GPS	global positioning system
HMI	human machine interface
HTTP	hypertext transfer protocol
HTTPS	hypertext transfer protocol secure
ICMP	Internet Control Message Protocol
ICS	industrial control system
IDS	intrusion detection system
IEC	International Electrotechnical Commission
IED	intelligent electronic device
IEEE	Institute of Electrical and Electronics Engineers
IF	interface
IP	internet protocol

IPP	independent power producer
IPS	intrusion prevention system
IPv4	internet protocol version 4
IPv6	internet protocol version 6
IISSU	in service software update
IT	information technology
IT-SDN	information technology software-defined network
LAN	local area network
LDAP	Lightweight Directory Access Protocol
MAC	media access control (address) – also known as a hardware address
MIB	Management Information Base
MITM	man-in-the-middle
MMS	manufacturing message specification
MPLS	multiprotocol label switching
NAC	network access control
NBI	northbound interface
NERC	North-American Electric Reliability Corporation
NFV	Network Function Virtualization
NIC	network interface card
NIST	National Institute of Science and Technology
NOS	network operating system
NSM	Network Security Monitoring
NTP	network time protocol
ONF	Open Networking Foundation
ONOS	Open Network Operating System
OP	operation (as in OP code)
OS	operating system
OSI	Open Systems Interconnection
OT	operational technology
OT-SDN	operational technology software-defined network
OvS	Open vSwitch
PCA	Protected Cyber Asset (a NERC term)
PLC	programmable logic controller
PNNL	Pacific Northwest National Laboratory
PRP	parallel redundancy protocol
PTP	precision time protocol, as described in IEEE Standard 1588
QoS	quality of service
REST	REpresentational State Transfer

RFC	Request for Comment
RIG	remote intelligent gateway
RS-232	Recommended Standard 232 (now Telecommunications Industry Association [TIA]-232) – a standard for OSI layer 2 serial communications
RSTP	Rapid Spanning Tree Protocol
RTU	remote terminal unit
SA	situational awareness
SBC	single-board computer
SBI	southbound interface
SCADA	supervisory control and data acquisition
SDN	software-defined network(ing)
SDN4EDS	software-defined networking for energy delivery systems
SDN-SAT	SDN Situational Awareness Tool
SD-WAN	Software-Defined Wide-Area Networking
SEL	Schweitzer Engineering Laboratories, Inc.
SIEM	security information and event management
SNC	Sierra Nevada Corporation
SNL	Sandia National Laboratories
SNMP	simple network management protocol
SPAN	Switched Port Analyzer
SV	IEC 61850 sampled values (sometimes referred to as measured sampled values or sampled measured values)
TCP	Transmission Control Protocol
TFTP	trivial file transfer protocol
TIA	Telecommunications Industry Association
TLS	transport layer security
TTL	time to live
UDP	user datagram protocol
VLAN	virtual local area network
VM	virtual machine
WAN	wide-area network
YANG	Yet Another Next Generation

Contents

Figures

Tables

1.0 Introduction

The software-defined networking (SDN) and Watchdog projects sponsored by the U. S. Department of Energy (DOE) Cybersecurity for Energy Delivery Systems (CEDS) program resulted in purpose-built SDN technology being commercialized by Schweitzer Engineering Laboratories (SEL) in 2016. The SDN flow controller and switch developed under these previous efforts are beginning to be deployed in energy delivery system (EDS) communication infrastructures. The SDN technology itself has provided a reduced cyberattack surface, increased network availability, and laid the groundwork for the work performed by the Pacific Northwest National Laboratory (PNNL) and its partners in the Software-defined Networking for Energy Delivery Systems (SDN4EDS) project. This blueprint document, a primary deliverable of the SDN4EDS project, is designed to identify how SDN can be used and expanded to: 1) natively provide new security benefits, 2) provide the foundation on which additional security applications can be developed, 3) introduce the technology for those who are new to SDN, and 4) provide a number of use cases to show how SDN can support current and future operational concepts. Its recommendations on how SDN can be deployed for EDS initially target the electric utility subsector but will be expandable to other energy subsectors including oil and gas.

This blueprint initially comprises background materials and industry use cases including electric transmission inter-station and intra-station protection and control. It can be expanded to include other data sharing or control environments. It captures the discussions and test results from each phase of the SDN4EDS project and has been updated and augmented as additional use cases are identified and as needs evolve.

One benefit of this use case-based requirements approach is to clearly identify and document areas where SDN technology may not be suitable (e.g., SDN-enabled moving target defense mechanisms for remedial action scheme communication), so that the project can focus work on use cases that clearly benefit from SDN technology. It also identifies areas for future research and analysis.

Most SDN implementations in the electric utility sector to date have focused on LAN technologies, primarily Ethernet. This blueprint addresses expanding upon the LAN uses of SDN, as well as examining the impacts of SDN technology on wide-area networking (WAN) communications.

The SDN blueprint document serves as a reference for energy companies of all sizes to deploy networks based on SDN technology to improve reliability, reduce cybersecurity attack surface and to ease mitigation of adversarial behavior. The blueprint also contains guidance for end users on how to evaluate SDN technologies to ensure that business and cybersecurity needs are identified and met.

The SDN blueprint is intended to be a living document that will continue to be updated as new technologies and uses for SDN are developed and deployed. Although the primary set of use cases for the blueprint are based on SDN deployments for control of the transmission portion of the electricity subsector, as use cases are developed for other aspects of the electricity subsector and additional energy sector components like oil refining and pipelines, the blueprint can be updated or modified to address different uses. Additionally, the blueprint can be updated to include use of SDN technology to control distributed energy resources and microgrids, specifically as they interface with other non-energy-related networks and functions.

During the project, the SDN blueprint was updated capturing each project milestone to reflect the lessons learned from that phase of the project, as well as to provide any necessary course corrections resulting from unexpected testing results or to capture new requirements. In particular, the blueprint was updated to reflect the results of the Red Team exercises; identification of analytic requirements and metrics availability; analysis of the use of SDN networks in hybrid environments (i.e., networks with both SDN and non-SDN components); analysis and testing of trust models, interoperability, protocol enforcement, and performance testing; results of cybersecurity requirements testing; and holistic testing of the entire reference architecture (containing all of the components developed during the life of the project).

2.0 Software-Defined Networking Basics

2.1 SDN Background

SDN is a proven approach to the management, configuration, and operation of network systems. This architectural change is revolutionizing the management of large-scale enterprise networks, cloud infrastructures, and data-center networks to better support the dynamic changes required many times a day. The reasons for the wide adoption of SDNs in the corporate world also are why we believe it can have a significant impact in the management of control system networks. SDN allows a programmatic change control platform, which allows an entire network to be managed as a single asset; simplifies the understanding of the network; and enables continuous monitoring in more detail than traditional networks. Control system networks are often more static, while the corporate world is more dynamic. That is, control system flows are more consistent and continuous than the ever-changing nature of a corporate network flow snapshot. The primary reason for this is that the control system is made up of machine-to-machine communications, while corporate communications are mostly people to machine. This means that the SDN architecture is applied differently for control system or operational technology (OT) networks. However, the good news is that SDN architecture is able to optimize for both, and this flexibility is one reason SDN technology is beginning to be deployed in OT networks. The fundamental shift in networking brought by SDN is the decoupling of the systems that decide where traffic is sent (i.e., the control plane) from the systems that forward the traffic in the network (i.e., the data plane).

The traditional network deployment process begins with designing the topology, configuring the various network devices, and finally, setting up the required network services. To achieve the optimal usage of network resources, application data must flow in the direction of the routes determined by the routing and switching protocols. In large networks, trying to match the network-discovered path with an application-desired data path may involve changing configurations in hundreds of devices with a variety of features and configuration parameters. In addition, network administrators often need to reconfigure the network to avoid loops, gain route convergence speed, and prioritize a certain class of applications.

This complexity in management arises from the fact that each network device (e.g., a switch or router) has control logic and data-forwarding logic integrated together. For example, in a network router, routing protocols such as Routing Information Protocol or Open Shortest Path First Protocol constitute the control logic that determines how a packet should be forwarded. The paths determined by the routing protocol are encoded in routing tables that then are used to forward packets. Similarly, in a Layer 2 device such as a network bridge (or network switch), configuration parameters or spanning tree algorithm constitute the control logic that determines the path of the frames [1]. Thus, the control plane in a traditional network is distributed in the switching fabric (network devices), and as a consequence, changing the forwarding behavior of a network involves changing configurations of many (potentially all) network devices.

SDN is a new architecture in networking that simplifies network management by abstracting the control plane from the data forwarding plane. Figure 2-1 shows the building blocks of SDN, which are discussed in the following subsections.

[1] Packets are used to describe OSI Layer 3 traffic; frames are used to describe OSI Layer 2 traffic.

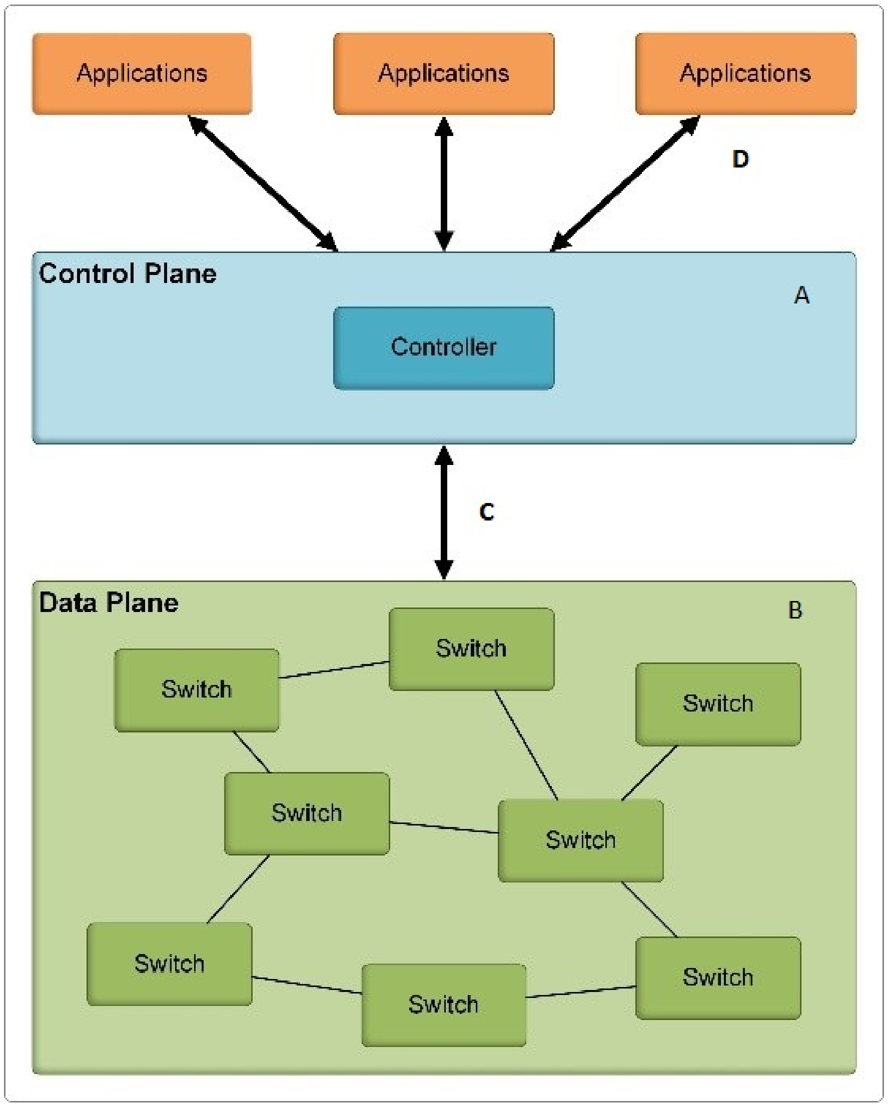


Figure 2-1. SDN Building Blocks[2]

A. Control Plane

At the heart of SDN is a controller that embodies the control plane. Specifically, controller software determines how packets (or frames) should flow (or be forwarded) in the network. The controller communicates this information to the network devices, which constitute the data plane, by setting their forwarding tables. This enables centralized configuration and management of a network. Many open-source controllers such as Floodlight,[3] NOX,[4] and Ryu,[5] to name a few, are now readily available.

[2] Based on [Bobba 2014].

[3] http://www.projectfloodlight.org/floodlight/

[4] http://www.noxrepo.org

[5] http://osrg.github.io/ryu/

B. Data Plane

The data plane consists of network devices that replace switches and routers. In SDN, these devices are very simple Ethernet frame-forwarding devices with a communications interface to the controller to receive forwarded information. Many vendors today provide frame-forwarding devices that are SDN-enabled.

C. Control and Data Plane Interface

SDN requires a communications interface between network devices and the controller. A standardized interface allows a controller to interoperate with different types of network devices and vice versa. The OpenFlow protocol, managed by the Open Networking Foundation (ONF), is one such standardized interface and has been adopted by major switch and router vendors. However, it should be noted that OpenFlow is just a building block in the SDN architecture, and there are other open Internet Engineering Task Force standards or vendor-specific standards that are either already available or are being developed.

D. SDN Services

In SDN architecture, the controller can expose an application programming interface (API) that services can use to configure the network. In this scenario, the controller may act just as an interface to the switching fabric while the control logic resides in the services using the controller. Depending on the SDN controller being used, the interfaces may be different. Controllers and their application interfaces can be tailored to meet the needs of an application domain. A controller that is designed and optimized for data centers, for example, may not be suitable for control networks in the electric sector and vice versa. The application domain specific to the industry it is used in will determine the overall system requirements. Trade-offs between optimizations like single instruction speed or parallel processing determine the best interfaces to use.

While SDN typically is used to monitor and programmatically change network configurations, the centralized nature of SDN also is well suited to meet the security, performance, and operational requirements of control system networks. Control system networks are designed to do specific jobs for many years with as little change as possible. With the help of SDN, operators can take advantage of this knowledge to preconfigure network paths and effectively create virtual circuits on a packet switching network. Power companies can design the virtual circuits they require for communication between certain devices and lock down the communications path. This type of approach can enhance security by reducing the attack surface and provide a clear approved baseline that can be continually monitored to make sure that it is never changed.

2.2 SDN Basics

Operating an enterprise class network comes with numerous challenges—shrinking budgets, supporting real-time applications, more and more traffic leaving the LAN for the cloud, cyber risks and demanding users that expect constant reliability and stability. Until now, a traditional campus network has maintained the control plane and the data plane in the same piece(s) of hardware. Resiliency is delivered by some combination of active-active or active-passive equipment in the core and distribution layers of the campus networks, which is expensive because it requires at least two pieces of matching hardware. It is not uncommon to see two data centers with each having one border router, firewall, reverse proxy, remote access gateway, IP address management scheme, and the access layer with dual supervisors.

Engineers employ control plane policing to protect against broadcast, unicast, multicast (BUM) traffic. Mitigating cyber vulnerabilities via software updates requires taking down the control plane and the data plane. In Service Software Update (ISSU) is still not hitless and, at least in the experience of this author, has the potential of rendering network equipment unbootable.

SDN breaks the mold by separating the control plane from the data plane, thereby making white box switches (i.e., generic switches, sometimes called "open switches," built with off-the-shelf parts with a generic operating system (OS) but without proprietary software, and that don't require proprietary management software) more attractive. The control plane can reside far away and perform all the thinking like functions—quality of service (QoS), control plane policing, flow control—and tell the forwarding plane what to do. The promise of SDN is so attractive that the likes of Google [Bailey 2016], Facebook[6], and Microsoft[7] are developing their own products because the market is unable to provide them a solution.

A dated *Network World* article from 2014[8] states that using SDN technology increases the security vectors that need to be hardened. This article may be accurate for SDN technology deployed without security objectives in mind. However, the Watchdog and SDN projects demonstrated that SDN technology deployed in a deny-by-default method, to require mutual trust between SDN components, and to provide secure user access to the SDN flow controller, actually reduces the attack surface. The SDN4EDS project followed proven, secure SDN deployment methodologies.

SDN supports the ability for optional third-party applications to interface with the SDN flow controller. Note that the SDN environment will function properly without these applications. The control plane needs to communicate northbound with business applications (via the northbound interface [NBI]) and southbound with the data plane (via the southbound interface [SBI]). These north- and southbound communication channels may provide attractive targets for man-in-the-middle (MITM) attacks to hijack and poison sessions. The controller is potentially susceptible to vulnerabilities in the host OS hardware and software as well as the controller code itself. Denial of service (DoS) may be a threat.

Through careful planning and strategic thinking as demonstrated by the SDN and Watchdog projects, the cyber threat can be mitigated, and all the promises of SDN (such as agility in patching and reliable and simplified operations) can be realized. Features in the controller need to be granular and well defined so that only the "minimum viable set" is provisioned, and unknown features do not lead to vulnerable systems like the one in Cisco[9] and a backdoor in Juniper.[10]

[6] https://code.facebook.com/posts/717010588413497/introducing-6-pack-the-first-open-hardware-modular-switch/ (Accessed September 17, 2021)
[7] http://www.businessinsider.com/microsoft-gives-away-sdn-software-sonic-through-open-compute-project-2016-3 (Accessed September 17, 2021)
[8] https://www.networkworld.com/article/2840273/sdn/sdn-security-attack-vectors-and-sdn-hardening.html (Accessed September 17, 2021)
[9] https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20170214-smi (Accessed September 17, 2021)
[10] https://www.cnn.com/2015/12/18/politics/juniper-networks-us-government-security-hack/index.html (Accessed September 17, 2021)

2.3 SDN Layers, Architecture Terminology, and Addressing

The layers and architecture for SDN is more formally defined in Request for Comment (RFC) 7426, *Software-Defined Networking (SDN): Layers and Architecture Terminology* [Haleplidis, 2015]. This RFC is an informational RFC that describes a formalized structured approach to designing SDN networks by defining three key concepts covering network communications planes, connecting interfaces, and abstractions for accessing SDN network functions. RFC 7426 exists to formalize, standardize, and clarify SDN terminology as its concepts were developed and evolved from academia and industry starting in 2008. Researchers and contributors used different terminology to describe the architecture of SDN, leading to confusion and lack of interoperability. RFC 7426 defines these three concepts as:

- Network Communication Planes
  - *Forwarding (Data) Plane* – Responsible for handling (forwarding) frames of data between network and application devices as determined sets of instructions (rules) from the Control Plane
  - *Operations Plane* – Responsible for managing and maintaining the operational state of devices, such as network device interface port state, link state, and status. Operations functions are intrinsic to network device hardware and software with configuration control under the Management Plane (e.g., port speed, duplex, and maximum transmission unit) and rapid decision making by the Control Plane (e.g., rerouting after a link failure event)
  - *Control Plane* – Responsible for making decisions and issuing instructions on how frames of data are forwarded between devices and pushing those instructions down to network devices for execution. The Control Plane mainly tunes and configures the forwarding table in network devices as device states, link states, and frame communications flow requirements change over time.
  - *Management Plane* – Responsible for configuring, monitoring, and maintaining network devices. The Management Plane is used to configure the Forwarding Plane but does not control it as the Management Plane is too coarse and too slow.
  - *Application Plane* – Applications and services running in the network that react or interact with the network infrastructure. SDN applications define, characterize, or support network behavior and mission. The Application Plane does not necessarily include end-user applications as they run in the Forwarding Plane. Applications that are used to control or manage network devices are distinct from the Application Plane, residing in the Control and Management Planes.
- *Interfaces* – Network planes interconnect via physical or virtual interfaces. For example, the SBI of the SDN flow controller defines the network connection between the SDN flow controller and the distribution of SDN network devices. The service interface or NBI of the SDN flow controller defines the network connection between the Application Plane and Control and Management Planes of the SDN flow controller. Similarly, the NBI of the SDN network devices reports connectivity status and statistics to the SDN flow controller.
- *Abstraction Layers* – Abstraction layers define how SDN network device services and functions are accessed. Abstraction layers include the Device and resource Abstraction Layer (DAL) which abstracts functions for inspection and control of the Forwarding and Operations planes via the SBI and the Network Services Abstraction Layer (NSAL) which abstracts functions for access to the control and management planes via the service or NBI. Abstraction layers will not be discussed further as they are more appropriate to the design and features of

application programmer interfaces (APIs) for SDN network devices themselves over the design, architecture, and security features of a SDN network.

2.3.1 CAP Theorem

RFC 7426 also makes use of the CAP Theorem, also called Brewer's Theorem, which was first proposed, presented, and conjectured by theoretical computer scientist Eric Brewer of University of California, Berkeley, from 1998 to 2000 [Brewer 2000]. Seth Gilbert and Nancy Lynch of MIT published a formal proof in 2002 [Gilbert 2002], rendering Brewer's conjecture a theorem. The theorem states that any network distributed system can simultaneously provide no more than two of three guarantees: 1) Consistency, 2) Availability, and 3) Partition Tolerance.

- *Consistency* – Every receive is the most recent write or an error (flow consistency)
- *Availability* – Every request receives a response, which may or may not be the most recent write (synchronous or asynchronous)
- *Partition Tolerance* – The system continues to operate despite loss or delay by the network, between its nodes (schism)

When a network partition event occurs, affected distributed nodes have one of two choices: 1) they can cancel outstanding read and write operations, thus maintaining consistency between nodes, sacrificing availability, or 2) they can continue their operations to maintain availability but risk inconsistency between nodes, leaving nodes operating on incomplete or missing information.

SDN seeks to satisfy all three guarantees, and largely achieves them except in the case where the SDN network devices become partitioned from the SDN flow controller in the Control Plane or their peers. Lacking an SDN flow controller, existing network flows may continue to operate (if the forwarding plane has not partitioned also), maintaining consistency and availability for them, but new flows will fail to form, thus sacrificing availability. Recovering from loss of an SDN flow controller, the SDN network may employ a backup SDN flow controller to take over control functions from the primary SDN flow controller. However, the backup SDN flow controller may not quite be up to date with the operating state of the network or new flows may be temporarily blocked while the backup SDN flow controller comes fully on-line, temporarily impacting availability.

Some SDN solutions, such as SEL OT-SDN, also implement flow control and operations intelligence directly within their switches, allowing for the network to run at least semi-autonomously when connectivity to the SDN flow controller otherwise lost.

2.3.2 SDN Communications Planes

Table 2-1 shows the various communication planes that are implemented in an SDN environment. These are shown graphically in Figure 2-2.

Table 2-1. SDN Communication Planes

			Attributes			
Plane	**Primary Interface**	**Flow Path**	**Timescale**	**Persistency**	**Locality**	**CAP Guarantees**
Control Plane	Southbound (control)	East-Westbound (inter-switch)	Very Small (microseconds)	Highly Dynamic	Local Area within the SDN environment	Consistency, Availability, Intolerant to partitioning
Management Plane	Southbound (configuration)	Northbound (services, alerts, logging, external)	Large (seconds)	Static or slow to change	Wide Area, Wide Distribution, Centralized Management from a Remote Location	Consistency, Availability, can be tolerant of partitioning
Forwarding (Data) Plane	East-Westbound (inter-switch)	East-Westbound (inters-witch)	Very Small (microseconds)	Highly Dynamic	Local Area within the SDN environment	Consistency, Availability
Operations Plane	East-Westbound (inter-switch)	East-Westbound (inter-switch)	Very Small (microseconds)	Highly Dynamic	Local Area within the SDN environment	Consistency, Availability
Application Plane	Data Port	East-Westbound (inter-switch) to/from network services	Variable	Dynamic	Local Area within the SDN environment	Consistency, Availability

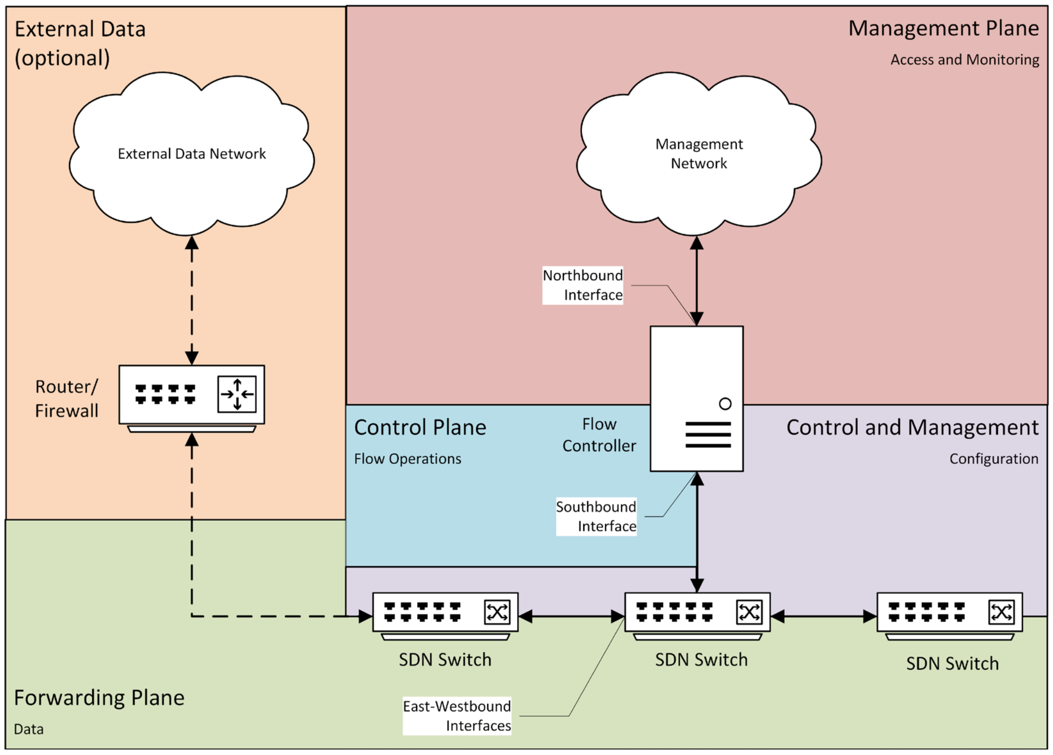


Figure 2-2. SDN Communication Planes

2.3.3 SDN Security and Performance Considerations

SDN implementations must consider a number of factors during their design and operations. These include:

- *Timescale Latency* – The network should be designed to meet or exceed the requirements of its mission. For example, in a power control system network, a protection relay will need to respond to an electric fault within microseconds to prevent potential loss of life and property. A well-architected, local SDN network can meet such a requirement. However, if the needed SDN flow controller involvement is not locally present or is situated too far away, say in an operations center 100 miles away from the station or if the network is congested by traffic causing head-of-line issues on interfaces, the protection relay may be unable to react to the event in time. The loss of locality shifts response time from microseconds to milliseconds or worse.
- *SDN Flow Controller Availability* – The SDN flow controller needs at least the SBI in order to function. Singly connected in that manner, the same physical interface must handle all services (i.e., control, management, and frame inspection processing) required for operation of the SDN network. Security and performance gains can be made by adding a second or third interface, thereby separating services so they cannot interfere with each other at the network level. In this way, SDN network inspection traffic cannot interfere with control and management frames and vice versa. In a network storm situation, the SDN network would effectively cease to function as the SDN flow controller becomes overwhelmed. SBI and NBI functions also are quite different in terms of timescale. For the SDN flow controller, control and inspection require very small timescales, depending on the mission they may require the SDN flow controller to be able to receive, inspect, analyze, compute, and issue new control instructions to SDN network devices in microseconds. Management is on a completely different timescale, usually measured in seconds. Separation of management and control interfaces prevents their network traffic from interfering with each other, increasing the flow controller’s ability to perform its most critical network functions (control and inspection) without possible unnecessary interference coming from the direction of the management plane, such as network security scans, run amok Simple Network Management Protocol (SNMP) queries, network probes, unsolicited service advertisements, network penetration tests, unwanted traffic, etc.

  Without access to the SDN flow controller, the network becomes static. Existing flows can continue to run, but neighbor discovery and new flow formation cease, with loss of recoverability, such as during link state failure. On the other hand, when the implementation includes autonomous flow control and operation plane intelligence in the switches themselves such as in the SEL OT-SDN implementation, then they only need the SDN flow controller for provisioning and configuration management.
- *SDN Flow Controller Capacity* – Like any other end-node, an SDN flow controller has limited network resources bounded by hardware and software. For example, an SDN flow controller with its SBI connected to an SDN fabric using Gigabit Ethernet is physically limited to receiving approximately 1.4 million frames or 1 gigabit per second, whichever comes first. SDN network devices may be instructed to forward unknown Ethernet traffic coming from end-nodes to the SDN flow controller SBI for inspection by Control Plane functions inside the SDN flow controller. If the unknown traffic frames should be forwarded, the SDN flow controller sends new instructions or spoof response messages as appropriate on behalf of a destination. The SDN flow controller also is limited by its internal hardware architecture such as memory bandwidth, central processing unit (CPU) speed, thermal tolerance, and interrupt

handling rates. Once the SDN flow controller's computational resources exhaust, new flows fail to form, again sacrificing availability.

In well-behaved static environments such as those found in OT implementations, the switches can be instructed to drop (and optionally count) unknown traffic, thus minimizing the need for SDN flow controller communication or considerations of SDN flow controller capacity.

Consideration should also be given as to how the SDN flow controller is provisioned in the network. An SDN flow controller can be virtualized, that is exist as a virtual machine (VM) running within a physical computer host, governed by a hypervisor. Incoming frames to the SBI packet inspection interface must be processed by physical hardware and then passed from driver into physical memory of the hypervisor. From there, incoming frames are copied from the hypervisor into memory reserved for the VM's Ethernet driver and copied yet again as frames pass from the virtual device driver and OS into flow controller program memory space. As these incoming frames are copied, more memory bandwidth and CPU processor time are consumed, eventually reaching the limit where the virtual flow controller cannot meet demand within the SDN network. A virtualized SDN flow controller could also be impacted when it must share the same physical machine with other VMs running on the same hardware.

- *Predictability* – That the network device always reliably and correctly forwards frames from source to destination through the distributed SDN network in a consistent and timely manner.
- *Head of Line Queue* – Queue processing within the SDN switches can contributes to jitter, dispersion, latency (due to signal distance and switching gate delay), and lag (queue wait or frame processing delay). As packets arrive on an interface, they are enqueued for processing, either first-in first-out, or using a weighted priority scheme such as QoS or Random Early Detection where higher priority packets moved to the head of the queue before lower priority packets. Large Ethernet frames also take longer to move from media to interface to memory than small ones. Large volumes of small frames can cause the Ethernet driver to interrupt the CPU continuously to the point where there are no more compute cycles available.
- *Network Addressing* – The fine-grained structure and control SDN can bring to networking can render some basic network engineering concepts and practices unnecessary. For example, in IP the network mask, parameter (netmask) is used to define what bits of a network address are the network ID and which bits are the host ID. The IP network module in end-nodes and routers uses netmasks to make basic decisions regarding the locality of source and destination addresses that intercommunicate with one another. That is, whether a neighboring host is adjacent locally (they share the same network ID) and can therefore communicate directly over their common network media (e.g., Ethernet over SDN) or in a different network, in which case packets bound for a remote destination must be forwarded through an adjacent router (i.e., a gateway). An SDN network, however, can directly define and control how end-nodes intercommunicate across the SDN switch fabric regardless of IP address mask. This can make the IP netmask irrelevant. However, once an end-node exists in the network with more than one IP interface, correct assignment of netmasks become vital once again. Incorrect netmasks can create situations where end-nodes and routers believe networks overlap which can lead to communications failures and incitement of packet storms. Intense packet storms have the potential to consume network device resources or overwhelm the SDN flow controller, and risk failing the network's mission. It also can lead to confusing results when using traditional network diagnostic techniques or equipment.

2.3.4 SDN Addressing Scheme Recommendations

SDN implementations can be very flexible and configurable, to the point that “normal” network configuration rules do not necessarily need to be followed. For example, two different nodes with the same IP address can coexist in the same SDN infrastructure as long as the SDN flow rules maintain separation between data flows from or to them. However, this has the possibility of creating confusion when attempting to address network connectivity issues or assessing network traffic using traditional network monitoring tools such as Wireshark[11]. When configuring networks in an SDN environment, the following considerations are recommended:

- Define a unique network number address and mask for the control, management and forwarding planes of the SDN network. When assigning network numbers or subdividing networks into smaller subnetworks, use classless interdomain routing (CIDR) scheme described in RFC1519 [Fuller 1993], obsoleted by RFC 4632 [Fuller 2006], particularly for IPv4 (internet protocol version 4). CIDR can also be applied to IPv6 (internet protocol version 6),as described in RFC 4291 [Hinden 2006]. When using IPv6 over Ethernet with SDN the use of the simplified /64 CIDR is recommended.
- Assign host interfaces with unique host addresses in their assigned network number with no overlap. Avoid using dynamic host configuration protocol (DHCP) to assign network addresses, especially if the dynamic nature of DHCP could lead to inconsistency with address-based SDN flow rules.
- Avoid “cutting corners” with addressing as SDN flow rules would otherwise allow. Hosts connecting to SDN networks will still follow IP routing standards regarding routing, network IDs, network masks and locality, particularly if they have more than one interface. Disagreements between them and SDN flow rules could coopt the network.
- Consider using non-routed private address spaces as described in RFC1918 [Rekhter 1998] updated by RFC 6761[12] [Cheshire 2013] for the control and management planes of the SDN that are never routed outside the SDN environment or elsewhere into the organization’s internal networks, or externally with the internet, Control and management plane address assignments should never overlap with the forward/data plane.

2.4 Comparing Old Versus New Switch Technology

To define the environment for comparing old (traditional or managed switch) versus new (SDN switch) technology, the high-speed substation network designed by a domestic electric utility shown in Figure 2-3 (based on [Johnson 2008]) was selected. The design provides for the full separation of operational (e.g., supervisory control and data acquisition [SCADA] or synchrophasor) and non-operational (e.g., voice over IP, data from relays or programmable logic controllers (PLCs) not used for operations) data. Expanding on their design and introducing an SDN switch as the substation LAN cloud enhances their vision by providing true priority and QoS in both normal and degraded operational environments.

[11] Wireshark is a freely downloadable network analysis tool. See https://www.wireshark.org/ for additional information. (Accessed September 17, 2021)
[12] RFC 6761 updates and clarifies the definition and use cases of special domain names for mapping RFC 1918 address spaces in DNS to fully qualified domain names.

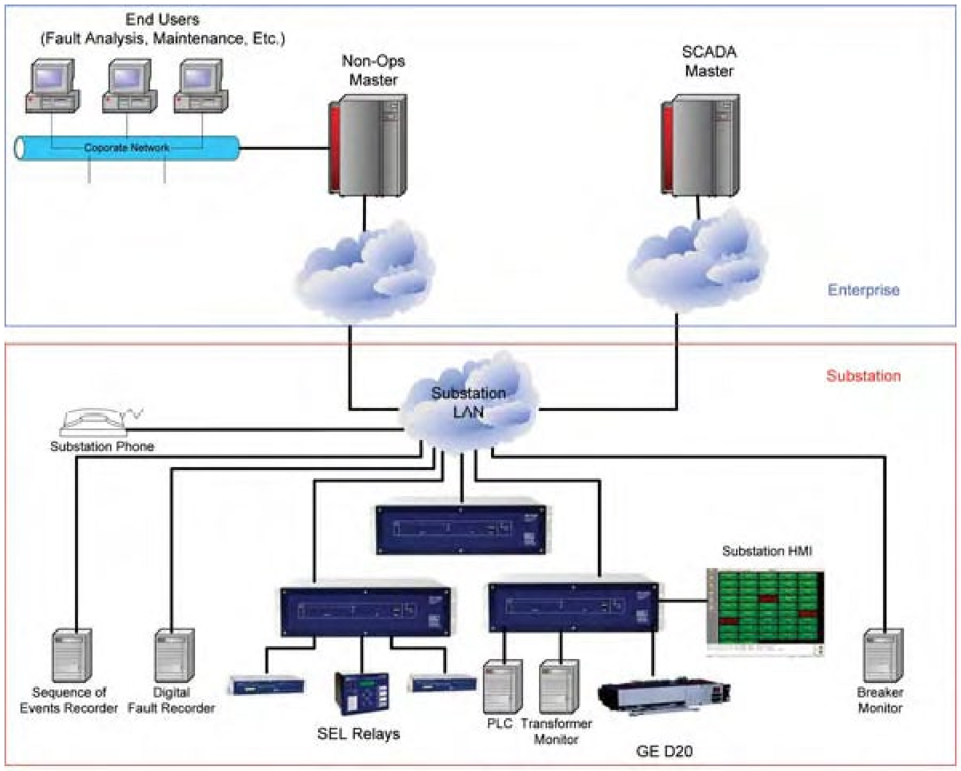


Figure 2-3. Utility Network Architecture

This design with SDN provides other benefits not envisioned by the utility as well. For example, it can support the addition of new applications (e.g., demand response, meter reading, remote switching, web-based engineering access (EA), etc.) and redundant communication media in an active-active mode without modification of the infrastructure.

Using this design as the basis for the comparison, the following categories were defined and reviewed with PNNL's network engineers and the SEL SDN project team. Table 2-2 identifies similarities and differences between the switch technologies for each category. The categories are intended to cover the full life cycle of the switch (commissioning, operations, and maintenance), security features, situational awareness (SA) capabilities, and general North American Electric Reliability Corporation (NERC) Critical Infrastructure Protection (CIP) requirements.

Table 2-2. Traditional and SDN Switch Comparison

Category	Traditional Managed Switch	SDN Switch
Commissioning a switch	The objective while commissioning a switch for both managed and SDN environments is to establish trust with the switch. The process is quite similar – ensuring firmware is updated and configurations are installed. The mechanics of the process will differ by vendors for both environments. SDN allows for mutual authentication of the SDN flow controller and SDN switch, making it difficult for rogue switches or SDN flow controllers to be used on the network.	
User authentication to switch or control software	Authentication is at the traditional switch and may utilize a variety of methods and protocols.	Authentication is performed at the SDN flow controller and is one way complexity is reduced in the SDN switch. The SDN flow controller and SDN switch may implicitly or through cryptographic means authenticate each other.

Category	Traditional Managed Switch	SDN Switch
Firmware updates	Copied to the traditional switch using Trivial File Transfer Protocol (TFTP) or some form of file transfer into the traditional switch on a manual or third-party schedule.	The SDN flow controller manages firmware upgrades, schedules, and can scale from a single SDN switch to thousands.
Firmware monitored for changes	Manual process is used to compare active and backup firmware versions	The SDN flow controller monitors the firmware version of the SDN switch for changes on a defined schedule. Any detected anomalies result in the firmware version being restored and the generation of alerts.
Configuration backup	Backing up the traditional switch configuration is performed through vendor-specific or third-party applications.	The SDN flow controller maintains the master copy of the configuration for each SDN switch. In addition, these can be duplicated onto a redundant SDN flow controller.
Configuration monitored for change	Manual process to compare active and backup configurations for changes	The SDN flow controller monitors the configuration of the SDN switch for changes on a defined schedule. Any detected anomalies result in the configuration being restored and the generation of alerts.
Support for multi-vendor switch environment	Organizations typically deploy single vendor proprietary switch solutions for ease of maintenance and support.	Most functions supported by open-source versions of the SDN flow controller and SDN switches to ensure interoperability.
Inherent switch security capabilities	Access control lists, whitelisting, and blacklisting capabilities.	SDN provides multiple layers of security including drop by default and whitelisting to ensure only permitted communications occur. In addition, SDN provides for protocol behavior to be enforced for both Information Technology (IT) and SCADA protocols by leveraging SDN attributes and intrusion prevention systems (IPS) (e.g., Snort in active mode).
Redundant deployment	Support for active – passive redundant communication links.	Support for active – active redundant communication links.
Preventing loops	Requires the use of Rapid Spanning Tree Protocol (RSTP) to prevent network loops.	Networks loops prevented by default with SDN switch technology.
Recovery from communication failure	RSTP used to recover from communication link failure. The length of time to failover varies and results in frame loss until passive link becomes active.	SDN actively monitors link status and can fail over up to 20 times faster than RSTP with only a single frame lost (generally the frame being transmitted at the time of failure).
Quality and priority of service	Provides QoS and PoS for a limited set of data types (e.g., voice over IP, video, audio)	QoS and PoS can be implemented for any port, protocol, IP address, data type, etc. Also supports QoS and PoS for fully functional and degraded communication scenarios.
Change control	Adding a new traditional switch, upgrading firmware, and making configuration changes typically require a network outage.	Adding a new SDN switch, upgrading firmware, and making configuration changes may be done without a network outage. Firmware updates without an outage require that a redundant SDN switch configuration or multiple path support is deployed.
Performing maintenance	Distributed through the management interface on the managed switch. May be available centrally if a switch management network is available. Switch is taken out of service for maintenance activities.	Centralized through the SDN flow controller application for all SDN switches. Includes configuration changes and firmware updates. The SDN switch does not need to be taken out of service and network outages can be avoided.

Category	Traditional Managed Switch	SDN Switch
Securing the control plane	With managed switches, securing the control plane is done to prevent the route processor from unwanted traffic that could cause a DoS. Securing the control plane is done through a combination if access control lists and Bridge Data Protocol Units (BPDU) frames. BPDU frames are special multicast frames that are communicated in an unsecured manner. These approaches only partially secure the control plane.	The control plane has been removed from the SDN switch and is now centralized on the SDN flow controller. Separating the control plane from the data plane makes network management more flexible. One could argue that SDN allows the control plane to be secured for the first time.
Define unauthorized traffic	Managed switches use access control lists where both the denied and allowed protocols must be defined. The rules are applied to IP address.	The SDN flow controller is used to define which traffic flows are permitted on the network. All other flows are denied by default. The SDN rules can be applied to IP addresses, individual switch ports, or groups of ports. SDN also enables unexpected traffic to be sent to the SDN flow controller for further analysis. This function may be performed by the SDN flow controller itself or delegated to an IPS (e.g., snort in active mode) if full packet inspection is required.
SCADA or industrial control system (ICS) traffic awareness	Not supported natively by the switch but may be supported by third-party application.	The ability to understand any protocol (SCADA or otherwise) is integrated into the SDN environment.
Decision-making time	Managed switches respond more slowly, especially in situations where network events occur more quickly than the personnel managing the switches can respond. Human analysis of network data may be required to respond to the event.	SDN switches make immediate decisions based upon observed traffic. The traffic engineering activities required to deploy SDN switches enable rapid response to cyber and physical events on the network.
Situational awareness (SA)	Typically requires the use of third-party SNMP applications to measure changes in link performance or behavior, test for acceptable performance during faults, and provide status on link availability.	The ability to measure changes in link performance or behavior, test for acceptable performance during faults, and provide status on link availability is integrated into SDN environments.

2.5 Characteristics of an Operational Technology Software-defined Network

OT network environments are different than traditional IT networks, therefore the OT implementation of an SDN environment also must be different. Although SDN was initially developed and implemented in an information technology software-defined networking (IT-SDN), its features are well suited for use in OT environments, with several different assumptions and characteristics. Characteristics of an operational technology software-defined network (OT-SDN) compared with those of an IT-SDN are shown in Table 2-3:

Table 2-3. IT and OT Network Comparison

IT Networks	OT Networks
IT networks tend to be dynamic in nature. Individual nodes like laptops can migrate from one location to another and can be disconnected from the network during the move or when taken home at night.	OT networks tend to be static in nature. Once equipment is placed and configured, it rarely moves from one location or another or is temporarily removed from service.

IT Networks	OT Networks
The dynamic nature of IT networks requires a dynamic method of provisioning IP addresses, such as the Dynamic Host Control Protocol, that allows nodes to be re-assigned new addresses as they move about the infrastructure.	OT networks are static, and addresses can be (and in some cases must be) hard coded in the device configurations.
IT networks must be flexible in the workloads they support.	OT networks have very static and predictable traffic patterns, connection pairs, and protocols.
IT networks must allow dynamic protocol use.	OT networks are similarly predictable in the protocols they need to support.
The flexibility of IT network loads requires the ability to dynamically perform name to address lookups.	OT network flows are static and can be managed without requiring host name to address lookups.

Because of these differences, different assumptions about network behavior can be configured in an OT-SDN environment:

- Physical ports in an OT-SDN environment should be configured to specifically allow only known media access control (MAC) addresses to successfully connect to the physical port. By configuring flow rule filters to block any traffic not from a configured MAC address (in certain cases, MAC addresses representing both unicast and multicast traffic may need to be configured), rogue devices that may be otherwise properly configured cannot connect to the network.
- Similarly, physical ports in an OT-SDN environment should be configured to specifically allow only known IP addresses to successfully connect to the physical port. The combination of MAC and IP address filtering greatly reduces the ability of either rogue or misconfigured devices to connect to the network.
- Since traffic flow patterns (i.e., communicating node pairs) is static in an OT-SDN environment. Flow rules to match on the destination address (IP for Open Systems Interconnection [OSI] layer 3 and above traffic, and MAC for OSI layer 2 traffic) should be enabled to ensure that a compromised device is unable to establish communication with unauthorized devices on the network. Note, however, that the flow rules will need to accommodate communications to backup and redundant devices to allow continued operation in the event of a primary node failure.
- Because the protocols used in an OT-SDN environment also are static, flow rules that match on acceptable protocols (including User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) for layer 4 traffic, Address Resolution Protocol (ARP) and Internet Control Message Protocol (ICMP) for layer 3 traffic, and EtherType and virtual local area network (VLAN) tags for layer 2 traffic) can be used to reject any unauthorized protocols. In some cases, a combination of protocol and address can be used to further restrict unauthorized traffic. For example, a relay may be configured to allow IEC 61850 SV (sampled value) and GOOSE (generic object-oriented substation event) traffic as well as distributed network protocol version 3 (DNP3) [IEEE 1815] traffic, but flow rules can be created that restrict the IEC 61850 traffic to other IEC 61850 devices and restrict the DNP3 traffic to the substation gateway that communicates with the control center.
- The static nature of the OT-SDN environment allows the SDN flow controller to be disconnected while the network operates or be configured as a passive observer providing network monitoring and SA capabilities.

3.0 Requirements for Software-Defined Networks in Energy Delivery Systems

This section identifies the requirements to deploy SDN technology for control system networks and also expand on SDN deployments to provide additional levels of security. SDN is a unique technology because it natively improves cybersecurity of control system networks and also enables new security technologies on both the NBI and SBI of the SDN switch. SDN and a deny-by-default implementation methodology would defeat recent malware exploits that targeted foreign energy infrastructure. The ability to use SDN as an enabler will positively impact many aspects of security, from SA to enforcing SCADA protocol behavior. The requirements are organized into the following sections. The requirements already addressed by the SDN and Watchdog projects will be highlighted in the requirements table.

1. *Reliability and continuous operation* – The first requirements category is in the planning, design, and testing stages of new projects. The control systems that make up our critical energy infrastructure are purpose-built systems requiring the highest levels of reliability and continuous operation. These systems depend on the network to communicate between the devices doing the monitoring and control as well as between the operators and the control devices. All of these actions are pre-engineered and must strictly follow policy. The networks that carry these critical messages need to match the pre-engineered policy enforcement, high-reliability model. Designers must engineer each communications circuit and failover circuit, prove through professional engineering principles the reliability, and methodically test to make sure the system will perform all desired actions before going live.

2. *Managing change* – The second requirements category deals with change control and scalability of the network after it has been deployed and commissioned. It is desirable for energy sector control systems to minimize the number of changes required to keep the system operational. When changes are required, there needs to be a programmatic way to make these changes system-wide at a desired time while having the smallest impact possible to the larger system.

3. *Performance* – The third requirements category addresses engineering the communications circuits and the required performance, as well as the tools to monitor and guard this performance. The desire is to engineer the complete forwarding path the way we engineer power delivery circuits and their failover circuits, ensuring that we do not overload any segment of the circuit. Pre-engineering forwarding circuits for all communications also brings an expectation that the forwarding path will have the same latency, providing a baseline to calculate the deterministic parameters of the messages and validate they are met for the system. Traditional networking on a switched packet infrastructure takes the approach that more bandwidth and application retries make best-effort delivery good enough. This unknown cloud approach is not acceptable for critical infrastructure. There also is a desire to maximize networking asset utilization, eliminating blocking or other degradation technologies.

4. *Network monitoring* –- The fourth requirements category includes continuous supervision and visualization of the entire network for operational monitoring and management. Control system operators need to monitor and respond to network conditions like they do power system conditions. To do this, they need to understand the flows on the system and the expected behavior, be alerted when those behaviors change, and have the tools and training to know what to do to get the system back to normal operating conditions.

5. *Cybersecurity* – The fifth requirements category is the cybersecurity of the network. Control system networks are unmanned networks that often exist in places that are difficult to access physically. The engineers who design and deploy these systems want the capability to approve all services running on the network and deny all other flows by default. Any new communications flow should be approved before being allowed to connect.

Requirements for designing and deploying SDN technology are summarized in Table 3-1.

Table 3-1. Requirements

Requirement Number	Requirement Text	Requirements Category	Exists in SDN and Watchdog projects
General			
G1	Each SDN switch must be autonomous, and capable of starting, restarting, and running without an active SDN flow controller management mode directing it.	1	X
G1a	The SDN flow controller should maintain at least two configurations—a current running version, and a last-known-good version.	2	X
G1b	The power-up configuration on a switch should be the last-known-good version.	1, 2	
G2	Each SDN switch should provide SA information via the NBI in a trusted, interoperable manner.	4	X
G3	Each SDN flow controller or SDN switch should provide security event logging in a common format such as syslog.	5	X
Performance			
P1	Extending SDN to enable EDS protocol behavior will not adversely impact performance.	3	
P2	Moving target defense on WAN communications should allow one to distinguish between generator control and market data.	3	
P3	Moving target defense on WAN communications should not introduce jitter or unacceptable latency.	3	
P4	Each SDN switch must be capable of making route or path update decisions in less than 1 millisecond.	3	X
Operations			
O1	The SDN traffic engineering process must define communication pathways in normal and degraded states, accounting for multiple failure scenarios, and options for degradation of non-essential data flows.	3	X
O2	The SDN configuration will support redundant SDN flow controllers to ensure availability of the networking infrastructure and utilize a secure mechanism to exchange information between the SDN flow controllers.	3	
O3	The SDN environment will support EDS networks that use a primary and backup control center.	3	
Security			
S1	The SDN flow controller must authorize all SDN network components including the switch, links, hosts, users, settings, changes, health, etc.	5	
S2	The northbound and southbound communications should be through a cryptographically secure communications channel.	5	X
S3	Default passwords configured on the SDN controller and switches should be changed to a secure password.	5	

Requirement Number	Requirement Text	Requirements Category	Exists in SDN and Watchdog projects
S4	Firmware images should be signed and verified digitally during updates	5	
S5	The SDN flow controller itself needs to be secured against unauthorized access.	5	
S6	The SDN fabric should prevent lateral movement and generate an alert if lateral movement is attempted.	5	
S7	The SDN configuration should be flexible enough to enable “deep packet inspection techniques” to enforce EDS protocol behavior.	5	

4.0 Using SDN to Achieve Specific Security Requirements

This section was revised in the report ".Software-Defined Networks for Energy Delivery Systems: Business Function Use Cases" in November 2020,

This section of the blueprint architecture discusses how various aspects of SDN technology can be applied to addressing security requirements for the networking infrastructure, securing access to the components of the networking infrastructure, or protecting the attached end devices from attack.

4.1 Securing the SDN Flow Controller

Research[13] indicates that the SDN flow controller itself is a valuable target for cyberattack against SDN networks. Figure 4-1 illustrates a rogue SDN flow controller, direct attacks on the SDN flow controller, and attacks on the communication to SDN switches.

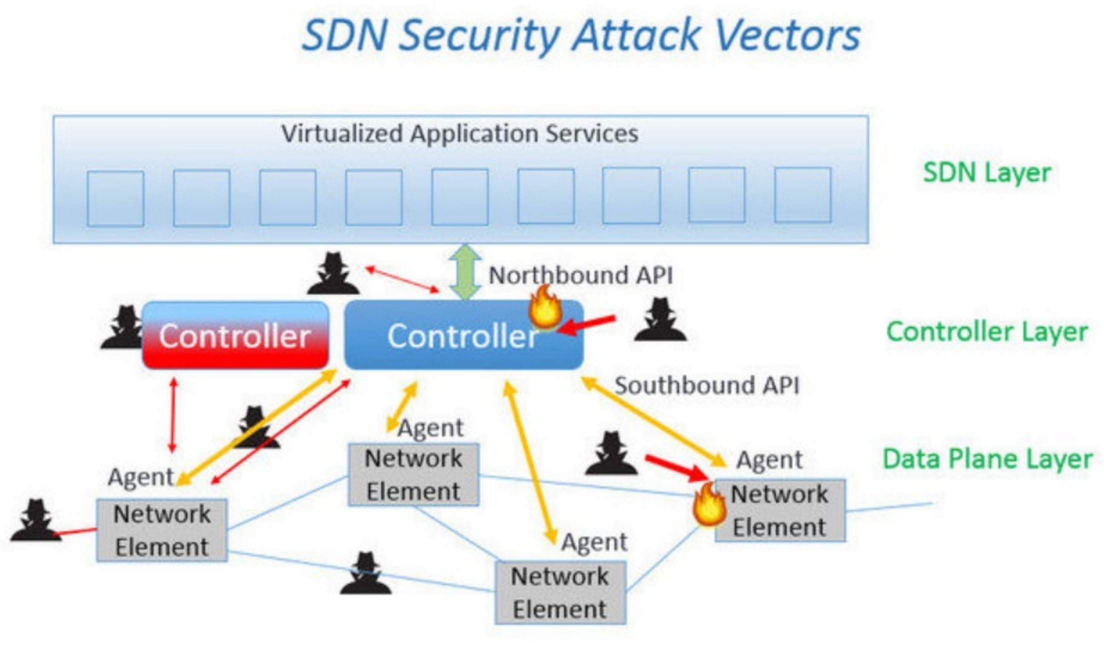


Figure 4-1. SDN Security Attack Vectors

To reduce the attack surface of an SDN flow controller and address the attacks described in Figure 4-1, the SDN4EDS project recommends the following architectural elements:

- Configure deny-by-default rules to limit which devices can communicate with the SDN flow controller, how they communicate with the SDN flow controller, and potentially when they communicate with the SDN flow controller.
- Implement cryptographic protections to secure all applications communicating with the NBI on the SDN flow controller.

[13] https://www.networkworld.com/article/2840273/sdn/sdn-security-attack-vectors-and-sdn-hardening.html (Accessed September 17, 2021)

- Configure deny-by-default rules to limit which devices can communicate with the SDN switch, how they communicate with the SDN switch, and potentially when they communicate with the SDN switch.
- Implement cryptographic protections to provide mutual trust between the legitimate SDN flow controller and SDN switch to eliminate rogue SDN flow controllers or SDN switches over the SBI.
- Configure each SDN switch with unique cryptographic keys to provide mutual authentication with the SDN flow controller.
- Encrypt all traffic that flows between the SDN flow controller and SDN switch over the SBI and authenticate it using algorithms approved by the National Institute of Science and Technology.
- Log flows received and installed on the SDN switches to detect attempted rogue SDN flow controller connections.
- Avoid reactive SDN flow controller installations, if possible, to prevent DoS attacks on the SDN flow controller.
- Implement SDN flow controller fail-over or backup options in the event a compromised SDN flow controller needs to be restored.

Note that in a hybrid environment, (i.e., an Ethernet environment that contains both SDN switches and traditional network switches), the provisions described can be used to protect the SDN environment, including the interface between the SDN and traditional environments, but will have little impact on the security of the traditional switches and the devices connected to them.

The SEL 5056 SDN flow controllers can be configured to communicate with SDN switches using communications secured with transport layer security (TLS) based encryption and authenticated using digital certificates. This configuration is allowed by the OpenFlow 1.3 specification but is not required by it. By using this encrypted and authorized configuration, SDN switches will only accept commands to modify their SDN flow rules from authorized SDN flow controllers. In an OT-SDN environment, all SDN flow rules are proactively configured (i.e., configured in advance), so the SDN flow controller has no role in the real-time operations of the SDN switch, thus minimizing the impact of a DoS attack against the controller

4.2 Securing Access to the Network

The first line of defense for securing an SDN environment is to only allow authorized nodes to connect to the network. This is accomplished by flow rules that implement ingress filtering. This filtering can be implemented by matching by MAC address, IP address, IP protocol (e.g., TCP, UDP), VLAN tag, source and destination TCP or UDP port, etc. the most common methods include filtering by IP address and TCP or UDP ports. If VLANs are implemented, they are also used to provide filtering and traffic management similar to VLAN implementations in traditional networks.

Providing MAC address filtering in ingress filters can provide an increase insecurity by detecting rogue devices that are configured with the same IP addresses and traffic patterns (i.e., TCP and UDP traffic usage), however, there are two problems with implementing MAC address filtering.

The first issue with MAC address filtering is an operational consideration. Since each end device has a unique legitimate address, replacement of a failed devices will present a different MAC address to the SDN switch. Unless the flow rule is updated to accept the new MAC address (and reject the MAC address from the failed equipment), the replacement equipment will not be able to participate in any network communications. The updating of the SDN flow rule will require additional actions to perform the replacement, and may require additional staff, which will cause the equipment outage to last longer than it would have in a traditional network environment. This may be acceptable in some environments, but not in others.

The second issue with MAC address filtering is that while MAC addresses are intended to be unique to each ethernet controller, many ethernet controllers support the ability to modify the MAC address. This can be used to mitigate the maintenance issue discussed above – the replacement equipment can be configured to have the same MAC address as the failed equipment thereby not requiring any SDN flow rule changes; however, it can also be used by rogue devices to masquerade as legitimate devices in the network.

Thus, while MAC address filtering appears to resolve the issue of rogue device addressing, it introduces a false sense of security, and should therefore not be relied upon to prevent rogue or masquerading devices access to the SDN environment.

4.3 Preventing Lateral Movement

Lateral movement is a term used to describe how an adversary traverses a network from one compromised system with the objective of gaining additional points of control. An adversary may conduct reconnaissance from a compromised system to identify available ports and services to target for their next exploit[14]. In an EDS network, let’s assume that an adversary has compromised the Historian. From the Historian, the adversary would like to pivot to another device such as the Human Machine Interface (HMI) in order to manipulate the system.

To minimize the ability of an adversary to move laterally in the EDS network, SDN flow rules are used. The default configuration needs to be “deny all communication” unless explicitly permitted. Next, flow rules are created that define both the type and direction of communication. Figure 4-2 and Figure 4-3 identify how fields are matched using processing defined by the OpenFlow specification [OpenFlow 2012] in a SEL SEL-2740S SDN switch. One or more rules will be created to allow a SCADA server or other device to initiate communication with the Data Historian.[15] With no rule created to allow the Data Historian to communicate with these devices, it cannot initiate communication, so the adversary is unable to conduct further reconnaissance of the network. If an adversary were to attempt scanning or lateral movement, the traffic would not match an SDN flow rule and would be dropped and logged. The beneficial result is the Data Historian cannot be used effectively as a launch point to move laterally through the EDS network.

[14] see http://www.securityweek.com/lateral-movement-when-cyber-attacks-go-sideways (Accessed September 17, 2021)

[15] A Data Historian is a software program that records and retrieves production and process data by time; it stores the information in a time series database that can efficiently store data with minimal disk space and fast retrieval.

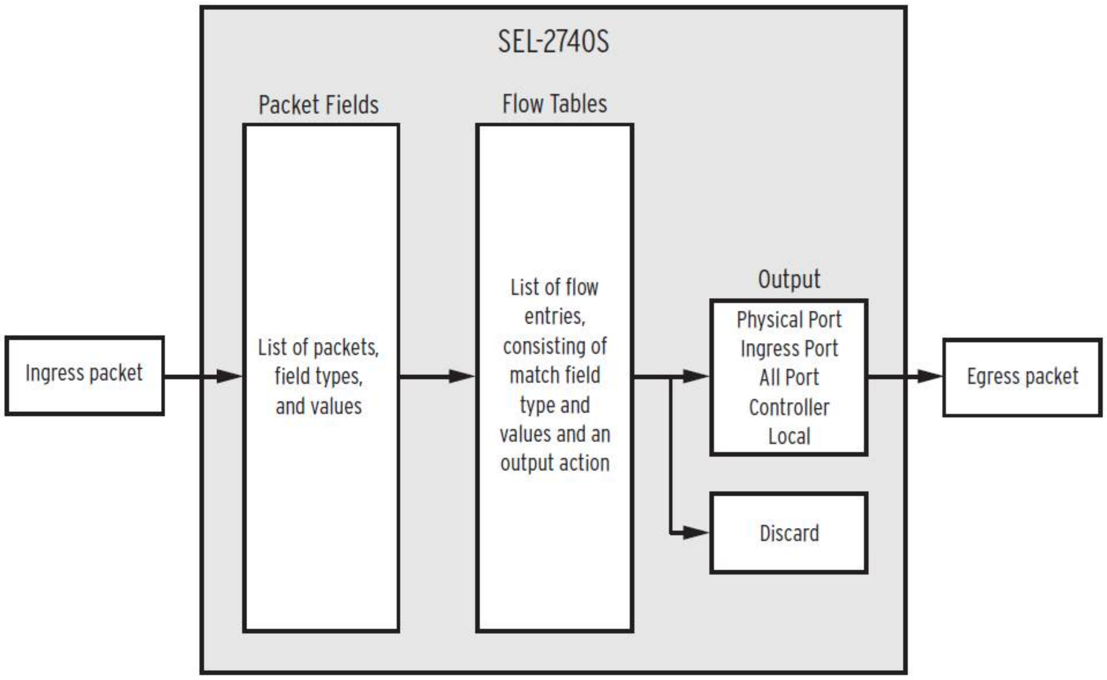

SEL-2740S
Packet Fields
Flow Tables
Ingress packet
List of packets, field types, and values
List of flow entries, consisting of match field type and values and an output action
Output
Physical Port
Ingress Port
All Port
Controller
Local
Discard
Egress packet

Figure 4-2. SDN Processing in a SEL-2740S Switch[16]

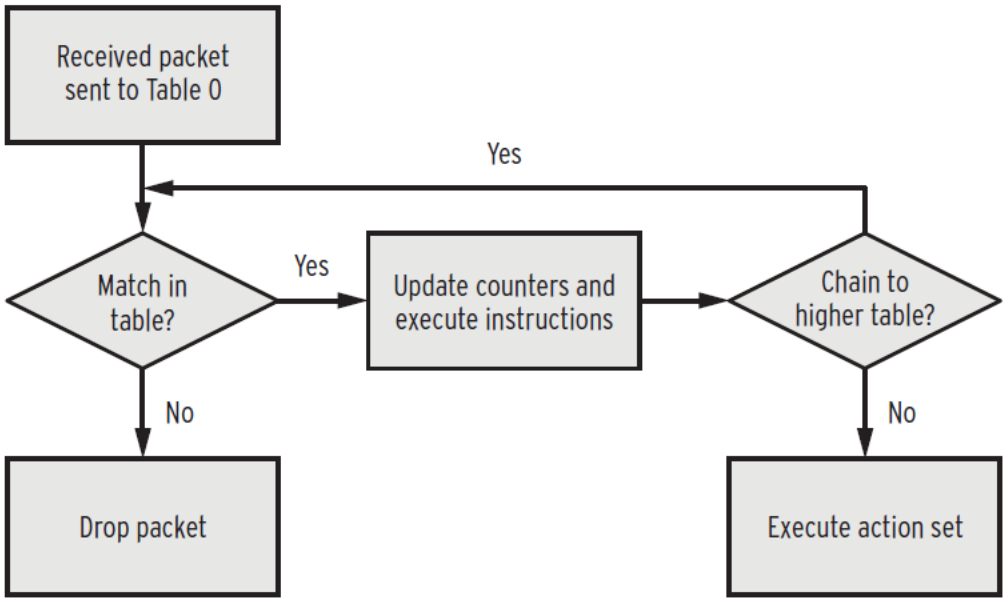


Figure 4-3. Flow Table Processing[17]

To reduce the ability to move laterally through an EDS network, the SDN4EDS project recommends the following architectural elements:

[16] Source:selinc.com. Used with permission
[17] Source:selinc.com. Used with permission

- Deny-by-default rules are created that only permit the network communication required to support the applications that monitor and control the physical process or grid.
- SA is provided through the NBI to identify which devices are attempting to communicate using unapproved methods.
- SDN metrics are used to identify changes in network communications.
- Provide interfaces for installation of additional layers of defense, such as intrusion detection systems (IDS), antivirus, intrusion prevention systems (IPS) to monitor and detect anomalous behavior.
- Utilize ability of SDN to capture anomalous traffic and send to IDS or storage for analysis.

As shown in Figure 4-4, SDN flow rules are comprised of several fields, all of which are used to control the behavior of the SDN switch and play a role in preventing lateral movement.

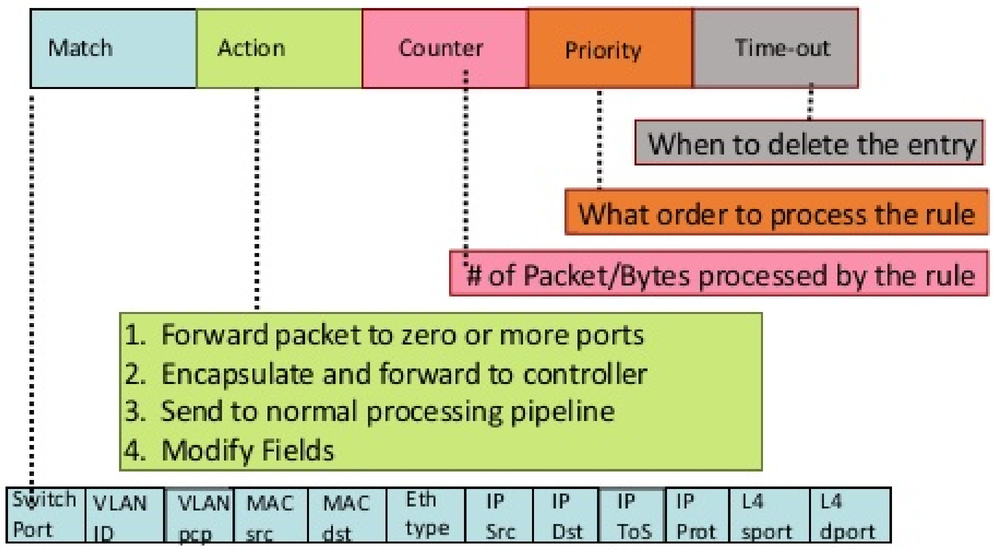

Match
Action
Counter
Priority
Time-out
When to delete the entry
What order to process the rule
# of Packet/Bytes processed by the rule
1. Forward packet to zero or more ports
2. Encapsulate and forward to controller
3. Send to normal processing pipeline
4. Modify Fields
Switch Port
VLAN ID
VLAN pcp
MAC src
MAC dst
Eth type
IP Src
IP Dst
IP ToS
IP Prot
L4 sport
L4 dport

Figure 4-4. Anatomy of a Flow Table Entry[18]

The fields in an OpenFlow flow table entry are:

- *Match*: This field specifies what portions of the incoming frame are to be inspected to determine whether the frame should be processed or not. The available match fields are commonly found entries from in OSI layers 2, 3, and 4. If the frame does not match any flow table entry, it is dropped. Match fields can include wild cards to either simplify processing or allow multiple actions to be taken against the same frame by different flow rules.
- *Action*: This field specifies what should happen to the frame if it matches the entry in the match table. A typical action is to forward the frame to one or more physical ports.
- *Counter*: This field is incremented every time a frame matches a flow table entry match field to allow performance statistics to be kept on a flow table entry basis.

[18] https://www.slideshare.net/SharifulIslam22/introductionto-sdn-69428319 slide 35 (Accessed November 5, 2020)

- *Priority*: This field specifies the flow table entry priority. Higher priority entries are processed first, allowing frames to be processed by multiple flow table entries in a specific order.
- *Time-out*: This field specifies an inactivity timeout for when the SDN switch is to automatically delete the flow table entry.

4.4 Securing WAN Communications

The use of the internet and other WANs in the energy sector is increasing for many functions, including providing outage information to customers, as a mechanism for customers to submit payments, and as a low-cost communications medium. This increased use comes at an increased risk of cyberattack resulting in degraded or disrupted communications. With the rise of botnets, the likelihood of a distributed denial of service (DDoS) attack against a utility is a distinct possibility. The Mirai botnet was used in this manner to cause a DDoS against the “Krebs on Security” website in 2016[19]. It is also becoming more common for the internet to be used as the communication pathway for independent power producer (IPP) to market operator or generation scheduler communications. Historically a remote intelligent gateway (RIG) was used to securely exchange SCADA or market data. When initially introduced the RIG was an expensive solution, but since then, inexpensive solutions have been developed.

WAN communications also are used for operational communications, including communications from a control center to another control center (either within a specific utility or between utilities), or between a control center and field locations such as substations or generation plants. Over the last 25 years, generation not owned by a traditional utility, referred to as IPP generation, has grown significantly. Unlike utility-owned generation, these IPP generation sites often do not connect with the utility control centers using internal private networks, but rather use shared networks, or in some cases, the public internet. Securing critical control communications is essential for both power system reliability and for financial (market) purposes.

Traditionally, communication security has relied on encrypting the traffic and authenticating both the sender and receiver, but traffic flow patterns, including message frequency and source and destination locations, can still be monitored in the WAN infrastructure. Recent advances in dynamic traffic management have made this monitoring more difficult by changing the physical pathways used in the WAN without adversely compromising the performance of the end-to-end communication.

An emerging use of WAN communications is for transporting routable versions of International Electrotechnical Commission (IEC) 61850 GOOSE and SV messages. While there are standard protections available via IEC 62351 for protecting the authenticity and confidentiality of the messages, analysis of traffic patterns could be a potential source of cybersecurity concerns.

SDN has been identified as one method to defeat DDoS attacks.[20] SDN supports multiple methods to respond to DDoS events, such as dropping unwanted traffic using SDN flow rules or redirecting prioritized communication over secondary or tertiary communication pathways to ensure availability of control.

---

[19] https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/ (Accessed September 17, 2021)
[20] https://www.networkworld.com/article/3156344/internet/2017-widespread-sdn-adoption-and-ddos-attack-mitigation.html (Accessed October 14, 2020)

An approach from Juniper Networks[21] is a software-defined–wide-area network (SD-WAN) architecture that is flexible and uses an SD-WAN controller to act as an "orchestrator" to control and manage traffic flows as the traffic is routed from site to site. As shown in Figure 4-5, a basic multi-site SD–WAN architecture includes just a few basic elements:

- Multiple connections between sites that form the underlay network providing for load balancing and resiliency in the event of a link or transport network failure
- Multiple overlay tunnels that provide a logical view of how the traffic flows between the sites
- A controller that configures and manages the SD-WAN gateway devices and the overlay tunnels.

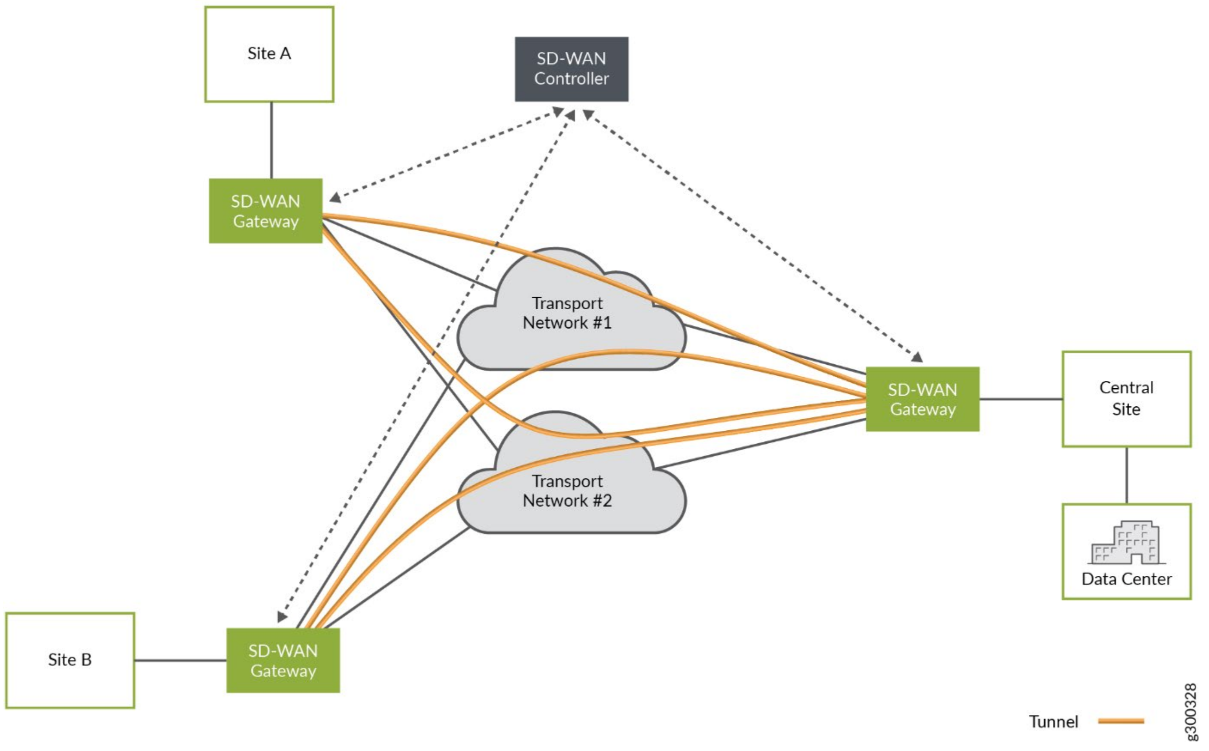


Figure 4-5. Juniper SD-WAN Architecture[22]

To reduce the impact of WAN-based cyberattacks on EDS network, the SDN4EDS project recommends the following architectural elements:

- Configure deny-by-default rules that only permit the network communication required to support the applications that monitor and control the physical process or electric grid. All other traffic will be discarded and logged.
- Deploy SDN flow rules to prioritize EDS communications on secondary or tertiary communication pathways in the event the primary pathway becomes degraded or fails.

[21] https://www.juniper.net/documentation/en_US/cso5.1.2/topics/concept/sd-wan-deployment-architectures.html (Accessed November 7, 2020)
[22] https://www.juniper.net/documentation/images/g300000-g300999/g300328.png (Accessed November 7, 2020)

- Provide SA capability through the NBI to identify discarded traffic and to inform operators of potential actions.
- Incorporate a moving target defense mechanism to reduce the amount of data available to MITM attacks.
- Use generic consumer-grade internet connections to ensure that bandwidth representative of a field location is used.
- Use the ability of SDN to capture anomalous traffic and send to an IDS or storage for analysis.

4.5 Securing Engineering Access

Remote EA to substation devices provides asset owners the ability to validate firmware versions, check or change configuration settings, and perform other maintenance functions on devices without the expense and time required to perform these functions on-site. This capability reduces cost and improves efficiency, but it also potentially introduces an attack vector. The ICS-CERT Alert on the 2015 cyberattack in the Ukraine[23] identified that firmware on serial to Ethernet devices in substations was corrupted, rendering the devices inoperable. It is assessed that this action was done to interfere with restoration efforts.

To reduce the attack surface of remote maintenance activities, SDN flow rules can be configured to only permit EA for remote maintenance when it is necessary. Separation of duties between the management of flow rules and the engineer performing maintenance will help reduce insider threats.

The SDN4EDS project recommends the following architectural elements:

- Configure deny-by-default rules to prohibit remote EA when it is not needed.
- Enable EA only for a specific engineering workstation. Activate the SDN flow rule only when maintenance activities are scheduled to be performed.
- Implement the SDN flow rule to incorporate a timer and auto-expire, reverting to the deny-by-default configuration when maintenance activities are complete.
- Implement SDN flow-based encryption to secure EA communication while in transit.

4.6 Enforcing EDS Protocol Behavior

According to the background section of NERC's June 2004 Deliverables and Work Schedules report issued by the Critical Infrastructure Protection Advisory Group for the Process Control System Security Task Force, "Control Systems are the 'brains' of the control and monitoring of the bulk electric system [BES] and other critical infrastructures, but they were designed for functionality and performance, not security. Most control systems assume an environment of complete and implicit trust."[24] The implicit trust attribute applies to many EDS protocols, where the ability to cryptographically ensure message integrity is not available natively in the protocol. EDS protocols are full of interesting features used to monitor and control expensive physical

[23] https://ics-cert.us-cert.gov/alerts/IR-ALERT-H-16-056-01 (Accessed September 17, 2021)
[24] *CIPAG WorkPlan PCSS TF - Final1.doc*, dated June 7, 2004 initially accessed at the following website: http://www.nerc.com/comm/CIPC/Pages/Control%20Systems%20Security%20Working%20Group%20CSSWG/Control-Systems-Security-Working-Group-CSSWG.aspx (not available online)

assets. Incorrect operation of these assets can lead to cascading failures or damaged equipment.

In addition to providing native security controls, SDN enables the deployment of new security controls. By leveraging attributes of SDN communication, the vision is to develop a proof-of-concept system to centrally define expected EDS protocol behavior and enforce that behavior in a distributed manner. The ability to limit protocol functionality can take many forms including:

- Defining which function codes are permitted
- Limiting which function codes or commands can be issued to multicast addresses
- Limiting the ranges or registers addressed by a command
- Limiting the frequency of control commands
- Providing awareness of EDS protocol use that is outside of expected behaviors

To reduce performance impacts upon SDN switch hardware, a separate system, appliance, or device will be used to enforce the defined protocol behavior.

The SDN4EDS project recommends the following architectural elements:

- Define flow rules to ensure EDS communications occur between authorized devices.
- Define a flow rule to ensure all EDS traffic is examined by the protocol enforcement technology.
- Provide SA of EDS communications through the NBI.

This document includes sample SDN flow rules to enable the enforcement of EDS protocol behavior. The EDS protocols for which the end user will be able to define desired behavior include DNP3 over Internet Protocol (IP)—referred to as DNP3/IP, IEC 61850, Institute of Electrical and Electronics Engineers (IEEE) C37.118, and Modbus over TCP—referred to as Modbus/TCP.

Most EDS protocols are logically point-to-point (i.e., in a substation) a remote terminal unit (RTU) or substation data gateway will communicate with individual intelligent electronic devices (IED) one at a time, each with their own protocol address. SDN flow rules can be established to limit which SDN switch ports participate in each conversation, thereby preventing rogue devices from connecting to the SDN switch and transmitting or receiving malicious traffic.

However, some protocols like IEC 61850 are intended to be multicast to allow a single message sent by a publisher to be received and processed by multiple subscribers. This allows, for example, the measured SV data from a single sensing device (known as an IEC 61850 merging unit) to be processed by multiple IED relays. Additional IEDs can be added to the configuration without increasing the number of network messages, making a change to the publishing device, or adding any network connections other than that to the added IED. The same processing and configurations are used for GOOSE messages.

For individual IEDs to receive the SV or GOOSE messages, the IEDs must be configured to subscribe to the individual data streams. This is most often accomplished by the use of multicast Ethernet addresses. The publishing node sends the data with a multicast destination, and each subscribing node configures its Ethernet adapter to receive messages sent to the multicast address. In a traditional Ethernet network, the multicast messages are copied to each physical port because the switch cannot determine where the multicast receiver nodes will be. However, in an SDN environment, the SDN switch can be configured with SDN flow rules to receive the multicast traffic only on ports that are configured to publish it and forward the traffic only to ports that are configured to subscribe to it. This results in the inability of rogue devices to spoof publishing of malicious traffic and minimizes the receipt of the traffic by rogue subscribers or intercepting and retransmitting the traffic in a MITM attack.

4.7 Establishing Trust between SDN Hardware and Software Components

The ability to establish trust between the hardware and software components of SDN product offerings is crucial to the security of SDN networks. This trust must also be established in a manner that enables interoperability of hardware and software from multiple vendors. While proprietary vendor-specific solutions may establish trust, they may not easily support an environment with diverse hardware and software components.

The vision is to define a method to establish trust between SDN components that supports interoperability. The method(s) will be shared with the SDN community to gain support by other vendors.

The SDN4EDS project recommends the following architectural elements:

- Implement cryptographic protections to secure all applications communicating with the NBI on the SDN flow controller.
- Implement cryptographic protections to provide mutual trust between a legitimate SDN flow controller and SDN Switch to eliminate rogue SDN flow controllers or SDN switches over the SBI.
- Configure each SDN switch with unique cryptographic keys to provide mutual authentication.
- Research is required to provide guidance about how to securely configure, review, design, approve, use all SDN interfaces (NBI, SBI, and east/west interface).

This topic is further discussed in Section 6.0.

4.8 Providing Situational Awareness

This SDN4EDS task requires that the project team research and explain where technology and data reside – in the SDN switch, the SDN flow controller, in third-party applications and algorithms that interface with the NBI, or other locations such as an IDS – to provide an accurate cybersecurity model. For the automated analytics task, SA information is securely provided by a SDN flow controller through the NBI (see Figure 2-1). Metrics or accounting data can be used by analytical algorithms to identify changes in behavior. For example, these algorithms can detect if a SCADA server is sending DNP/IP commands at half the expected or normal rate.

During the SDN4EDS project, analysis was done on hybrid infrastructures where a combination of SDN switches and legacy managed switches are used in the EDS network. The rationale for this approach is that a deployment of SDN technology will be phased, and there may be certain network segments that continue to use the traditional managed switch infrastructure. We need to learn how to analyze SDN metrics and net flow data together. This project also explored how SDN flow tables can inform net flow analysis.

The SDN4EDS project recommends the following architectural elements:

1. Establish a trusted connection between the analytics engine and the NBI of the SDN flow controller. This requirement was met by following SDN4EDS trust guidance.
2. Develop analytics, or borrow analytics from the Chess Master Project[25], if appropriate, to identify changes in approved SDN traffic behavior.
3. Develop hybrid analytics based upon SDN-provided metrics and flow rules.
4. Present outcomes of hybrid analytics to SDN flow controller or other monitoring tool.

Section 6.0 provides information on how trusted connectivity between an SDN flow controller and an analytics engine can be implemented in an SDN environment.

4.9 NERC CIP Discussion

Current NERC CIP standards (as of the summer of 2021) use the concept of an Electronic Security Perimeter (ESP) and an Electronic Access Control or Monitoring System (EACMS) (which is generally implemented as a filtering firewall) to manage logical access to critical control system components (called BES Cyber Assets [BCA] or BES Cyber Systems [BCS]). The concept requires that all access to the BCA must pass through and be filtered by rules in the EACMS. One could argue that the SDN flow rules in an SDN switch can perform this filtering capability by distributing the EACMS filtering to the network core rather than the border. However, the current wording of the NERC CIP standards implicitly requires that the EACMS be a device on the network border (the ESP) rather than a function in the network core, so an interpretation that allows SDN flow rules to serve as EACMS filters would be needed. The previously mentioned work of the Standards Drafting Team would appear to be moving in this direction, but absent the specific language of an approved NERC standard, it is imprudent to assume that interpretation.

The language of the CIP standards also assumes implicit trust of all components within the ESP, whether they play a role in reliable operations (and are therefore identified as BCA), or whether they are ancillary to reliability (and are identified as Protected Cyber Assets [PCA]). The current version of the standard does not restrict network flows but does require that all the PCAs within the ESP implement the same cybersecurity provisions as the BCAs within the ESP. This prevents an otherwise unprotected device inside the security perimeter from becoming a pivot point to attack the BCAs inside the ESP.

In early 2021, during a development update webinar, the NERC Standards Drafting Team proposed changes to the CIP Standards to allow virtualization and zero-trust implementations. SDN is a possible implementation that supports and enables the concept of zero trust networking and can be used to enforce virtual networking through carefully crafted SDN flow rules. These SDN flow rules would limit network traffic to and between individual BCAs, allowing the functions of the EACMS to move strictly from the border of the OT network into the core of the protected network. SDN flow rules could be used to improve the cybersecurity of the core

[25] Note – ChessMaster is a DOE funded project designed to use SDN to combat Ukraine style cyberattacks. See https://energy.gov/sites/prod/files/2017/05/f34/SEL_ChessMaster_FactSheet.pdf (Accessed September 17, 2021)

network by allowing network flows only between specific BCAs both by node address and protocol. It will also increase the cybersecurity by implementing similar restrictions on how the PCAs can interact with the BCAs. Properly crafted SDN flow rules will essentially eliminate the implicit trust currently assumed for communications within the ESP.

The SDN flow rules can also be used to provide the border protections currently provided by the EACMS by restricting which BCAs can interact with external by node address and protocol, similar to how most EACMS firewalls are currently configured.

Using SDN's deny-by-default model, any rogue device with a node address that would otherwise be allowed in a traditional network would be denied access to any of the BCAs or PCAs since no SDN flow rules would be enabled to allow network traffic from the rogue device. If MAC addresses are configured in the SDN flow rules, additional protections beyond what is already proposed could be implemented to deter any rogue devices within the ESP that attempt to masquerade nodes with valid node addresses.

Although these changes have not been finalized, SDN can still play a role in additional enforcement of network security within an ESP by implementing zero-trust network concepts in addition to the requirements implemented using EACMS border devices. Once these changes have been finalized and adopted, the EACMS border devices may no longer be required, but could still be left in place as an additional layer of security supporting network defense in depth concepts.

4.10 Zero-day and Known Vulnerability Mitigations

Both zero-day and known vulnerabilities can take several forms. They may be exploitable by flaws in how EDS protocols are implemented by EDS host devices like RTUs and IEDs, or they may be exploitable by unnecessary or misconfigured software on the EDS host device. SDN technology can assist in the mitigation efforts to prevent these vulnerabilities from being exploited in several ways (many of which will be discussed in later sections):

- Establish SDN flow rules to limit the protocols (by TCP or UDP port number) that can be sent to the EDS host device. If a particular service implementation in the EDS host such as Hypertext Transfer Protocol (HTTP) contains a vulnerability, but that service is not used by the utility, all HTTP traffic can be blocked from being sent to the EDS host device.
- Establish SDN flow rules to limit which hosts (by IP address) can communicate with the EDS host device.
- Establish SDN flow rules to limit which EDS hosts can use which protocols when communicating. For example, if an EDS host is configured using a web browser, SDN flow rules can be established to 1) disable HTTP and 2) allow Hypertext Transfer Protocol Secure (HTTPS) traffic only from an authorized engineering workstation. Because configuration is not a real-time function, this SDN flow rule should only be enabled when device configurations are being performed and disabled after the configuration operation is complete.
- Establish SDN flow rules to forward EDS traffic through an IPS that can detect protocol anomalies (such as buffer overflow attempts) that may represent vulnerability exploits. The IPS can also be used to detect and prevent the use of dangerous commands (like factory reset or firmware install) unless enabled through a separate mechanism.

5.0 Scalability

The reference architecture was used to measure scalability and redundancy attributes of EDS SDN infrastructures. The metrics may be based upon the number of traffic flows, the number of SDN switches, or the volume of traffic. Two additional aspects of scalability that were addressed in this section include change control and how to enact multiple changes that result in the safest and most reliable change set to the networking infrastructure.

5.1 Control Plane Scalability

The typical OpenFlow model of one centralized SDN Flow Controller to many SDN switches may result in several complications as the size of the network grows. In a paper comparing the performance differences of proactive vs reactive OpenFlow models [Fernandez 2013] (see section 8.4.3 for additional information), Marcial Fernandez describes that the OpenFlow protocol can fall victim to this scenario resulting in several concerns including the abundance of OpenFlow control messages that must be processed by the controller as the number of SDN switches grow, an increase of network diameter and setup of a new SDN switch depending on the vicinity it lies from the SDN flow controller, and the scaling of new traffic flows over the SDN as limited by the processing power of the SDN flow controller as the size of the network grows.

One advantage of the use of OpenFlow in an OT network is the use of proactive flows to decrease the amount of OpenFlow control messages in the network, yielding for better performance from the SDN flow controller. Fernandez was able to demonstrate in Mininet networks of 100 and 200 OpenFlow SDN virtual switches, the performance of the proactively programmed network was always higher than their reactively programmed counterparts. One still must consider the linear scale at which the number of OpenFlow control messages will increase as the number of SDN switches increase, despite no reactionary decision overhead. The second point made is also a concern, as the diameter of the network grows wider, the route traversed of the OpenFlow control message from the SDN Flow Controller to a new additional SDN switch on the outside may take longer and thus introduce a new latency in programming these new SDN switches.

A possible approach to address control traffic bottleneck is the strategic placement of multiple SEL SDN flow controllers that would be designated to, and manage a given zone, for example a substation or a plant. Using this method, the quantity of OpenFlow control messages is limited to the scope of those SDN switches in that zone. This has an advantage of being able to manage multiple zones on a smaller scale rather than on a large enterprise level. The disadvantages to this method are that the SA across the entire network is lost, with views only available in each individual network, and that there are more networks to now manage.

Other scalability concerns regarding the SDN flow controller or control plane include the lack of an active redundant mode of operation by the SDN flow controller itself. At the time of this writing, there are currently no redundancy options for the SEL-5056 SDN flow controller. This means that a single active SDN flow controller is used to control the SDN regardless of the number of SDN switches that it contains. SDN flow controller redundancy is achieved by restoring a copy of the SDN flow table database on another SDN flow controller node and establishing that new node as the active SDN flow controller.

While it is true that in this proactive model, that if the SDN flow controller is taken offline, the network will still function as per its last programmed state, operators lose site of the SA into their network, along with any ability to disable, delete, modify, or add existing or new flow rules.

Ad-hoc work has been done by PNNL to explore how a failover functionality could work and be implemented. By utilizing virtualization technologies, a primary VM hosting the SDN flow controller can be mapped to the physical network, while at the same time leveraging the use of the hypervisor to periodically make backups via the SEL-5056 REST (REpresentational State Transfer) API and send those over to a secondary SDN flow controller VM that will sit on standby. After a disconnect of the primary SDN flow controller either logically or physically, a script can trigger the failover to the secondary VM, run another script that uploads the backups to the new controller, and establish itself as the new SDN flow controller of the network. From the point of the data plane, this SDN flow controller has all the same characteristics (IP address, physical port connection) as the previous and so the observability and control of the network is restored.

In a related project by Sandia National Laboratories. the use of containers with a container orchestration tool such as Kubernetes has also been considered and demonstrated to provide a degree of redundancy that would be more low-cost than the use of traditional VMs. At the time of this writing there is no containerized version of the SEL-5056.

These solutions may not be realistic for environments that cannot host a hypervisor server or support containerization or virtualization technologies. Additional considerations to make the solution more robust and production ready must also be made prior to deployment due to the current ad-hoc nature of these approaches.

5.2 Flow Rule Scalability

A SEL-2740S SDN switch running firmware version SEL-2740S-R106-V0-Z001001-D20210122 supports four OpenFlow tables (following zero based numbering from 0 to 3), and 2000 flows per table, supporting a total of 8000 supported flow rules[26]. Additionally, 256 OpenFlow group entries are also supported.

In a production environment, these specifications may not be enough to support that actual number of flow rules required, which grow quite rapidly. To put into perspective a usual flow creation workflow; for two devices that speak Layer 3 or higher, two rules (bidirectional) must be added for ARP and two for each relevant protocol. When we take into consideration that by default, CST entries enable the creation of failover flows, this number is now multiplied by two, so 8 total flow rules for two devices. In a realistic scenario, a device is not just limited to communicating with one other device and may have multiple connections with many other devices. Furthermore, it should be anticipated that intermediary switches between source and destination devices may also have to support flow rules that are used to forward traffic between endpoints, meaning that switches will have flows that are not directly affecting those end devices attached to them.

SEL has developed a method of addressing this “blow-up” of flow rules by using flow aggregation[27]. By specifying the ingress management across the four flow tables, abstract device filtering can be defined before handling detailed flows across the remaining tables in

[26] SEL-2740S Software-Defined Network Switch Instruction Manual
[27] Flow Aggregation Feature, Jason Dearian, Schweitzer Engineering Laboratories. (Not available online).

case the criteria for table 0 is not fulfilled. VLAN tagging is leveraged to support this method. By tagging the traffic with the VLAN ID of the destination switch on the source port switch, the intermediary switches can forward traffic relative to the destination rather than the detailed content. This implementation will be provided in SEL's Flow Aggregation Feature tool that is currently still under development.

5.3 Flow Rule Granularity

SDN flow rules can be implemented with various levels of granularity, which can impact the security, maintainability, and scalability of the network. A system with low levels of granularity may be easier to maintain and have less impacts on future scalability but may result in a less secure network environment. On the other hand, a system with high levels of granularity will be more secure by implementing a finer-grained set of allowable flows, but that level of detail in the SDN flow rules will be more difficult to maintain and may lead to future scalability issues as additional devices or flows are added to the environment.

As has been discussed previously, SDN flow rules can be written to match on a variety of fields and then take one or more actions. Figure 5-1 depicts the matching process for OpenFlow 1.0. SDN flow rules can be written to match on any of the fields along the bottom row, but if a specific field is not included in the SDN flow rule match table, a wildcard match is implemented. As an example, if the source Ethernet MAC address is not specified but an IP source address is specified, then any device with the configured IP source address can access the network.

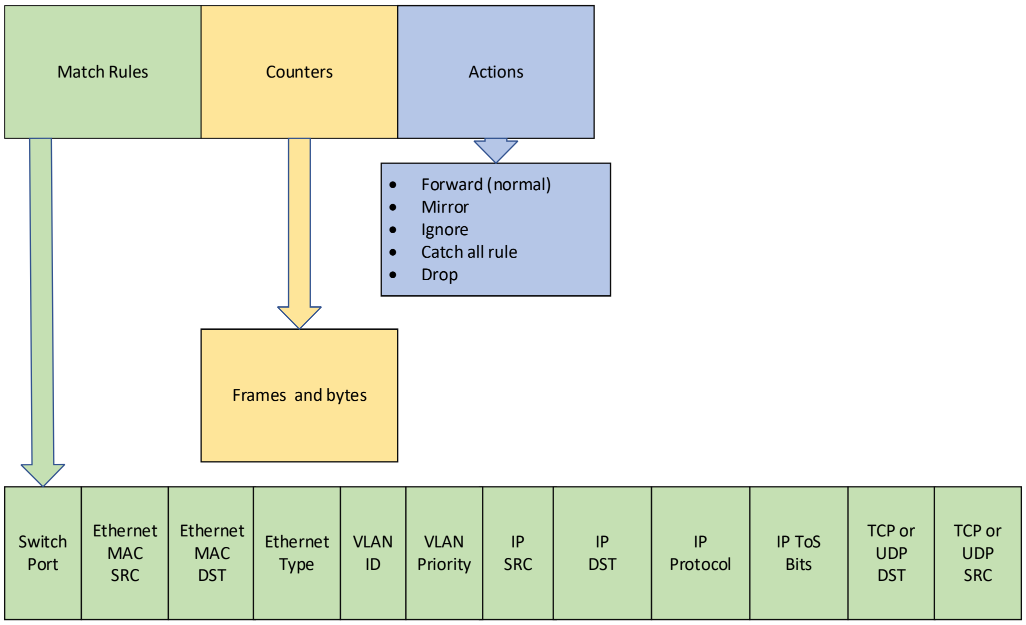
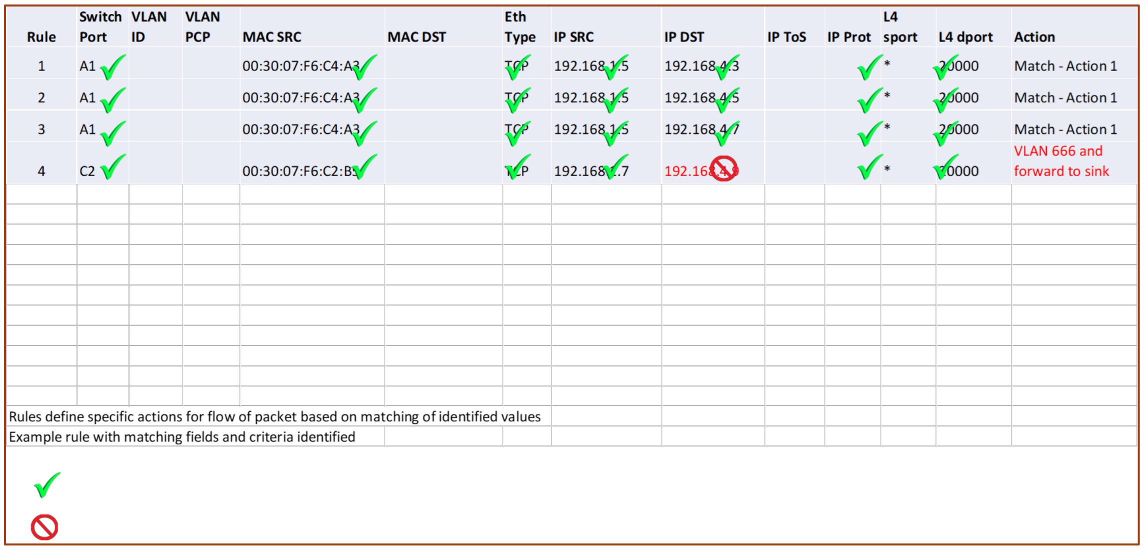


Figure 5-1. Flow Rule Processing

When creating SDN flow rules, especially for retro-fit implementations in a brownfield environment, understanding the existing environment and its expected flows is important. Capturing actual traffic using a network analysis tool can assist in determining what flow rules are appropriate. Figure 5-2 shows an example table showing the flows that have been created or denied based on traffic analysis and understanding the network environment. In this example, rule 4 has been implemented to block an undesired traffic flow.



Rule	Switch Port	VLAN ID	VLAN PCP	MAC SRC	MAC DST	Eth Type	IP SRC	IP DST	IP ToS	IP Prot	L4 sport	L4 dport	Action
1	A1 ✓			00:30:07:F6:C4:A3 ✓		TCP ✓	192.168.1.5 ✓	192.168.4.3 ✓		✓	*	20000 ✓	Match - Action 1
2	A1 ✓			00:30:07:F6:C4:A3 ✓		TCP ✓	192.168.1.5 ✓	192.168.4.5 ✓		✓	*	20000 ✓	Match - Action 1
3	A1 ✓			00:30:07:F6:C4:A3 ✓		TCP ✓	192.168.1.5 ✓	192.168.4.7 ✓		✓	*	20000 ✓	Match - Action 1
4	C2 ✓			00:30:07:F6:C2:B3 ✓		TCP ✓	192.168.1.7 ✓	192.168.4.9 🚫		✓	*	20000 ✓	VLAN 666 and forward to sink

Rules define specific actions for flow of packet based on matching of identified values

Example rule with matching fields and criteria identified

✓

🚫

Figure 5-2. Traffic Analysis

In addition to address and protocol-based filtering, flow rules can also be turned on and off to allow control over actions that occur sporadically, such as maintenance access, or occur infrequently but on a regular basis such as monthly compliance scanning. These are referred to as temporal rules, and they can be enabled manually or automatically by an application program interfacing with the SDN flow controller.

Table 5-1 shows some of the trade-offs in implementing varying levels of SDN flow rule granularity in an environment.

Table 5-1. Operational and Security Trade-offs

Flow Rule Matching	Operational Use Case	Security Considerations	Potential Mitigations
Match only on physical port or IP address	Tie point with managed switch network Mirroring traffic	Protection provided by defense in depth outside of OT-SDN configuration	Utilize OT-SDN metrics to monitor for changes in expected behavior
Match on IP addresses and protocol	Reduces the labor cost to replace a failed protection device.	Some MITM attacks may be enabled with this configuration	For MITM attacks, look at the Sync warnings available in syslogs for indication that a switch has become desynchronized with the flow controller. Utilize the API to monitor specific flows for changes in expected behavior.
Match on IP and MAC addresses and protocol	Granular matching on physical port, source IP, source MAC, and destination port can be used for “ignore” rules that send expected but unwanted communication to a data store. While MITM attacks may be mitigated, the operational cost to	Matching on as many flow rules as possible increases security but will also increase operational costs.	Use SA tools to monitor the frequency and volume of data captured by the ignore rule.

Flow Rule Matching	Operational Use Case	Security Considerations	Potential Mitigations
	replace a failed field device is higher because field devices may not support modifying the MAC address. This limitation requires modifying and pushing updated flow rules to the network.		
Add temporal element	Engineering access Assured Compliance Assessment Solution (ACAS) Vulnerability Scanning (mandated by the UI.S. Department of Defense for certain systems)		Utilize OT-SDN metrics, potentially in combination with traffic mirroring, to monitor for changes in expected behavior

6.0 Methods to Establish Trust

Contents of this section was originally published as the report "Software-Defined Networks for Energy Delivery Systems: Identification of Methods to Establish Trust Between the Human, Hardware, and Software Components of Software-Defined Networks" in January 2020

6.1 Introduction

This section of the blueprint architecture identifies methods that can be used to establish trust between the human, hardware, and software components in an implementation of an SDN.

SDNs have dramatically changed the commissioning, management, and operation of network infrastructures used in EDSs. The SDN concept of separating the management function of the infrastructure (e.g., the management plane or control plane) from the data transport function (e.g., the data plane) in a network has introduced significant dynamic flexibility and control in how data frames are transmitted through the network.

Unlike traditional networks and switches, SDN networks and switches do not immediately start forwarding traffic as soon as they are connected. Rather, the individual switches must be adopted by the controller software, and individual flow rules must be applied to each physical port for each logical network flow in the network. These actions are performed in the separate control plane.

SDN flow rules control how frames are forwarded by the switches, but management of how the flow rules are established requires that a secure and trusted process be used, otherwise the flow rules could be modified to introduce unexpected or malicious behavior. Figure 6-2 (included in Section 6.3.1) provides a high-level overview of the trust relationships in an OT-SDN environment. To protect the configuration of an SDN network, the control plane interfaces must be trusted. Establishing this trust involves the following steps:

1. Trusting the human using the interface to configure the SDN flow rules (or that the flow rule modification requests are issued by an authorized application)
2. Trusting the SDN controller software accepts configuration commands from authorized sources, and processes them properly
3. Trusting the communication channel used to transmit the flow rules from the SDN controller software to the SDN switch
4. Trusting that the SDN switch will only allow control commands that modify the flow rules if they come from an authorized and trusted source.

6.2 Trust Requirements

Many implementations of SDN controller software rely on simple OS authentication and authorization to access the control software and provide for implicit trust of the control plane. This may make sense in a traditional IT SDN environment in which a robust access scheme exists for accessing the node hosting the controller software and the control plane is on a physically separate and controlled out-of-band control network is used. In this case, only authorized control software nodes are physically connected to a network that allows access to dedicated control plane network ports on the SDN switches.

However, in many OT-SDN environments, particularly those that span large geographic areas like electric power, the control plane may use in-band communications that utilize the same physical cables and hardware ports as the data plane. This requires additional levels of authentication, security, and trust used to access the control plane function on the data plane infrastructure.

Whether to choose an in-band or out-of-band control plane requires an analysis of several factors, including network performance, security, and availability of a separate communications infrastructure. If the communications requirements of the data plane are already stressing the bandwidth of the existing network, the addition of control plane traffic may result in decreased performance for both the data-plane and control-plane applications. While this may be unlikely in many networks, high-volume and high-speed data applications with minimal latency and jitter requirements may not be able to tolerate the additional non-deterministic nature of the control-plane traffic. In a large geographically diverse environment such as electric power transmission, the availability of separate network infrastructure for the control plane may present a challenge, especially at small very remote sites.

Obviously, the trust considerations between an in-band control plane and an out-of-band control plane are dramatically different. Since an out-of-band control plane uses a separate and isolated physical infrastructure for managing the network, it is less susceptible to compromise from rogue devices in the data plane, but a physical compromise of the control plane network could still occur. The out-of-band control plane network could either be an SDN in-band control plane network, or a traditional network (since an out-of-band SDN environment would require its own separate control plane infrastructure). Regardless of how it is implemented, an out-of-band control plane requires additional hardware and management support.

In either case, there is often implicit trust between the control software and the SDN switch. This is unacceptable in an OT environment; therefore, additional levels of trust are required. These additional trust requirements may lead to interoperability issues between different controller, application and switch implementations that must be addressed.

The diagram shown in Figure 6-1 provides an overview of the trust zones in an SDN environment. Five trust zones are depicted in the figure. These zones and relationships will be discussed in more detail in the following sections.

1. The human trust zone
2. The SDN controller NBI trust zone, including interfaces with third-party applications
3. The SDN control plane, which consists of interfaces from the SDN controller, as well as interfaces from third-party applications
4. The SDN switches themselves
5. The SDN data plane.

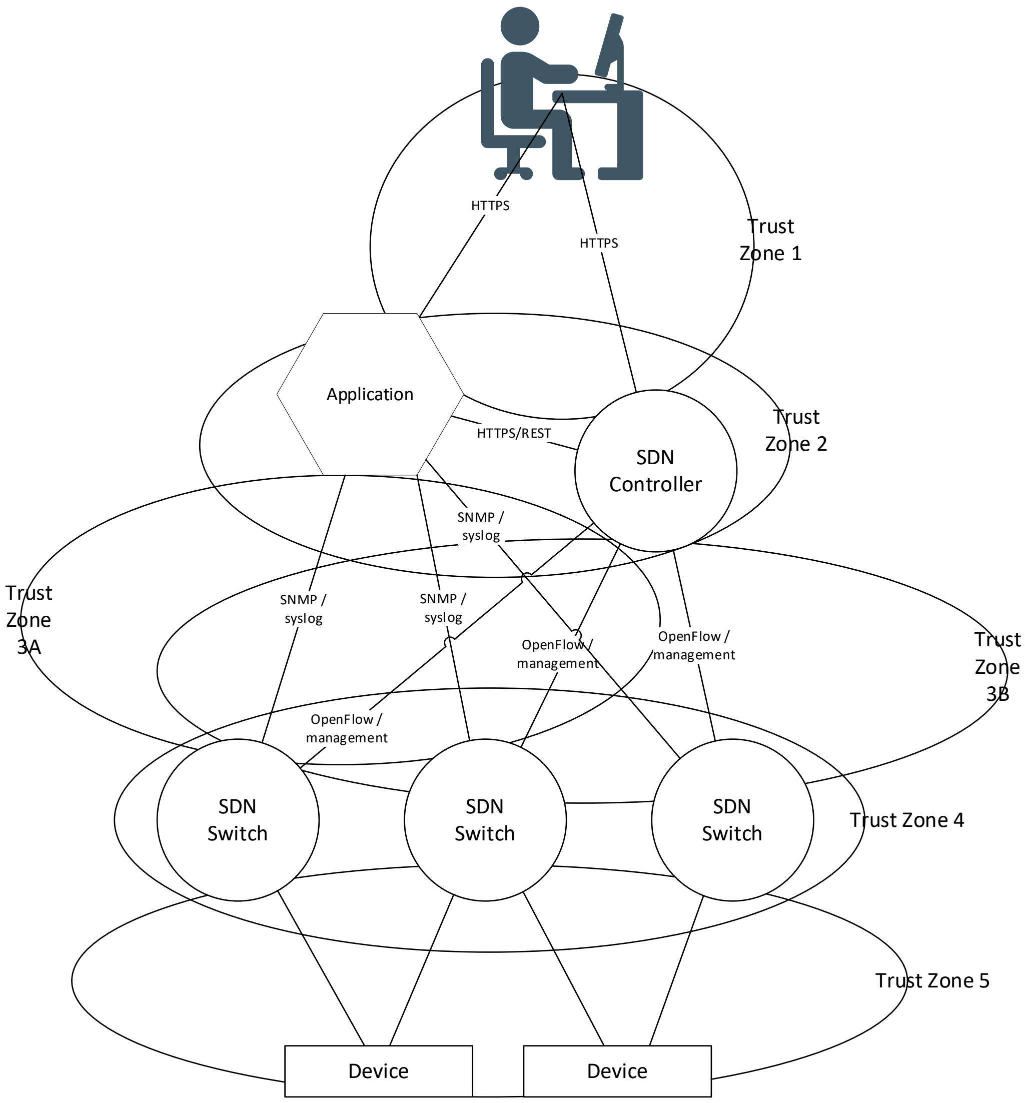


Figure 6-1. Trust Zones

6.2.1 Trusting the Human

The first trust zone relates to establishing trust between a human operator or engineer (i.e., the user) and the SDN environment. Human users are involved both in configurating the SDN environment and in monitoring its performance and health. Trusting the user generally involves authenticating each user and ensuring that the authenticated users are authorized to perform the configuration function. An individual user may be a valid (authenticated) user, but may be authorized to only monitor network statistics, or the user may be authorized to make changes to the SDN configuration.

Users can authenticate to the controller using a simple username and password, but a preferred approach would be to use a two-factor authentication system, such as an RSA SecureID token and a personal identification number.

Individual roles for a user should also be controlled, following a least-privilege model. Roles such as configuration, diagnosis, and monitoring are typical, with each requiring different levels of access to information and statistics and the ability to update the configuration or simply monitor the performance of the network. Additional roles may be required to maintain users and their associated roles and permissions. Existing role-based access systems such as Microsoft Active Directory, Remote Authentication Dial-In User Service, or Lightweight Directory Access Protocol (LDAP) are used often.

6.2.2 Trusting Northbound Interface Components

The second trust zone includes the management and monitoring interfaces that communicate with the SDN controller to configure or modify the software-defined data center environment. This would include some components of the human trust zone when using a human-machine interface to interact with the SDN controller but could also include an interface from a third-party application to provide additional monitoring, statistical analysis, or autonomously request configuration changes.

6.2.2.1 Trusting Applications

Like human users, application programs can interface with the SDN controller software to autonomously make configuration changes to the SDN switches. These changes are made as requests to the controller to update flow rules or other configurations on the SDN switches.

Applications may reside anywhere in the network. Some applications interface directly with the SDN controller to request configuration updates or query the OpenFlow statistics, while others may use a more traditional network operations interface such as SNMP to query port status and statistics for display by a converged network monitoring application.

Rather than applications issuing OpenFlow commands directly to SDN switches, they should interface with the SDN controller to request configuration changes. This practice allows the controller to maintain the complete and correct configuration for all the SDN switches.

Because applications operate autonomously, multi-factor authentication that requires human input cannot be used. Rather, most applications use a certificate-based authentication system to identify to the controller that they are authorized to request configuration changes. Protecting the keys from unauthorized access or misuse must be addressed by each application.

6.2.2.2 Trusting the Controller Software

Because the SDN controller is key to managing and monitoring the OT-SDN environment, care must be taken to ensure it is not maliciously compromised or corrupted. The operating environment (e.g., Microsoft Windows) in which the controller software runs should be hardened following the best practices for the specific environment and version. In addition, the controller software executable itself should be protected from tampering. Secure boot processes or run-time software verification processes can be used to help detect compromised software.

The controller software maintains a database of all flow rules for all SDN switches it manages. The database content must be protected from unauthorized access (i.e., to prevent access to configuration information by unauthorized users) and unauthorized modification (i.e., to prevent an improper configuration from being downloaded to an SDN switch during a synchronization operation).

The SDN controller software must protect the configuration data, generally by storing it in an encrypted datastore and only allowing access to it by authorized applications or users. This is especially important for situations in which the SDN controller is inactive and unattended allowing the OT-SDN environment to operate autonomously. Because the SDN controller configuration database contains all the flow rules, an adversary could access the flow rule and gain information about how the network is configured and what devices and protocols are in use, and then use the information to subvert the configuration. Similarly, the SDN configuration database could be modified so that when the SDN controller reconnects to an SDN switch, maliciously modified flow rules could be installed into the SDN switch, leading to subverting the network and possibly allowing an adversary access to the network.

The performance and security tradeoffs should be investigated to determine the impact of using encrypted data stores for archival purposes only, or whether the data store should be encrypted while in use.

The SDN controller also is responsible for managing updates to the SDN switch firmware. The SDN controller should verify the integrity and authenticity of any firmware before allowing it to be installed and use secure communication channels to download the firmware into the SDN switches. Firmware updates are not specified in the OpenFlow protocol, so the SDN controller’s management interface to the SDN switch is used to download and install firmware updates.

6.2.3 Trusting the Control Plane Communication Channel

The third trust zone comprises the interfaces from the SDN controller to the SDN switches primarily using the OpenFlow protocol and other management protocols (e.g., for collecting event information or updating firmware). Other applications may be used to directly query the SDN controller or the SDN switches for performance metrics using an SNMP interface or to collect event logs using a syslog interface.

The communications channel used for the control plane must be protected from malicious access and be initiated from a trusted source. Unauthorized writes to the control channel may result in the creation of malicious flow rules, leading to loss of node connectivity or possibly duplication of traffic to a monitoring port allowing data leakage from the data plane. Unauthorized monitoring (e.g., read access) of the control plane could result in leakage of performance or other SA data that could be used in future malicious modifications of the flow rules. Mutual authentication of both the controller and switches is desired.

Communications between the SDN controller and each SDN switch should be cryptographically protected by encrypting and authenticating the communications using unique cryptographic keys for each switch. These unique cryptographic keys should be generated when the SDN switch is commissioned to prevent default (and therefore well-known) keys from being used in the SDN environment. In this way, compromise of a single controller-to-switch exchange will not lead to a more wide-spread compromise of the entire control plane.

The OpenFlow 1.3 specification states that the data plane communication “… channel is usually encrypted using TLS but may be run directly over TCP.”[28] It is highly recommended that all control plane communications be encrypted and authenticated using TLS to minimize the ability for unauthorized users to configure the SDN switches or monitor the control plane traffic.

[28] See [OpenFlow 2012] pg. 21

Once provisioned, SDN switches periodically "call home" to the SDN controller that provisioned them using the OpenFlow protocol. SDN switches should only communicate with authorized SDN controllers using a logical communications channel that is encrypted and authenticated using TLS 1.3, This practice minimizes the ability for a rogue device to emulate the SDN controller application and install rogue flow rules to the SDN switches or perform other maintenance activity. All the traffic to the SDN controller also flows through the existing SDN flow rule processing to further minimize the ability for a rogue controller to interact with the SDN switch.

For communications with third-party applications, some protocols (e.g., older versions of SNMP and syslog) may not support encryption, so caution should be used when transmitting potentially sensitive data such as event records. This is especially true in an in-band management environment that converges the control and data planes onto a single infrastructure. Even in an out-of-band management environment that uses separate physical infrastructure for the control plane, a compromise of the control plane network could reveal information about the environment if the transmissions are not encrypted.

6.2.4 Trusting the SDN Switch

The fourth trust zone involves the SDN switches. The SDN switches are trusted because they can be modified only from valid and authorized control sources that present valid credentials to the configuration port on the switch. Unlike traditional Ethernet switches that can be configured by access from the unified data plane interfaces or via local configuration ports, SDN switches can be modified only from a configuration plane interface after presenting a valid configuration plane credential. This practice dramatically reduces the attack vector against the switch by eliminating rogue or compromised devices on the data plane from modifying the flow rules in the switch.

6.2.5 Trust in the Data Plane

The fifth trust zone is the data plane. The key trust advantage that an SDN environment provides over a traditional network environment is that, assuming a proper implementation of the SDN switch software, the data plane cannot be used to configure the SDN switches; therefore, a compromised node residing in the data plane cannot be used to modify the network configuration. This assumes that the control plane is either a physically separate network, or in an in-band management environment, the control pane communications are secured against unauthorized access (e.g., by encrypting the traffic, and authenticating each connection). If the control plane traffic cannot be secured (i.e., if unencrypted versions of management protocols must be used), properly configured SDN flow rules can mitigate unauthorized access to the traffic, but it may still be subject to observation (e.g., if the ethernet cables are physically tapped).

Note that any additional sensitive management traffic (e.g., SNMP and syslog messages from devices in the data plane) that may also be forwarded to the same monitoring applications as the SDN monitoring traffic is only protected from observation by SDN flow rules implemented in the SDN switches.

6.2.6 Additional Considerations

The introduction of the trust elements may introduce some interesting scalability and architectural issues.

For the SDN environment to function, there needs to be a "single source of truth" about all the configurations and flow rules. Flow rules need to be coordinated among various SDN switches along a logical path and allowing ad hoc modifications to the SDN configurations without the knowledge of the SDN controller should not be allowed. This results in the SDN controller (more specifically the configuration database in the SDN controller) becoming the single source of truth about the SDN environment. If multiple SDN controllers exist in an SDN environment, the configuration database must be synchronized across all the controllers to allow any one of them to be able to configure the entire network. This allows any SDN controller to be able to reload the configuration to an SDN switch that has become desynchronized with the SDN controller's database, adopt and replace a failed SDN switch, or add a new switch into the environment and integrate it and its flow rules with the existing SDN switches.

A key difference between IT-SDN and OT-SDN is that the OT-SDN switches do not require that the controller be active in order to function. This has the advantage of the SDN controller not being a single point-of-failure but requires that the SDN switches store their configuration (e.g., flow rules) in non-volatile memory so they are available after a power cycle. Traditional IT-SDN switches that do not store their configuration in non-volatile memory require and require that the controller reload them following a power cycle.

Because an SDN switch can be configured only from a single controller, that controller represents a single point-of-failure, albeit one that does not have an immediate impact to the network operations in an OT-SDN environment (because in an OT-SDN environment, the controller is not required for the network to operate). However, because the controller is the single "source of truth" about the flow rules for the entire SDN infrastructure, ensuring that the SDN flow controller configuration database is backed up and can be restored in the event of a hardware or software failure is essential to ongoing operations.

The integrity of the SDN controller's database also must be maintained. A compromise of the SDN controller's database represents a potential attack vector if the flow rules for switches can be compromised by manipulating the database. The SDN controller expects that its database is the single source of truth and would not be able to determine if it was compromised (assuming the contents of the database is properly formatted). The controller will assume that the database is correct moving forward.

In an environment containing multiple controllers in a clustered configuration[29], or a hierarchical controller structure (for example a "centralized" controller located at a control center, and a "distributed" controller instance located in a substation), the centralized controller may not necessarily be synchronized with any changes made by the distributed controller (i.e., a rule change made in response to a detected intrusion into the substation network). If a rule change is made by a distributed controller, it must be reflected in the central controller's database for it to remain the single source of truth.

However, if the distributed controller is used for monitoring and SA only, the central controller may still be the only source of truth because the distributed controller does not autonomously make any changes to the SDN switch configurations.

[29] Multiple controllers for reliability and load balancing were introduced in OpenFlow Version 1.4 (see [Open Flow 2013] section 6.3.4), but the SEL-5056 controller supports only OpenFlow Version 1.3. No publicly available OpenFlow specification discusses hierarchical controllers.

If multiple controllers are required for scalability reasons, a single database containing all the flow rules should be implemented so that any of the controllers can access the SDN flow controller database representing the source of truth for the network.

6.3 Trust in Software-Defined Network Implementations

Typical trust management is between the machine and the human owner or between machines. SDN introduces abstraction concepts that create new trust boundaries. These new trust boundaries must be integrated into the trust management design. For critical infrastructure implementations, trust management has the following goals:

- Protect the people, system, process, policies, and technology
- Default or initial state has no trust
- Require little management and educational burden on the end-user as possible
- Support long lifetimes
- Fail safe.

6.3.1 Details

SDN's abstraction creates new trust boundaries as functions that have typically been integrated into a single product that runs abstracted as a central piece of software (the SDN flow controller) that communicates management or control commands across a communication channel to the switches. This channel may be highly trusted (in the case of an out-of-band control plane) or indeterminately trusted with additional levels of security such as TLS authentication and encryption (in the case of an in-band control plane). This has created a hybrid trust management that demands the trust between the human and software (the SDN flow controller) to also carry over to the trust between the software to the switch (machine to machine).

SEL started the design for this hybrid trust management approach by documenting all the touch points used by the SDN technology. The solution includes everything the system owner needs to successfully configure, deploy, and maintain an SDN environment. The design shows the different functional blocks and the communications are happening between the function blocks. These functional block relationships are shown in Figure 6-2[30].

[30] The figure is adapted from Figure 7.1 of the SEL SEL-2740S Switch and SEL-5056 SDN Flow Controller Instruction Manual [SEL 2019] versions from 2019 or later

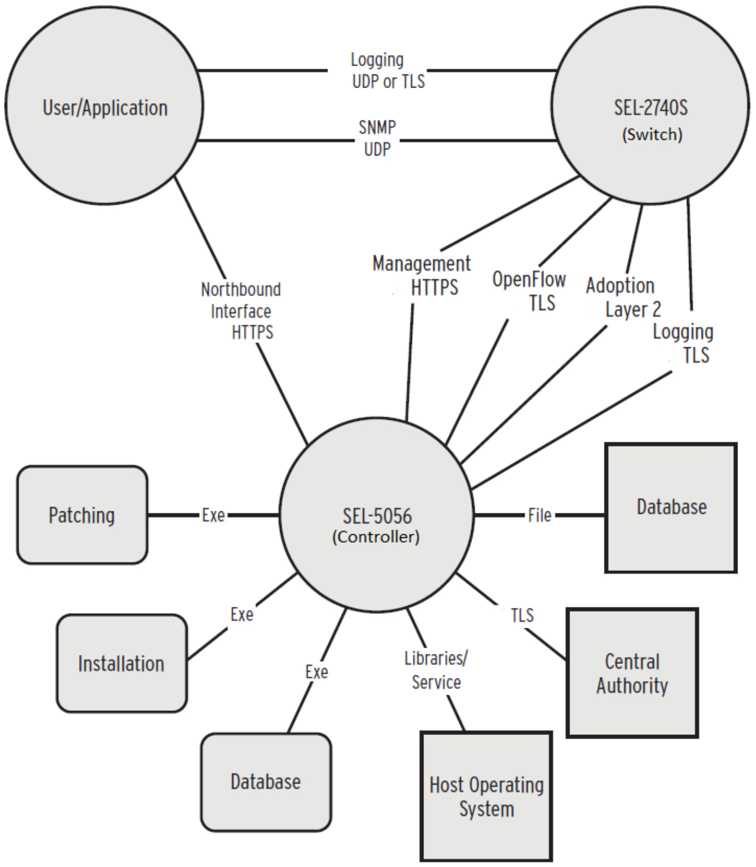


Figure 6-2. SEL-5056 and SEL-2740S Trust Relationships

When initially commissioned, the SEL-5056 SDN flow controller and all its components were digitally signed and run once by the end user after they had verified that the software's digital signature is from SEL. The database and OS are on the same machine as the SDN flow controller and are typically protected by hardening the operating environment. An additional security measure available for OT-SDN environments is that once all the switches are configured, an OT-SDN environment can operate without the SDN flow controller, allowing it to be disconnected from the operational network. The downside to disconnecting the SDN flow controller is a loss of SA.

The design used the REST API because of the trust management available with bearer tokens. These tokens provide scalable trust as the actions move between the different components of the system. The user interfaces with a browser that in turn interfaces with the SDN flow controller's REST API. This interface is scalable and allows multiple users to access the SDN flow controller at the same time. As long as the user can make a network connection to the SDN flow controller, they will be presented with an authentication landing page. Users can then authenticate using a local application-based authentication or can use common central authentication methods such as LDAP or Remote Authentication Dial-In User Service. These methods offload the authentication process to a central server where it can be used for other purposes and provides speed and ease of access.

The SDN flow controller's API is also used for scalable applications that perform actions for or on behalf of the user. The trust between the SDN flow controller and the applications can be through either user credentials or token-based trust, where the exchange of certificates is performed when registering the application with the SDN flow controller for the first time. Because certificates are used, there is a time element that must be managed by ensuring the computer's time is set and synchronized with the node hosting the application, and the certificate management authentication process is performed before the certificate expires. The advantage to this approach is that the application can perform actions without user credentials.

The SDN flow controller establishes mutual trust with the switch through certificates as well. Both the controller and the switch, when commissioned, exchange certificates, and when communications are established, both sides authenticate each other and encrypt and authenticate every packet of the communications. SEL sets the certificates for a 10-year lifetime to support the longer lifetimes of the critical infrastructure, but certificates can be updated at any time.

Thus, this design has trust established at every boundary of the system but requires the user to manage certificates and user accounts (i.e., typically usernames and passwords) with group membership for privilege management.

The challenges with this design include:

- The individual user accounts must be maintained and updated as staff are hired, terminate, or change jobs.
- Users manage their own passwords.
- The use of certificates demand the system owner either accept self-signed certificates or stand up their own certificate authority and make sure those certificates are updated before they expire.

The benefits of the design include:

1. Very little training is required for users.
2. Once commissioned and the certificate exchange is completed at first registration, trust between machines is transparent to the users.
3. All inputs and outputs that travel across untrusted communication channels are mutually authenticated and encrypted.

SEL and PNNL designed the OT-SDN technology to support proactive traffic engineering by maintaining the switch flow rules through a power cycle. This means if the trust management fails, the SDN infrastructure "fails safe" with the switches continuing to forward traffic on the data plane based on the most recent flow rules installed.

6.3.2 Conclusion

SDN has more complicated trust management because of its abstraction architecture, but there are solutions that minimize burden on the system owner and provide security at all touch points. Mutual authentication and encryption are important to keep the data and control authenticated, authorized, and confidential. Availability of the control plane is less important if the solution is designed with proactive and fail-safe traffic engineering. Parts of the system can be taken offline if cybersecurity concerns warrant (i.e., the SDN flow controller can be taken offline after the switches are configured). The flexibility inherent in the SDN ecosystem provides a scalable and easy-to-use design that has been proven through its use by Google, Facebook, and Amazon.

7.0 Metrics and Existing Gap Areas

Contents of this section was initially published as the report "Software Developed Networks for Energy Delivery Systems: Metrics and Existing Gap Areas" in May 2019

7.1 Introduction

This section of the blueprint architecture summarizes the metrics that pertain to SDN irrespective of, whether specified in the OpenFlow standard or in traditional networking environment. Also identified are tools that can be used to collect those metrics. Gaps in the capabilities of those tools and gaps between metrics that are applicable to traditional networking versus those that are only applicable to SDN are also noted.

Network monitoring is vital to any operational network and can help identify expected and unexpected traffic behavior on the network. This is especially crucial in terms of network security where the metrics gathered allow operators to identify and address the sources of malicious traffic. The centralized nature of an SDN provides the benefit of a global view of the network by enabling a higher degree of SA such as identifying existing nodes on the networks, traffic flow characteristics between the nodes, etc. Improved SA is accomplished by monitoring end-to-end flows, or "conversations," rather than simply monitoring frame flows from a single switch.

SDNs decouple the systems that decide where the traffic is sent (i.e., the control plane) from the systems that perform the forwarding of the traffic in the network (i.e., the data plane). An SDN controller, or a cluster of controllers, is responsible for managing the flow control to the devices below and integrates with applications such as load balancers and firewalls, above. The SDN controller provides NBI, via NBI APIs, to external applications. A SBI of the controller, via SBI APIs, is used to manage the data plane of the forwarding network devices below. OpenFlow protocol-based controller is the most common SDN controller in use.

The SEL 5056 SDN controller[31] used in this project provides many of the metrics defined in the OpenFlow protocol, such as raw packet and byte counts of a particular traffic flow and traditional switch port statistics. The controller's NBI (control-plane) interface also can be leveraged through the development of an application using the REST API to further process metrics particular to their use case. Other vendors may choose to implement SDN differently than SEL which may affect available metrics.

The goal of this document is to identify both existing and potential metrics of interest that would be applicable and appropriate to an SDN network. These metrics should contribute to the SA capabilities that SDN offers and should allow for functions such as detecting link and device failures while pinpointing nodes in which reconfigurations may be needed. Additionally, metrics should aid in the process of network monitoring for performance and potential network anomalies. By ingesting this data into a Security Information and Event Management (SIEM) system, such as Splunk, or through a NBI application, it is possible to alert operators of significant events and potentially automate the appropriate responses by blocking or re-routing suspect traffic.

[31] See https://selinc.com/products/5056/ (Accessed September 17, 2021)

7.2 Available Metrics

The OpenFlow protocol provides methods for collecting a variety of metrics on both network data plane and control plane. OpenFlow is arguably the most common implementation of SDN, therefore we reviewed tools pertaining to the OpenFlow protocol that may not be applicable for other SDN solutions.

The SEL 5056 is an SDN controller that is specifically designed for OT networks and uses OpenFlow 1.3 as its SBI protocol. Due to its use of the OpenFlow protocol, this implies that metrics available for retrieval by the protocol also will be available in the SEL 5056. This includes data such as flow statistics, group statistics, meter statistics, and port statistics among others.

The metrics described in this section are grouped into three categories. The first category contains metrics defined in the OpenFlow 1.3 specification. All metrics described in the OpenFlow 1.3 specification should be available from all controllers supporting the 1.3 version of the protocol.

The second set of metrics uses the standard SNMP interface. The SEL 2740S switch used in the reference architecture only supports the interface management information base (MIB) – IF MIB. Other switches may support different SNMP blocks, including the IF MIB, or may not support SNMP at all. The metrics available from the SEL 2704S switch are described in Section 7.2.3.

The third set of metrics are available from the REST API. Metrics available from this interface will likely be vendor (and perhaps product) dependent. The metrics available from the SEL 5056 SDN controller are described in Section 7.2.4.

7.2.1 OpenFlow 1.3 Counter Metrics

Counters for certain metrics are described in the OpenFlow 1.3 specification and are often available via the controller's NBI (i.e., its REST API). They contain counts for fields pertaining to flow rules or port statistics to name a few. Counters are unsigned values that wrap around and have no overflow indicator. Additional information on the data structures containing the counters is available in the OpenFlow specification [OpenFlow 2014]. The following are the available counters, separated by category.

Per-Flow Table

- *Reference Count* – Number of active (table) entries
- *Packet Lookups* – Number of packets looked up in table
- *Packet Matches* – Number of packets that hit table.

Per-Flow Entry

- Received Packets – Packet count in flow
- Received Bytes – Byte count in flow
- Duration (Seconds) – Time flow has been alive in seconds
- Duration (Nanoseconds) – Time flow has been alive in nanoseconds.

Note: Additional information such as the Flow Instructions and Match fields can be extracted from the OpenFlow message structure, OFPMP_FLOW.

Table miss flow entries are flow rules that will process unmatched packets in a flow table. These may be configured to increment a counter whenever they are triggered, allowing metrics to be gathered for packets that do not match any other entries, and therefore represent unexpected or unauthorized traffic.

Per-Port

- *Received Packets* – Number of received packets
- *Transmitted Packets* – Number of transmitted packets
- *Received Bytes* – Number of received bytes
- *Transmitted Bytes* – Number of transmitted bytes
- *Receive Drops* – Number of packets dropped by Received
- *Transmit Drops* – Number of packets dropped by Transmit
- *Receive Errors* – Number of receive errors, super-set of more specific receive errors and should be greater than or equal to the sum of all Receive Error values
- *Transmit Errors* – Number of transmit errors, super-set of more specific transmit errors and should be greater than or equal to the sum of all Transmit Error values (None currently defined)
- *Receive Frame Alignment Errors* – Number of frame alignment errors
- *Receive Overrun Errors* – Number of packets with Receive overruns
- *Receive Cyclic Redundancy Check Errors* – Number of Cyclic Redundancy Check errors
- *Collisions* – Number of collisions
- *Duration (seconds)* – Time port has been alive in seconds
- *Duration (nanoseconds)* – Time port has been alive in nanoseconds beyond Duration (Seconds).

Per-Queue

- *Transmitted Bytes* – Number of transmitted bytes
- *Transmitted Packets* – Number of transmitted packets
- *Transmit Overrun Errors* – Number of packets dropped due to overrun
- *Duration (seconds)* – Time queue has been alive in seconds
- *Duration (nanoseconds)* – Time queue has been alive in nanoseconds beyond Duration (seconds).

Per-Group

- *Reference Count* – Number of flows or groups that directly forwarded to this group
- *Packet Count* – Number of packets processed by group

- *Byte Count* – Number of bytes processed by group
- *Duration (seconds)* – Time group has been alive in seconds
- *Duration (nanoseconds)* – Time group has been alive in nanoseconds beyond Duration (seconds).

Per-Group Bucket

- *Packet Count* – Number of packets processed by bucket
- *Byte Count* – Number of bytes processed by bucket.

Per-Meter

- *Flow Count* – Number of flows bound to meter
- *Input Packet Count* – Number of packets in input
- *Input Byte Count* – Number of bytes in input
- *Duration (seconds)* – Time meter has been alive in seconds
- *Duration (nanoseconds)* – Time meter has been alive in nanoseconds beyond Duration (seconds).

Per-Meter Band

- *In Band Packet Count* – Number of packets in band
- *In Band Byte Count* – Number of bytes in band.

7.2.2 Available Metrics in OpenFlow 1.4 and Future Versions

There are several differences in the metrics data structures between OpenFlow 1.3 and OpenFlow 1.4; not all of them appear to be backwards compatible. The equipment used in this blueprint reference architecture currently only supports OpenFlow 1.3. However, the collection and use of OpenFlow metrics will need to be revisited if equipment using OpenFlow 1.4 or newer is used.

7.2.3 SNMP Available Management Plane Metrics

SNMP is a core IP standard protocol used for collecting and managing network devices in an IP network. SNMP organizes the data it manages in a MIB that is used to describe network device configuration and provide status and metrics. The structure of many SNMP MIBs is standardized by the Internet Engineering Task Force in several RFC documents (the name given to documents used by the task force to define protocols for the internet), including RFC 1155 [Rose 1990], RFC 1213 [McCloghrie 1991], and RFC 1157 [Case 1990].

If an OpenFlow switch also supports SNMP as a management protocol, metrics related to the diagnostics of the switch would be available for collection and processing. Currently, the only known OT-SDN switch used commercially is the SEL-2740S, which supports the SNMP IF MIB (ports, traffic up, traffic down…). Specific details of the IF MIB are available in [McCloghrie 2000]. Currently, should the switch be power cycled, there is no guarantee that the same counter will be found at the same index; however, using the *ifDescr* field as a reference should allow metrics to be correlated across power cycle events.

As noted in the SEL 5056 reference manual[32]:

> The port status and other port information and diagnostics can be accessed remotely though the ifTable (1.3.6.1.2.1.2.2) in the IF MIB. *Table F.1* shows the relationship between the back-port number and the ifDescr of the ifTable. The ifIndex should not be used as the ifIndex relationship because port number is dynamic and may change. If the module is not present, then the corresponding ports do not appear in the ifTable.

7.2.4 REST API Query Metrics

The REST API is a mechanism used to create web services. Web services that conform to the REST architectural style, are called *RESTful* Web services. Similar to SNMP MIBs, information available in a REST API is called a resource. Nearly any information that can be named can be a resource—a document or image, a temporal service, a collection of other resources, a non-virtual object (e.g., a person), etc. REST uses a resource identifier to identify the particular resource involved in an interaction between components.

The SEL 5056 SDN controller uses a REST API to connect the web-based configuration tool to the core controller software [SEL 2020]. In addition to configuration actions, metrics can be extracted using the REST API.

The following diagnostic resources are available from the SEL 5056 controller:

- module
- modulePort
- coProcessorStatistics
- unsortedDiagnostics
- powerSupply
- hmi
- systemInformation
- disk
- qualityCounter
- ubi
- fileHandles
- memory
- cpuLoad
- realTimeClock
- ntp
- frontPort
- transactions

[32] See [SEL 2019] Appendix F

- properties.

7.2.5 SYSLOG Messages

While not specifically considered metrics, the SEL 5056 controller can collect syslog event messages for a number of OpenFlow events from SEL 2704S switches. According to the SEL 2704S manual, the supported syslog messages are:

- Flow entry added
- Flow Entry deleted
- Flow entry modified
- Flow entry deleted because of time-out
- Group entry added
- Group entry deleted
- Group entry modified
- Meter entry added
- Meter entry deleted
- Meter entry modified
- OpenFlow port added
- OpenFlow port deleted
- OpenFlow port modified.

Other switches may support a different set of syslog message types.

7.3 Example Existing Metric Use Cases

This section includes OpenFlow data and traditional networking use cases that can generate more elaborate metrics for consumption.

7.3.1 OpenFlow Counter Use Cases

7.3.1.1 Measuring Packet Drops

There are three available metrics that count packet drops: 1) Receive Drops, 2) Transmit Drops (from Per-Port statistics), and 3) Tx Overrun Errors, which counts the number of packet drops due to overrun (from Per-Queue statistics). It also is possible to count bytes or packets dropped by the OpenFlow switch at a per-flow level by creating a flow rule specific to the type of packets to be dropped.

7.3.1.2 Bytes or Packets per Second, per Minute, or per Hour

Counters per-flow entry monitor flow volume by counting the number of packets or bytes over a specified amount of time. These data can be graphed over time through periodic polling of the controller. These counters also are available on a per-port basis.

7.3.1.3 Network Utilization

The number of transmitted and received bytes per-second described in section 7.3.1.2 for each physical port can be summed, and the sum divided by the speed of the link, which is available from the SNMP ifSpeed element of the IF MIB for each port, provides an indication of network utilization as a percentage of total capacity.

7.3.2 Control-Plane Monitoring

Metrics involving the performance of the control plane of SDN include number or volume of SBI messages, flow rule add time, flow rule update time, and processing time. The intent is to be able to measure performance of SBI communications between switches and controllers. Currently, collecting control-plane metrics via the controller's NBI is not standard practice. The controllers that belong in this category include the SEL 5056, Ryu, Floodlight, and Open Daylight (the latter three being open source). Open Network Operating System (ONOS) is another open-source SDN controller that can run a NBI application called the Control Plane Management Application, which enables the monitoring of control-plane metrics including statistical information for six types of OpenFlow messages.

Widely accepted and used benchmarking tools used by the wider SDN community do exist for these measurements; however, they are several years old.

- *OFLOPS* – OpenFlow Switch benchmarker. "Special" controller that sends and receives messages to or from an OpenFlow switch to characterize its performance and observes responses from the switch.
- *Cbench* – OpenFlow Controller benchmarker. Emulates a variable number of switches, sends PACKET_IN messages to the controller under test from the switches, and observes responses from the controller.

The primary concern with these two tools is that they are intended for testing an OpenFlow implementation (switch and controller) and do not seem adequate for control-plane performance monitoring of a live SDN network.

However, because control-plane communications are implemented as flows by the switch, flow rules and counters can be used to gather metrics associated with control-plane communications using the same tools and techniques as are used for the data plane.

7.3.3 Malformed Packets

OpenFlow matches are based on the packet headers (L2–L4), so deeper analysis would be needed to find malformed packets, a feature that is not available in OpenFlow natively. A potential approach would be to use analyzers such as Wireshark or Snort In-Line to capture instances of malformed packets and obtain metrics based on those counts. Potentially, another option would be to build an application that can inspect packets if they are sent to the controller from the switch. This approach would take additional processing time and introduce control-plane latency that could affect performance.

7.3.4 Clustering

OpenFlow 1.3 has no features that provides the capability of monitoring flow rules of switches being managed by multiple controllers other than extracting performance statistics from the multiple controllers and monitoring them outside of any specific controller. This capability is specified in OpenFlow 1.4 via the Flow Monitor message that monitors changes for a subset of flows in a flow table. Monitoring is done on the switch side—anytime a change is made an event is sent to the controller. It is primarily used to detect changes to a flow made by another controller in a multi-controller environment. Unless clustering is critical for OT-SDN networks, this gap is expected to be minor.

7.4 Metrics of Interest

To properly monitor and manage a network, it is important to understand how it should perform. Network metrics provide the raw data needed for SA to assess network performance, identify bottlenecks, and plan for capacity expansion. Network metrics also can be used to identify and monitor traffic patterns and flows, diagnose connection and performance problems, and spot anomalous or rogue behavior. This is especially true in an SDN environment because flows put the packets in context of the conversations to which they belong, thereby quickly giving the operator an understanding of who is participating in a conversation and the content of the conversation.

Conventional networks have a control plane, management plane, and a data plane. In SDN, the management plane is a subset of the control plane. Thus, metrics for the control plane and the data plane are important and should be maintained separately.

Metrics can generally have several different uses, including monitoring performance and behavior. Performance monitoring includes metrics such as number of frames sent or received, number of bytes sent or received, number of dropped packets, and CPU usage on the switch. Behavior monitoring includes the number of invalid packets, changes in expected packet flow rates or volume, and the number of flow rule changes.

Metrics should be periodically captured and stored as a time series, which could be used for a SA graphical display or behavior analysis. Additional analytics can be programmed using data collected from switches and controller(s). For example, upper and lower thresholds can be established for each metric, and an alarm can be generated if the metric drifts from its normal range. While most of the metrics apply connections and flows in the data plane, similar metrics for the control plane can be captured and analyzed.

The following metrics have been identified as potentially useful for monitoring SDN performance and behavior and providing SA for anomalous or rogue behavior.

- *Number of frames sent* – This metric is tracked on a per-port, per-flow, per-meter, per-group, or per-table basis, and can be used to assess the performance of the switch and the network. This metric should be tracked for both the data plane (east-west communication) and the control plane (north-south communication).
- *Number of frames received* – This metric is tracked on a per-port, per-flow, per-meter, per-group, or per-table basis, and can be used to assess network performance, specifically tracking whether the network is approaching a capacity limit. This metric should be tracked for both the data plane (east-west communication) and the control plane (north-south communication).
- *Number of bytes sent* – This metric is tracked on a per-port, per-flow, per-meter, per-group, or per-table basis, and can be used to assess the performance of the switch and the network. It should be tracked for both the data plane (east-west communication) and the control plane (north-south communication).

- *Number of bytes received* – This metrics is tracked on a per-port, per-flow, per-meter, per-group, or per-table basis, and can be used to assess the performance of the network, specifically tracking whether the network is approaching a capacity limit. It should be tracked for both the data plane (east-west communication) and the control plane (north-south communication).
- *Number of transmit errors* – This metric typically is tracked on a per-port basis but could also be tracked on a per-flow, per-meter, per-group, or per-table basis. It is used to track the number of transmit errors detected during frame transmission. Transmit errors generally are an indication of hardware failure somewhere on the physical link.
- *Number of receive errors* – This metric typically is tracked on a per-port basis, but also could be tracked on a per-flow, per-meter, per-group, or per-table basis. It is used to track the number of transmit errors detected during frame receipt. Receive errors generally are an indication of hardware failure somewhere on the physical link.
- *Use of a priority queue* – This metric is used to ensure higher priority packets are processed before those with lower priority.
- *Number of unique endpoints in the network* – In an OT environment, this metric normally should not change. It can be tracked by the endpoint MAC address or the endpoint IP address. Any change should be associated with a new function or a maintenance action.
- *Number of conversations* – This metric identifies the number of unique send or receive endpoint pairs in the network. In an OT environment, this metric normally should not change. Any change should be associated with a new function or a maintenance action.
- *Number of frames sent per conversations* – This metric shows the number of packets transmitted in either direction between the two endpoints of conversations. It can be used to establish a baseline against which future communication is assessed to determine network behavior changes.
- *Number of bytes sent per conversations* – This metric shows the number of bytes transmitted in either direction between the two endpoints of conversations. It can be used to establish a baseline against which future communication is assessed to determine network behavior changes.
- *Top endpoint senders by packet count* – This metric assesses all endpoints in the network and tracks them in order of the number of packets transmitted by the endpoint.
- *Top endpoint senders by byte count* – This metric assesses all endpoints in the network and tracks them in order of the number of bytes transmitted by the endpoint.
- *Top endpoint receivers by packet count* – This metric assesses all endpoints in the network and tracks them in order of the number of packets received by the endpoint.
- *Top endpoint receivers by byte count* – This metric assesses all endpoints in the network and tracks them in order of the number of bytes received by the endpoint.
- *Number of frames that match a flow rule* – This metric is tracked on a per-port and per-flow rule basis and represents the amount of valid traffic received on a port.
- *Number of frames that fail a flow rule* – This metric is tracked on a per-port basis and represents the amount of invalid traffic detected and rejected by the SDN switch. If available, tracking the number of mismatched fields in flow rule also would be useful.

- *Number of incorrect MAC or ARP responses* – This metric tracks the number of spoofed, invalid, or unexpected MAC addresses detected. This could be associated with equipment swap outs, and therefore a valid different MAC address, or could indicate a rogue device plugged into an active port.
- *Number of link-up events* – This metric is the number of times an inactive port becomes active. If the port is normally not active, this represents a new device being plugged into the network.
- *Number of link-down events* – This metric is the number of times an active port becomes inactive. The reason might be loss of power by the end-node device or a cable being unplugged. Note that a link-down followed by a link-up with a different MAC address may represent an equipment swap or may represent a rogue device masquerading as the actual device.
- *Number of frames sent where flows do not exist* – In an OT-SDN environment, this metric should not exist, but may be an artifact of a hybrid network.
- *Switch CPU usage* – This metric tracks the CPU utilization for each switch. Changes in CPU utilization that cannot be directly tied to increased traffic flows or the introduction of new flow rules may indicate a compromised switch.
- *Switch memory usage* – This metric tracks the memory utilization for each switch. Changes in memory utilization that cannot be directly tied to increased traffic flows or the introduction of new flow rules may indicate a compromised switch.
- *Controller CPU usage* – This metric tracks the CPU utilization of the SDN flow controller. If any additional computers are used in analytic processing, their CPU usage should be monitored as well.
- *Controller memory usage* – This metric tracks the memory utilization of the SDN flow controller. If any additional computers are used in analytic processing, their CPU usage should be monitored as well.
- *Number of flow rule changes* – This metric tracks the number of changes made to the flow tables. Each switch should track changes made on itself, and the controller(s) should track the number of changes made there (whether the changes are made manually or under program control). The two should match.
- *Frequency of flow rule changes* – This metric tracks how often flow rules are changed.
- *Date and time of flow rule change detection* – This metric tracks when flow rules are changed.
- *Switch up event* – This metric can be detected by the controller, or possibly could be inferred by correlating “link-up” events from other connected switches.
- *Switch down event* – This metric can be detected by the controller, or possibly could be inferred by correlating “link-down” events from other connected switches.

7.5 Conclusions

The metrics available in an SDN network generally are comparable to those in traditional networking but with the added advantage of flow-context metrics. There is a gap in terms of available and up to date tools that can take advantage of metrics in an SDN network. Tools that are available to measure statistics are not inherently available in the OpenFlow protocol are mostly intended for benchmarking during controller development rather than monitoring production environment. It should also be noted that most of the SDN-specific tools have been developed for OpenFlow implementations. It is possible that an SDN implementation using some other SBI protocol may not be properly addressed by some of the listed tools.

Some tools are research-based and have not been used in operational SDN networks. Additionally, the OpenFlow metrics available are primarily raw count data (e.g., packets and byte count) that must be processed to yield additional and meaningful information. Most of these gaps can potentially be alleviated through further development of NBI applications that capable of leveraging data from the controller. The trade-off of this is additional throughput and processing time that is created in NBI communications which, depending on the use case and application, may not be viable.

A tool that integrates information from traditional network monitoring functions (i.e., metrics available from a traditional SNMP interface) with the new information available from SDN OpenFlow metrics to provide a single holistic view would be desirable and considered for future development. This tool would be beneficial for monitoring both traditional or SDN hybrid networks and native SDN networks.

8.0 Defining Desired System Protocol Behavior

Contents of this section was initially published in the report "Software-Defined Networks for Energy Delivery Systems: Methods to Define Desired System Protocol Behavior" in May 2020

8.1 Introduction

This section of the blueprint architecture is to review selected user experiences in implementing SDN environments and to propose methods for defining and enforcing behaviors of EDS protocols in an SDN.

Current network environments can be broadly separated by their support of IT or OT. IT networks often include campus networks that rely on self-discovery and automatic provisioning of services for connected clients, while OT applies to networks of control systems that have much stricter requirements for client's connectivity.

In traditional Ethernet-based IT networks, network access control (NAC) typically is implemented using the IEEE 802.1x framework. SDN improves upon traditional NAC (without actually using IEEE 802.1x) through its deny-by-default approach and strict flow rules. These new controls rely on packet attributes from OSI layer 1 (physical port), OSI layer 2 (data link), OSI layer 3 (network; e.g., IP), and OSI layer 4 (transport, e.g., UDP or TCP).

An SDN designed to meet requirements of OT—such as maintaining flow rules through a power cycle, autonomous restart, and autonomous operation—is referred to as OT-SDN. OT-SDN also provides detailed meters on packet count and byte counts per network flow, group, meter, port, and action bucket. With PTP [IEEE 1588] sub-microsecond accuracy for time-stamped events can be achieved for better visibility and context.

To provide the resilient operation and low latency required in OT environments, the SDN switch equipment only inspects OSI layers 1 through 4; however, when combined with management plane-based intelligent application support, it can process data in all OSI layers, including layer 7 (application; e.g., DNP3) containing EDS specific payloads.

When we look at how we can apply these new security controls to SCADA systems, combined with user experiences in implementing or using SDN provided by the SDN4EDS project partners and PNNL, some very exciting and new opportunities for increased protocol inspection and behavior enforcement can be identified.

This report provides an overview of user experiences and expectations and also details use cases for further research in the SDN4EDS project for EDS protocol enforcement.

8.2 User Experiences

The information describing user experiences was collected from project partner staff who have worked with SDN technologies within laboratory environments and operational environments. We also included information from end users who have deployed SDN within operational environments. Both positive and negative experiences were reported on the SDN equipment.

While the SDN4EDS project is primarily focused on the use of an SDN environment from SEL comprising their SEL-5056 SDN flow controller and SEL-2740S SDN Switch, we also documented experience reported from users of a broader range of equipment from different manufacturers.

The SDN devices include:

- SDN flow controllers
  1. Ryu – An open-source SDN flow controller with a Python codebase.[33]
  2. OpenDaylight – An open-source SDN flow controller with a Java codebase.[34]
  3. SEL-5056 – A freely available SDN flow controller produced by SEL.[35]
  4. Faucet – An open-source SDN flow controller environment, derived from Ryu.[36]
  5. Big Switch SDN flow controller – A commercial SDN flow controller for data centers.[37]
  6. ONOS – An open-source SDN flow controller written in Java and running within a Java Virtual Machine.[38]
- SDN Switches
  1. Open vSwitch (OvS) – An open-source SDN software switch that can be deployed without specialized hardware.[39]
  2. SEL-2740S – A commercial SDN hardware switch that is produced by SEL and targeted to be deployed within OT environments.[40]
  3. Aruba 2920 Hewlett Packard Enterprise – A commercial SDN hardware switch that is produced by Hewlett Packard and targeted to be deployed within IT environments[41].
  4. Aruba Hewlett Packard Enterprise 2530 Managed Switch – A commercial IT or enterprise SDN switch.[42]
  5. Dell S6000-ON – A commercial data-center-grade SDN-capable switch.[43]
  6. PICA8 Pronto 3290 – A commercial IT SDN-capable switch, based on the PICOS network operating system (NOS).[44]
  7. Allied Telesis IE210L-10GP Industrial Gigabit Ethernet Switch – A commercial industrial Ethernet switch that supports both traditional and SDN switching technology.[45]

[33] https://github.com/faucetsdn/ryu (Accessed September 17, 2021)
[34] https://www.opendaylight.org (Accessed September 17, 2021)
[35] https://selinc.com/products/5056/ (Accessed September 17, 2021)
[36] https://faucet.nz/ (Accessed September 17, 2021)
[37] https://www.bigswitch.com/ (no longer available)
[38] https://www.opennetworking.org/onos/ (Accessed September 17, 2021)
[39] https://www.openvswitch.org (Accessed September 17, 2021)
[40] https://selinc.com/products/2740S/ (Accessed September 17, 2021)
[41] https://www.arubanetworks.com/assets/ds/DS_2920SwitchSeries.pdf (Accessed September 17, 2021)
[42] https://www.arubanetworks.com/assets/ds/DS_2530SwitchSeries.pdf (Accessed September 17, 2021)
[43] https://i.dell.com/sites/csdocuments/Shared-Content_data-Sheets_Documents/en/Dell_Networking_S6000_ON_Spec_Sheet.pdf (Accessed September 17, 2021)
[44] https://www.pica8.com/ (Accessed September 17, 2021)
[45] https://www.alliedtelesis.com/en/datasheet/ie210l-series (Accessed September 17, 2021)

From the SDN flow controllers and switches listed above, information from user experience was gathered and documented (with technologies associated with those comments listed where applicable). Positive experiences of deploying SDN include the following:

- Modeling and simulation environments can easily be created and integrated in either stand-alone or operational networks. OvS was used within the modeling and simulation environment.
- Only a minimal set of software tools is needed to configure and install SDN flow rules. The OvS *ovs-ofctl* program was used to manage flow rules and SDN switches.
- The environment can be configured and managed with various SDN flow controllers. OpenDaylight and Ryu were confirmed to work with all switches. The SEL SDN flow controller was only tested with the SEL-2740S and Allied Telesis IE210 switch(es).
- SDN environments can be used seamlessly with multiple virtualization software packages and technologies (e.g., QEMU,[46] Docker,[47] Mininet, OvS) to create or simulate large network environments.
- Endpoints and VLANs can easily be reconfigured within an SDN environment (e.g., OvS).
- SDN infrastructures and configurations are easy to document, tear down, modify on the fly, and restore when needed (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056). This is especially true when automation is added; initial setup of bridges, ports, VLANs, etc. can be scripted across an entire experiment network.
- Network errors are straightforward to troubleshoot; network flow rule misses can be logged to help identify missing rules (e.g., Ryu, OpenDaylight, SEL-5056).
- Advanced switch functionalities can be supported (e.g., port mirroring, tunneling) by configuring additional network flow rules (e.g., OvS, SEL-2740S, Aruba 2920).
- Network topology is easy to visualize using SDN flow controller software. Topology information can include connections, network flows, and node information (i.e., IP address, MAC address). The OpenDaylight, ONOS, and SEL SDN flow controllers have visual front-ends.
- The network can be managed either centrally or in a distributed manner. OpenDaylight and ONOS were demonstrated to work with clustering a set of SDN flow controllers. The SEL-5056 SDN flow controller was not tested for clustering.
- SDN provides fine-grained access control of network flows. Network flows can be specified based on the physical port on the switch, source and destination MAC addresses, source and destination IP addresses, and protocol. SDN flows can be configured and activated or deactivated based on time windows (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Network flow rules and configuration can be changed in real-time with minimal impact on existing or on-going network flows. All switches were tested for flow-updates in real-time (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Network flow rules can be installed into SDN switches proactively or reactively. All switches have been tested for the flow installation mode (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).

[46] https://www.qemu.org/ (Accessed September 17, 2021)
[47] https://www.docker.com/ (Accessed September 17, 2021)

- Logical networks can be created from existing physical components. Logical networks that shared physical components can remain isolated from each other (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Deny-by-default is a powerful security protection against rogue devices and network flows (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Failover between SDN switches can be fast when preinstalled network flow rules are configured (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- SDN can be largely transparent to end users and end-use applications once installed (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056). Most applications only need to know the ‘true’ endpoint address and therefore the flows working at the SDN layer are transparent.
- Network operators must know their network and data flows in order to configure an SDN environment when using proactive flow installation mode. This is beneficial for network administrators to identify and understand all devices communicating on their networks (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Predefined network flows in network topologies greatly reduces convergence times often associated with common network protocols (e.g., BGP, Open Shortest Path First, etc.) (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Multiple SDN flow controllers may be leveraged to provide resiliency (“hot-swap”) (e.g., SEL-5056).
- Can easily support heterogeneous switching or routing devices as long as the OpenFlow protocol is supported (e.g., Ryu, OpenDaylight, SEL-5056).
- Can operate in “hybrid” modes to get the best of both worlds (that is, passing packets to common switching protocols) (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Multiple OpenFlow tables may be used to develop complex pipelines to deal with diverse traffic and protocol requirements (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Metering and counters can be used on flows to actively modify traffic routing and QoS (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- “Whitebox” switches allow forgoing of paid subscription-required NOS in lieu of open-source or other NOS.
- With customized flow rules, traditional security flaws associated with common routing and switching protocols may be averted (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).

The users were also asked about experiences with SDN technologies where there is still opportunity for improvement and growth. Those experiences include the following:

- Network flows and flow rules can be complex and confusing when managing many flows across a network. The administrative burden is increased to understand all communication patterns between two endpoints (for example, ARP communications is a prerequisite for many communications) (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).

- Configuring an SDN environment can require complicated bridging architectures when the control plane and data plane share the same physical network (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- For example, porting an existing OvS experiment network to SDN (using Ryu for example), requires each endpoint interface having its own personal bridge. In addition, all individual bridges need to be wired into the experiment control plane (separate from the SDN control plane) and, therefore, needs pairs of virtual Ethernet ports set up and configured correctly. When considering automation, this setup increases complexity and increases the time needed for initial configuration.
- SDN environments can be challenging to troubleshoot and setup when using secure communications (i.e., using TLS) between the SDN flow controller and switches. Staff must have sufficient knowledge to import and export certificates into switches and SDN flow controllers (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- There are expensive “startup costs” associated with setting up the right physical network topology, getting the SDN flow controller to pair with the switches, and configuring allowed network flows. Together, these costs limit the number of devices an SDN network can service and remain secure (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
  - In an enterprise network with a large number of switches and endpoints, this might not be manageable from a single SDN flow controller.
  - There is a limit to the number of switches a single SDN flow controller infrastructure can effectively manage. This is especially true in reactive SDN flow controllers.
  - There is a limit to the number of network flow rules that can be installed into an SDN switch. Additionally, depending on the type of memory, the number of times to which memory can be written is limited.
- SDN configuration is not necessarily a plug-and-play action, particularly in an OT-SDN environment. When introducing a new device into the network, SDN flows need to be analyzed and configured for the new device to communicate on the network to other devices (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
  - Most end users do not have the experience required to use SDN and will need to contact an administrator to add new devices into the network, thus increasing the administrative overhead. Traditional switches have different overhead costs. While a network consisting of traditional switches is initially easier to configure, most OT environments require documentation for all network flows and behaviors, including performance and fault condition recovery. This after-the-fact documentation may take as much time and effort as the up-front design and engineering required in an SDN environment.
  - Additionally, the lack of fine-grained control during the design and implementation process coupled with the “it just works” result of most traditional switches may lead to inconsistent or unexpected performance issues during operation, as well as unanticipated failure-recovery scenarios. Additionally, the ease of operations allows improper configurations and the introduction rogue devices on the network with the same ease as expected configurations and legitimate devices.
- Troubleshooting network problems might be difficult. When in hybrid environments that use SDN and non-SDN compatible network devices, it can be challenging to determine if a communication issue is attributed to a routing configuration problem on the host node or network router, or if it is SDN related. For example, if two endpoints cannot communicate,

both the SDN flow rules and the network routing algorithms may have to be evaluated for misconfigurations (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).

- SDN may increase the complexity of networking. It requires understanding how SDN concepts meshes with old network concepts and may require additional training for end users in dynamic networks that routinely change. OT environments are fairly stable, but industrial internet of things will disrupt this paradigm (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- SDN also leverages some Ethernet protocols, such as Link Layer Discovery Protocol for link and host health discovery that were intended to exist only between the attached device and the switch. In SDN, the protocol's packets can be forwarded to the SDN flow controller, thereby increasing the attack surface (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- Troubleshooting SDNs embedded in conventional network architectures may be challenging (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056). Because of the relative newness of SDNs, there is not as much support and help available.
- Configuration requires compatibility between the SDN protocols installed on the SDN flow controller and the capabilities of the SDN switch (e.g., correct version of OpenFlow support) (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
  - OpenFlow versions are not always compatible. For example, in addition to different features and capabilities, OpenFlow command packet structures changed between specification versions 1.3 and 1.4.
  - Additionally, different SDN switches and controllers may implement different sets of features and options such as write-actions and apply-actions. For example, Hewlett Packard only supports apply-actions, while SEL only supports write-actions (even though write-actions are required for OpenFlow and apply-actions are optional in the standard)
- Setting up and configuring the SDN flow controller software to work correctly can take time, especially for open-source SDN flow controllers like OpenDaylight and Ryu. As SDN continues to be adopted, this issue will be addressed.
- SDN technologies (e.g., SDN flow controller software, host environments supporting SDN flow controllers, and SDN switch software) and implementations (e.g., network flow rules) themselves should be assessed for security so that there is not a false sense of security in using SDN (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).
- SDN, particularly OT-SDN, is a rapidly advancing technology, which may lead to software incompatibilities in delivered equipment. For example, in one case, a commercial switch did not work out of the box and required a return-to-factory service so the correct firmware could be installed. In other cases, old versions of the SDN flow controller software were required to connect to and download firmware upgrades on switches running older versions of firmware (e.g., SEL-2740S).
- In certain instances, the SDN flow controller software regularly experienced unstable operations (e.g., SEL-2740S, Aruba 2920, SEL-5056, OpenDaylight):
  - Appearance of devices on the SDN flow controller graphical user interface was inconsistent.
  - Sometimes adoptions of devices failed, and the solution was to reset the SDN switch to factory settings.

- Communication issues between the SDN flow controller and SDN switches resulted in issues in which stale network information in the SDN flow controller conflicted with current information in the switches. Manual resynchronization was required to update the SDN flow controller (e.g., SEL-2740S, SEL-5056).
- To configure an SDN environment, network operators must know their network and data flows (e.g., OvS, SEL-2740S, Aruba 2920, Ryu, OpenDaylight, SEL-5056).

  This is both a positive and a negative aspect of SDN networking. In most OT environments with static devices and network flows this is generally not an issue, but with the introduction of mobile test equipment and the proliferation of industrial internet of things devices, network operators will need to be aware of an increasing number of devices and manage a corresponding increase in data flows.

In summary, the users enjoyed the benefits and control of SDN but did experience several difficulties when configuring the SDN flow controllers and switches.

The benefits mainly focused on cybersecurity and involved the following:

- The fine-grained control
- Added network cybersecurity such as the secured control plane and the elimination of MAC table spoofing
- Interoperability
- SA of devices communicating on the network.

Areas for improvement mainly focused on:

- Improving the configuration and setup of the SDN technologies to communicate securely
- Installing all needed network flow rules so only the necessary communications are configured
- Interoperating with traditional network devices.

Because of the inconsistencies and instability when using different SDN switches and controllers together, one project opted against adopting an SDN deployment. However, user experiences in using SDN technologies were positive overall after installation and configuration were complete.

8.2.1 User Expectations

Most users expected the integration of SDN technologies to be similar to that of traditional networking devices. The configuration was time consuming for all users who shared their experiences.

One group mentioned that the expected time to complete an SDN configuration was originally planned for 2 to 3 days. However, the configuration took over 3 weeks because of the granular network flow rules being installed. As new communications would be discovered or network flows were omitted by error (e.g., network flows to allow ARP communications), those network flows would have to be manually added to the SDN switches. The user’s expectations were for the initial configuration to be an automated or user-assisted process instead of an entirely manual process.

Users also expected SDN technologies to be interoperable with one another. This expectation was met for the most part; however, it did take more time than expected to correctly configure the SDN flow controllers so that they could communicate with different SDN switches securely using TLS.

Another expectation that was noted but difficult to implement was for an easy way to implement a switched port analyzer (SPAN) port similar to a traditional switch (also known as a mirrored port). This task required more effort than configuring a mirrored port on a non-SDN switch. Implementation in an SDN switch required all network flow rules to be manually modified to forward all traffic to the designated mirrored port. A feature of an SDN implementation that is not easily accomplished with traditional switches is the ability to mirror only specific traffic (e.g., only a specific VLAN, traffic from a set of physical ports, or traffic from a specific TCP or UDP port) reducing the amount of traffic that would be captured or monitored on the mirror port.

A different experience resulted from difficulties in integrating an SDN environment into an existing traditional network. The SDN flow controller repeatedly was confused by the amount of traffic seen at the connection points between the traditional and SDN networks resulting in repeated loss of synchronization and observability between the SDN switches and SDN flow controller. A minimal set of network flow rules was installed in the SDN switches to allow some traffic flow, largely turning the SDN switches into “TCP hubs” whereby all incoming traffic seen for specific TCP ports was forwarded to all physical ports. While this is a severely insecure configuration, using essentially none of the security or fine-grained configuration capabilities of SDN and that could potentially lead to performance issues on a heavily loaded network, it allows the user to gain some experience with the monitoring capabilities of the SDN flow controller and gain familiarity with the SDN switch hardware. This configuration was implemented as a temporary measure until more experience and upgraded SDN flow controller software and switch firmware is available to address some of the issues observed. When new software and firmware are available, the configuration will be revisited to more fully implement a fine-grained SDN environment. The goal of the final environment is to transition it from a traditional network core with VLANs to an SDN network core with a small number of repurposed traditional Ethernet hubs connecting non-critical devices to the network.

Some of the positive user expectations that were met were the fast failover capabilities with preconfigured network flow rules, the deny-by-default benefits, and the ability to quickly log and throttle network flow rule misses.

One user has had very positive experiences deploying SDN in experimental networks. Expectations to have fine-grained control, observation and modification of network traffic and flows, and also modifying packet attributes were met and aided immeasurably to meet experiment requirements and also provide avenues to modify network state (that would have been almost impossible using standard networking protocols).Once configured and operational, most user expectations were met; however, user expectations for ease of configuring and installing the SDN technologies into the network were not met.

There are generally two approaches used when users implemented an SDN environment. They either started from a blank slate, a process generally called a “greenfield” deployment, or they attempted to integrate SDN into an existing operational network, a process generally called a “brownfield” deployment. The approaches used for these deployments are generically described in the following sections.

8.2.1.1 Implementing SDN in a New Environment

When implementing a new network (i.e., a greenfield deployment), there are no expectations for existing flows, and the network can be implemented in a “deny-by-default” configuration that requires each flow be authorized and allowed. The initial set of flows can be designed and implemented based on expected flows. For example, in a substation that uses DNP3 to communicate with a central station through a substation data gateway, the SDN flows could be configured to allow DNP3 traffic between the substation gateway and each DNP3 outstation device in the substation. Other flows such as ARP and ICMP used for network and device management also can be implemented.

Once the initial set of expected flows is implemented in the SDN switches, any traffic that does not match an existing flow rule will not flow through the network. Flow rules that track unmatched flow rules can then be flagged and assessed either as configuration errors, remedied by reconfiguring the devices to not generate the traffic, or determined to be legitimate traffic and remedied by creating new flow rules to allow the traffic to flow. During this initial flow rule generation phase, it is important to completely exercise all expected operations including boundary, exception, and infrequent operating conditions to verify that all the necessary flow rules are created.

Once all the expected traffic flow rules have been implemented, the network will still need to be monitored for unexpected but valid traffic. The SDN switches will contain a “catch-all” rule that can count the number of packets that it processes and forward them to the SDN controller for analysis. In a greenfield deployment, the number of packets found in this manner should be small and may be the result of device misconfiguration or malfunction.

Some flows, such as flows associated with infrequent maintenance, do not need to be enabled all the time, but could be enabled as part of the maintenance procedure, and then disabled when the maintenance activity is complete. This minimizes the capability to exploit using maintenance network traffic outside of a maintenance activity.

8.2.1.2 Retrofitting an Existing Network with SDN

When retrofitting an existing network with a new SDN environment (i.e., a “brownfield” deployment), the basic expectation is that the network is operational and is performing properly, even though there may be additional unauthorized traffic. Because the network is operational (and in the case of an OT network, likely controlling a physical process), it is appropriate to apply the medical concept of “first do no harm.” In this case, deny-by-default will cause all network traffic to cease until flow rules are created and implemented to allow flow of the authorized traffic. In the case of a brownfield implementation, a better approach would be “allow-by-default” with traffic logging and monitoring that allows all traffic flow to continue while recording all network flows for analysis.

When implementing a brownfield deployment, existing network flows should be assessed with multiple approaches, including:

- Using strategic network traffic captures that capture normal traffic flows
- Exercising equipment to ensure that expected normal operations that generate traffic flows are captured

- Exercising operations and equipment that cause infrequent (e.g., monthly, annual) communications (if possible)
- Exercising conditions that cause communication path and device failover (if possible)
- Using automated traffic analysis tools. If possible, use tools that can automatically analyze the traffic captures and generate SDN flow rules.

Tools for assessing network traffic are being developed and released by both SDN component vendors and others. SEL, for example, has released a feature they call "load and lock" that can capture network traffic, generate flow rules to all the traffic to pass through the network, and automatically install them to the appropriate SDN switches. Once a near-complete set of flow rules has been established, the feature can be turned off (i.e., the automatic flow rule creation is "locked"), requiring manual addition of additional flow rules. Once locked, the existing flow rules can also be analyzed to determine if any unwanted flows were inadvertently enabled.

Other tools that can analyze network traffic captures and generate flow rules may exist but will required additional manual processes to implement them in the SDN environment. For example, existing tools like GRASSMARLIN[48] could be augmented to create flow rules from network traffic captures.

Once the traffic has been analyzed and a set of flow rules that represent the traffic as seen in the operational environment has been created, the flow rules should be tested in a laboratory environment with end-user participation to exercise the network, including failover and infrequent communications to ensure that the network will perform as expected during normal and infrequent operating conditions.

Once the network has been completely tested in the laboratory environment, the SDN equipment and flow rules can be migrated into the operational environment with a "catch-all" rule inserted as the lowest-priority flow rule to capture and log any traffic that does not match one of the pre-engineered flow rules. Any traffic not already caught by a higher-priority flow rule and forwarded or explicitly dropped is caught by the catch-all rule and should be assessed against the following criteria:

1. Is the traffic legitimate? If so, associated flow rules should be created and implemented in the network.
2. Is the traffic the result of a configuration error? If so, the sending node should be reconfigured to eliminate the traffic.
3. Is the traffic benign? If so, it can be dropped by the SDN switch.
4. Is the traffic malicious? If so, further investigation is warranted.

During the allow-by-default implementation phase, the traffic processed by the catch-all rule should be forwarded on to its final destination.

This process continues until no additional traffic is observed by the catch-all rule.

This approach will allow the network to continue to operate while a set of flow rules matching the existing traffic flows can be created and installed. During this process, each flow rule should be verified and validated against an expected data flow, and any flows that cannot be justified

[48] https://github.com/nsacyber/GRASSMARLIN/ (Accessed September 17, 2021)

should be investigated to determine if they are the result of configuration errors, lack of complete understanding of what flows are actually required in the network or represent illicit traffic. (Very often, existing operational network devices contain configuration errors that create benign traffic on the network or contain unnecessary or compromised devices that create illicit traffic.) Once all legitimate network traffic has been captured and analyzed, and the set of final flow rules is implemented, the network can be converted to a “deny-by-default” operations mode.

Note, however, that locking down the flow rules too early in the analysis process may lead to unexpected service disruptions for cases in which critical but infrequent flows are required. These infrequent flows could be time based (i.e., some flows only occur monthly, quarterly, or annually) so the longer actual flows are monitored before the deny-by-default mode is enabled, the fewer disruptions to legitimate traffic will be encountered.

Other flows happen infrequently, such as responding to specific failure conditions or in response to emergency or upset conditions. Emergency or upset conditions cannot be scheduled (although they could be simulated during a maintenance window) and may occur while the network is being observed. For this reason, the deny-by-default mode should only be implemented after careful consideration. An extended scheduled or unscheduled maintenance outage may be required to simulate all known possible emergency conditions to accurately capture infrequent data flows.

As with the greenfield approach, flows associated with infrequent maintenance could be enabled as part of the maintenance procedure, and then disabled when the maintenance activity is complete.

8.3 Overview of SDN Flow Rule Actions

Implementing an SDN infrastructure in an EDS can enable significantly more control over the behavior of the network and the communications that take place within it than can be implemented in a traditional network environment, even when using VLANs. SDN, specifically OpenFlow V1.3, allows for the creation of network flow rules that can act based on inspection of physical connections and protocol header information, specifically including matching any combination of:

- Source physical port
- Source MAC address
- Destination MAC address
- The VLAN tag
- The VLAN priority
- The Ethernet type (EtherType) field
- Source IP address (if specified)[49]
- Destination IP address (if specified)[49]
- IP type (e.g., TCP, UDP, or ICMP) [49]

[49] Note – The SEL-2740S only supports IP Version 4

- Source TCP or UDP port
- Destination TCP or UDP port.

In addition to these, some additional match fields could include flags and fields in the IP, TCP, or UDP packet headers. For example, a match on the time to live (TTL) field could be used in conjunction with additional action processing to modify the TTL value to allow additional network hops within the SDN environment for packet filtering and processing without impacting the TTL processing.

Other match processing could use wildcards or ranges to match. Examples include matching on the manufacturer's code in the first 24 bits of the MAC address or matching on a range of IP address or an IP subnet. Options for deep packet inspection on the data within a packet also would be useful (as long as users are aware of the resulting performance impacts).

The network flow rules can be programmed to take a number of actions. The significant actions can be categorized as either "write-actions" or "apply-actions." Write-actions append or replace a set of an actions to an action set for the packet. A "clear-action" is a special type of write-action that removes all actions from the action set. Apply-actions execute the action set immediately (bypassing the remaining flow rules). Unless an "apply-action" is specified, the actions in the action set are executed at the end of the match processing for the packet.

Typical actions are below:

- Forward the packet to a destination physical port on the same or different switch in the SDN environment using the OpenFlow OUTPUT action.
- Re-write portions of the packet (e.g., the destination MAC address, VLAN tag), and forward the packet to a destination physical port on the same or different switch.[50] Additional packet field rewriting options could be explored (e.g., rewriting the TTL).
- Take no action (i.e., drop the packet).[51]
- Send the packet to the SDN flow controller for further guidance.[52] In an OT-SDN environment, this introduces a potential reliability issue associated with overloading the SDN controller with requests. It therefore should be avoided if possible.

If the packets are sent to the SDN flow controller, the SDN flow controller, in conjunction with additional application processing, can then take any of the following actions:

- Send the packet back to the switch and allow it to be forwarded.
- Create and install a new network flow rule in the switches to allow the packet (and future similar packets) to flow normally through the SDN environment.
- Further analyze the packet using more advanced analysis techniques such as deeper inspection of the packet, contextual analysis of the packet, and contextual analysis of the communications stream to determine what should be done with the packet.

[50] Note – The SEL-2740S only supports addition, removal, and modification of VLAN tags and VLAN priorities
[51] Note – This is the default action if the packet does not match any of the flow rules
[52] Note – Technically, this is the same as the OUTPUT action, however, inserting the SDN controller into the OUTPUT processing allows additional decision-making processing that cannot be performed inside the SDN switch

- Take no action (i.e., drop the packet).

To meet performance requirements, SDN switch software and hardware and SDN flow controller implementations are limited in in their ability to match on fields and their ability to take actions. For example, the switch must be able to rapidly find a particular field for matching. Protocols with variable payload lengths and locations, especially if they require processing length pointers to fields of interest, may cause performance processing issues (e.g., latency or jitter). Any packet re-write actions may require “checksum values” to be recalculated and written into packets before transmission, and any re-writes that impact the length of the packet will cause the entire packet to be re-written to adjust length indicators and checksums values, some of which may be in the application (OSI layer 7) portion of the packet. For these reasons, implementing some additional match rules and actions may not be practical within the SDN switches and controllers but may make limited sense in application layer processing.

Another issue with packet inspection is fragmentation. When using normal Ethernet, each Ethernet data frame is restricted to 1500 octets of payload data, including the IP and TCP headers. If the TCP packet is larger than can be transmitted in a single Ethernet frame (i.e., it is larger than 1500 octets minus the length of the TCP header information and the length of the IP header information), it is fragmented by the IP layer into multiple frames. For the receiving node to reassemble the frames into a complete packet to be sent to the TCP layer, the IP header is repeated in each frame with a “more fragments” flag set in all but the last frame. However, the TCP header is only included in the first frame. The switch must be able to perform all of its processing on a single frame and cannot wait for additional frames to determine how the initial frame should be processed. This would normally not be a problem because the destination MAC address (contained in the Ethernet header), and the IP address (contained in the IP header) are all that is needed to send the complete set of fragments to their destination. However, if the SDN switch also needs information from the TCP header, it cannot process any frame except the first one.

This condition can be overcome by ensuring that all devices in the network manage their data sizes to ensure that the IP layer does not have to perform any fragmentation processing and configuring SDN flow rules to check for and discard any fragmented packets seen on the network.

8.4 Protocol Behavior Enforcement using SDN

8.4.1 EDS Protocol Characteristics

As previously noted, a dated quote from NERC states, “Control Systems are the ‘brains’ of the control and monitoring of the BES and other critical infrastructures, but they were designed for functionality and performance, not security. Most Control Systems assume an environment of complete and implicit trust.”[24] The implicit trust attribute applies to many EDS protocols in which the ability to cryptographically ensure message integrity is not available natively in the protocol. EDS protocols have many interesting features that are used to monitor and control expensive physical assets. Incorrect operation of these assets can lead to cascading failures or damaged equipment.

In addition to providing native security controls, SDN enables the deployment of new security controls. By leveraging attributes of SDN communication, the vision is to develop a proof-of-concept system to centrally define the expected EDS protocol behavior and enforce that behavior in a distributed manner. While native SDN implementations do not support the activities associated with protocol behavior enforcement, an enhanced SDN environment could support the ability to limit EDS protocol functionality or physical path. This can take many forms including the following.

- Defining which EDS protocol function codes are permitted.

  Investigate how or whether the OpenFlow standards should be modified to support this kind of inspection activity. A possible approach could define behavior similar to the *packet-in* and *packet-out* processing to new functions in the switch itself, perhaps using co-processors or daughter cards with more general-purpose computing capability.
- Limiting which physical ports can accept specific EDS protocol function codes.
- Limiting which EDS protocol function codes or commands can be issued to multicast addresses, including limiting which multicast devices can receive those codes or commands.
- Limiting the ranges or registers addressed by an EDS protocol command.
- Limiting the frequency of EDS protocol control commands.
- Providing awareness of EDS protocol use that is outside of expected behaviors.
- Preferring or enforcing physical paths or devices to use.

SCADA communication (a specific form of EDS communication that often occurs at “human” speeds) is a multi-purpose conversation between a central system (typically found in a control center) and field remote stations (in an electric system environment, typically found at substations and generation plants). SCADA consists of two different functions: 1) control (the SC portion of the SCADA term) and 2) data acquisition (the data acquisition portion of the term). When used for the data acquisition portion, communication messages are pre-engineered to work on a specific polling cycle, although there may be exceptions for “integrity” scans or “demand” scans that may be operator initiated. The poll (request) and the response are typically the exact same amount of data in each direction every time for each different type of scan. This allows OT-SDN system owners to baseline and monitor that SCADA polls are happening within the expected time windows and the poll and response are the proper format and length. Any additional packets or polls out of cycle are potential indicators that a problem exists that should be investigated.

Furthermore, the system owner could shut off the network flows between polls using the OpenFlow disable or enable actions (although this may have hardware design impacts on the SDN switch if the actions are performed frequently). The SCADA packets are then only forwarded during the time they are meant to be on the system. For non-periodic or non-predictable behavior (such as control actions or ad hoc integrity scans), a separate command channel to the SDN flow controller or switch could allow (or enable) these infrequent or unscheduled communications to flow only when needed and then disable them after the flow concluded.

OT-SDN can enforce directionality and on specific physical connections and cables. This way system owners know and control which physical central site systems can poll. They also know and control the remote stations that can respond and can monitor for SCADA packets entering the network from other locations or representing illogical communication exchanges (like a remote station issuing a poll command).

OT-SDN primarily inspects the Ethernet, IP, or TCP headers. SCADA, however, uses well-structured protocols such as DNP3 with data sets in the payload, which could extend these security controls even further into opcodes, and state model enforcements. Using the opcodes, OT-SDN system owners could control which operations within the DNP3 protocol are allowed, when they are allowed, or which direction they are allowed.

The EDS protocol enforcement processing described in the remainder of this report focuses primarily on analyzing and enforcing application level (OSI layer 7) data payloads.

8.4.2 EDS Protocol Enforcement Processing

While SDN flow rule processing is very effective at processing the lower-level OSI layers used for physical access, network communication, inter-network routing, and session control (i.e., OSI layers 1 through 4), OpenFlow V1.3 does not provide for inspection (field matching) or processing in higher levels such as the application layer containing the data payload.

Many EDS protocols were developed before the widespread use of networking and internetworking technologies, and therefore contain their own addressing mechanisms within the application layer (OSI layer 7), as well as additional inter-device routing for specific commands such as telemetry and control point addressing. OpenFlow V1.3 does not provide for monitoring of fields in the application layer, so alternative mechanisms must be used to provide the same kinds of decision-making and forwarding enforcement that are provided for OSI layers 1 through 4. Because many of these protocols can also run in non-networked environments (e.g., point-to-point direct lines using modem technology), some fields such as packet length and error detection or correction codes (e.g., checksum fields) perform duplicate functions to those provided in the network (e.g., IP, TCP) headers.

To reduce performance impacts on SDN switch hardware, a separate system, appliance, or device can be used to enforce the defined protocol behavior. Alternatively, these processing features could be incorporated into the switch itself either as extensions to the basic SDN processing already provided or by taking advantage of extensions to the SDN specification to provide this capability.

The approach recommended includes the following architectural elements:

1. *Define network flow rules to ensure EDS communications occur between authorized devices* – This processing would be performed by the SDN switch to enforce basic network configurations (e.g., packets arriving from proper physical, MAC and IP addresses destined for proper IP addresses) using existing configured flow rule processing.
2. *Define a network flow or flow rule to ensure all EDS traffic is examined by the protocol enforcement technology* – This processing would be performed by forwarding the packet for inspection by an external application.
3. *Provide SA of EDS communications by monitoring counters maintained in the SDN switches through the NBI or via SNMP queries* – This is accomplished by maintaining counters of received, rejected, analyzed, and forwarded packets. Counters also could be maintained for flow rule matches, including some flow rules created solely for maintaining counters.

The SDN4EDS project proposes that packets could be forwarded from the switch to enforcement processing either through the NBI to the SDN flow controller or through flow rules to application nodes to enforce EDS protocol behavior.

The protocol behavior enforcement approach is described in the remainder of this report and contains the following discussions in subsequent sections:

1. *Protocol validation* – Verifying that the format and fields contained in the data payload are valid according to the protocol description by looking at individual packets. This is further described in Section 8.5.
2. *Configuration validation* – Verifying that the telemetry and control points specified in the data payload are valid and exist for the target device by looking at individual packets. This is further described in Section 8.6.
3. *Behavior validation* – Verifying that the interactions between the end-device and other devices are logical by looking at multiple packets. This is further described in Section 8.7.
4. *Dynamic behavior modification* – Dynamically changing SDN flow rules based on expected data flows. This is further described in Section 8.8.
5. *Proposed SDN flow controller architecture* – A proposed architecture for implementing the validations in a large, distributed environment using a hierarchical approach. This is further described in Section 8.9.

The SDN4EDS project proposes to implement protocol behavior enforcement for the DNP3/IP protocol. Additional protocol enforcement of other protocols, such as IEC 60870, IEC 61850, IEEE C37.118, and Modbus/TCP could be implemented in follow-on activities.

8.4.3 Overview of SDN Proactive and Reactive Message Forwarding

SDN environments can operate in one of three modes: 1) proactive, 2) reactive, or 3) hybrid.

In proactive mode, all network flows are pre-engineered, and network flow rules are preinstalled in the switches. This allows the switches to operate autonomously without external coordination from an SDN flow controller or an application. This is the primary operating mechanism for an OT-SDN environment.

In reactive mode, the network switches do not have pre-engineered flow rules, and the SDN switches need to query the SDN flow controller to determine how to handle a packet. The SDN flow controller can instruct the SDN switch how to process the packet. As an option, the controller can create a permanent flow rule so that the next time the SDN switch sees a similar packet, it can act proactively and process the packet without interrogating the SDN flow controller, thus allowing the switch to build up a list of network flow entries over time. In a completely reactive environment, SDN switches will build up flow rules slowly as traffic is encountered, potentially causing performance issues for time sensitive protocols (like Network Time Protocol [NTP]) as the flow rules are created. Additionally, a process needs to be implemented to identify and remove old, unneeded flow rules to avoid filling up the flow rule table, which would lead to performance issues associated with processing through large flow tables and also the inability to add new flow rules when the table is full. An attacker could exploit this by vulnerability by crafting packets to artificially fill the flow rule table, thereby blocking addition of legitimate flow rules. This mode is generally not applicable in an OT-SDN environment.

A hybrid mode is a combination of the two approaches.

For EDS protocol enforcement, the hybrid approach is more appropriate, whereby routine and expected non-EDS protocols (or EDS protocols that are not using active protocol enforcement) flow through the switch using pre-engineered flow rules, while the EDS protocols with active protocol enforcement use the reactive approach or an augmented hybrid approach. Unknown protocols or unconfigured protocols (e.g., EDS protocols appearing on unconfigured physical ports or from unconfigured MAC or IP addresses) can still be dropped without contacting the SDN flow controller.

The purely reactive approach requires that packets be forwarded from the SDN switch to an SDN flow controller for further analysis to determine if they will be forwarded or dropped. The process uses the *packet-in* and *packet-out* capabilities of the OpenFlow protocol. The SDN switch will contain a flow rule to forward the EDS protocols to the SDN flow controller using the *packet-in* (i.e., packets flowing in to the SDN flow controller) command. The SDN flow controller will then be programmed to pass the packet to an application for EDS protocol-specific processing. If the packet is to be forwarded, the application will return the packet to the SDN flow controller, which in turn forwards it to the SDN switch using a *packet-out* (i.e., packets flowing out from the SDN flow controller) command and includes the actions that the switch is to take with the packet (i.e., where to forward the packet).

The augmented hybrid approach uses pre-engineered flow rules to forward packets through the SDN to application nodes that are connected to SDN switch ports on their way to the end-devices. The application nodes perform the same analysis as in the purely reactive approach. If the application determines that a particular flow is acceptable, it can request flow rules be created temporarily to allow the flow directly without having each packet be processed. When the flow sequence is over, the flow rule can be deleted or disabled.

Implementing either approach involves additional processing to manage the flow rules, at a minimum, including the following actions:

- *Flow timeout* – Rules for flows that are triggered in an appropriate amount of time should be removed (trimmed) from the SDN switch.
- *Flow end* – If the logical end of a particular flow can be detected, the rules associated with that particular flow can be deleted from the SDN switch. An example of this would be a set of flow rules performing firmware updates. A process would tell the validation software to enable a set of flow rules to allow the firmware download paths. The validation software would monitor the packets looking for the end of the firmware download and then automatically disable the firmware download flow rules. Note, however, that some protocols or flow patterns do not have actionable endings so an external process would be required to disable the flow rules.
- *Flow intercept* – This action is used for some protocols such as TCP that do not exchange OSI layer 7 data until the flow has moved to the ESTABLISHED state. TCP packets sent before the ESTABLISHED state are said to be embryonic. If an attacker's embryonic packets are sent to a destination, the attacker can stall it, for example. With intercept, the application node proxies the TCP responses at the destination and then inspects the data as it moves into ESTABLISHED. If denied, the application node sends a TCP RST (i.e., a reset) to the destination and drops the source without replying as the attacker otherwise could solicit a reflexive attack against spoofed sources.

Note that inserting the SDN flow controller (or potentially any other process) into the communication stream will introduce delays in packet delivery to the final destination. This will cause problems in applications that are sensitive to communication latency. Because the processing involved in inspecting the packet may be different depending on the contents of the data payload, packet delivery may also be subject to jitter.

Another issue with inserting the SDN flow controller into the communication stream is that it becomes a potential single point of failure and a chokepoint. If too many packets arrive on different switch ports, even though they may be slated for different destinations, the SDN flow controller and its associated processing will need to inspect each packet. If the inspection process becomes overloaded, the increase in latency for an individual packet may result in a DoS.

These are the primary reasons that a distributed hierarchical approach to SDN flow controller implementation is being proposed. Distributing the inspection processing as far down into the network and as close as possible to the final destination minimizes the latency introduced by long distance or slow communication. Additionally, the distributed nature of inspection processing can allow the inspection to be performed in parallel at different field sites. Monitoring the performance of the nodes performing inspection process could be in indicator of local attacks.

For these reasons, the SDN4EDS project implemented the protocol behavior on a simple unencrypted[53] DNP3 communication stream operating at “human speeds” (i.e., scan rates of tens of points every 2 seconds, and occasional control commands). The project investigated the impact of latency and jitter in the communications introduced by the inspection process and make recommendations on how the impact of the inspection might be reduced.

Investigating the performance of this approach with SCADA-speed protocols like DNP3 can help inform how other protocols would be affected by inspection processing. High-speed, low latency protocols and applications like IEC 61850 GOOSE and SV, or IEEE C37.118.2 Synchrophasor Data Transfer may not be suited for the approach proposed in this report.

8.5 Protocol Validation

Validating that a given packet data payload conforms to the protocol specification is the base level of validation that could be performed. This could include actions such as validating checksum and length fields that are typically included in the packets, looking for out-of-range fields or other anomalous packets that could potentially be used to cause device malfunctions.

Typical IDS processing validates that the payload portion of a packet conforms to the protocol specification and issues an alert when malformed packets are detected. Most IDSs can detect packets containing invalid operation (OP) codes, or missing required OP code data, packet length errors (too long, too short, not as indicated in the length field of the packet), and checksum errors. Many IDSs also can detect and send alerts on malicious actions such as firmware download or device reset commands that while being valid commands, are infrequently used in real environments and generally indicate that malicious actors have gained access to the communications system. Because IDSs typically only “detect” suspicious or malicious activity, they generally only provide SA to a network operator who must act, often long after malicious actions have occurred.

To overcome the passive nature of an IDS, an IPS can block malformed and malicious packets from reaching end-devices. For example, packets that are too long may cause buffer overflow errors in end-devices as they are processed. An IDS can also be used to prevent malicious packets such as firmware updates and device resets from reaching end-devices. IPS

[53] To inspect the OSI layer 7 data, encrypted data transfers will need to be decrypted.

processing fails when critical, infrequent, and potentially dangerous operations must be performed (e.g., if firmware must be updated remotely) but the packets are blocked by the IPS.

An IPS may also block infrequently seen traffic associated with critical functions such as GOOSE and SV traffic used in protection relaying applications. Blocking or delaying this traffic may cause device malfunction, equipment damage, or customer outages. Excessively delaying streaming data such as IEEE C37.118.2 synchrophasor data may result in the data being perpetually untimely resulting in its diminished use for real-time decision making.

Because end-devices often are located in remote locations sometime requiring seemingly malicious actions (such as firmware updates and device resets) and they perform critical functions that may require updating and resetting to continue operating, many EDS operators are hesitant to allow the “prevention” component of an IPS to function.

8.6 Configuration Validation

Configuration validation differs from protocol validation in that packets that are formed perfectly in terms of a protocol specification still may contain illogical flows for defined configurations. These typically are benign and ignored when received by an end-device because the end-device is not configured to process them, but they still use network bandwidth and processing. Even though their end result is no action, they should be identified because they could be an indication of legitimate configuration errors or attempts at scanning or other malicious actions.

8.7 Behavior Validation

Behavior validation differs from protocol and configuration validation by looking at the interaction and relationships among multiple packets. This will require a more in-depth understanding of the protocol and application to minimize the number of false positive triggers for uncommon but still legitimate behaviors. It also is significantly more challenging because the processing may allow some packets through while blocking others. These approaches may be suited for machine learning or artificial intelligence processing to manage the behavior validation processing.

There are three approaches associated with behavior validation: 1) logical or sequence validation, 2) performance validation, and 3) out-of-band permissive validation.

8.7.1 Logical or Sequence Validation

An example of logical or sequence validation would be for example, if a DNP3 control point requires a “select” command to be received and processed before an “operate” command can be executed, the validation processing could drop any “operate” commands that are not preceded by a “select” command. The validation would allow the “select” command to proceed to the end-device in anticipation of receiving and processing the following “operate” command, internally noting that a subsequent “operate” command should be allowed. Once the logical sequence of a “select” command followed by an “operate” command were processed, the validation software would reset waiting for the next “select-operate” command sequence. If a “select” command were received but no “operate” command were received, the validation application would need to wait for a protocol-specified (or device configured) timeframe before resetting the select indication.

8.7.2 Performance Validation

An example of performance validation would be for example, if telemetry scanning should operate on a 10-second cycle, validation processing could detect and drop scan requests that occur at significantly faster times. Note, however, that while a 1-second scan rate may be excessive (i.e., a ten-fold increase in the number of requests), an 8- or 9-second scan rate (i.e., nearly the same as the expected scan rate) may simply be a central station catching up from a previous processing delay and likely should be allowed.

8.7.3 Out-of-Bound Permissive Validation

An example of out-of-band permissive validation would be blocking firmware download requests that, if allowed, would legitimately happen rarely. An approach to allow such requests would be to implement an operation similar to the “select-before-operate” scenario used for control points. The firmware download option could be “enabled” by sending an “out-of-band” command to the validation application to allow the firmware download command temporarily. This would tell the validation software that a legitimate firmware download operation was about to happen, and it should not drop the firmware download requests. This out-of-band command should be issued outside of the normal protocol and should be communicated over an authenticated and potentially encrypted channel to minimize the ability of an attacker to exploit the action. This would most likely be a manual procedure since the application that performs the firmware download would not be aware of the need to enable the traffic in the SDN environment. The validation software would continue to monitor the packets looking for the end of the firmware download and then automatically re-disable firmware download commands. Because the validation process operates on a per-end-device basis, the firmware download would only be successful for the selected end-device.

8.8 Dynamic Behavior Modification

The concept of dynamic behavior modification takes the behavior validation step further by determining how to dynamically modify network flow rules in the SDN switch to only allow packets to be passed during certain times or based on certain conditions. This concept can be used to offload the SDN flow controller’s processing of individual packets to determine if they should be passed or dropped, possibly improving the performance of the network traffic flows.

In essence, the SDN environment would not allow any traffic to flow (at least for EDS protocols) unless and until it is specifically allowed, and when the traffic has flowed, further packets would be disallowed until they are again specifically allowed.

There are two basic approaches to dynamic behavior modification:

- Time-based enable
- Dynamic enable (note that some vendors provide inputs [contacts] or similar interfaces to trigger a dynamic event, allowing predetermined flows to be added based on field inputs).

These approaches and behavior monitoring are discussed in the following sections.

8.8.1 Time-Based Enable

A time-based enable assumes that specific communication sequences occur on very strict and predictable frequencies, for example, every 10 seconds. The local application could enable (or create) a flow rule just before the communication sequence is scheduled to happen, then allow the communication to flow through the SDN environment normally, and finally, when complete, automatically disable (or delete) the flow rule.

8.8.2 Dynamic Enable

The dynamic enable approach is used when the schedule of the communications flows cannot be determined. A primary example of this is a control command in which controls often are based on operator requests in response to other conditions and could happen at any time, although infrequently.

In both the time-based enable and dynamic enable approaches, communications between the application node and the SDN switch (through the SDN flow controller) is the same.

8.8.3 Behavior Monitoring

To monitor the behavior and effectiveness of the dynamic behavior modification, the application node should collect information about when the flow rules are enabled, when the EDS protocols transit through the switch, and when the flow rules are disabled. For the time-based approach, this could help tune the processing to indicate the tolerance required for opening and closing the communication and help detect and alert for clock drift or other time-based processing.

Monitoring also could check and alert for spurious communications that occur when flow rules are enabled; for example, if multiple scans or replies occur during a time-based periodic scan when a single scan reply is expected or if multiple communications occur during a dynamic enabled scan.

Use of machine learning in the application node may assist the analysis of these data in the field and provide additional insights.

8.9 Implementing EDS Protocol Behavior Processing in an SDN Environment

Traditional SDN implementations were developed primarily for data center environments with a relatively small number of switches with each containing many ports and multiple high-bandwidth (e.g., 1 Gbit or 10 Gbit) links interconnecting them. This implementation allowed significant flexibility for dynamic node discovery and movement; the ability to adapt to dynamic traffic patterns using different protocols; and use in situations in which network performance was paramount with application or node redundancy implemented through virtual reconfiguration of virtual or real compute and storage resources.

OT environments, on the other hand, typically are very large, geographically dispersed environments comprised of a large number of individually small sites containing tens, or in limited cases, hundreds of discrete devices containing specialized hardware and software. Redundancy and resiliency are of paramount importance, and redundancy is implemented through a “no single point of failure” approach in all aspects of the environment from individual physical sensors throughout the network to application processors. In an OT environment, a minimum of two hardware nodes capable of performing the same function and configured to take over when one node fails is a hard and fast requirement.

In an OT environment, two network switches, at a minimum, are required for redundancy (this applies to both traditional networks and SDN environments). Additional switches are added to provide additional application separation (e.g., to separate and isolate different voltage levels in an electric power substation; or in an IEC 61850 environment, to physically isolate a process bus from a station bus) or to allow the connection of more devices than a single switch (or set of switches) can support. Network redundancy to individual end-devices most often is achieved by multiple network interfaces from the end-devices with each device connected to a separate network switch, which allows continued operation after a network interface failure, cable failure, or switch failure.

In an electric transmission environment, these individual sites (transmission or distribution stations) are internally connected by high-speed (often 100 Mbit or higher), redundant, and resilient links but are connected to a central site (i.e., a control center) by lower-speed, less reliable, often non-redundant wide-area connections to provide data acquisition, supervisory control, and SA. These wide-area connections are often long-distance connections using a combination of private and public infrastructure (including fiber optic, microwave, and leased copper circuits) that is not necessarily designed for low latency, low jitter, highly deterministic communications. There could be tens to hundreds, or in some cases, multiple thousands of remote sites, each with a minimum of two network switches, resulting in a combined infrastructure of anywhere from several hundred to nearly 10,000 (or more) switches. Current SDN environments have not envisioned a single infrastructure consisting of that many individual SDN switches but managed and monitored at individual site (e.g., substation) level using a hierarchical SDN flow controller architecture.

8.9.1 Protocol Enforcement Processing

To implement protocol enforcement, individual packets will need to be passed through various inspection processing to determine if the packet should be forwarded through the network.

As part of the overall inspection and enforcement processing, a separate state must be maintained for each logical conversation. This implementation will allow some interactions (e.g., acknowledgments to be quickly forwarded on, while other packets will need to undergo additional inspection). Because EDS protocols are generally time sensitive and susceptible to latency and jitter disturbances, the inspection and enforcement processing must be as efficient as possible.

Using the DNP3 protocol as an example, the following processing would take place:

1. A packet would arrive on a port of the SDN switch using port TCP/20000 from an external source, such as a master station at a SCADA control center or a substation gateway.
2. The basic addressing of the packet would be verified against existing SDN flow rules to ensure that the physical port, MAC address, and source and destination IP addresses are acceptable. If the packet is improperly addressed (e.g., it does not match the physical port, MAC address, IP address, or TCP or UDP port), it will not match an existing flow rule. The SDN switch will drop the packet and no further processing takes place. If the packet is accepted by the switch (i.e., it matches a flow rule associated with the address fields), it is processed by the next step.
3. The packet would be sent to an application or network appliance to inspect the packet for protocol violations, similar to the processing available in commercial and open-source IDS or IPS solutions (the Protocol Validation processing described in Section 8.5).

4. The packet then is inspected by a behavior enforcement node to determine if it contains any illogical (but syntactically valid) commands. This requires detailed knowledge of all possible protocol variations, including optional, non-specified order, and variable length fields in the protocol. This also requires knowledge of the capabilities and features of the target device; for example, from a configuration file (the Configuration Validation processing described in Section 8.6).
5. The packet is then inspected by a behavior enforcement node to determine if the commands or responses in the packet make logical sense given the context of the packet. The processing also stores context for future processing; for example, to verify that a scan reply is expected, a preceding scan request needs to be captured (the Behavior Validation processing described in Section 8.7).
6. If the packet is deemed to be valid (i.e., syntactically correct, logically correct, and contextually valid), it is forwarded to its end destination. If it is not valid, it is dropped.[54]
7. The behavior enforcement node could also receive context messages outside the scope of the SCADA protocol being inspected (the Dynamic Enable processing described in Section 8.8.2). For example, the local application could receive notification that a firmware download is about to happen. Normally, download packets would be dropped as invalid behavior but could be enabled by a maintenance application that tells the local application node that the download operation is valid. This would allow the local application to be aware contextually that firmware downloads are valid and allow them to be processed through to the end node. If the protocol supports an “end of download” packet, this could be processed to disable firmware downloads until another permissive request was received. Alternatively, a timeout counter could be started to allow the download packets for a set time from the initial packet or a set time between packets to accommodate slow or unpredictable performance communication links, or after a period of packet inactivity, or the maintenance application could send an indication once the firmware download is complete.
8. The behavior enforcement node could also perform additional context-based (the Dynamic Enable processing described in Section 8.8.2) or time-based (the Time-Based Enable processing described in Section 8.8.1) processing to allow packets to proceed, either through context processing, or by enabling and disabling seldom-used commands or command sequences by modifying network flow rules in the SDN switches. It would do this by making requests to the SDN flow controller to create, delete, or update specific network flow rules.

The concepts of using 1) an external application to perform the inspection [Comer 2019], 2) a distributed or hierarchical SDN flow controller environment [Shah 2018], [Koshibe 2014], [Amiri 2019], [Giatsios 2019], 3) configuration files to configure SDN flow rules [O’Raw 2017], and 4) modifying the SDN flow rules based on process or protocol behavior [Nivethan 2016 – 1] and [Nivethan 2016 – 2] are not novel. The implementation proposed in this report combines all four concepts into a single solution.

Note that much of the processing performed by the application node will require access to the payload portion of the data packet, which means that use of encrypted secure protocol variants will need to include the capability to decrypt the payload for inspection.

[54] This can get quite complicated. For example, if the packet is part of a TCP communication, additional handshaking at the TCP layer will be required so that the dropped packet is not retransmitted. The application layer (e.g., DNP3) may also include handshaking and acknowledgement that must be processed so that the application does not attempt to re-transmit the packet.

8.9.1.1 Protocol Enforcement Processing Using Flow Rules

Several approaches to protocol enforcement processing were proposed, including using the SDN Flow Controller to intercept and pass all frames to an application processing node for analysis, but this approach was determined to be impractical from a resilience and performance standpoint.

A preferred approach to integrating the protocol enforcement applications into the SDN environment as described in [SEL 2017] would be to create flow rules to forward traffic to the applications located within the data plane. This is shown graphically in Figure 8-1.

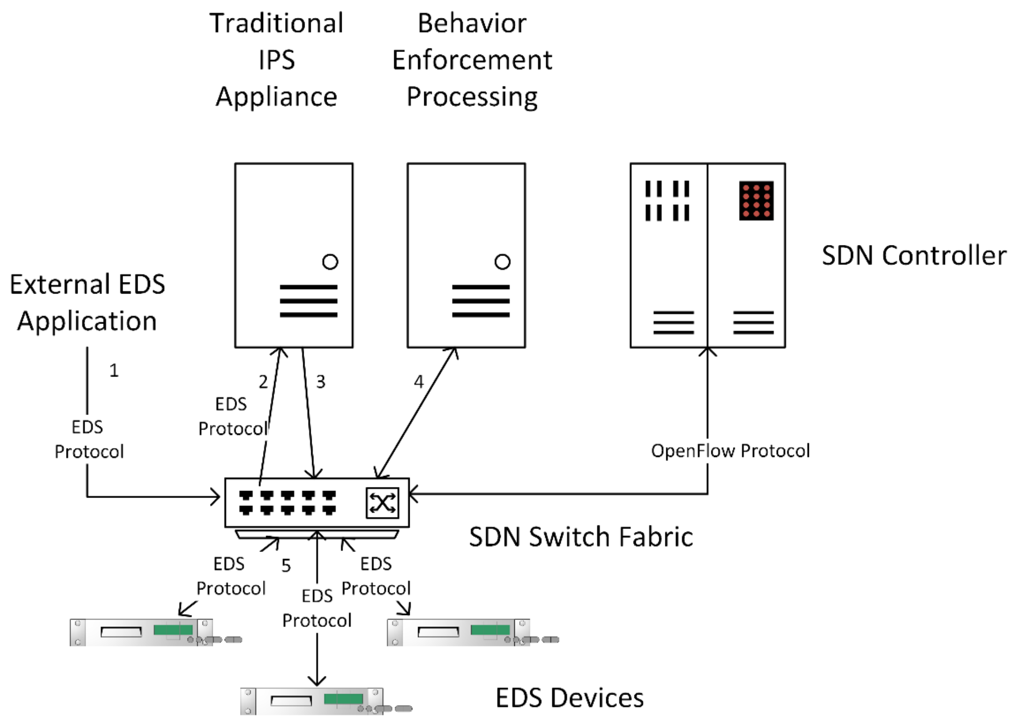


Figure 8-1. Application Node Integration

In this approach, flow rules within the SDN switch are used to forward the EDS protocol packets for analysis, rather than sending them to the SDN flow controller for processing. Enforcement processing for this approach would be the same as noted in the DNP3 example above but using existing SDN flow rules rather than the OpenFlow *packet-in* and *packet-out* processing. In this approach, the SDN flow controller remains a passive component of data flow processing.

1. A packet would arrive on a port of the SDN switch using port TCP/20000 (arrow #1 in Figure 8-1).
2. An existing flow rule in the SDN switch would forward the packet to the IDS appliance (arrow #2 in Figure 8-1).
3. The IDS appliance would inspect the packet and perform its “normal” inspection and processing to determine of the packet should be forwarded or dropped (the Protocol Validation processing described in Section 8.5).

4. If the IDS determines that the packet is legitimate, it sends it back to the SDN switch (arrow #3 in Figure 8-1). If the packet is improperly formatted or contains basic protocol errors, it is logged, but no further processing occurs, and the packet is dropped.[55]
   - Note – The IDS appliance may expect that the two interfaces are on different IP networks and have different addresses so network translation may need to occur. This will require further investigation and may be dependent on the particular IDS device used.
   - Note – In this case, the IDS received the entire packet directly, so any fields or options present in OSI layers 2, 3, or 4 of the frame are available for additional inspection.
5. Another existing flow rule would receive the packet on the SDN switch and forward it to the behavior enforcement processing node for further analysis (arrow #4 in Figure 8-1).
6. The behavior enforcement processing node then examines the packet to determine if it contains any illogical (but syntactically valid) commands (the Configuration Validation processing described in Section 8.6), or if the packet contents does not make sense given the context of the packet (the Behavior Validation processing in Section 8.7). The behavior enforcement processing node also could receive context messages outside the scope of the SCADA protocol being inspected (the Dynamic Enable processing described in Section 8.8.2).
7. The behavior enforcement processing node then inspects the packet to determine if it contains any illogical (but syntactically valid) commands. This requires knowledge of the capabilities and features of the target device (e.g., from a configuration file) (the Configuration Validation processing described in Section 8.6).
   - If the packet contains illogical commands, it is logged, but no further processing occurs, and the packet is dropped.[55]
   - The protocol analysis processing can be very complex, especially for protocols that contain optional and variable fields or are embedded within other protocols. A permissive mode may be required whereby questionably logical commands are logged but allowed to be forwarded while further manual assessment is performed.
8. The behavior enforcement processing node then inspects the packet to determine if the commands or responses in the packet make logical sense given the context of the packet. The processing also stores context for future processing (e.g., to verify that a scan reply is expected, a preceding scan request needs to be captured) (the Behavior Validation processing in Section 8.7).

   There will need to be a permissive mode for this processing in the event that individual packets are missed, or the specific behavior of a device is not well understood. Permissive mode will assess and log potential illogical packets but will not block them from proceeding to the destination node. This will allow the packets to proceed through to their destination, especially those sent to build context for follow-on processing.
   - The packet state is stored for future processing.
   - If the packet does not make sense from a contextual standpoint, the packet is logged and potentially dropped if not in permissive mode.
9. If the packet is deemed to be valid (i.e., syntactically correct, logically correct, and contextually valid), it is sent back to the SDN switch, which forwards it to its end destination.

[55] using the same handshaking processing as in footnote [54]

- Additional coordination may be required if the packet needs to transit across multiple local SDN switches.
- Processing will be needed to make sure that rogue commands injected directly into the SDN infrastructure not from the external interface also are properly filtered.

10. If the packet contains illogical commands, it is logged, but no further processing occurs, and the packet is dropped.[55]
11. Since the packet is deemed to be valid, it is returned to the SDN switch using the same switch port (arrow #4 in Figure 8-1)[56].
12. The SDN switch would receive the packet and use existing flow rules to forward the packet to the destination EDS device (arrows #5 in Figure 8-1).
13. The behavior enforcement processing node could also receive context messages outside the scope of the SCADA protocol being inspected (the Dynamic Enable processing described in Section 8.8.2). For example, the behavior enforcement processing node could receive notification that a firmware download is about to happen. Normally, download packets would be dropped as invalid behavior, but could be enabled by a maintenance application that tells the behavior enforcement processing node that the download operation is valid. This would allow the behavior enforcement processing node to be aware contextually that firmware downloads are valid and allow them to be processed through to the end node. If the protocol supports an "end of download" packet, this could be processed to disable firmware downloads until another permissive request was received. Alternatively, a timeout counter could be started to allow the download packets for a set time from the initial packet or a set time between packets to accommodate slow or unpredictable performance communication links, or after a period of packet inactivity, or the maintenance application could send an indication once the firmware download is complete.
14. The behavior enforcement processing node could also perform additional context-based (the Dynamic Enable processing described in Section 8.8.2) or time-based (the Time-Based Enable processing described in Section 8.8.1) processing to allow packets to proceed, either through its context processing or by enabling and disabling seldom-used commands or command sequences by modifying network flow rules in the SDN switches. It would do this by making requests to the SDN flow controller to enable, create, disable, delete, or update specific network flow rules.

This approach has the advantage of using existing SDN capabilities without requiring modification of the SDN flow controller. It is also a more robust solution than alternatives which do not depend on inserting the SDN flow controller into the processing stream using fragile and inefficient REST APIs and does not introduce a single point of failure or chokepoint of the single interface from the SDN fabric to the SDN flow controller.

Another advantage of this process is that the inspection nodes will most likely be running a full IP (and TCP) software stack and will automatically reassemble packets that were fragmented because they were larger than a single Ethernet frame. This allows the entire logical payload to be assessed at one time. Because the fragments will be arriving in very close time intervals, the re-assembly delay should introduce a negligible delay. If the packets need to be re-fragmented before being sent to the end node, the IP software in the inspection node will automatically perform the re-fragmentation without any input from the inspection application.

[56] Although the same physical connection is shown, different implementations of the behavior processing node may use a separate connection to return the packets to the SDN switch.

However, the approach does have several disadvantages. Additional flow rule entries will be needed to move the packets through the SDN fabric to the behavior monitoring nodes and back to the target EDS devices. This is especially true in environments with multiple SDN switches where the EDS devices will be dispersed among multiple SDN switches, but only one set of protocol behavior enforcement nodes will be implemented. It also has the minor disadvantage of using additional network interface ports on an SDN switch.

However, the advantages far outweigh the disadvantages, thus making this the preferred approach.

This approach should work well for both SCADA-oriented protocols that generally operate on human speeds (e.g., seconds) like DNP3 and the manufacturing message specification (MMS) component of IEC 61850, as well as autonomous protocols like IEC 61850 GOOSE and SV that operate at faster speeds (e.g., milliseconds).

Both approaches using SCADA protocols also can take advantage of more customizable software in central stations that can be re-programmed to issue dynamic modification requests. Code changes in IEC 61850 relays and merging units, for example, to dynamically allow GOOSE messages, are highly unlikely unless the standards are updated, and even then, a corrupted device could still request illicit GOOSE messages. This could also potentially mitigate performance problems associated when misconfigured IEC 61850 devices generate extraneous GOOSE messages.

8.10 Conclusions

Traditional SDN implementations in general provide significant amounts of control over how data flows through a network, specifically for information contained in the lower layers (layers 1, 2, 3, and 4) of the OSI model.

Using SDN capabilities to intercept and process frames transparently while in transit between source and destination nodes, allows significantly more fine-grained assessment of the individual packets and provides the capability to provide additional enforcement of legitimate and contextually logical data and control exchange.

This additional control enhances cybersecurity and brings the control of network access to new levels. These features also facilitate the ability of SDN to apply new security automation (e.g., protocol validation).

Leveraging the well-known and deterministic behaviors of SCADA protocols can allow use of new security controls in OT-SDN environments. This implementation provides more reliability for SCADA systems while also improving SA for system owners. SDN4EDS research is discovering additional means, such as OP codes, for using the technology to improve cybersecurity in OT systems.

9.0 Distributing Behavior Processing to Field Locations

Contents of this section was initially published in the report "Software-Defined Networks for Energy Delivery Systems: Methods to Distribute Behavior Processing to Field Locations" in October 2020.

9.1 Introduction

This section of the blueprint architecture is to describe the approaches used by the SDN4EDS project to implement protocol inspection and enforcement in a laboratory environment. The approach used two separate IPS in series to process and filter network conversations using the DNP3 protocol. Using two separate IPS implements a level of defense-in-depth to the protocol inspection.

In a real implementation, additional resiliency and failure recovery procedures will be needed to ensure that communications, in particular inspected communications, can continue in the event of a failed inspection node. Some approaches may allow recovery from a failed inspection node by bypassing the inspection processing, thus allowing continued operations at a reduced security level. Other implementations will require redundant inspection nodes to allow continued operation of the network after an inspection node failure.

While not limited to use in EDS, OT-SDN was designed and purposed for EDS using Ethernet frame-based communications.[57] Being deny-by-default and by carefully traffic engineering restrictive flow rules, it is easy to see that OT-SDN Local Area Network (LAN) facilitates an IDS or IPS. Restricting the frame types being forwarded by the OT-SDN Ethernet switches not only reduces the amount of expected traffic needing to be processed by an IDS or IPS but also dramatically reduces the number of unexpected frames to be processed. A few of the additional methods that an OT-SDN LAN could use to augment the capabilities of an IDS or IPS are described below.

9.1.1 VLAN Tagging of Traffic by Circuit Categories

During traffic engineering of flow rules that define the circuits of communication in an OT-SDN LAN, the technician also can predetermine and tag each circuits' frames with a VLAN tag according to a category. Examples of allowed or engineered traffic categories for an OT-SDN LAN can be but are not limited to the following:

1. *SCADA Circuits* – These frames are designated as control and system state packets typically between a server system that aggregates control system data and IEDs that make automated decisions and actuate on the EDS. SCADA server systems poll and collect system state data from IEDs and also can issue a state change based on a larger understanding of the EDS. Ethernet frame protocols such as DNP3, Modbus, or IEC 61850 MMS are examples of types of packets that would be categorized as SCADA traffic.

[57] Note –OSI layer 2 communications happen at the "frame" level, while upper-layer communications such as IP at the OSI layer 3 level communicate using packets. For small messages, there is a one-to-one correlation between packets and frames; however, for large messages, the packet may be split into multiple frames for transmission. Layer 2 devices such as Ethernet switches can look only at individual frames. If inspection requires looking into packets that may be longer than individual Ethernet frames, an Ethernet switch cannot be used for that inspection.

2. *EDS Control Circuits* – These frames also are control and system state packets but are not between a mediator or server device and IEDs. Instead, they are directly between embedded IEDs for immediate and automated decisions. Because control data between IEDs is meant to be deterministic, control system communications typically are not over an Ethernet medium. Because OT-SDN removes most of the non-deterministic aspects of Ethernet, EDS control circuits can be packet-based in some circumstances. IEC 61850 GOOSE and CodeSys' Network Global Variables are two types of protocols that could be categorized as control traffic.
3. *Engineering Access (EA) Circuits* – These frames are for situations in which an EDS engineer or technician needs to interact with the IEDs in the OT-SDN LAN directly. The technician typically will use a local network computing workstation or a specified transient device such as a laptop or smart phone for EA functions. This type of communications in an EDS is considered rare and is typically planned and scheduled prior to taking place. These planned events will include technician functions such as the following:
   - Updating EDS devices with new firmware
   - Changing settings or updating configurations in EDS devices
   - Collecting event reports or system diagnostics data
   - Adding new devices or upgrading several devices on the system.

   Because they are not part of the normal operations of the EDS, these types of circuits can be ephemeral in the OT-SDN LAN, only being available during planned and scheduled function(s). *Cryptographically* encapsulated Telnet, File Transfer Protocol (FTP), or HTTP and authenticated and encrypted secure shell are example protocols for EA circuits. EA access may be enabled on a functional basis rather than allowing all EA-related access; for example, allowing the collection of event reports but not allowing changing configurations.
4. *Event Collection Circuits* – These frames are typically unsolicited in nature and include both cybersecurity events such as when an EA session occurs or during EDS state change events. *These* packets are typically generated at the IED level and sent to an aggregator and can be either expected or unexpected. These event collection circuits can be their own new category or can be tied to a pre-existing operational category. For example, expected cybersecurity events are already tied to EA functions so these event collection circuits can be categorized as EA circuits, whether or not the packets generated were expected. Subsequently, expected EDS state change events are already tied to SCADA, so packets created for expected or unexpected an EDS state change could be categorized as SCADA circuits. In addition to the already mentioned EA and SCADA protocols, other protocols including cryptographically encapsulated SNMP or Syslog can be tied to existing or new event collection circuits.
5. *Undefined Flow Rule Circuits* – Unexpected frames captured by an OT-SDN LAN switch without a pre-defined traffic engineered flow rule for forwarding are considered "rogue" frames. Rogue frames typically are captured by OT-SDN switches on unassigned or supposedly unused physical ports. Instead of disabling unused or unassigned physical ports for cybersecurity reasons as is done on traditional Ethernet networks, an OT-SDN switch can have special rules for unassigned physical ports. Rogue frames captured in an OT-SDN network can be sent directly to a designated IDS sensor or to an OT-SDN flow controller for further processing.

6. *Rogue Frames* – Rogue frames also can be sent from legitimate hosts using unexpected protocols or to unexpected destinations. These rogue frames can be handled in a similar manner to frames received on unexpected physical ports.

Additional categories can be considered for this enhanced intrusion detection method. Once a frame is received and categorized, it can be tagged with a specified VLAN such as ‘666’ for “Undefined.” Special consideration then could be given to frames received by an IDS or IPS using that VLAN tag. Frames with specific VLANs can be preprocessed by an IDS or IPS sensor. Tagged frames can be dropped for no further consideration, elevated in priority, or forwarded for additional scrutiny and correlated with additional events by the IDS or IPS. This enhanced method can improve both the pre filtering as well as time to execute IDS or IPS functions.

9.1.2 Interrogating Rogue Devices Connected to the OT-SDN LAN

This method builds upon the fact that unused or unassigned physical Ethernet switch ports can now be left enabled and open on an OT-SDN LAN switch. Unexpected rogue devices that connect into the OT-SDN LAN open ports typically will send out ARP requests immediately after connecting. As these frames do not have pre-engineered flow rules for forwarding, they can be instead sent on to the IDS for initial notification and awareness of existence. An IDS system or subsystem can respond to the ARP as well as to subsequent frames as the rogue device continues to enumerate and function on the network. Continued monitoring, mock responses, and interrogation of the rogue device by the IDS can provide additional information in a short time of the type of device and its intended purpose on the network. While this method is like a honey pot on a traditional Ethernet network, the OT-SDN network with a smart IDS can more readily and efficiently establish a fingerprint and purpose of the rogue device, leading to faster decision and response by the EDS owners.

9.1.3 Flow Rule Baseline and Monitoring

While it is possible to baseline the typical traffic on a traditional Ethernet network, the deny-by-default and traffic engineering aspect of the OT-SDN for EDS enables a specific “allowed only list” frame baseline with fewer false positives on rogue frames. With pre-engineered traffic and baselining, changes to the flow rules can now be easily detected and alerts can be made. This method is not waiting for an IDS to detect a malicious or new frame type in the network such as in traditional Ethernet networks. Secondary monitoring on a change in flow rules will be more advanced and alerts of potential harmful frames occurring inside the EDS network will be faster.

9.1.4 Traffic Metric and Metering Baseline and Monitoring

Unlike typical IT networks, EDS network traffic is constant and static in nature. Under nominal conditions, automated communications between the IED devices that are controlling the system and aggregating the data can be characterized and accounted for in an OT-SDN network. Static attributes of frame frequency, size, and their designated paths can be monitored in a pre-engineered OT-SDN network. If there is a change in the system or system state requiring secondary communications to occur or a change in frequency in present traffic, then in an OT-SDN network, this can be correlated to the event and baselined as normal EDS response. An IDS can learn and begin to capture or baseline what is normal versus what needs additional review and understanding. A mature IDS engineered as part of the EDS can now also be considered as purposed for the EDS. An EDS IDS can now more efficiently monitor and quickly alert on OT-SDN network traffic for the following types of traffic characteristics:

1. Changes in the frame frequency in a window of time for a specific flow given the EDS state
2. Changes in the frame size for a specific flow and frequency compared to other frame sizes for the same flow with the specific EDS states
3. Timing end-to-end through the network for a specific frame of a specific size
4. Timing of a frame per hop for its priority through the network given the present network load versus what is expected (precise time aware OT-SDN with Precision Time Protocol (PTP) can potentially detect the injection of a tap between devices simply sniffing network traffic)
5. Changes of frame frequency and time for the given part of the day or other outside variables such as weather and time of year.

These characteristics can enable an IDS with additional baselining and input for determining if a potential threat whether malicious or natural may be present in the system.

9.2 Architecture

SDN flow rules allow frames to be forwarded through the SDN switch fabric based on a number of parameters, including physical port, VLAN tag, IP address, and UDP or TCP port (identifying the specific protocol).

9.2.1 Addressing Concerns

In a traditional IP-based network, having a packet (or message) be forwarded through different computers for analysis requires that each computer have a unique IP address, and if the packets are expected to flow in on one interface and out on another, the two ethernet ports need to be on different IP networks. This means that each inspection step along the way must at a minimum change the IP address of the packet, potentially to a different IP network for the OS of the inspection node to properly forward the packets to the next step along the way. This requires address configuration changes to nodes at one or both ends of the communication link (i.e., the master station located at the central control center or the outstation located in the field) when inspection nodes are added to the communication channel.

Figure 9-1 shows a network containing two nodes communicating on the same IP network. In this figure, a master station named NODE 1 at IP address 192.168.100.1 communicates with an outstation named NODE 2 at address 192.168.100.2. The outstation responds to the master station at address 182.168.100.1.

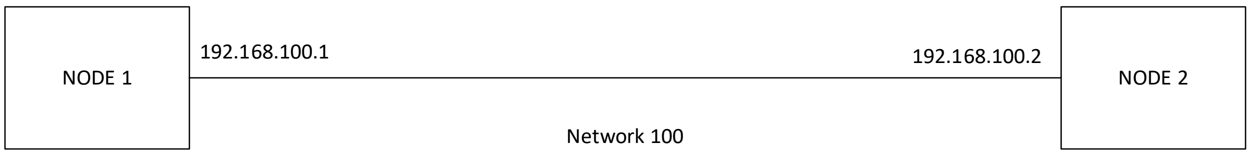


Figure 9-1. Original Network

When a traditional firewall is inserted between the nodes, it acts as a router requiring that each side of the firewall must reside in a different IP network as shown in Figure 9-2. This allows the traditional firewall to act as a filtering router between the two IP networks. Firewalls can act as either a network proxy or a network gateway. In this figure, the master station named NODE 1 has been re-addressed to allow the traditional firewall to be inserted in the communication path.

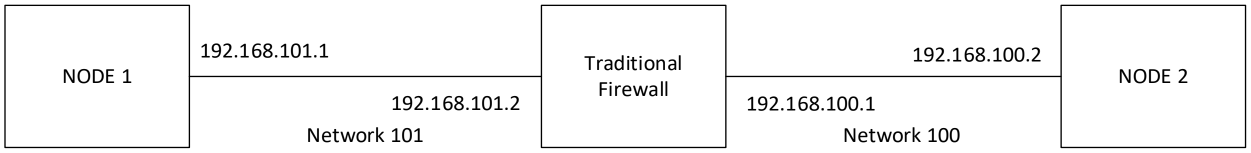


Figure 9-2. Network with Traditional Firewall

If the firewall is acting as a network gateway, the master station NODE1 specifies that the traditional firewall act as the network gateway between IP networks 192.169.101.0 and 192.168.100.0. NODE 1 sends all packets destined for any address on network 192.168.100.0 to the firewall address 192.168.101.2, and the gateway processing in the firewall forwards the packet to the outstation NODE 2 on its 192.168.100.1 port. NODE2 also specifies the traditional firewall as the network gateway between the same two IP networks. Responses from NODE2 to NODE 1 are sent to the firewall at address 192.168.100.1, and the same gateway processing in the firewall to forwards the packet to NODE1 from its 182.168.101.2 port.

If the firewall is acting as a network proxy, the master station named NODE 1 has a new address (192.168.101.1) and appears to communicate with the outstation named NODE 2 at address 192.168.101.2, but in reality, it is communicating with the traditional firewall. The traditional firewall then inspects the protocol packet, and re-formats its IP address as if it were coming from a master station at address 192.168.100.1 and forwards it to the outstation at address 192.168.100.2. The outstation then responds to the traditional firewall as if it were the master station using address 192.168.100.1. The traditional firewall inspects the packet, adjusts the addressing, and sends it to the master station at address 192.168.101.1.

If the firewall processing is being retrofitted into an existing configuration, a network re-addressing exercise is required to adjust addressing or specify a network gateway. Although there are fewer addresses to adjust if the changes are made at the master station, inserting the traditional firewall in a running system can be a complicated process, often involving a slow migration from the old addressing scheme to a new scheme and extensive testing to ensure that all the expected communication paths still work. If using a gateway firewall, all nodes will need to be modified to specify the network gateway.

However, because SDN operates at layer 2, it can have flow rules written to inspect the traffic arriving on a specific physical switch port and forward it through the SDN fabric without the need to change IP addressing. This is implemented using a “bridging firewall” or a “transparent firewall,” and requires support from the inspection nodes to bypass the IP protocol stack and address or network management functions in the node’s OS. This requires implementation of additional software in the application to decode the network and session (IP and TCP/UDP) layers to properly handle the frames. Note that if the IP stack is bypassed in the operation system, the IP processing in the application will need to re-assemble multi-frame packets into a single packet for further inspection, and then re-fragment any packets that will be sent on to the next node in the sequence.

The major advantage of bypassing the host node’s IP and TCP stack is that the inspection node operates as a transparent device to the network. It can receive and transmit packets independently of IP address and does not need to follow network routing rules where different interfaces on the inspection nodes need to be assigned to different IP networks allowing the inspection node to operate as a router.

When implementing a bridging firewall in an SDN environment, the SDN flow rules need to be carefully crafted to segregate support traffic (e.g., ARP requests) from one side of the firewall from similar traffic on the other side.

Figure 9-3 shows how a transparent firewall is implemented with minimal disruption to the existing network addressing. The figure shows the original two nodes communicating with each pother using the same addresses. In this case, the transparent firewall masquerades as the source or destination for traffic between the nodes. Neither node requires any configuration or address changing for the traffic to flow from NODE 1 to NODE 2, but all traffic passes through the transparent firewall. The resulting configuration consists of two distinct and separate networks (“Network 100 – A” and “Network 100 – B”), each containing nodes with addresses 192.168.100.1 and 192.168.100.2. However, the ARP tables on each distinct network contain different MAC addresses associated with the IP addresses. The isolation would typically be implemented using physically isolated network switches, a direct cable connection on one side of the transparent firewall, or a switch-enforced VLAN configuration. The SDN environment having a shared control plane with access to both logical networks by the SDN controller must allow for this, and not inadvertently create SDN flow rules that pass traffic from one network to the other.

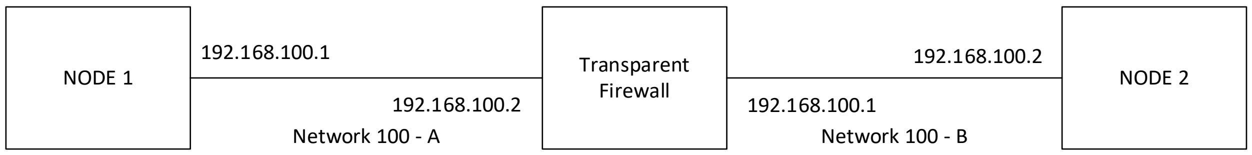


Figure 9-3. Network with Transparent Firewall

9.2.2 Resiliency Concerns

In an operational environment, availability is a key consideration. Any architecture that introduces a single point of failure must be modified to eliminate that failure point. This is typically accomplished through the use of redundant or backup processing or communications, such as a standby processor or interface that monitors the health of the primary and, when a failure is detected, takes over the failed function, or a redundant process or interface that shares the load but is capable of assuming full capability in the event of the failure of one process. Both of these require either sharing the “state” of the process being recovered or allowing a re-start of the state processing during recovery.

In some cases, especially those for which continued operation of the process is deemed more important than recovering the failed function, processing for the failed component may be routed around the failed component bypassing it. If additional components can assume some of the functions, the process continues with minimal impact; however, if the failed function cannot be subsumed by other components, the overall process can continue in a diminished capacity. Determining whether this is acceptable to the process must be done on a case-by-case basis.

In the case of an intrusion detection function, the overall impact to the process is negligible, but the process continues without SA of potential attacks. In the case of an IPS or firewall function, the process can continue but in an unprotected state. If multiple IPS or firewall devices are configured in series (i.e., valid packets pass through all the devices, but invalid packets are stripped off by the first device that detects the invalidity), then the failure of one device can be mitigated by including similar filters in the others.

SDN flow rules can be used to aid in the recovery of failed firewall nodes. In the case of the transparent firewall, because the firewall node appears to either end device (e.g., NODE 1 and NODE 2) as using the IP address of the other device, neither end device needs to have any addressing changes to communicate with the other device.

For the traditional firewall, an alternate stand-by firewall configured with the same IP addresses as the primary firewall can be configured and enabled by the SDN failure recovery processing. Because both the primary and alternate firewall have the same addressing and rulesets, no address updating is required in the end devices (e.g., NODE 1 and NODE 2) to allow continued operations.

In either case, the ARP tables on the end devices (e.g., NODE 1 and NODE 2) will need to be updated following a recovery action taken by the flow rules, unless supplemental MAC addresses (e.g., multicast MAC accesses) are used by the firewall nodes.

9.2.3 Intrusion Detection System Configuration

In a traditional network environment, the individual Ethernet switches need to be configured to forward all traffic to a SPAN port on the switch. In a single switch environment, the IDS can be directly connected to the SPAN port and see all the traffic in the switch. In an environment with many switches, the SPAN ports all need to be connected to a second network with the same SPAN port configuration in order to use a single IDS.

For a simple detection application, scaling of the traffic for inspection can lead to performance and availability issues. All traffic for an entire switch will be sent to the SPAN port, and if a monitoring network is used, all traffic from all the monitored switches will appear on the monitoring network and be further consolidated onto the monitoring network's SPAN port, potentially saturating the network, making the network appear to be performing differently than it actually is, and forcing some packets to be dropped. This may present network performance issues if the amount of traffic exceeds the capabilities of either the SPAN port or any component of the monitoring network resulting in the switches dropping traffic to the IDS.

To overcome this, the monitoring network would need to be segregated into multiple networks connected to multiple separate IDS nodes. Each IDS node would inspect traffic from its set of traffic, but event correlation between different network segments would need to be performed external to the IDS. If network performance is not an issue, traffic could be replicated and sent to different IDSs, each specializing in different protocols. This would allow, for example, an IDS that is designed to monitor web (i.e., HTTP) traffic to receive all the traffic and ignore all but the HTTP traffic, while an IDS that is designed to monitor DNP3 traffic would also see all the traffic and ignore all but the DNP3 traffic.

In environments with redundant networks either using dual or redundant networks, or those using Parallel Redundancy Protocol (PRP) or High-availability Seamless Redundancy for delivery, the redundancy requirements likely mean that multiple IDS nodes will be required to ensure that the IDS sees all the traffic regardless of which physical or logical network it flows on.

An SDN environment can use flow rule processing to simplify inspection requirements and forwarding traffic for inspection throughout the SDN environment allows more flexible and potentially more efficient implementation of an IDS.

An SDN environment normally allows traffic to be rejected at the edge of the network based on flow rules. By using carefully crafted SDN flow rules, a significant amount of unnecessary traffic can be kept from the network. SDN flow rules allow inspecting and matching of the physical switch port, MAC address, IP address, and TCP/UDP port number before allowing any traffic to enter the SDN infrastructure, although wild card matches are allowed. If any frames do not match the flow rules, the frame is rejected. This means that if a command is received with the wrong IP address or arrives on the wrong port, it will not be processed. This can be used to eliminate simple rogue devices such as unconfigured master station IP addresses or rogue devices connected to the wrong physical switch port.

It also means that if there is no rule for how to process web (HTTP port TCP/80) traffic, it will not be accepted into the network so there should be no HTTP traffic to inspect. This can greatly reduce the amount of traffic on the network that needs to be inspected in an OT, where typically a limited number of protocols are allowed.

By using SDN flow rules, the traffic can be copied and forwarded to an IDS connected to the SDN data plane with all the traffic flowing "in band" without requiring additional hardware or a separate "monitoring" plane. This eliminates the cost of installing separate hardware and managing a separate monitoring network, while using the SDN flow rules to maintain segregation of the monitoring traffic from the real traffic in the data plane.

In the event of performance issues, adding an IDS node could be as simple as adjusting some of the flow rules to forward different subsets of the traffic to different IDSs on separate switch ports.

The IDS could also easily be moved from one switch port to another with some simple SDN flow rule updates, for example, to attach the IDS to a 1-Gb or 10-Gb port to mitigate some of the network saturation issues.

9.2.4 Intrusion Prevention System Configuration

While an IDS only need to observe network traffic to analyze it, by their nature, IPSs must see all traffic to be able to take actions to block or drop it to prevent it from being forwarded. Because of this, the IPS must be inserted in-line with the communication so it can see all the traffic and can assess all the packets in both directions and maintain the status of responses to requests.

IPSs in traditional environments cannot be easily expanded by adding parallel processing nodes to overcome performance issues. Similarly, an IPS must be see and inspect traffic for all protocols in the communication. Unlike the parallel processing discussed above, a single IPS cannot simply ignore traffic it is not designed to inspect – it must either process all traffic, or multiple IPS nodes in series must be implemented to inspect and process all traffic.

By using SDN flow rules to segregate traffic, different IPSs can be implemented for different traffic to block malicious traffic. For example, a flow rule can be set up to forward all DNP3 traffic to a DNP3 IPS, all Modbus traffic to a Modbus IPS (if different than the DNP3 IPS), and all engineering traffic to an IPS that is more general purpose (i.e., IT-centric) for handling command line and web access.

SDN flow rules can also be implemented that allow multiple different IPSs to inspect the same communication streams and protocols, implementing a defense-in-depth approach to provide additional inspection for critical protocols.

SDN flow rules that match on TCP/UDP port numbers allow protocols to be processed differently by the frame forwarding rules, even if the messages arrive on the same physical switch port from the same valid IP address (and even *to* the same IP address number), the frames can be forwarded to the appropriate IPS node based on the TCP or UDP port number.

9.2.5 Intrusion Detection System or Intrusion Protection System Selection Considerations

The same SDN flow rules that support the IDS or IPS configuration can be beneficial when considering how to implement a “best of breed” IDS or IPS in the SDN environment. In many cases, especially in a traditional ethernet environment, for the IDS or IPS to perform its function, it must be configured to see all traffic. Since a traditional network cannot easily parse traffic based on protocol, for overall performance reasons a single IDS or IPS is often implemented to minimize latencies introduced by the packet inspection, meaning that the IPS is selected as a compromise to detect all malicious traffic for a variety of different protocols and uses. Even though some IDSs or IPSs perform better for OT (e.g., SCADA) traffic, yet others perform better for traditional traffic, the selected IDS or IPS needs to perform “well enough” for both.

Using SDN flow rules to parse and forward the traffic also would allow different rulesets or different IDS or IPS technologies or implementations based on the kinds of traffic that might be expected to be seen allowing the use of the best IDS or IPS for each protocol or data flow type. For example, one implementation of one IDS or IPS could focus on the operational protocols like DNP3 or Modbus using those protocols and ports, while implementation of a different IDS or IPS could focus on EA and management functions using more IT-like protocols and ports. Depending on the specifics of the protocols used or the preference of the utility, the two IDSs could be the same model with different ruleset focuses, or the IDSs or IPSs could be different models or from different manufacturers.

9.2.6 Overall Implementation

Protocol enforcement for the project uses two IDS or firewall devices configured in series so both IDS devices can potentially assess and block any traffic. The implementation places the Binary Armor (BA) IPS first in line to block any guaranteed malicious traffic (e.g., device resets) or traffic known to be unsupported by the end DNP3 devices (e.g., “direct operate” in cases where “select before operate” is expected). The configuration then places a Suricata IPS to inspect all the traffic that the BA IPS allows through. Similar rulesets are configured in both IPSs to implement a defense-in-depth inspection approach, while allowing each IPS to take advantage of different inspection approaches and processing.

The BA device can be configured to block traffic based on specific function and group codes with little additional programming. While the BA device can be configured to allow (or block) upon receipt of an “override” code from the configuration utility, most rule changes require a device reset to implement any rule updates. This means that there is no communication until the BA device has completed its reset.

On the other hand, the ruleset for the Suricata device can be modified on the fly, and the interface for updating rules is not dependent on the supplied configuration utility, making it more “nimble” and programmable outside of supplied tools. The Suricata device also can be configured on a more finely grained basis without significant additional programming effort.

9.3 Experimental Setup

The experimental setup used for this report uses two of SEL 2740S SDN switches of the five available in the laboratory setup. These switches will be referred to as S1 and S2. The SDN switches are interconnected with a cable connected between port D2 of switch S2 , and port D1 of switch S1. By convention, we refer to these switch-port pairs as “switch:port” so the interconnection is from S2:D2 to S1:D1.

The SEL 5056 SDN Controller (version 2.2), compatible with OpenFlow v. 1.3, running on a Windows Server. The SEL 5056 SDN controller communicates with SDN switches using an in-band management plane in the 192.168.10.x/24 subnet. All experimental devices are connected in the control plane on the 192.168.1.x/24 subnet.

A Raspberry Pi is configured to serve as a DNP3 master station with IP Address 192.168.1.17 connected to the SDN network at S2:C2. Another Raspberry Pi is configured to serve as the DNP3 outstation with IP address 192.168.1.18 connected at S1:B3. The two Raspberry Pi devices exchange data using the DNP3 protocol using port TCP/20000 using a script to generate the traffic simulating typical interactions between a DNP3 master station and its outstation.

Two separate IPS installations inserted between the DNP3 master station and the DNP3 outstation inspect all traffic flowing between the devices. Either IPS can reject or drop traffic that is determined to be unacceptable, meaning that if the first IPS drops traffic, it will not be seen by the second IPS or the destination station. However, if the first IPS passes the traffic, the second IPS still has the opportunity to drop it if it is unacceptable. This provides for defense-in-depth for the inspection process by requiring all valid traffic to be passed through two separate inspection engines with different rulesets.

SDN flow rules allow management of traffic by routing it from the DNP3 master station through the first IPS, then through the second IPS, and finally to the DNP3 outstation. The reverse traffic is routed from the DNP3 outstation through the second IPS and then through the first IPS, and finally to the DNP3 master station.

The following sections describe the setup of the two IPS devices and the SDN flow rule configurations. Section 9.4 describes the setup and configuration of the first IPS, the Binary Armor SCADA Network Guard. Section 9.5 describes the setup and configuration of the second IPS, the Suricata IPS. Section 9.6 describes the SDN flow rules used to manage traffic flows through the SDN network fabric.

Note that the emphasis of this report is on the ability of an SDN environment to insert multiple IPS devices into the logical flow between the DNP3 master station and the DNP3 outstation. While some issues and gaps associated with protocol inspection are noted, they are not the emphasis of this report.

9.4 Binary Armor SCADA Network Guard Intrusion Prevention System

The BA SCADA Network Guard IPS from Sierra Nevada Corporation (SNC)[58] provides a secure robust platform suitable for deployment in remote field locations like substations. It is "… designed to be installed in-line between PLCs, RTUs, IEDs or controllers and the WAN/LAN."[59] It can also be deployed as an IDS. It is a deep-packet inspection system, that "processes every byte of every message."[60]

The BA device has two network interfaces, referred to as "HIGH" and "LOW." The HIGH side interface connects to the devices (e.g., IEDs or relays) that are to be protected, while the LOW side connects to external networks containing devices that are not protected. In a substation environment, the HIGH side connects to the substation LAN, while the LOW side connects to a WAN interface.

The BA device acts as a transparent firewall replicating IP addresses and allowing the BA device to be inserted into an existing network infrastructure without creating additional IP networks or performing any end-node re-addressing as discussed in Section 9.2.1.

9.5 Suricata Intrusion Prevention System

According to the Suricata website,[61] "Suricata is a free and open source, mature, fast, and robust network threat detection engine … [that] is capable of real time intrusion detection (IDS), inline intrusion prevention (IPS), Network Security Monitoring (NSM) and offline PCAP processing." It is a multithreaded software package that supports integration with tools such as existing SIEM systems, Splunk, LogStash, Kibana, and other databases. The scalable TCP/IP engine within Suricata contains a flow engine, numerous protocol parsers, and application protocol decoders. The detection signature engine within Suricata is compatible with most open-source Snort Signatures as well as containing an advanced Lua scripting engine to allow the implementation of advanced detection and flow controls rules.

Suricata can be configured as an IPS in a router configuration using Netfilter rules or may be configured as a layer-2 IPS using an AF_PACKET interface. As a layer-2 IPS, Suricata may be implemented as a transparent bridge within a network environment to protect devices and appliances.

9.6 Configuring Multiple IPS

In some cases, a single IPS cannot easily accomplish all the filtering that is desired or may be difficult to configure or customize. Some IPSs may offer better filtering in one area (such as detecting malformed packets), while other IPSs may provide a more flexible deep packet inspection. Some IDSs may easily be able to modify rulesets, to respond to dynamic conditions, while other are more rigid.

[58] https://binaryarmor.com/, (Accessed 10/13/2020)
[59] See https://binaryarmor.com/product/binary-armor-scada-network-guard/ (Accessed October 8, 2020)
[60] BinaryArmor_SCADA-SpecSheet_05-18-20.pdf, available on binaryarmor.com website after entering contact information (Accessed August 27, 2020)
[61] https://suricata-ids.org/, (Accessed August 17, 2020)

To address this issue, the SDN4EDS project implemented two different IPSs in series to test how a single SDN environment could be adapted to mimic what would normally be three separate OSI layer-2 LAN environments allowing multiple IPS instantiations to operate on the same DNP3 data stream. The basic setup of the IPSs is shown in Figure 9-4. Note that the configuration shown is not resilient, but SDN flow rules could be created to provide recovery from the failure of an individual IPS by detecting the failure and forwarding the frames around the failed IPS.

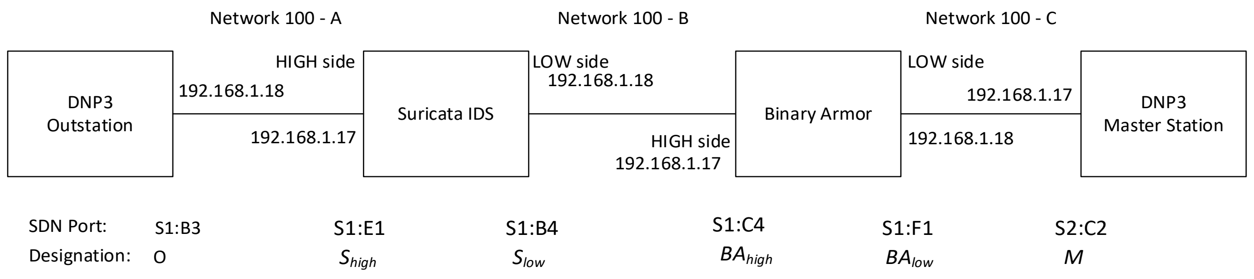


Figure 9-4. Multiple IPS Configuration

These connections are shown in table form in Table 9-1, and the physical cable layout in the SDN4EDS laboratory is shown in Figure 9-5.

Table 9-1. Port and IP Address Assignments

Device	Role	SDN switch:port	IP Address (if applicable)
Binary Armor - VM	BA Configuration Manager	In-band Controller	192.168.10.4
Binary Armor - LOW	Unprotected LAN or WAN	S1:F1	192.168.10.100 (Mgmt port) 192.168.1.18 (DNP3 traffic)
Binary Armor - HIGH	Protected SCADA or ICS	S1:C4	NA
DNP3 master station	DNP3 client	S2:C2	192.168.1.17
DNP3 outstation	DNP3 server	S1:B3	192.168.1.18
Suricata - LOW	Bridged IPS	S1:B4	NA
Suricata - HIGH	Bridged IPS	S1:E1	NA
Switch2 to Switch1	Trunk	S2:D2 to S1:D1	NA

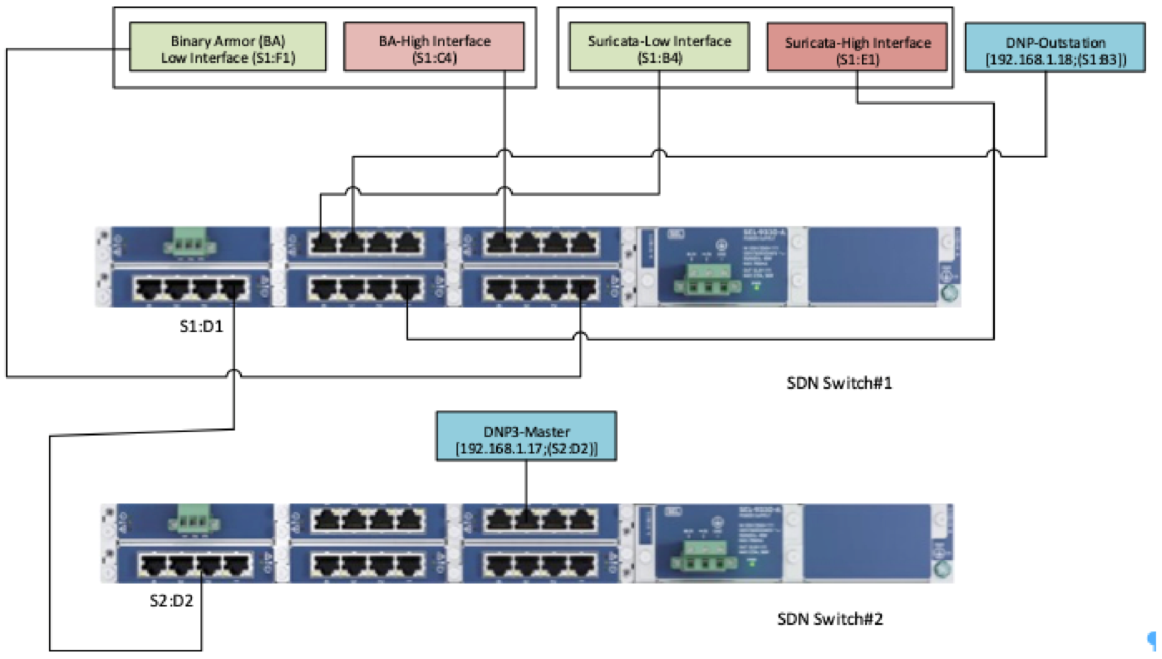

Binary Armor (BA) Low Interface (S1:F1)
BA-High Interface (S1:C4)
Suricata-Low Interface (S1:B4)
Suricata-High Interface (S1:E1)
DNP-Outstation [192.168.1.18;(S1:B3)]
S1:D1
SDN Switch#1
DNP3-Master [192.168.1.17;(S2:D2)]
S2:D2
SDN Switch#2

Figure 9-5. Physical Cable Configuration

The SDN flow rules for the configuration were created to allow traffic to flow between the DNP master station and the DNP3 outstation through both the BA IPS and the Suricata IPS without requiring any addressing or configuration changes on either the DNP3 master station or the DNP3 outstation. The SDN flow rules allow both IPS devices to operate as transparent bridge nodes, effectively performing MITM address masquerading and creating three logically separate LAN environments as shown in Figure 9-4. Each of the LAN segments represents an independent “ARP domain,” allowing the TCP/IP protocol stacks on the DNP3 end devices to operate normally. That is, the DNP3 outstation sees the Suricata HIGH side port as IP address 192.168.1.17, while simultaneously allowing the Suricata LOW side to see the Binary Armor HIGH side port as IP address 192.168.1.17 and allowing the BA LOW side port to see the DNP3 master station as IP address 192.168.1.17. The rules were developed using the OpenFlow syntax and comprised of matching packets by their ARP requests for ARP resolution, the physical switch ports the devices are connected to, the devices IP addresses, the TCP protocol, and the TCP port number in which the standard DNP3 protocol operates over.

Because of the nature of operation of the BA and Suricata devices, rules had to be made between end points and the devices that sat between the DNP3 nodes to forward traffic to and from BA and Suricata devices, but also resolving the DNP3 communication by forwarding the appropriate request and responses back. Given a set of devices *D* that contains the elements $D = \{BA_{low}, BA_{high}, S_{low}, S_{high}, M, O\}$ where BA_{low} is the BA device’s “LOW” interface, BA_{high} is the BA device’s “HIGH” interface, S_{low} is the Suricata device’s “LOW” interface, S_{high} is the device’s “HIGH” interface, *M* is the DNP3 Master station device, and *O* is the DNP3 outstation device, the high-level flow of traffic is determined be the following configurations, assuming that *M* initiates communication:

1. M sends and receives ARP to BAlow
2. BAlow sends and receives ARP to M (impersonating as O, thus acting as a “Man in the middle”)
3. M sends and receives DNP3 traffic to BA_{low}

4. BA_{low} sends and receives DNP3 traffic to M. Upon receiving DNP3 traffic destined to O, BA_{low} forwards the traffic to BA_{high} via an internal bridge between the BA device's two interfaces
5. BA_{high} (impersonating as M) is configured to forward and receive the DNP3 traffic to the S_{low} interface
6. S_{low} forwards incoming traffic to the interface S_{high} via an internal bridge
7. S_{high} forwards the incoming traffic to O. This observation effectively means that between M and O sits four elements, with the BA_{high} , S_{low} , and S_{high} impersonating M.
8. *O* sends and receives ARP and DNP3 traffic to S_{high}
9. S_{high} forwards the matched traffic to its other interface S_{low}
10. S_{low} forwards the incoming traffic to BA_{high}
11. BA_{high} forwards the matched traffic to its other interface BA_{low}
12. As specified in configurations 1 through 4, M and BA_{low} will communicate as if M was communicating directly with O.

SDN allows this configuration by creating flow rules that match not only in physical MAC address and IP address but also on the physical switch port the traffic arrives on. In OpenFlow, traffic that is "matched" based on these criteria have several options, among these is the *output* action. Using this action, frames can be directed to specific physical ports regardless of where the protocol packet headers (i.e., IP addressing information) would indicate the frame be sent (a difference from standard layer-2 network switching). Thus, the true path to which the traffic is forwarded can be abstracted from the endpoints while allowing middle devices to perform the necessary operations, in this case checking for DNP3 function codes and determining whether they are appropriate to forward or not.

9.7 Conclusions

We have reported just an overview of simple IDS and IPS methods that are available for EDS OT-SDN networks. Not only should these be researched more, but there are additional methods that can be realized through research. In addition, this brief does not cover or uncover the additional OT-SDN controller applications that also could be considered and researched for augmenting an IDS' or IPS' capabilities on an OT-SDN network. There also are the aspects of additional research and ideas regarding Advanced Persistent Threats. Behavioral monitoring and inducing an adversary to "*readily*" reveal their tactics, techniques, and procedures is another aspect of an OT-SDN system's capability that should be researched.

10.0 Decision Process

This section of the blueprint architecture discusses the decision process that a utility could use to determine where SDN could be used within their environment. It contains example questions to be used internally to determine where SDN can be deployed, what network characteristics make sense for an SDN deployment, how flow rules should be developed and created, etc.

Also included in this section is a discussion of an example process that could be used to gradually roll out an SDN environment into a large existing traditional networking environment, and a discussion of how the migration or rollout could be accomplished.

The Brownfield Commissioning discussion in Section 11.24 provides some additional information on migrating from a traditional environment to an SDN environment.

10.1 Migrating to SDN

Implementing SDN is highly contextual based on the infrastructure and services identified for virtualization. To prepare for the transition to SDN, some version of the following questions need to be answered:

- What are my goals for migrating to open SDN?
- What are the migration options available to achieve my goal?
- What steps and what sequence of those steps should be taken to achieve my goals?

For the purpose of this section, we propose the goal to be set as:

> Establishing a secure, virtualized, scalable control plane for the purpose of adopting standardized, packet-switched technologies for data communications within, to, and from the electrical distribution edge.

The migration options can be described as legacy to greenfield and legacy to hybrid. For the purposes of this paper, the legacy to hybrid (i.e., a network containing both SDN and legacy equipment) approach will be assumed.

The steps and sequence of steps is to introduce and migrate critical control flows from legacy to the new hybrid network can be further defined by careful definition of the following variables:

- What is the current state of the soon-to-be hybridized network?

  There is a very valid argument that, while a stand-alone SDN controller, serving a specific function (i.e., abstract forwarding plane, defining logical flows, instantiating VMs, or optimizing multiprotocol label switched [MPLS] paths) can be implemented for the purposes of achieving a specific goal; an automation backplane platform underpinning said SDN controller and other complementary controllers can provide business value via the ability to provide a common set of APIs, common provisioning language, a common set of networking primitives, a common big data database, and a common time series database. Such a platform could serve as a virtual underlay for automation, telemetry, orchestration, and management functions for future deployment of similar SDN technologies but also existing element management, work order, and business logic systems.

- What are the core requirements of the new hybrid network architecture? How are the requirements prioritized?
- What security paradigm will be implemented for the new, hybrid SDN environment? How will it be firewalled off from traditional business application infrastructure? Is there a framework to ensure trust if additional SDN controllers are added?
- What is the plan to implement a phased introduction of and migration to SDN elements within the data or control center, the WAN, and the distribution edge (substation)? What substation has been identified for the first semi-live deployment?
- To what extent will the new control plane be physically distributed to ensure resilience and reliability?
- How will the initial deployment be staged and tested prior to deployment into production substations? What are the criteria for success?
- How will each phase of the design and testing be validated before moving to the next phase?

10.2 Installing and Maintaining an SDN Environment

As an overlay technology, SDN is the proper first step for introducing next generation technology into the control network. Automating the underlay would be extremely difficult to accomplish and would not benefit from the common APIs, common Service Primitive, and common provisioning language strategy that is employed for SDN overlay technologies.

- *Order and Fulfillment*: Self-care portal provisions services based on existing repeatable templates to deliver services in minutes.
- *Control*: Customers have substantial visibility into and control over their services, giving them the ability and flexibility required to activate, modify, remove, and relocate services. Requested changes are automatically configured in the network with fewer errors.
- *Security*: Automated security detects malicious traffic and enforces policies designed to safeguard network access.
- *Policies*: Policy-based service management adjusts network resources, including bandwidth and traffic priority, allowing the network to dynamically provide differentiated services and role-based access.
- *Assurance*: Proactive error detection and fault reporting provide insights that enable network operations to reroute traffic and limit service disruptions.
- *Performance*: Automation provides active traffic management while maintaining service performance objectives.
- *Analytics*: Analytics capabilities enable service data to be collected and analyzed from across the network domain for network optimization purposes.
- *Usage and Reporting*: Reporting features record and measure usage patterns, traffic volume, and any specific usage of network resources for network planning purposes.

11.0 Operational Use Cases

Contents of this section was revised in Version 3 (September 15, 2018), and further revised in the report ".Software-Defined Networks for Energy Delivery Systems: Business Function Use Cases" in November 2020,

Security-focused use cases are provided in Section 4.0 to illustrate how SDN can provide cybersecurity natively or as additional security protections. The use cases described below explain how SDN can meet operational expectations and requirements of an EDS. Concepts such as isolation and testing, patching, substation automation, and prioritization of communications are addressed.

11.1 IEC 61850 traffic stream separation and failure recovery

The use of IEC-61850 currently is limited but growing in use in the United States and is more widely used in Europe and other parts of the world. This operational use case is focused on the ability of SDN technology to support using all available communication pathways simultaneously during normal operations, assigning separate protocols or application communication to each communication media, designing failover and priority when a communication failure occurs, and recovery from the failure. The illustration of SDN capabilities could have been made using DNP3, IEEE C37.118, and metering data and their separate or shared communication infrastructure. The concepts described are transferable. The SDN4EDS project team acknowledges the previous work establishing the PRP and views SDN as an enabler of PRP, not a competitor. The following discussion applies equally to both PRP and non-PRP networks.

This use case examines how SDN can recover from equipment failure with minimal or no impact on operations.

- As shown in Figure 11-1, IEC 61850 supports three different protocols: 1) SV, 2) GOOSE, and 3) MMS. To make the applications work to specifications, PTP network time synchronization or the use of out-of-band Inter Range Instrumentation Group time synchronization is required.
- We explored how SDN can prioritize the communication of each sub-protocol during normal as well as degraded operating environments, an example of which is shown in Figure 11-2. This work will include testing the capabilities to only deliver the packets the end device wants to process when the sources have individually tagged traffic (different VLANs) and, when all sources are on the same VLAN. The goal is to only deliver SV and GOOSE packets to the subscribers that want them and not burden other devices. We will also test how packet injection of multicast control messages impacts the system while documenting the attack surface.

  SV and GOOSE messages will have highest priority followed by time (second), MMS (third), and EA (lowest). The team will evaluate improvements to priority through flow path planning and as well as options this conversation-based orchestration enables. DoS and other attacks can be applied, and results captured.
- Station bus and process bus also are common solutions in separation of traffic due to the limited capabilities in traditional networking for multicast traffic so the team will re-evaluate the possibilities with SDN and how that makes packet delivery more reliable and secure.

The team will test the impact of removing an SDN flow controller on the operational network.

- SDN also allows you to design failover and combine or stop and only send priority traffic in microseconds, the team will evaluate how this new level of performance supports and impacts the network design and work that is needed to be done. Network recovery time to guarantee GOOSE message delivery must be less than 4 milliseconds. Network recovery can be tested by capturing GOOSE traffic, looking at the sequence numbers in the GOOSE packets, and verifying that none are missing.
- The same approach applies to more commonly used domestic protocols such as DNP3, Inter-Control Center Communications Protocol, IEEE C37.118, and metering communications.
- Test and evaluate the solution that it is working as expected.

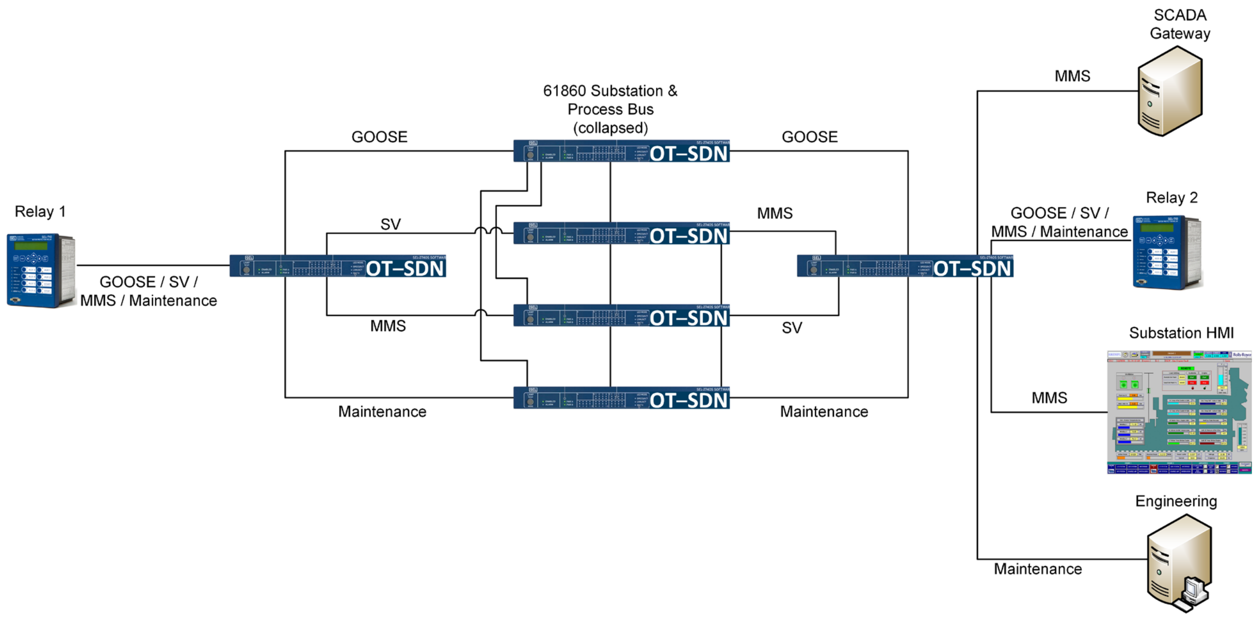


Figure 11-1. Traffic Stream Separation – Normal Case

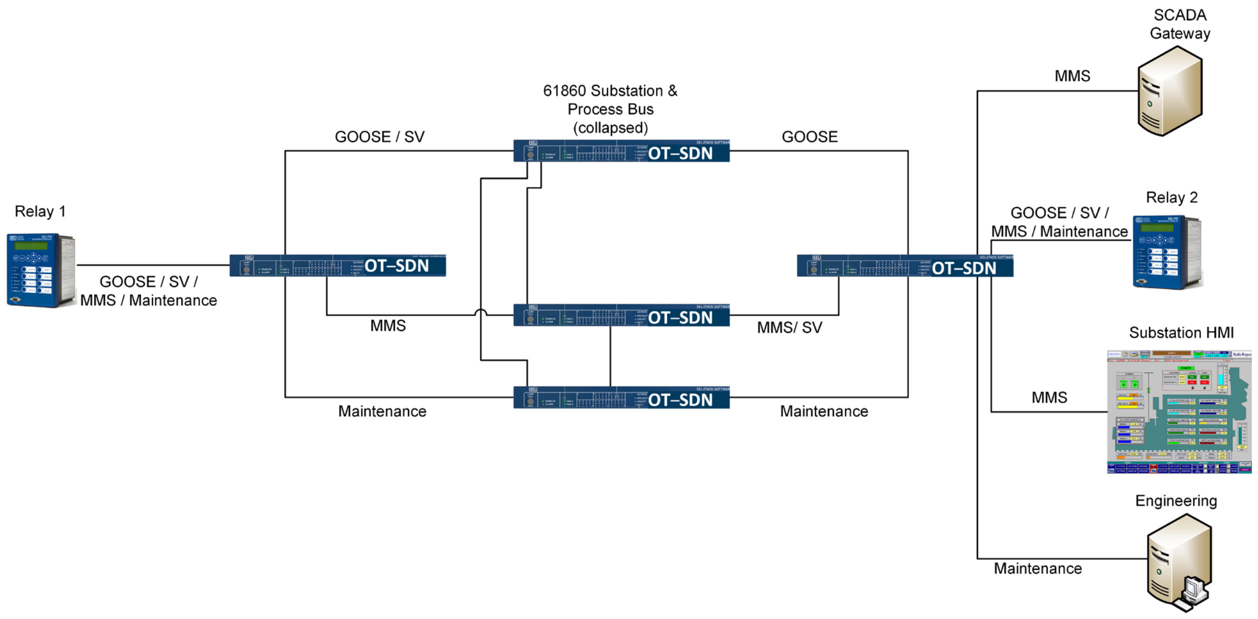


Figure 11-2. Traffic Stream Separation – With Equipment Failure

11.2 Testing or Isolation within a Substation

As the networking and communication infrastructure in a substation becomes more complex, the need to periodically test the functionality of the equipment becomes more important. The use of an SDN environment allows individual components to be isolated from the production network and be tested without impact. This functionally is equivalent to the process of removing a protection relay from service to allow a relay technician to calibrate and test its functionality while leaving the equipment physically mounted in the substation.

This use case examines how SDN allows equipment testing in an operational environment with minimal or no impact on operations.

- Within a mesh network configuration, logically isolate and forward traffic between two test relays through a specific test switch to ensure that traffic filtering can be tested without impacting other switches or traffic flows.
- Switches and relays not being tested will not see any difference in logical traffic flows (but the physical path may be different if the traffic would normally flow through the test switch).
- Traffic may be limited to just one type of traffic, traffic from a single port on a relay, or all traffic between two relays.
- Testing could be associated with new firmware, switches, or relay feature sets, or forwarding rules.
- Test change management and measure disruption of communications, add and remove switches as well as add and remove flows. The team will simulate network faults and measure the scope of the impact (i.e., does it impact just the flows on that wire or does it have a broader impact then this).

Given the network in Figure 11-3, isolate SDN switches 2 and 5 from the network to force all traffic to route around them, and insert test traffic between test points 2 and 5 to test the functionality of network communications between SDN switches 2 and 5.

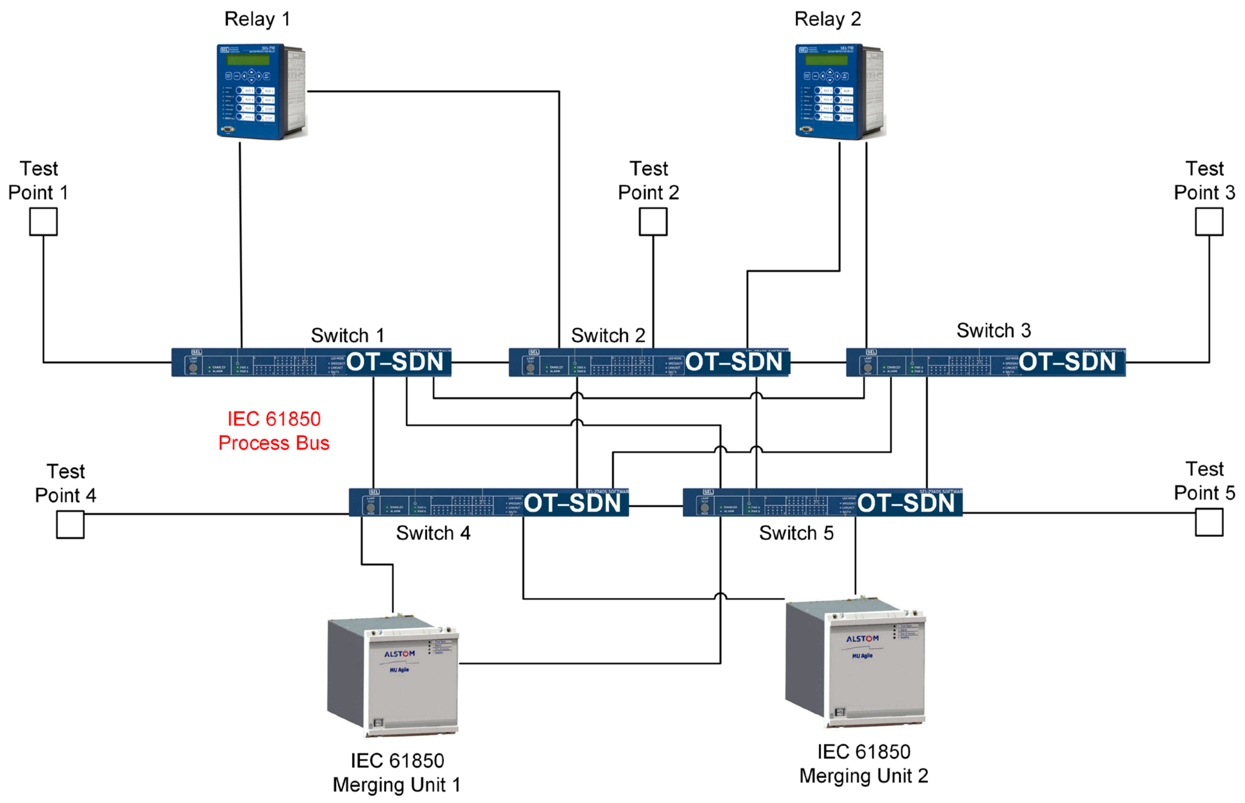


Figure 11-3. Test or Isolation Use Case

11.3 Operational Isolation within a Substation

The need to isolate one or more devices from being used in active protection of the power grid may arise for many reasons. Consider a device compromised through a cyber-attack, a hardware failure flooding a network with Ethernet traffic, the installation of a new smart grid application, or the need to update and test the protection scheme used by a protection relay.

This use case examines how SDN can enable these operational requirements.

- Using traffic flow patterns and data analytics available to the SDN controller, malicious activity could be detected on one node within the SDN fabric.
- The SDN flow controller could determine if the removal of the affected node would impact any critical data flows and alert an operator prior to taking any autonomous action. This could be the result of multiple failures (whether naturally occurring equipment failure, or a response to previously detected malicious traffic).
- The SDN flow controller could automatically and proactively disable the affected switch equipment, allowing the pre-existing flow rules to continue to allow traffic to flow.

Given the network in Figure 11-3, if malicious activity was detected on switch 2, isolate switch 2 from the network to force all traffic to route around it.

11.4 Firmware Update

A special use case for device isolation is the need to update firmware and ensure that the backup and primary protection relays function as desired after a firmware update is applied. This use case will examine how OT-SDN enables the isolation, update, and testing of protection devices, including the ability to roll back firmware if testing fails. This also will highlight the temporal nature of SDN rules and leverage the testing and failover capabilities listed in the Testing or Isolation within a Substation use case discussed in Section 11.2.

This use case examines how SDN can assist with software upgrade and maintenance without impacting other equipment.

- The firmware download channel can be secured by creating SDN flow rules that only connect the authorized source of the firmware to the destination device over the specified port. These SDN flow rules should probably not be installed but disabled on the SDN switch, but rather created when needed and removed afterward to avoid inadvertent enabling of the SDN flow rules. The SDN flow rules should be constructed to minimize the impact of the download to jitter and other performance characteristics of the operational network.
- Access to the engineering interface should be controlled through SDN flow rules and enabled *only* when it is needed. This will reduce the attack surface and help mitigate tactics, techniques, and procedures of malware. Access to the engineering interface should also be authenticated and through a secure channel.
- The relay to receive new firmware is isolated from the remainder of the system by configuring an EA path to the device and forwarding all other traffic to other paths.
- EA is enabled to the device to initiate the upgrade process.
- Firmware is downloaded and installed onto the device. This will generally require the device to be reset to complete the installation.
- The functionality of the new firmware on the device is tested using isolation procedures similar to use case 11.2 or 11.3.

Note that updating the firmware on an SDN switch can be a special case of this use case. To perform a transparent firmware update on an SDN switch, all flows through the SDN switch must have pre-engineered “fast failover” alternate path SDN flow rules that do not involve the SDN switch. This includes not only end devices, such as relays, but any flow paths between other SDN switches that transit through the switch being updated. The use of alternate paths with alternate SDN switch hardware platforms is good basic redundancy engineering that allows for automatic recovery from failed SDN switch hardware but could be problematic for end-devices with single Ethernet connections, or SDN switches that serve as gateway connections to single external networks (such as a single SCADA connection point). Note that fast failover capability will enable the network to continue operating while the SDN switch is being updated, but when the updated switch is restarted, some communication paths may revert back to their “primary” configurations even if the updated SDN switch is not completely configured and ready to process the traffic. For this reason, prior to performing the SDN switch firmware upgrade, use the SDN flow controller to reconfigure all the physical links on the SDN switch to disabled. This should result in each logical connection failing over to the alternate link one at a time in a controlled manner. During the disabled operation, the technician can monitor for unexpected behavior, such as a failover link not properly functioning, and take steps during this step to endure continued operation after the SDN switch has been isolated from operational traffic.

The technician can then download the new firmware to the SDN switch, install it, and reset it to enable the new firmware. The SDN switch should reboot with the new firmware image but maintain all physical ports in the disabled state. The technician can then re-enable ports associated with each logical connection one at a time and monitor the resulting traffic for any anomalies or irregularities. If any are found, the SDN switch ports can be re-disabled allowing the system to continue to operate, although in a less resilient mode, until the problems can be resolved (likely either by re-installing the previous firmware, working with the vendor to resolve new configurations, or in more extreme cases, applying a newer firmware update).

The new firmware also may include additional features that should be tested and may require additional configurations prior to returning the SDN switch back to full operational service. This can be done while the SDN switch is still in isolation (or partial isolation) mode to minimize the impact on the operational status of the network.

Note that while in the proactive SDN flow controller operation mode of the OT-SDN environment, the SDN flow controller is not required to be active because it would be in reactive mode. Therefore, the SDN flow controller software can be updated independently of the status of the OT-SDN network components. If the SDN flow controller is required to operate in reactive mode, it can be replicated or cloned, the cloned SDN flow controller upgraded and tested, and then the network re-directed to the upgraded SDN flow controller instance.

11.5 Engineering Access

A relay or automation technician for a substation will need limited access only for completing their EA duties in a substation. EA can include, but is not limited to, the following types of tasks:

- Managing firmware updates
- Event report or data collection
- Settings changes to existing hosts in the network (which may include network traffic engineering changes)
- Change out, removal, or addition of new end devices to the network (which will include network traffic engineering changes).

As technicians complete their tasks, there can be a process that system owners could use to define the network behavior. This process may include, but are not limited, to the following:

- EA conversations (logical flows and SDN flow rules) can be established, engineered, and vetted but disabled until a specified time or when the technician acknowledges presence in the substation and readiness to perform the tasks. This includes SDN flow rules associated with local access as well as remote access. This process prevents a rogue technician from being able to physically access the substation equipment, connect their laptop to the network, and access any devices.
- EA conversations for a given task are allowed by a different network technician or engineer after acknowledgement or authorization of the EA technician by using the SDN flow controller to enable only the required SDN flow rules in the SDN switches. The EA technician performing SDN flow rule modifications could be located at a central network operations center. All other EA conversations remain disabled until the task is completed and tested. The SDN flow rules for that task then are disabled by the network technician or engineer after acknowledgement or authorization from the EA technician that the work is complete and has been tested. EA conversations for the next task then are enabled and disabled using the same process. This process can take place without interrupting the technicians' work. This process ensures the following precautions are enforced:

- To complete the assigned and approved tasks, two different technicians are required to agree.
- Only approved tasks are being completed by the EA technician as verified by the second technician.
- The tasks can be completed in a pre-approved sequence, if necessary, with the sequence enforced by the EA technician performing the SDN flow rule updates.
- The EA technician is only able to complete the tasks on the appropriate host devices in the network based on the SDN flow rules that are active at the time.
- Additional conversations can be enabled or disabled in a controlled manner as removal or addition of host devices in the network.

- Address information (e.g., IP address, MAC address, SDN switch physical port, etc.) for the laptop or test device used by the EA technician is provided just prior to enabling conversations to ensure that only that device can perform the maintenance.
- Additional SDN flow rules for capturing all the traffic associated with the EA access and associated activity also can be generated or enabled (see Section 11.17) to provide a log of all EA activity performed.

11.6 Communications between a Market Participant and a Market Operator

A method to secure communication between market participants and operators uses a communications device sometimes referred to as a RIG. Historically, this approach has proven to be both costly and technically challenging. This operational use case will examine two approaches enabled by SDN to secure WAN communications. The first approach is a moving target defense supplied as a managed service. The second approach is using SD-WAN technology that enables the end users to establish secure virtual networks over the internet.

This use case examines how SDN can facilitate market participant communications.

- This scenario will use DNP3 for communication of status and control. Because only a subset of the features of DNP3 are needed for this communication, the specific communication or services required for this scenario need to be identified.
- Traffic is restricted to or from a market participant to specific protocols (e.g., DNP3).
- Traffic is restricted to or from a market participant to specific types of values (e.g., only analog values from specific DNP3 data tags).
- Alerts are generated if other traffic types are seen.
- Alerts can be generated if expected performance changes. Note, “normal” needs to be defined so that deviations can be measured.
- Traffic alerts and other performance or connectivity information can be fed to an analytical engine such as the ELK (Elasticsearch, Logstash, and Kibana) stack to assess traffic logs and anomalies and display them on a Kibana dashboard.
- For market communications, the SD-WAN orchestration should limit the traffic to U.S. (or North American) deflectors and paths.

11.7 Protection System Coordination between Transmission Substations

As Smart Grid technologies and new applications are deployed, traditional north-south SCADA communication is being augmented by east-west communication occurring between transmission substations. Examples of this communication include IEC 61850, which has one primary east-west data communication method: SV; and two options or modes for east-west control communication between substations: GOOSE and MMS.

- SV messages enable sharing of real-time measurements within a single substation or between substations to provide a digital emulation of continuous analog sensing.
- GOOSE messages enable fast exchange of binary or analog information between protection relays either in the same substation or between different remote substations as needed for the protection schemes deployed. GOOSE messages also are used to convey control actions or control block actions. GOOSE messages typically are generated autonomously by IEC 61850 IEDs.
- MMS SCADA messages enable exchange of binary or analog messages between the protection relays and an RTU or HMI within a substation, between RTU or HMIs in remote substations, and between RTU or HMI in one substation and a protection relay in another for telemetry and control. MMS messages are typically, but not always, generated in response to human actions.

Typical transmissions of IEC 61850 GOOSE or SV messages within a single substation are described here. GOOSE messages can be transmitted to another substation using extended OSI layer 2 LAN technology or IEC 61850-8-1 R-GOOSE (GOOSE messages encapsulated in a UDP packet and routed over IP, most commonly referred to as UDP/IP-network). SV messages can be transmitted using IEC 61850-9-2 R-SV (SV messages encapsulated and routed over a UDP/IP network). Both GOOSE and SV messages can be secured using IEC 62351-3 transport layer security or IEC 62351-4 T- or A- profile security. Theoretically, MMS messages also can be secured using IEC 62351-4 T- or A- profile security. While a standard exists for securing layer 2 GOOSE messages, it introduces significant performance delays, so usually, it is not implemented.

This operational use case will examine how SDN can secure traditional 61850 intra- and inter-substation communications without having to implement explicit security measures such as the ones outlined in IEC 62351, which are briefly discussed below:

- Testing to ensure typical protection schemes that rely on high-speed, inter-substation communications (often set up as dedicated point-to-point links) such as directional comparison blocking or unblocking, and permissive overreaching transfer trip schemes operate as expected when using IEC 61850 messaging and SDN networking. The team should also set up SV in a point to multi-point system and conduct the same testing. The team should set up the system to measure performance indices such as latencies and jitter under normal and stressed network conditions, then implement specific attack scenarios to measure the same performance indices and validate the performance against the baseline and also against the protection application-specific performance constraints.
- Forwarding IEC 61850 GOOSE or SV traffic associated with a specific transmission line across a LAN extension to a different substation's LAN based on packet content (which may include VLAN tag or MAC); that is, hardware address destination, but it also could be based on IEC 61850 data tag). This could be one subnet or layer 2 tunneling on MPLS networks.

11.8 Hybrid SDN – Traditional Infrastructure

Most SDN deployments will be made alongside traditional Ethernet networks. The deployments must be performed without significant disruption of the operational nature of the existing networks and be cognizant of existing capital investments in networking infrastructure. SDN deployments must therefore integrated into traditional networking infrastructure. Decisions on where best to implement SDN should be made based on the improvements that an SDN infrastructure can bring to the network.

This use case will examine how SDN technology can be integrated into an existing traditional networking infrastructure.

- This use case raises a number of questions that need to be answered before transitioning to an SDN environment.
  - How do you transition SDN into an existing architecture?
  - Where do you start?
  - Where is the first SDN switch installed and why?
  - What lessons have been learned from actual deployments?
  - Where should the first SDN switch not be installed?
  - That is, how do you introduce an SDN switch into a production environment that is using a traditional switch for operations?

  SEL has an application guide for integrating their SDN switch into a traditional RSTP environment.[62]

- Should a clear boundary be drawn between SDN for OT (control) networks and IT (traditional) networks for business applications?
  - Are there requirements (such as NERC CIP) that require the system to be separate?
  - Are there maintenance or performance reasons to keep the networks separate?
  - Are there business or maintenance needs to provide connections between the networks?

  If IT and OT networks are converged, they can be configured safely and reliably by managing each individual conversation to make sure that the flows for IT do not impact the flows for OT, and vice versa.

- The placement and architecture of the SDN flow controller(s) and the data plane need to be determined. Will the SDN flow controller be implemented in a distributed arrangement with a central SDN flow controller physically separate from a set of distributed SDN flow controllers that are collocated with the SDN switches (e.g., in a substation) or are the SDN flow controller(s) in a separate centralized physical location not located with the SDN switches?
- Should the SDN flow controller operate in a proactive or reactive mode?
  - If the SDN flow controller is configured to install flows reactively, then the SDN flow controller should be distributed and in separate locations than the SDN switches.
    - The SDN flow controller becomes a single point of failure and a focus of attacks.

[62] See https://selinc.com/api/download/121304/ (site requires a free account for access) (Accessed October 14, 2020).

- ○ The number of SDN flow controllers should consider potential DoS attacks when installing flows reactively.
- ○ Best practices on regularly performing security assessments of the SDN flow controller and SDN switch firmware itself also should be taken into consideration before deployment.

– If the SDN flow controller is configured to install only SDN flow rules proactively, then it should be protected against attacks that could modify the internal database of SDN flow rules that would then be installed when the SDN flow controller next updates configurations on the SDN switches.

11.9 Traffic Engineering Process

When new technology is deployed, owners and operators need to understand all aspects of the commissioning and management of the technology. The purpose of this operational use case is to identify and document an efficient and effective process to define how to configure SDN for both greenfield and existing infrastructures. Traffic engineering enables the network owner to have greater control over how the network operates and to maximize the capabilities of the network assets. No longer is there a need for dynamic negotiation protocols designating or blocking forwarding paths, but all physical ports can be used for forwarding packets. This helps balance bandwidth and segregating services, which maximizes the network asset potential.

This use case will examine the engineering process for deploying and maintaining an SDN environment.

- Traffic engineering focuses on expected behavior of applications and their communication flows. It takes into consideration the source and destination, roles, protocols, prioritization, etc. Each SDN flow controller vendor or supplier should provide information on how to define flows within. An example of traffic engineering methods is outlined in Section 6 of the SEL-2740S manual. [63]
- To identify hosts, document all the places each host needs to communicate, identify the protocols used for each flow, identify the match criteria for each protocol, provision conversation, and test the conversation.
- What the network must do to monitor and control the physical process should be determined. Nothing more and nothing less should be permitted or expected. Examples include multi-layer packet inspection at each hop, physical path planning, contingency planning and design, testing, documenting methods the SDN technology discovers and tracking the physical topology and changes that occur under steady-state and attack conditions.
- SEL has developed object extensions to the Microsoft Visio drawing tool that can be used to design SDN networks. The output of the Visio tool is a set of spreadsheets that can be used by automatic SDN flow rule generation tools integrated with the SDN flow controller.

The reality of configuration management is that over time because of changes made to the network, it is possible that there will be stale configurations that are no longer necessary or additions that were not correctly documented. Using the Visio drawing as the “single source of truth” about the network configuration will assist in determining if an SDN rule is no longer necessary, while providing documentation for the network. Using near-real-time SA monitoring,

[63] See https://selinc.com/api/download/117185/ (registration required) (Accessed October 14, 2020)

flow match counts can be used to determine how frequently an SDN flow rule is matched and when it was last matched. If the SDN flow rule has not matched for a configurable time period, then an alert can be raised to an operator to determine if the SDN flow rule is still valid.

11.10 Microgrid Applications

One of the first SDN deployments for the U.S. Department of Defense has been microgrid infrastructures. This operational use case will describe how SDN can be used to support microgrid communications.

This use case will show how SDN can be used to provide traffic isolation and control within a microgrid implementation.

Figure 11-4 shows the electrical configuration of a notional microgrid consisting of a campus microgrid and building microgrids.

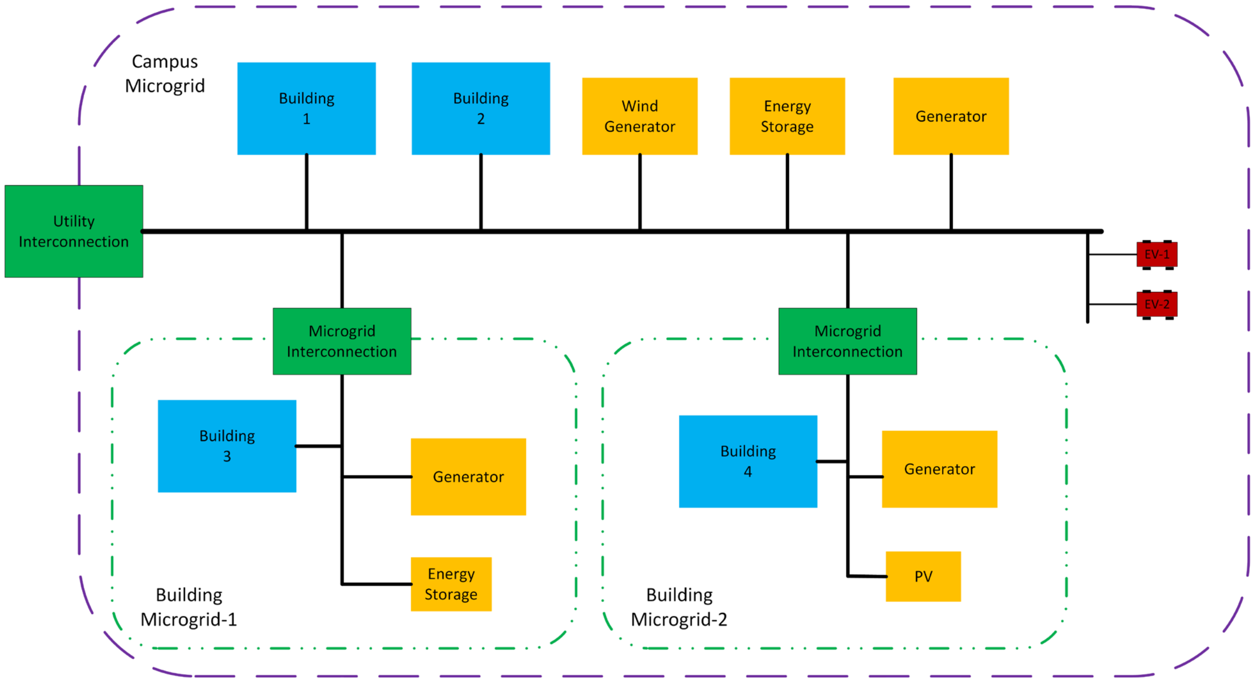


Figure 11-4. Notional Microgrid Architecture

- SDN flow rules can be created to manage traffic flows without the use of filtering firewalls that may introduce network latency or require network readdressing
- SDN is being explored to provide the networking infrastructure for Microgrid and heating, ventilation, and air conditioning systems. This raises several questions:
  - What requirements of a facility or campus microgrid can SDN meet?
  - Does the ability to install temporal flows to not burden an operator with additional administration provide value?
  - What requirements of a facility heating ventilation and air conditioning system can SDN meet?

11.11 Stopping Unauthorized Physical Changes

The need to bridge the gap between cyber and physical systems is a well understood security need for the electric sector. This operational use case will examine how SDN provides the ability to identify changes in hardware locations, the introduction of new hardware devices, and the removal of expected hardware systems.

This use case will show how SDN can be used to prevent unauthorized physical changes in a network.

- SDN switches differ from traditional network switches in that they do not pass traffic when initially turned on and require each port to be configured before passing traffic. This requires advance planning during the installation and commissioning of the network, but provides significant control over what devices, protocols, and conversations are allowed in the network.
- Each SDN switch physical port can be configured to accept traffic only from a specific MAC address, thus minimizing the ability of a rogue device to be inserted into the network.
- The SDN switch also can be programmed to generate an alarm if a physical link is disconnected and re-connected, even if the same MAC address is present. Note: this link-state monitoring will also generate an alarm for a power cycle or re-cable to an existing legitimate device.

11.12 Stopping Logical Changes

Establishing trust in network communication has been an unreachable goal. Many have tried with mathematical models, but these approaches are insufficient. How does one ensure that the network performs only the necessary functions and nothing else is the end goal? SDN provides a novel approach in which the network is programmed for desired behavior. Logical changes and attempted changes to the approved configuration can be identified in real time. This use case will examine how trust can be established and maintained with SDN technology and show how SDN can be used to prevent unauthorized logical changes in a network.

This use case will show how SDN can be used to prevent unauthorized logical changes in a network.

- SDN switches can be configured to monitor traffic as it flows through and is processed by the switch. This monitoring can include inspection of data within the application portion of the packet by forwarding the frames to an external IDS or IPS.
- The flow control rules of the SDN switch can be configured with specific valid logical data paths and data elements, which may be more finely grained than what is available in traditional switches.
- If an end device is compromised or misconfigured to communicate with a different logical end device or if it tries to access an unauthorized data point, the resulting traffic will not match the engineered traffic flows, and the frames will be dropped. SDN flow rules can be created to count the number of dropped frames; monitoring these counters can trigger alerts when excessive unauthorized traffic is detected.

11.13 Computer and Network Black Start or Brown Start

Planning and testing SDN deployment during contingency scenarios ensures that the network can be properly brought back online during recovery. Examples of scenarios where outages may occur include black start or brown start events. Black start events occur after a total outage is experienced and all systems on the network need to be restored. Brown start events are partial outages where a subset of the systems on the network need to be restored. In these scenarios, the order in which systems are restored and reconnected to the grid is important. The SDN switches may need to re-prioritize or install new flows outside of normal operations in the network to ensure that the network is gradually restored in the proper order.

This use case will demonstrate that the SDN deployment can be brought online during normal operating conditions as well as during contingency scenarios. Black start and brown start scenarios provide adversaries with unique conditions to exploit systems while cyber monitoring systems may not be fully recovered to detect the attacks.

A typical network black or brown start scenario is:

- The entire network will be powered off.
- The SDN flow controller is powered on and its configuration is verified.
- SDN switches are powered on individually and their configurations are verified or reloaded.
  - Individual SDN flow rules may be temporarily disabled to validate traffic flows when connecting individual end devices.
- Each of the end device systems will be brought back online and connected to the network.
  - SDN flow rules that were temporarily disabled previously may be re-enabled while verifying and validating traffic flows.
- As end devices are brought online, the proactive SDN flows will be available and installed so that communications can resume. If an adversary attempts to join the network during a network black start or brown start, the SDN flows will not match the adversaries MAC address and frames will be logged and dropped.

11.14 Network Monitoring

To analyze data and monitor network traffic, analysts rely on network taps that can mirror traffic to an analysis system. The network traffic can be used to capture metrics, perform security analyses, troubleshoot network problems, and for forensic purposes. Traditional network switches would accomplish this task by activating SPAN port that could duplicate frames traversing the switch and forward the copied frames to a separate system (e.g., an IDS) for analysis. Performing an equivalent task in an SDN switch requires network administrators to create SDN flow rules that will similarly replicate frames and forward those frames to a designated monitoring port. The SDN flow rules that replicate traffic will need to be performed on a per flow basis.

This use case will demonstrate that the SDN deployment can perform equivalent network functions that network administrators depend on for troubleshooting, metrics gathering, and security analysis.

- A monitoring port will be configured on the SDN switch to capture all traffic traversing it.

- The mirrored traffic will be forwarded to the SDN switches designated monitoring port.
  - Use of SDN group flow rules allows multiple actions to be performed on an individual frame, such as sending the frame to the monitoring port as well as normal forwarding.
- The mirrored traffic can be collected and analyzed on an analysis system such as an IDS.
- Metrics will be captured on the performance impacts (if any) of the network with and without the monitoring port enabled.

11.15 Enabling Situational Awareness

SDN provides a feature-rich environment for fine-grained monitoring of network conversations. Typically, using SNMP statistics, a network operator can see only aggregate byte and frame counts on a physical port, but those physical port counters may represent many different conversations.

Some network monitoring applications allow node (or possibly port) conversations; that is, the number of bytes or frames between node 1 and node 2, or between node 1 on port X with node 2 on port Y. SDN flow rule counters maintain byte and frame counts for each SDN flow rule, which may include more granularity. Examples could include matches on MAC address, EtherType, VLAN tags or priority, IP protocol, IP source or destination addresses, TCP or UDP source or destination ports, ARP Opcodes, or ARP sender or target protocol addresses.

These counters can be collected and processed by specialized tools, such as the SDN Situational Awareness Tool (SDN-SAT) from Spectrum Solutions, Inc., integrated into existing network management tools like Splunk, or imported into ELK instances and displayed using customized dashboards.

11.16 Multi-Vendor Integration – Common Network Model

Configuring SDN networks can be a complicated and time-consuming activity that is rife with possible errors and misconfigurations. Unlike a traditional Ethernet environment in which frames are forwarded based on hardware addressing and simply connecting a cable between a node and a switch is often sufficient to start conversations, SDN employs a “deny-by-default’ approach and only forwards frames that match specific criteria. This means that unless an SDN switch port is already configured to forward frames and the frames received on the physical port match the forwarding rule criterion, no traffic flows. The process is even more complicated for frames flowing between two different SDN switches, because not only do the ingress and egress ports (the ports directly connected to the end devices) need to be configured, the links between the various SDN switches also need to be configured to accept and forward the frames.

To add additional complexity, different SDN switches models or SDN switches from different manufacturers may have slightly different configuration and management requirements such as maintaining configurations through power cycles or deleting individual rules that have not been triggered in a set time frame (although standards like OpenFlow are supposed to provide vendor-neutral interfaces for configuration).

Some SDN flow controllers provide a centralized and graphical interface for designing and implementing SDN flow rules, but vendor-specific tools generally favor their own equipment, and generic tools do not recognize vendor-specific extensions.

Using the Visio tool mentioned in Section 11.9, objects for other SDN switches and configurations could be developed to extend the tool from being specific to SEL-2740S configurations to a more generic tool that can design and visualize network flows between end devices through an SDN environment.

The Visio tool not only models SDN switches but also models individual end-devices that connect to the SDN switches, including the protocols that are used, and the conversation endpoints. This level of detail is required to create fine-grained SDN flow rules that strictly limit what protocols and conversations are supported by each individual SDN switch port.

Data extracted from the Visio drawing objects can be processed by SEL-specific tools and fed into the SEL-5056 SDN flow controller for dissemination to the SDN switches under the SEL-5056's control. This could be extended to other vendors SDN switches in two ways:

- Other vendors' SDN switches that conform to the same OpenFlow protocol specification could be adopted into an SDN-5056 control environment, allowing the SEL-5056 to use the configurations as if they were designed for SEL-2740S SDN switches.
- The Visio output could be generalized either by retaining the current CSV (comma-separated value) format or by converting to an XML (Extensible Markup Language) or a JSON (JavaScript Object Notation) format that then is used by other SDN flow controller software to generate and install SDN flow rules into the SDN switches.

Either of these approaches would provide a design-based visualization of the SDN network, connections, and flows, providing a single point of truth that can be assessed and audited before being installed into the SDN switches.

An alternative could be to develop YANG (Yet Another Next Generation) models for the data objects and convert the Visio output into YANG format. The advantage of using the YANG format is it is a standardized method for describing data interactions that may be compatible with other SDN configuration software or SDN flow controllers.

11.17 Enhanced Port Mirroring

SEL has been issued a patent on selective port mirroring[64] that describes how SDN can be used to facilitate port mirroring and make the mirroring and eventual monitoring of specific network traffic more efficient. This patent describes how SDN flow rules can be used to tag, or color, specific Ethernet frames with, for example, special VLAN tags, and transport them in-band through an SDN environment and send them to a common port that can be used for monitoring. This approach has two advantages over traditional switch mirroring using SPAN technology:

- SDN flow rules can be used to only forward specific frames to the mirror port for inspection. For example, assume a network that uses a combination of DNP3/IP and Modbus/TCP for control purposes, and a combination of secure shells and HTTP or HTTPS for EA and maintenance. To diagnose a problem with the DNP3 communications, network monitoring is required to see the DNP3 packets and their interactions on various nodes of the network. A traditional SPAN port would see a combination of all of those protocols (along with any support protocols such as ARP and ICMP) requiring filtering by the monitoring process (e.g.,

[64] Patent No. US 10,785,189, "Selective Port Mirroring and in-Band Transport of Network Communications for Inspection," issued September 22, 2020.

Wireshark) to see only the DNP3 packets. In addition, on a large switch (e.g., a 48-port switch) and a heavily loaded network, it would be impossible to guarantee that all the traffic from 47 of the 48 ports to be successfully mirrored to the 48th port for monitoring; some of the traffic would have to be bypassed, and there is no guarantee that all the DNP3 traffic would be exempt from being bypassed for mirroring.

- Mirror ports are restricted to a single physical switch, so each switch must independently set up mirror ports for traffic contained on that switch. This requires that each switch forward its mirrored traffic toward a monitoring node. If a very small number of switches are involved (say no more than two or three), it is possible that a single node running the monitoring software may have enough physical ports to receive all the mirrored traffic. However, if there are more switches involved, or the individual switches are farther apart than a set of network cables can reach, a separate network must be established to transmit all the mirrored traffic to another Ethernet switch with its own mirrored port where the node with the monitoring software can be connected. This presents several problems. First, mirrored ports already have the possibility of being overloaded and drop packets, and by their nature use nearly all the bandwidth available for the physical port. Aggregating several already nearly overloaded ports into yet another switch will guarantee that some traffic of interest will be dropped. Second, the addition of a separate network for monitoring requires acquiring, installing, and monitoring yet another set of switches to support the monitoring efforts. These switches will take up rack space, require power, and use interconnection cabling infrastructure, all of which may be scarce resources already, especially in field locations.

The approach described in SEL's patent resolves these issues in the following ways:

- SDN flow rules can be established to only send traffic of interest to the monitoring node. In the example above, only DNP3 traffic (and perhaps some of the supporting protocols like ARP and ICMP) can be programmed to be the only frames forwarded to the mirror port. This will significantly reduce the amount of traffic the mirror port needs to process, greatly reducing the possibility of dropping any traffic of interest due to overloading the port. This also has the benefit of reducing the processing and storage load on the node running the monitoring software. Because most protocol analysis starts by filtering out traffic that is not of interest, this should have no detrimental impacts on the analytical process.
- SDN flow rules can augment the packets of interest with what SEL refers to as a "color" by pre-pending special debugging VLAN tags to the packets, allowing them to flow through the remainder of the SDN infrastructure and be flagged as debugging packets not sent to operational end devices. Because the packets remain inside the existing infrastructure, they can be forwarded between SDN switches without requiring any additional switch (SDN or traditional) or cabling infrastructure, using otherwise unused bandwidth on existing interconnection links. Because only traffic of interest is forwarded, the increased traffic on the switch interconnects is minimal (as opposed to significant if the same approach were required on a traditional network). Flow rules at the receiving SDN switch can re-write the packets to remove the coloring, or the monitoring software may be able to ignore the coloring.

In addition to these direct advantages, additional information can be encoded in the coloring that can assist in the diagnosis of network problems, such as indicating the ingress SDN switch or port. If this is the case, the egress processing should not remove the color indications; rather, it should let the monitoring or diagnostic software interpret them.

11.18 Point-to-Point Legacy Migration

Many legacy EDS devices still use point-to-point serial connections to exchange data. A distinct advantage of this approach is the source, destination, and path taken by the data are always known and fixed. However, it also leads to a detrimental side effect that if any component in the path fails or if the source or destination need to change (e.g., to recover from a failed end-node), separate pre-existing serial connections need to be engineered and built, or the communication between the source and destination will fail. Data transfer speeds of serial connections also tend to be lower than those of many modern Ethernet-based communication approaches (e.g., 1.2 kbps to 9.6 kbps versus 10 Mbps to 1 Gbps, respectively), thus limiting the amount of data that can be transferred. By designing and implementing specific flows through an SDN, the point-to-point behavior of the serial connection can be emulated, while adding the ability to pre-engineer recovery paths for link failure and node failover. Older legacy devices (i.e., devices that only support serial connections) can use conversion hardware to translate the serial frames to Ethernet frames and can encapsulate the frames inside TCP/IP packets for additional routing flexibility outside of an Ethernet LAN. Newer legacy devices may already have unused Ethernet ports that can be used instead of the serial (RS-232) ports, thus eliminating the need for the conversion hardware.

When conversion to an Ethernet-based communication network using hardware encapsulation is complete, it can also be used to integrate modern equipment (that expects Ethernet-only or TCP/IP-addressed frames) allowing a straightforward migration to replace old equipment with modern equipment with minimal re-engineering and hardware conversions. This will allow both migration of older equipment that may have support issues to newer supportable equipment and increased data capability (at smaller response times), providing potentially more functionality to be implemented in the EDS at low incremental cost.

An advantage that SDN has over traditional Ethernet networks in this case is the amount of control that an implementation has over where the data flows. Rather than the “broadcast” approach that exists in traditional networks, in which an attacker may be able to access the network and transmit or receive packets on what was previously a logical point-to-point network, using SDN flow rules allows the flexibility of an Ethernet infrastructure with the control of a legacy point-to-point network. SDN flow rules can be created that exactly mimic the control an implementer has over where the data flows, while simultaneously allowing the addition of additional sources or destinations (for diagnostic or expansion purposes) that were difficult with serial connections without the addition of hardware multiplexors or hardware repeaters (or implementing a multi-drop serial connection).

11.19 Legacy Ethernet to SDN

Many implementations have already moved to Ethernet-based communications to provide increased speed and flexibility over hard-wired and serial connections and to take advantage of communications capabilities provided in modern field equipment. Ethernet generally operates in a “plug and play” mode, whereby once an Ethernet environment is established, new nodes can be added simply by connecting them to a network switch, Ethernet protocols like ARP are used to map destination IP addresses to hardware (i.e., MAC) addresses, and the switches learn what devices are connected to each port by monitoring traffic. For example, when a new node is connected to an Ethernet switch and wants to establish communication to another node, it issues an ARP request asking for hardware-to-IP address mapping. The ARP request supplies the switch with the node’s MAC address, and the responding node supplies its MAC and IP addresses. Once both nodes know each other’s MAC address to IP address mapping, they can start communicating within the same Ethernet network. No additional configuration (or security) is required to communicate at the IP or TCP level. If a switch physical port fails, the node attached to the failed port simply moves to another port, and the switch will automatically detect the change in MAC address location and will continue the communication.

This presents a number of security concerns that can be addressed by SDN but raises the issue of how to migrate from an existing traditional or legacy Ethernet environment.

The most straightforward approach would be to connect the existing traditional Ethernet switches to ports on the SDN switch and configure the SDN switch with very broad rules to allow traffic to flow between the SDN and traditional environments. Once the two networks are interconnected, an SDN flow rule can be crafted one device at a time to migrate the individual end nodes from the traditional Ethernet environment to the SDN environment. Initially, the rules will all involve the connection(s) between the traditional environment and the SDN environment, but as additional devices migrate from the traditional environment to the SDN, fewer packets will flow between the traditional and SDN environments. Note that each time a device moves, SDN rules for devices that have already been moved may need to be adjusted in addition to the rules associated with the device being moved.

In some cases, some devices may be kept on traditional switches. This may be due to a limited number of ports available in the SDN environment or the inability to install or impracticality of installing an SDN switch at a location served by an existing traditional switch. If this is the case, SEL provides guidance on how to connect SDN and traditional resilient environments (i.e., RSTP) in SEL Application Guide AG2017-28, "Setting Up a Fully Redundant RSTP-to-SDN Tie Point."[65]

Engineering tools, such as the engineering tool created by SEL using Visio can assist in the design and implementation of SDN flow rule sets by engineering a small number of changes at a time and using the tool workflow to implement the changes in concert with the physical migration of the device connections. Using the tool will help ensure that all the required SDN flow rules will be generated and installed.

11.20 Knowledge, Skills, Abilities to Operate SDN

Network monitoring of an SDN environment is different than a traditional environment but not as different as management. Most SDN switches (including the SEL-2740S) can provide raw byte and frame counts for each port through SNMP queries, but SDN switches maintain more granular counters and metrics associated with individual SDN flow rules that are available to the SDN flow controller but not necessarily to SNMP. Tools such as the SDN-SAT can be used to query the SDN flow controller to obtain the individual counters and make them available for display natively, or they could be exported for display and processing in existing SIEM tools such as Splunk.

Network diagnostics also can be different. SDN can provide more flexibility and less potential overhead than traditional network diagnosis using the procedures described in Section 11.17.

[65] See https://selinc.com/api/download/121304/ (site requires a free account for access) (Accessed October 14, 2020)

11.21 Automated Commissioning Testing

One of the uses of output from the tool SEL developed using Visio (discussed in Section 11.16) is the ability to generate data for use by test scripts and test harness configurations that can simulate both authorized traffic and unauthorized traffic.

Using small, inexpensive single-board computers (SBC), such as a Raspberry Pi, network and protocol configurations can be loaded to simulate end devices, including both expected traffic and simulated unauthorized traffic. Because the complete valid configuration is available from the engineering tools, the SBC can be configured with the end device's IP address, an emulation of valid protocol packets, and a list of target destinations for each protocol. The SBC can log the tests it attempts allowing the test results to be uploaded and correlated with other monitoring information gathered from the SDN switches.

If supported by the SBC, actual-end device MAC addresses can be configured into the SBC to better emulate the real end device and allow MAC address filtering in the SDN flow rules.

For "positive testing," the SBC can generate emulated traffic and send it to the SDN switch, allowing the SDN switch to process the traffic and update traffic counters. The traffic counters from the SDN switches and logs kept in the SBC can be compared to verify that all traffic sent from the SBC was accepted by the SDN switch.

For "negative testing," there are two aspects that need to be tested. First, the SBC can be programmed to send packets associated with otherwise authorized protocols to addresses that do not appear in the configuration. The SDN switch should detect and drop these packets as not matching destination addresses. Second, the SBC can generate packets associated with unauthorized protocols and send them to both the authorized and unauthorized destination addresses. The SDN switch should similarly detect and drop these packets. Traffic counters from the SDN switch and logs kept in the SBC can be compared to verify that no unauthorized traffic was accepted by the SDN switch.

To perform end-to-end testing to verify that the entire SDN network allows accepted traffic to reach the proper destination, two SBCs are required—one at each end of the logical communication link. The SBCs are programmed to transmit and receive and then respond and receive network packets to validate that all the SDN switches are configured to allow traffic between the nodes. Note that because SBCs are used, the contents of the packets do not need to conform to the protocol. Only protocol headers and other information that may be used in SDN flow rule match fields need to be supplied. Packets of varying sizes should be included to verify that fragmentation is handled appropriately. If network resiliency and path redundancy are configured as part of the SDN configuration, this end-to-end testing can allow verification of those SDN capabilities before they are needed in the operational environment.

(Note that end-to-end testing also can be performed on traditional networks to validate network connectivity and resiliency, including through firewalls and routers, and on hybrid networks that include both SDN-enabled switches and traditional switches.)

11.22 ELK Integration for Situational Awareness

To integrate OT-SDN data into a SIEM tool such as Splunk or ELK, multiple methods exist to capture operational network data and health information for hardware and software components. This OT use case borrows heavily from work performed in IT environments. The focus will be on methods to ingest data such as SDN metrics, server health, and syslog messages produced by SDN hardware into an ELK environment.

ELK uses Beats, an open-source platform for single-purpose data shippers, to ingest data from machines into either Logstash or Elasticsearch. Documentation on Beats can be found on the elastic website.[66] Figure 11-5 illustrates how this traditionally IT-based solution can be used with OT-SDN. Two common shippers include PacketBeat to ship network data and FileBeat to ship logs and other data. After data ingest, ELK dashboards and other visualization capabilities can be used to monitor the data for anomalous behavior and generate alerts.

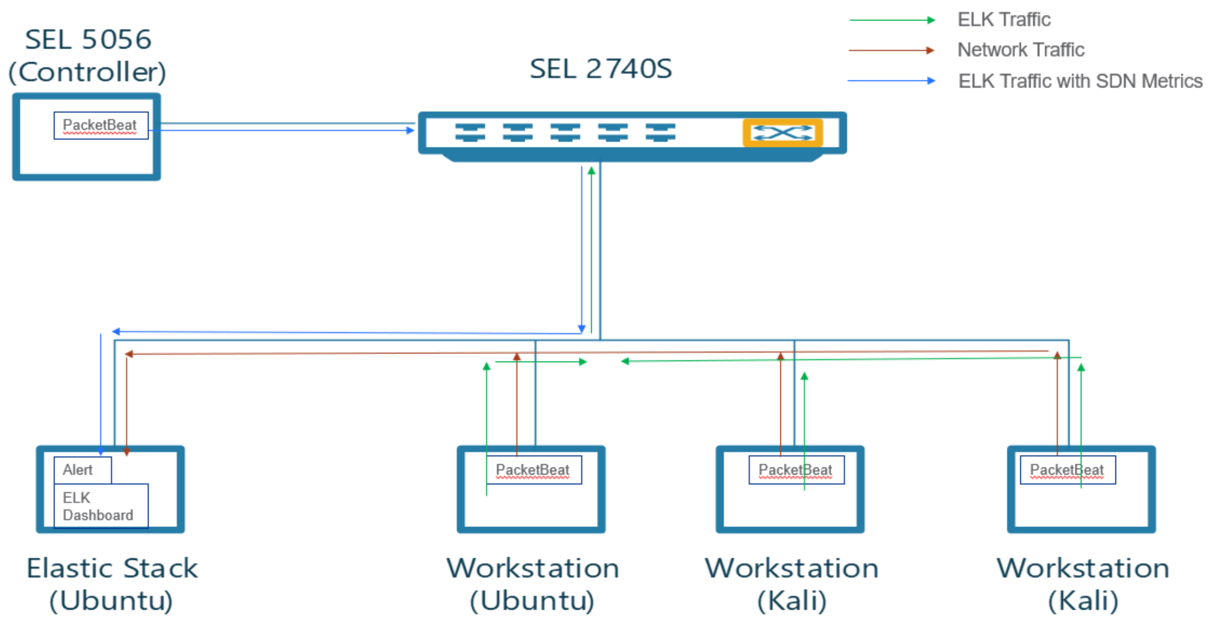


Figure 11-5. Use of ELK to Analyze SDN Traffic Metrics

Using a combination of Beats and ELK, the system owner can identify numerous events of interest. SDN metrics available through the API of the SDN flow controller can be used in conjunction with PacketBeat data to:

- Capture statistics for high-value communication flows and monitor the flow for changes in behavior (e.g., volume, periodicity).
- Identify attempts to connect an unauthorized device connecting to the network.

[66] See, https://www.elastic.co/beats/ (site requires a free account for access) (Accessed November 2, 2020)

- Monitor suspicious events such as table misses. When unauthorized traffic is identified on an SDN network, the default action associated with the deny-by-default model is to drop the traffic. However, SDN can be configured to apply a VLAN tag to this traffic and send it to a data store such as ELK. Please see the operational use case in Section 11.17 for more information on this method.

Operational information from the SDN hardware also can be obtained and incorporated into ELK. OT-SDN switches may generate syslog messages. Through FileBeats, this log information can also be ingested into ELK where hardware health data can be combined with analysis of network traffic.

11.23 Incorporate Brownfield Deployment Experience

The typical engineering process for technology deployments is depicted in Figure 11-6.

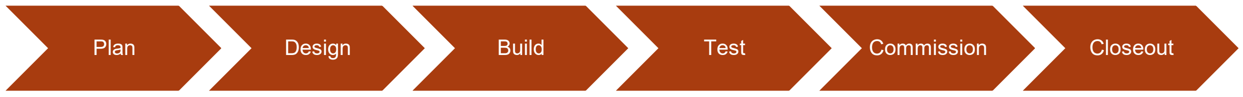


Figure 11-6. Typical Engineering Process for Technology Deployments

Each phase in the process consists of multiple activities that must be completed before moving to the next phase. This process works well for new deployments, but it must be altered when SDN technology is being deployed into existing Ethernet networks. These brownfield deployments require a new Site Assessment phase placed between the Plan and Design phases to ensure sufficient information is captured or collected from the existing environment to inform the traffic engineering process. Additional activities also must be performed in the Commission phase to address unexpected network traffic. A new Operations phase is added after Closeout to ensure that owners and operators of the system receive any additional training and all legal and contractual agreements are established. Activities to address quality assurance are added to the Commission phase to ensure the system owner is involved with approving any changes to the design identified during deployment. The new phases and activities are presented in Figure 11-7:

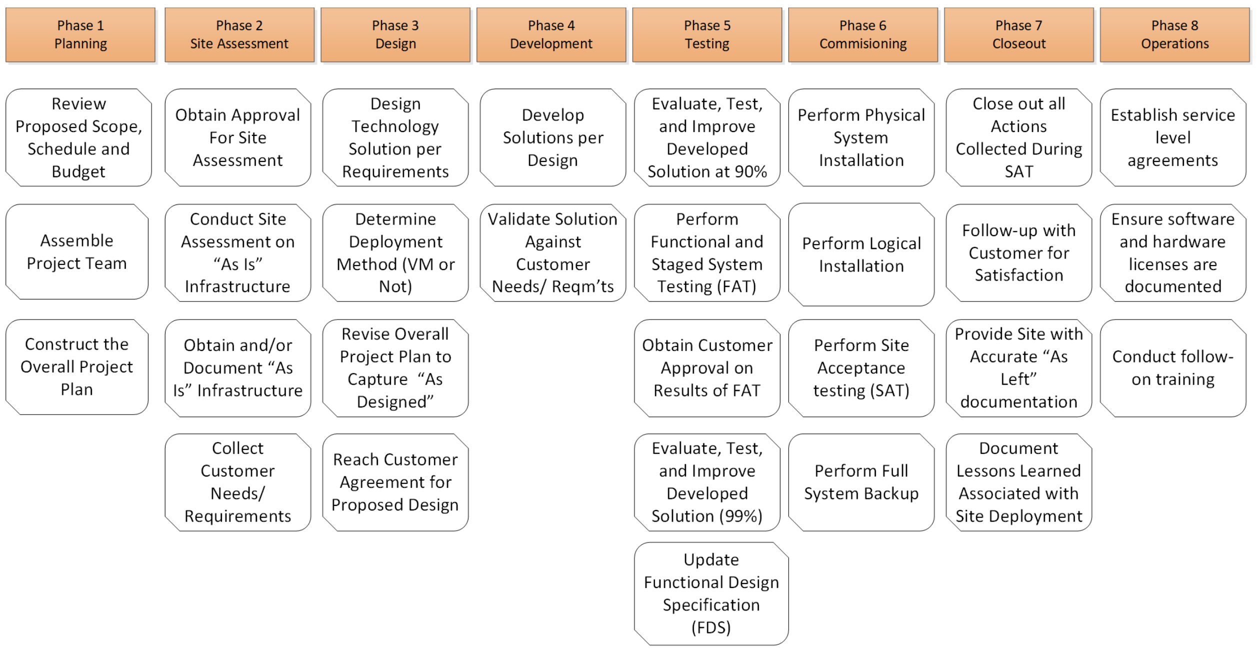

Phase 1
Planning
Phase 2
Site Assessment
Phase 3
Design
Phase 4
Development
Phase 5
Testing
Phase 6
Commisioning
Phase 7
Closeout
Phase 8
Operations
Review Proposed Scope, Schedule and Budget
Assemble Project Team
Construct the Overall Project Plan
Obtain Approval For Site Assessment
Conduct Site Assessment on "As Is" Infrastructure
Obtain and/or Document "As Is" Infrastructure
Collect Customer Needs/ Requirements
Design Technology Solution per Requirements
Determine Deployment Method (VM or Not)
Revise Overall Project Plan to Capture "As Designed"
Reach Customer Agreement for Proposed Design
Develop Solutions per Design
Validate Solution Against Customer Needs/ Reqm'ts
Evaluate, Test, and Improve Developed Solution at 90%
Perform Functional and Staged System Testing (FAT)
Obtain Customer Approval on Results of FAT
Evaluate, Test, and Improve Developed Solution (99%)
Update Functional Design Specification (FDS)
Perform Physical System Installation
Perform Logical Installation
Perform Site Acceptance testing (SAT)
Perform Full System Backup
Close out all Actions Collected During SAT
Follow-up with Customer for Satisfaction
Provide Site with Accurate "As Left" documentation
Document Lessons Learned Associated with Site Deployment
Establish service level agreements
Ensure software and hardware licenses are documented
Conduct follow-on training

Figure 11-7. Engineering Process for SDN Deployment

The checklist shown in Figure 11-8 was created by PNNL staff during multiple SDN deployments. The checklist illustrates the complexities of a brownfield deployment that should not be taken for granted. For the Planning Phase, the checklist also assumes that absolute knowledge regarding the devices and communications on the existing infrastructure are not known by the owner or operator. For the Data Collection Requirements section in the checklist, it is common for IT personnel to assume that a mirror or SPAN port exists on an Ethernet switch. For OT environments, the ability to mirror all ports or use a SPAN port may not exist, thus complicating data collection. Similarly, in IT infrastructures, network scanning tools frequently are used to enumerate a network. In OT environments, the use scanning tools may not be permitted by the system owner due to the potential harm the tools may cause. A combination of data sources provides two benefits. First, the data sources supplement the potential lack of complete traffic captures. Second, these data sources provide an opportunity to data validation and quality assurance activities.

Planning Phase – Brownfield Checklist (page 1)		
Site Name:______________________ Site Specific Lead: ______________________ Site Owner POC: ______________________ Supervisor: ________________________		
Identify Key Roles	**Initials**	**Date**
Site Assessment Roles: ☐ Identify leadership roles (system owner, approvals, or authorities to address technical difficulties and safety concerns) ☐ Data collection ☐ Data validation ☐ Identify stakeholders (e.g., utilities, vendors) ☐ Stakeholder technical lead(s) ☐ Safety oversight ☐ Contractor oversight ☐ Contract representative, (contractor) ______________________________ ☐ Contract representative, (site) ___________________________________		
Site Commissioning Roles: ☐ Stakeholder approval of changes to design ☐ Equipment purchased ☐ Equipment shipped ☐ Installation scheduled ☐ Equipment installed ☐ Site acceptance test or validation process ☐ Staffing needs		
Licenses and Certification Requirements: ☐ Licenses (e.g., professional engineer) ☐ Professional certification (e.g., CISSP, GIAC)		
Identify Requirements	**Initials**	**Date**
Scope and Schedule ☐ Identify initial project scope ☐ Identify desired completion date ☐ Identify anticipated cyber and infrastructure projects ☐ Identify future vision for critical infrastructure operations ☐ Identify requirements for ATO, red teaming, outreach		
Security Requirements ☐ Negotiate an NDA with all necessary parties ☐ Identify classification guidance and data samples ☐ Identify protection measure for data at rest ☐ Identify secure communication methods		

Planning Phase – Brownfield Checklist (page 2)		
Data Collection Requirements ☐ Network diagrams ☐ Device function description ☐ Router, firewall, or switch configuration files ☐ Relay configuration database ☐ IEC-61850 configuration files ☐ Primary to backup failover testing ☐ Packet capture ☐ Photographs (e.g., network racks, switches, media, etc.) ☐ E-MASS or Accreditation package ☐ Permitted tool use for network enumeration		
Quality Control Requirements ☐ Applicable standards for software and equipment ☐ Design review process expectations (PNNL, site, stakeholders, timing, method) ☐ Order of installation ☐ Change control processes or approvals ☐ Verify collected documentation matches install and label components ☐ Continency plans ☐ Records management		
General Safety Requirements ☐ Governing safety work control documentation ☐ Site-specific environment, health, or safety approvals or associated documentation ☐ Risk Management Plan ☐ Training (e.g., lock out tag out, low voltage electrical, high voltage electrical) ☐ Emergency preparedness ☐ Stop work authority ☐ Injury or Illness procedures		
Applicable Transportation Methods or Requirements for Remote Facilities ☐ Boat ☐ Commercial air carrier (CFR 121; e.g., Delta, United, etc.) ☐ Commercial aviation (CFR 135; e.g., local charter, float plane, etc.) ☐ All-terrain vehicle ☐ Passenger vehicle		
General Considerations ☐ Subcontractor plan ☐ Communication plan		

Planning Phase – Brownfield Checklist (page 3)		
Site Visit Planning	**Initials**	**Date**
Site Access ☐ Agreements or approvals (e.g., off-site work request) ☐ Badging ☐ Escorts ☐ Notifications ☐ Training ☐ COVID-19 requirements ☐ Other requirements		
Safety Equipment Needed ☐ Hearing protection ☐ Hard hat ☐ Eye protection ☐ Arc flash protection ☐ Footwear ☐ Ladder ☐ Other (specify)		
Approvals		
Point of Contact Signature		**Date**
Oversight Signature		**Date**

Figure 11-8. Sample Brownfield Installation Checklist

11.24 Brownfield Commissioning

The deny-by-default security posture provided by OT-SDN and the potential hazards faced when deploying technology in energized environments necessitate special considerations during the commissioning phase of an OT-SDN network. The following lessons learned were captured during the DOE-sponsored deployment of OT-SDN technology for various DOD, VA, and utility environments.

1. Performing work safely requires the substation be de-energized for OT-SDN commissioning.
    - If the substation or energy delivery system cannot be de-energized, the OT-SDN installers must follow appropriate safety standards and guidelines.
    - Staff performing the deployment of the OT-SDN network must be certified to work at the voltage level of the energized environment.

2. Staff may encounter other hazards while traveling to locations where OT-SDN equipment will be installed. Remote locations may require non-traditional methods of transportation such as boats, float planes, or all-terrain vehicles. Risks associated with each type of transportation need to be identified and either accepted or mitigated prior to travel.
3. Prior to commissioning, a factory acceptance test (FAT) will be conducted with participation from integrators, system owners, and other stakeholders. The purpose of the FAT is to test all expected communications identified during the traffic engineering process. Any changes or errors identified in the FAT will be documented in drawings and design documents. A more comprehensive FAT will introduce unwanted communications (e.g., negative testing) to verify that the deployed flow rules correctly handle unexpected traffic.
4. During commissioning deployment, the process shown in Figure 11-9 is used to monitor the deployed configuration. Each of these topics will be discussed below.

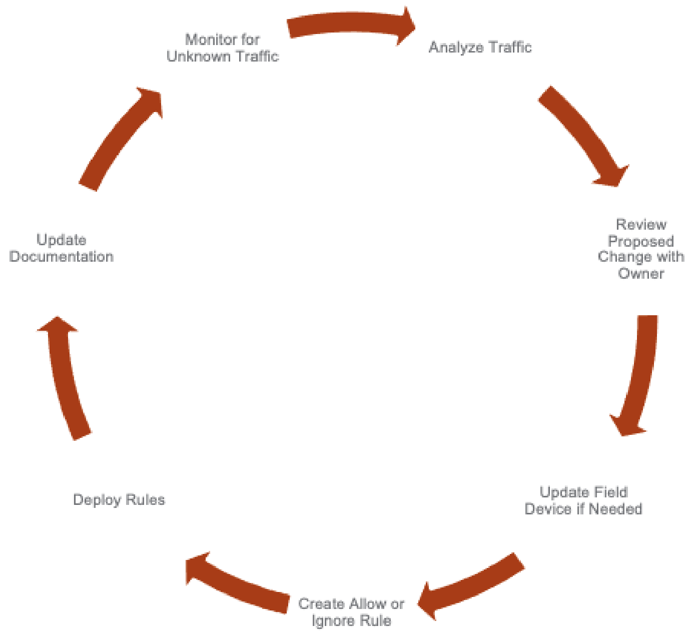

Monitor for Unknown Traffic
Analyze Traffic
Review Proposed Change with Owner
Update Field Device if Needed
Create Allow or Ignore Rule
Deploy Rules
Update Documentation

Figure 11-9. Deployment Lifecycle

5. Experience has demonstrated that the traffic engineering process will miss intermittent or infrequent communications due to incomplete data for analysis. This means that those commissioning the environment must monitor the network for unexpected communications for several reasons
   - Typical data captures are not of sufficient duration to capture all expected communications
   - Redundant / failover links are not exercised during the capture
6. OT-SDN supports the creation of a “catch all rule” that is added at the end of the flow tables. This rule will match on any traffic that does not meet the criteria of all other flow rules. A VLAN tag can be added to this traffic to enable forwarding it to a data store. During commissioning this configuration option will identify all missed traffic and provide system integrators and owners an opportunity to analyze any traffic that does not match expected communications. The catch all rule should be used to mitigate the anticipated authorized traffic not identified during the traffic engineering process. Figure 11-10 depicts the concept.

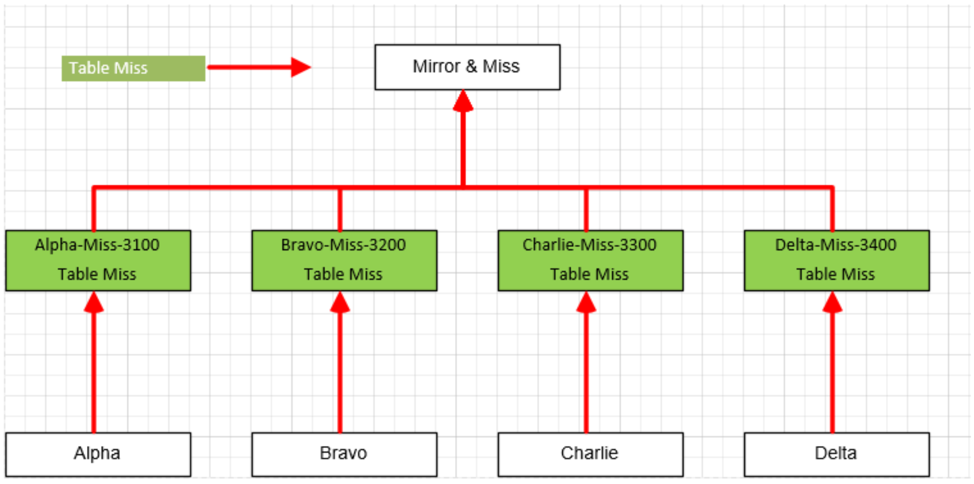


Figure 11-10: Table Miss Concept, provided by SEL

- Use Wireshark or some other repository to capture all mirror and miss traffic
- Run data capture tool on a separate system – not the flow controller
- Define the table miss logical network tap for each switch
- Define priority for table miss trafficcatch all flow rule with the lowest priority
- Place rule in the last flow table
- Deploy flow rule to all switches for all devices
- Over time the volume of traffic matching the catch all rule should decrease
- Consider using the catch all rule to provide SA after the system is operational to identify nefarious or unexpected communications

7. After analyzing traffic forwarded by the catch all rule to a data store to identify its purpose, discuss whether the traffic should be allowed on the network or ignored by the OT-SDN switch hardware. This step not only provides quality control and review by system owners of all deviations from design, but it also informs future traffic engineering efforts to ensure the type of traffic is not missed.
8. After a determination is made regarding the missed traffic from the previous step, two options exist for traffic that is not wanted on the network. An example is a device attempting to reach out to the internet to check for firmware updates. If this type of communication can be disabled through configuration settings, make the necessary change on the device or devices per manufacturer instructions.
9. The second option is to use a flow rule to capture all expected but unwanted traffic on the network. This approach enables an operator to monitor unwanted communications for changes in behavior using a dashboard in a product such as ELK or Splunk. Both options to eliminate unwanted traffic will reduce the number of false positives produced by SA tools. Lessons learned include:
   - Use a separate Wireshark instance or separate VLAN tag for expected communications that should be ignored
   - An ignore rule will reduce the volume of traffic that must be analyzed by the catch all data store

- A separate, granularly defined, ignore rule should be created for each device and conversation to ignore
- The ignore rules should be placed directly before the catch all rule

10. If the captured and analyzed traffic should be permitted on the network, create an appropriate flow rule to ensure the conversation is implemented.
11. Deploy ignore or allow rule, as required, by the system owner.
12. Repeat the process until minimal traffic matches the catch all rule
13. QA process
    - Update system documentation and drawings to reflect all changes to keep the design and deployed network configurations in sync
    - Capture lessons learned for each type of allowed or ignored traffic to enable the next site assessment process

12.0 Reference Architecture

This section was revised in Version 3 (September 15, 2018), and further revised for the Final Report.

12.1 High-level Notional Architecture

Figure 12-1 depicts a notional version of the reference architecture for communications within an organization based upon the requirements in Section 4.0. Intra-organization communications are deployed using an SD-WAN technology as described in section 4.5. SDN is deployed with deny-by-default rules to reduce the attack surface and prohibit lateral movement through the network. All communications are secured using TLS to defeat MITM attacks. The locations of the EDS protocol behavior and analytics technologies to be researched and demonstrated during SDN4EDS also are included.

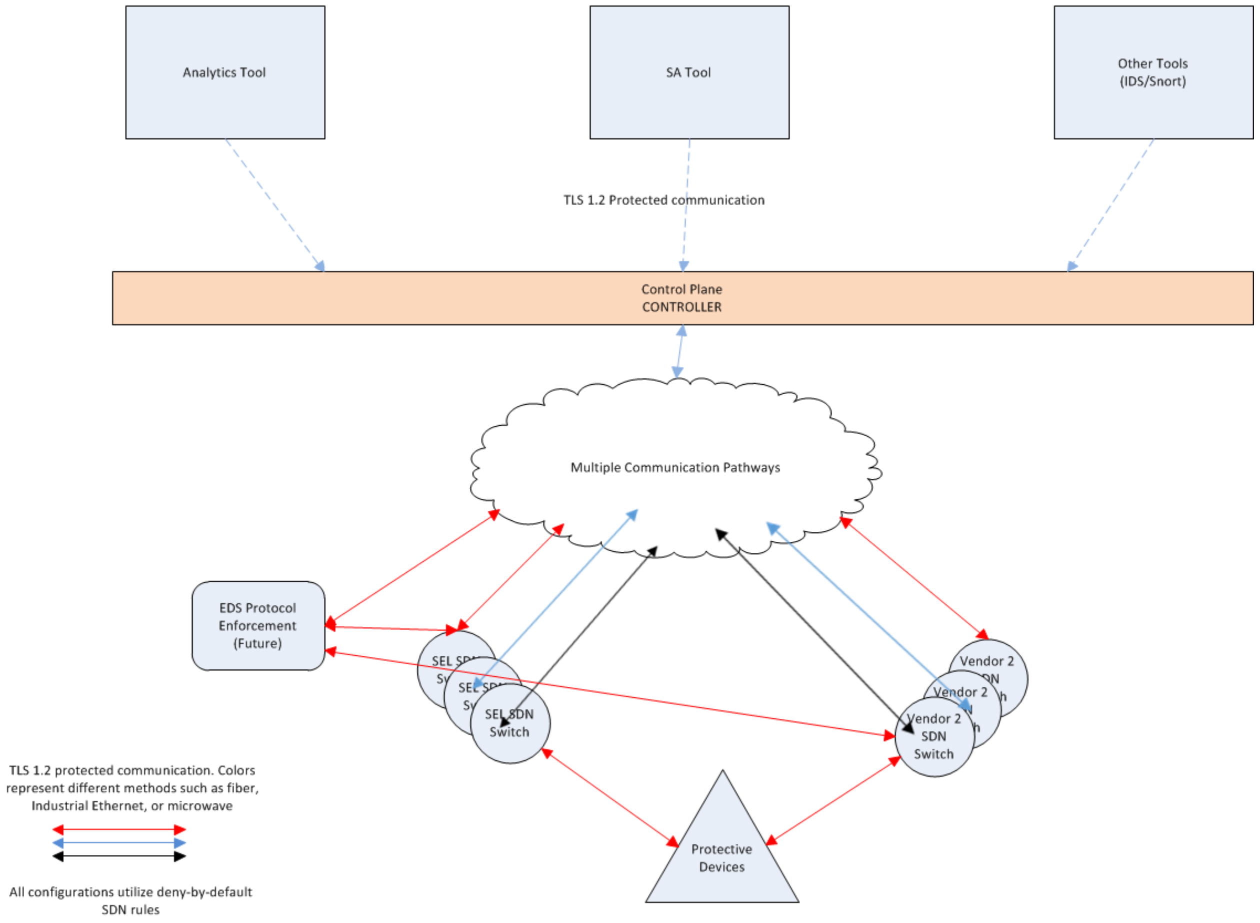
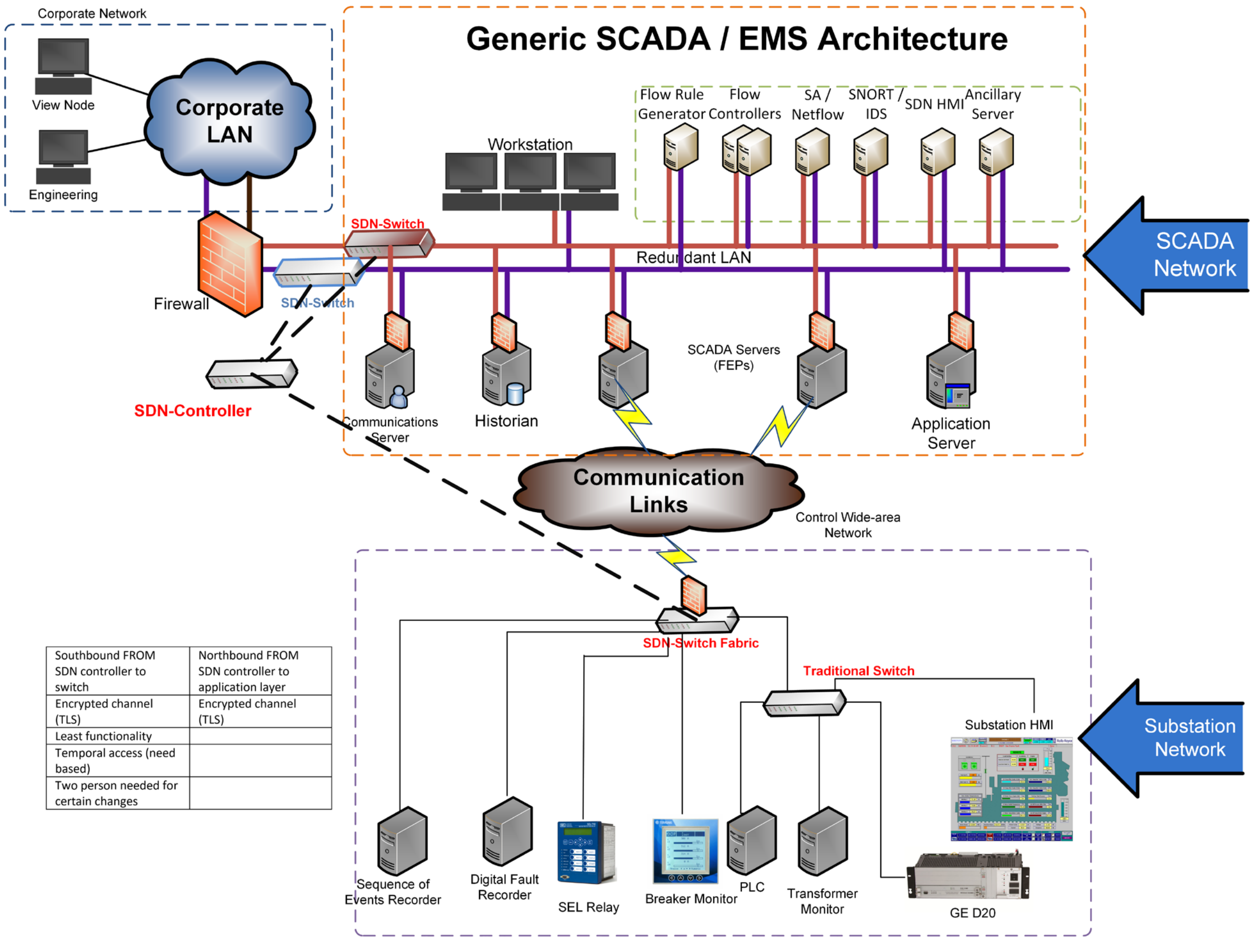

Analytics Tool
SA Tool
Other Tools (IDS/Snort)
TLS 1.2 Protected communication
Control Plane CONTROLLER
Multiple Communication Pathways
EDS Protocol Enforcement (Future)
SEL SDN Switch
Vendor 2 SDN Switch
Protective Devices
TLS 1.2 protected communication. Colors represent different methods such as fiber, Industrial Ethernet, or microwave
All configurations utilize deny-by-default SDN rules

Figure 12-1. Initial Reference Architecture

12.2 Detailed Conceptual Reference Architecture

Figure 12-2 shows a high-level overview of a notional electric power SCADA network, including a control center, a wide-area communications network, and a substation network. The WAN is often owned and operated (at least partially) by the utility, using fiber optic lines run along transmission lines, and private radio and microwave. Communications leased from a common carrier also may be used. Typically, an additional (backup) control center and many more substations would be connected to the network. Control centers in an electric power system usually have links to other control centers, such as neighboring transmission operators, generator operators, and reliability coordinators. These links are typically high-speed links leased from common carriers.

Also shown are the two logical networks that would be implemented in an SDN: 1) a traditional SCADA telemetry and control network and 2) an SDN control network. The traditional network transmits all the data and commands necessary to run the SCADA system, while the SDN network manages the configuration of the SDN switches and provides SA data and alerts to the SDN management nodes at the control center.



Southbound FROM SDN controller to switch	Northbound FROM SDN controller to application layer
Encrypted channel (TLS)	Encrypted channel (TLS)
Least functionality	
Temporal access (need based)	
Two person needed for certain changes	

Figure 12-2. Network Overview

12.2.1 Control Center Network Architecture

Figure 12-3 shows a notional control center network component from Figure 12-2. It is shown with a redundant physical network, as this is typically required for availability. The networks are shown using SDN switches, connected to an SDN controller node (not shown in this figure) that manages the configuration of the SDN switches. The control center network typically is connected to the corporate enterprise network through a traditional firewall (not shown in this figure). The SCADA servers communicate with field devices (typically substations and generating plants in an electrical system) over a WAN. The communications servers manage the communication and process data received or transmitted to other control centers using communication links not shown in the figure. The remaining nodes on the bottom of the figure, along with the workstations in the upper left, comprise the components of the control system typically found at an electric power control center.

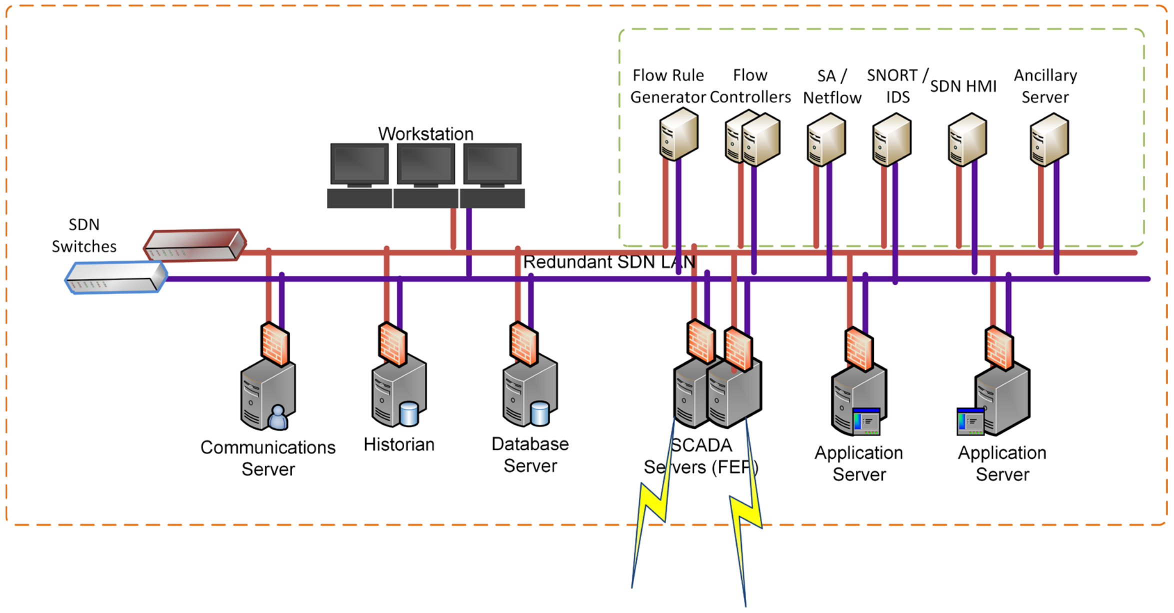


Figure 12-3. Control Center Network

The nodes in the upper right of the drawing, contained within the green hashed box, represent the SDN and support control components of the system these are located on the control center network for illustrative purposes, and could be located on another secure network. The Flow Rule Generator is used to engineer the network by designating what nodes in the SDN are allowed to communicate, and specifying the protocols are allowed. The flow rules, once generated, are sent to the SDN flow controller nodes that communicate directly with the SDN switches to populate the flow rule tables in the individual switches. Multiple SDN flow controller nodes are shown, to provide operational redundancy, as well as to allow the SDN network to scale. Although not shown, the database containing the flow rules also is redundant allowing for failure of a component of the storage infrastructure (i.e., a disk) to not impact the ability to update or install flow rules. Application node clustering and virtual storage networks could be used to increase the availability of the SDN flow controller and associated storage. The remaining nodes (SA or Netflow, Snort-IDS, and SDN HMI) are used to gather, process, and display network performance and statistic data. The SDN switches also capture link status change events on the SDN switches and send them to the SA node for processing to determine if nodes have been added or removed from the operational network. The SA function also could be able to detect if a new or changed MAC address were to start communicating on a physical port, even if the link status were unchanged.

All Control Plane traffic is encrypted to prevent eavesdropping on control or status messages sent through the SDN environment. The authentication server component of the SDN flow controller nodes provides the X.509 (or equivalent) credentials to secure the encryption. The authentication of Control Plane traffic can assist in the “deny-by-default” activity for control or configuration actions to the SDN network.

The SDN flow controller also contains an authentication server used to provide authentication services (e.g., X.509 certificates) when populating rulesets into the SDN switches, and for authenticating other access to the SDN switch fabric. An Ancillary Server provides other services to the SDN and operational environment, such as anti-virus and patching services.

Host-based firewalls are shown on the operational nodes of the control center to assist the filtering and IDS or IPS capability of the SDN environment in further protecting the operational servers from attack. The host-based firewall can also provide limited anti-virus processing if a full anti-virus solution cannot be installed on the operational nodes.

12.2.2 Substation Network Architecture

Figure 12-4 shows a notional substation network component from Figure 12-2 that includes a hybrid network consisting of an SDN component and a traditional component. While not shown, the network is typically a resilient network comprised of a set of redundant networks, redundant switches, and redundant equipment designed to operate together so that no single equipment failure (relay, switch, cable) can result in a loss of observability or control of the substation electrical equipment.

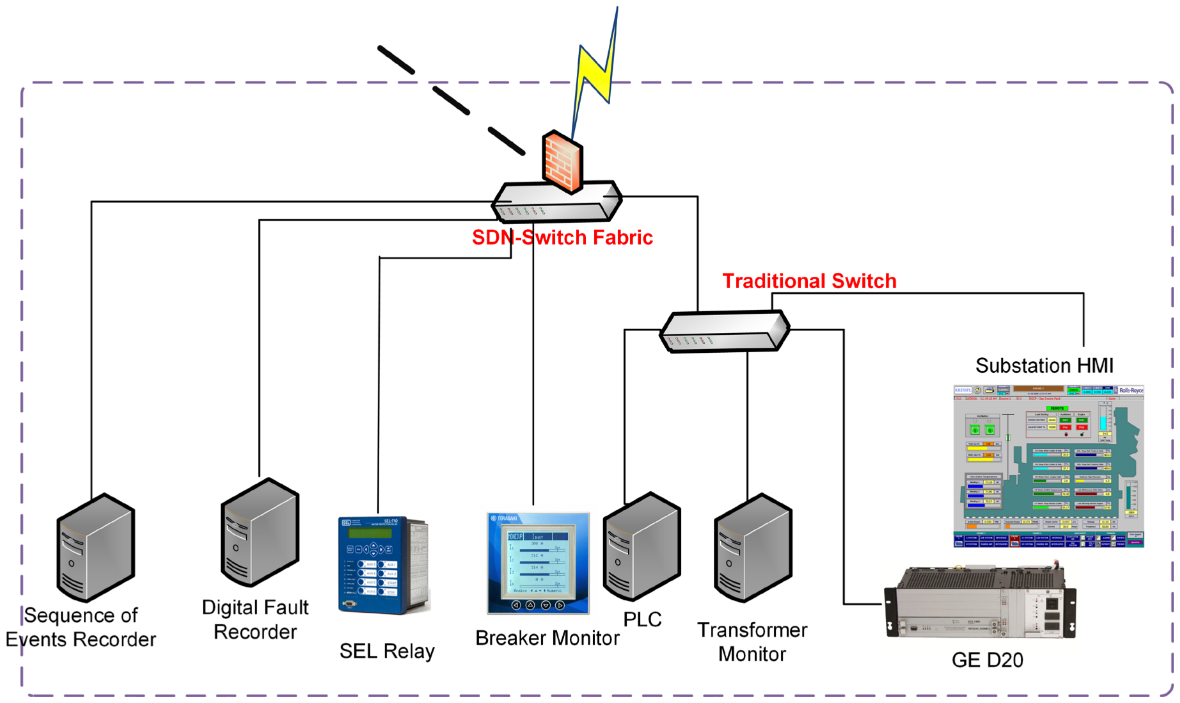


Figure 12-4. Notional Substation Network

Two external connections are shown: one for telemetry and control data for the power system (the lightning bolt), and the other for SDN control communications (the dashed line). These may be implemented on a single physical link but are logically separate. A firewall with IDS or IPS technology is implemented on the logical connection used for operational data transport to prevent large classes of unwanted network traffic from entering or leaving the substation environment, reducing the amount of traffic that must be processed by the SDN flow rules, and limiting the spread of any malicious traffic from a single compromised substation network. Because many devices in a substation environment cannot run traditional anti-virus software, this firewall can also perform network-based anti-virus scanning to minimize the introduction of malware into the environment.

Figure 12-5 shows a notional substation network deployed using the IEC 61850 substation automation protocol. IEC 61850 specifies two different types of network traffic, designated to run on either a “substation bus” or a “process bus”.

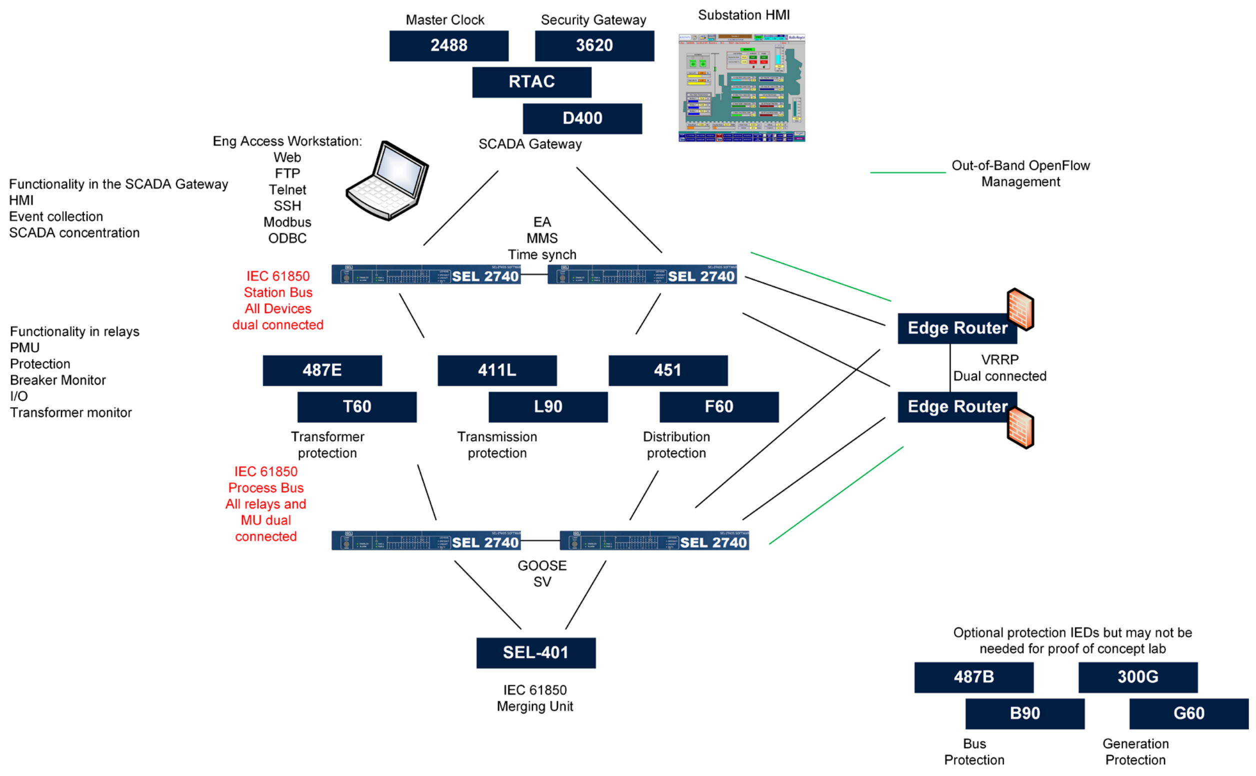


Figure 12-5. Substation Network

The substation bus connects all devices in the substation, providing substation-wide control and monitoring. The process bus provides the digital network equivalent of continuous analog signals to the protection relays. The data volume on the process bus typically does not allow it to span the entire substation, particularly for a very large substation, so multiple process busses are often seen. Both the substation bus and process bus are redundant networks, with the protection relays connecting to both networks, to receive constant data from the field equipment, as well as to coordinate with other relays in the substation.

Although not shown in the figure, the substation typically has a communication controller or “substation gateway” to manage communications to the control center over a WAN for telemetry and control, logically connected to the edge routers. An electric power substation also has WAN links (often point-to-point) to neighboring substations to coordinate line protection functions between the protection relays in the substations. In an SDN environment, it also has an SDN management link to the control center to receive flow rule updates and install them into the SDN switches, and to provide SA alerts and performance data to the SDN analytics functions also located at the control center.

Figure 12-6 shows a notional resilient SDN network architecture, with five SDN switches configured together for each network in the substation. Each switch is interconnected with four other switches, and the critical operational process elements (relays, merging units, etc.), are each connected to two different switches. Non-critical components are not connected to multiple switches. This configuration allows for any single network component (i.e., switch, network link) to fail without impacting the operation of the network. Operational resiliency would be accomplished by configuring multiple operational components and connecting them to different switches.

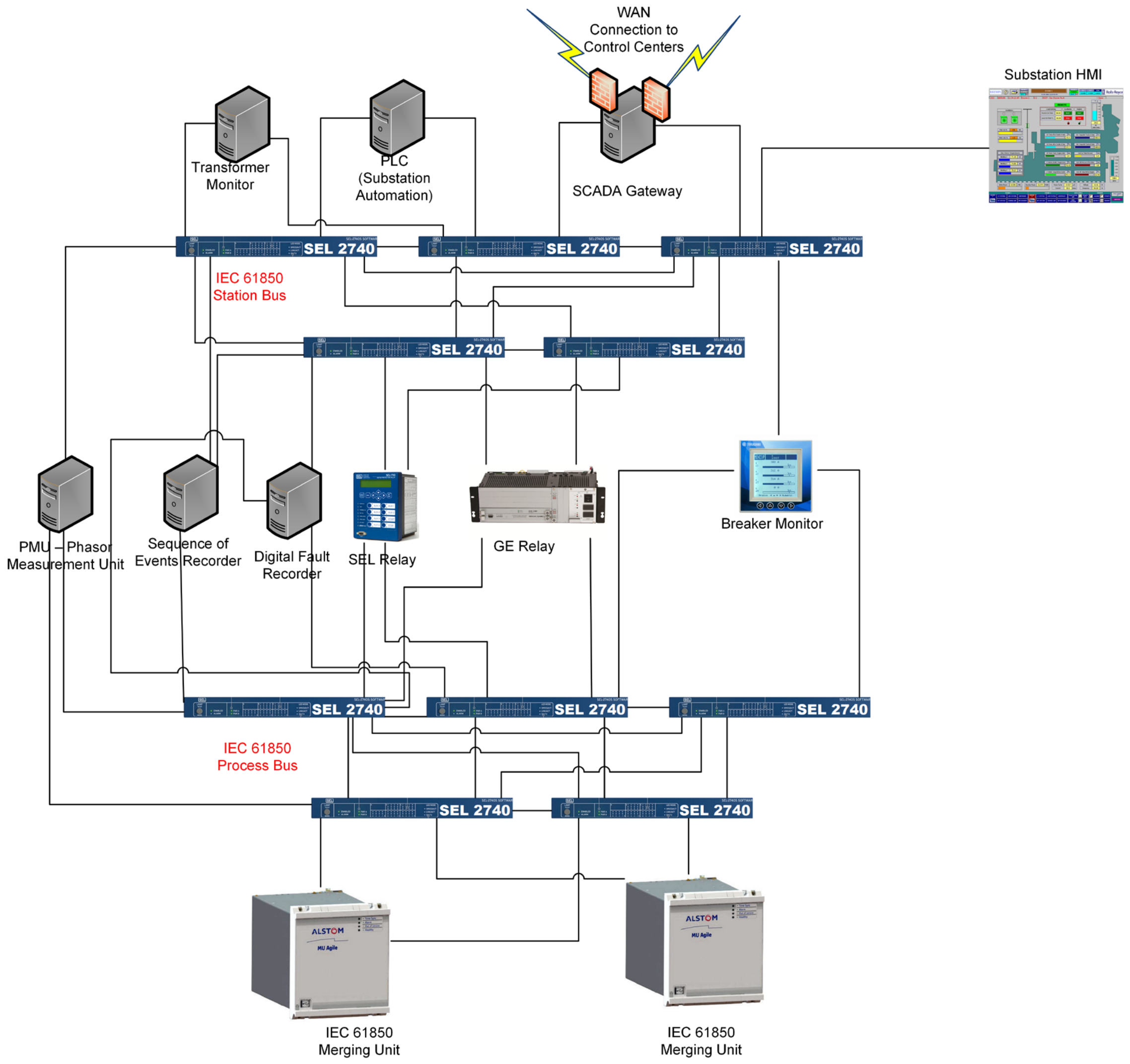


Figure 12-6. Redundant Network in Substation

12.2.3 Wide-Area Network Architecture

Figure 12-7 shows a notional WAN connecting the primary and backup control centers to a several substations. Each location is shown with a physical connection to the wide-area “cloud,” and logical connections from the control center to each substation, and logical connections connecting the substations. The connections from the control center are used for telemetry and control by the control center SCADA systems. The connections between the substations are used for autonomous protection and control coordination between the protection relays located at each end of a transmission line. Typically, elements of the WAN are replicated to minimize the impact of equipment failure. Each control center would have two physical links into the wide area cloud, terminating at different equipment (possibly at a different provider central office). Key or critical substations could also have redundant physical links, while less important substations may only have a single link. In some cases, separate WANs (sometimes provided by different carriers) are used.

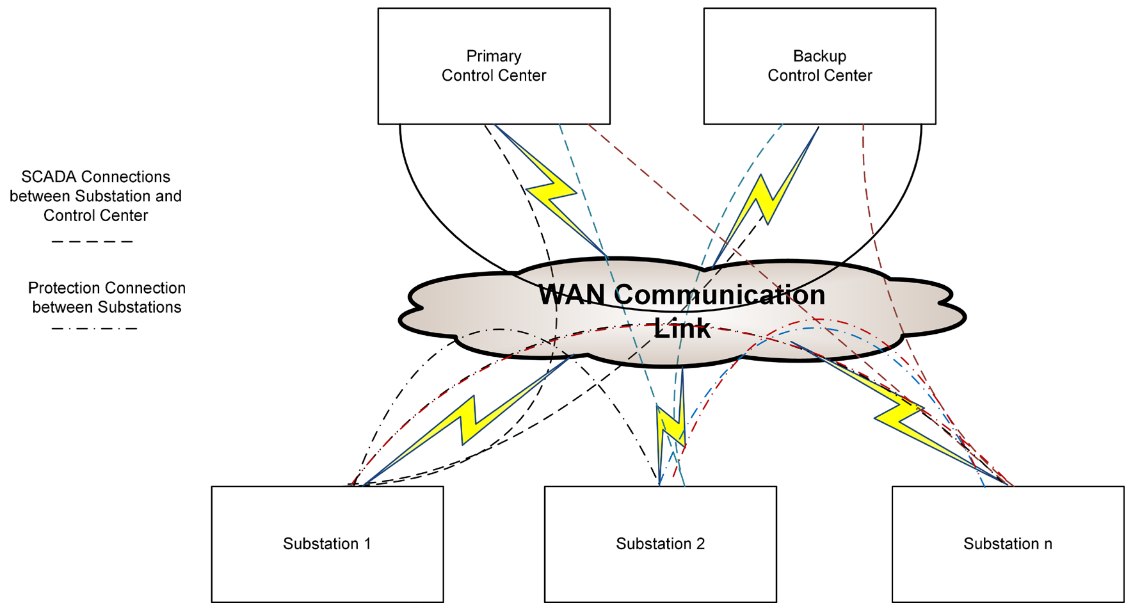


Figure 12-7. Wide-Area Network

As with other aspects of the SDN environment, the WAN traffic is engineered to only allow traffic flows (protocols and addresses) that are specifically allowed. A compromised substation network would still only be able to communicate with other networks or nodes if allowed by the configuration, and only using protocols that were configured; it would not be able to communicate *ad hoc* to other locations or use non-configured protocols.

12.2.4 Control Plane Architectures

The control plane of an SDN can be configured either as “in-band” or “out-of-band” as shown in Figure 12-8 and Figure 12-9. An in-band configuration uses the SDN network itself as the communication mechanism between the SDN Flow Controller and the SDN switches, while an out-of-band configuration uses a separate network.

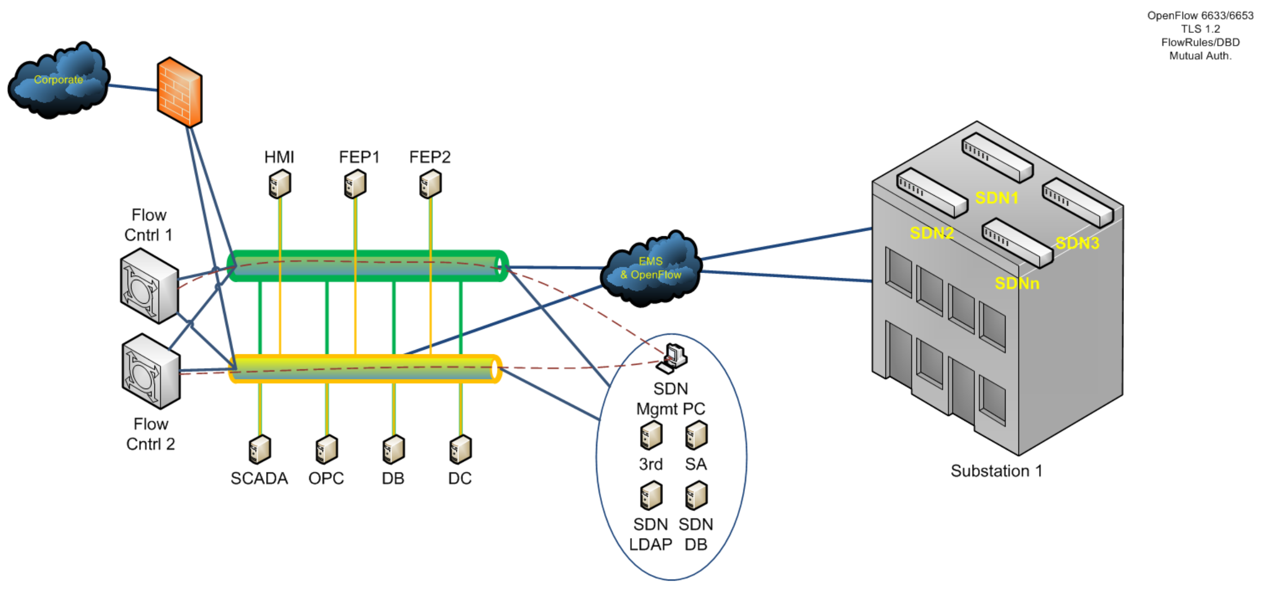


Figure 12-8. SDN Logical In-Band Controller

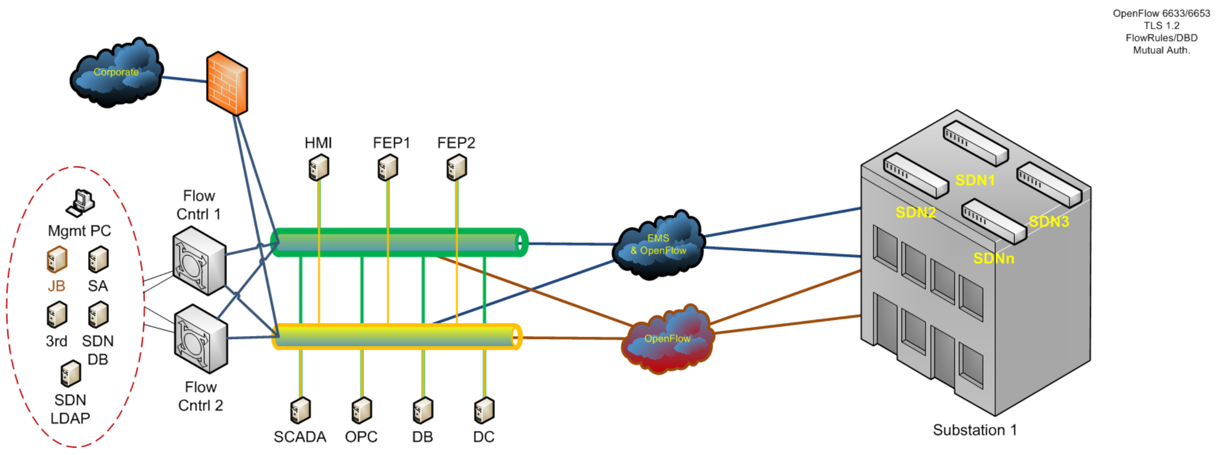


Figure 12-9. SDN Logical Out-of-Band Controller

12.3 Test Environment Architecture

The SDN4EDS test environment network consists of six SEL 2740S SDN switches, one Allied Telesis SDN switch, a SEL 5056 SDN Flow Controller, 16 Raspberry Pi SBCs simulating various types of OT traffic and protocols, four SEL protection relays, one SEL IEC 61850 merging unit, a global positioning system (GPS)-synchronized clock, one Binary Armor IDS, one Suricata IDS, three Juniper SD-WAN routers, a VMware VM infrastructure, and several routers and switches that are used to connect the test environment to the PNNL corporate network and facilitate external access.

A high-level overview of the SDN4EDS test environment is illustrated in Figure 12-10. It is logically separated from the PNNL campus network. This diagram shows the SDN test network environment in the lower right section of the drawing, along with connections to PNNL's corporate network in the upper right section of the drawing, and the location of the Red Team test equipment in the left section of the diagram.

More detailed diagrams showing the equipment configured as part of the SDN test network are shown in Figure 12-11 and Figure 12-12. Figure 12-11 shows the configuration of an example substation network containing a mixture of real and simulated substation equipment connected to a set of five Schweitzer Engineering Laboratories, Inc.(SEL) 2740S SDN switches. The diagram in Figure 12-12 shows a simulated control center environment, in this configuration with direct Ethernet SDN connections to the substation network. This control center network would typically contain an energy management system or a s SCADA system for controlling the substation, and contains the SDN Flow Controller, the interface to the SD-WAN environment that provides a simulated DER generator communicating back to the SDN test environment using DNP3, and the connection to the corporate network.

Figure 9-4 and Figure 9-5 show how the Binary Armor and Suricata IDS were installed into the environment. Outbound traffic flows from the DNP3 master station first through the Suricata IDS, then to the Binary Armor IDS, and finally to the DNP3 out station. Traffic from the DNP3 out station flows first to the Binary Armor IDS, then to the Suricata IDS, and finally going to the DNP3 master station.

Figure 12-13 shows the SD-WAN network that was used to connect the test environment at the PNNL lab campus in Richland, WA to a simulated DNP3 outstation located at the National Renewable Energy Laboratory (NREL) campus in Golden, Colorado.

The substation environment also contains an out-of-band network (not shown in the figure) that was installed due to the restrictions placed on in-person laboratory access during the COVID-19 pandemic. This network allows access to primarily the Raspberry Pi devices in the event that inadvertent SDN mis-configurations prevented access to those devices and provides unrestricted access so that they can be reconfigured or reset remotely. This network would not be present in a real environment but was useful in the test environment to troubleshoot configurations.

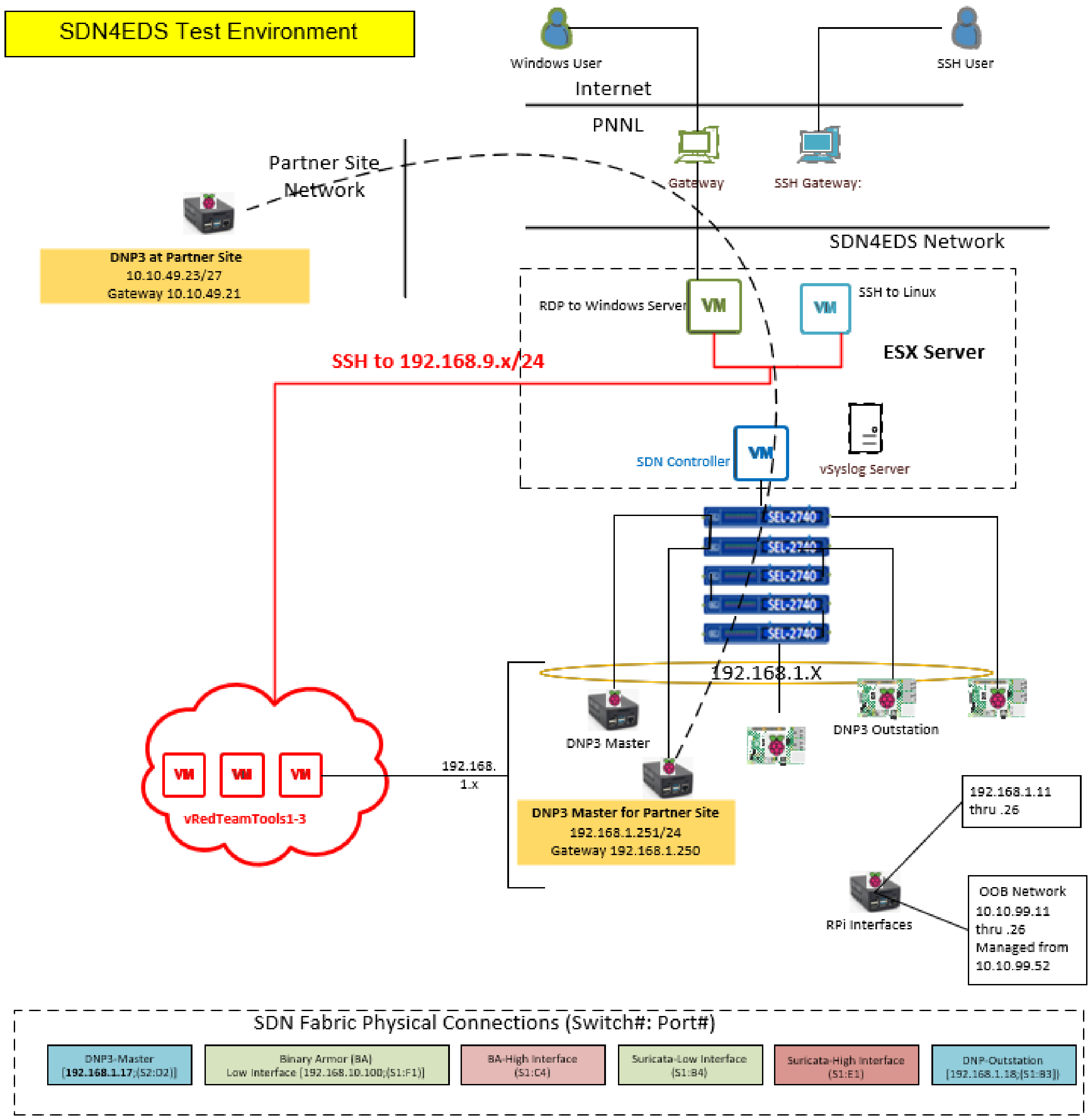


Figure 12-10. SDN4EDS Laboratory Environment - 1

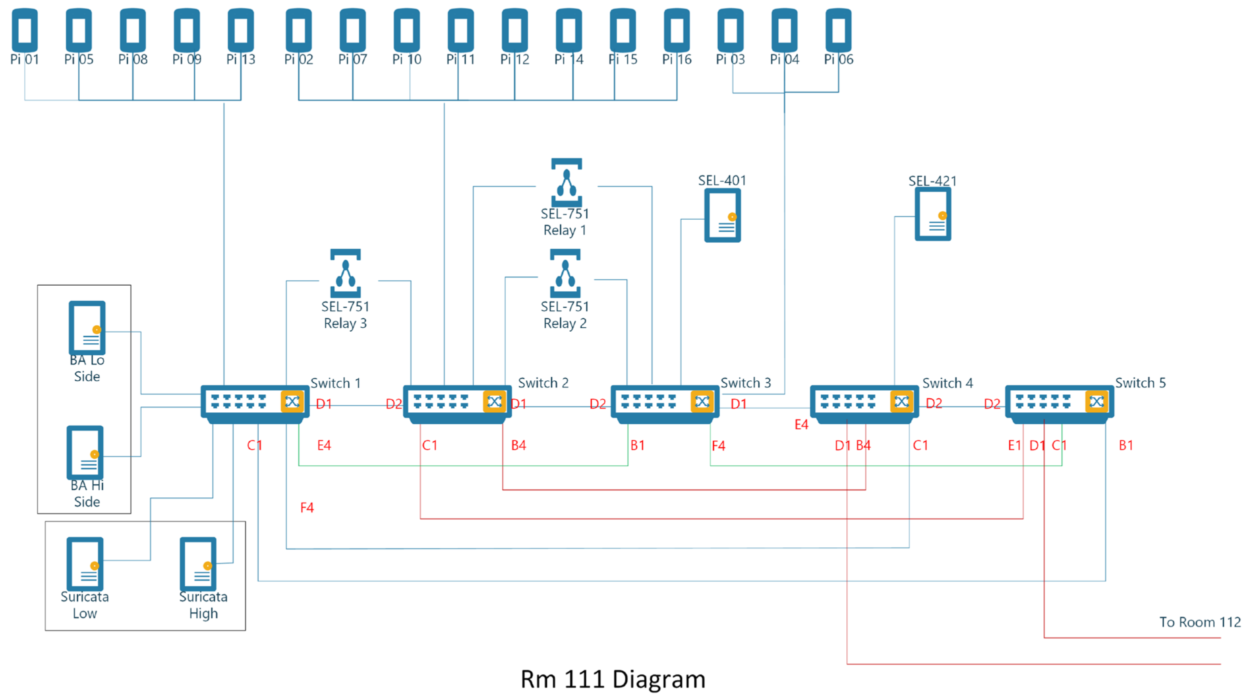


Figure 12-11. SDN4EDS Laboratory Environment - 2

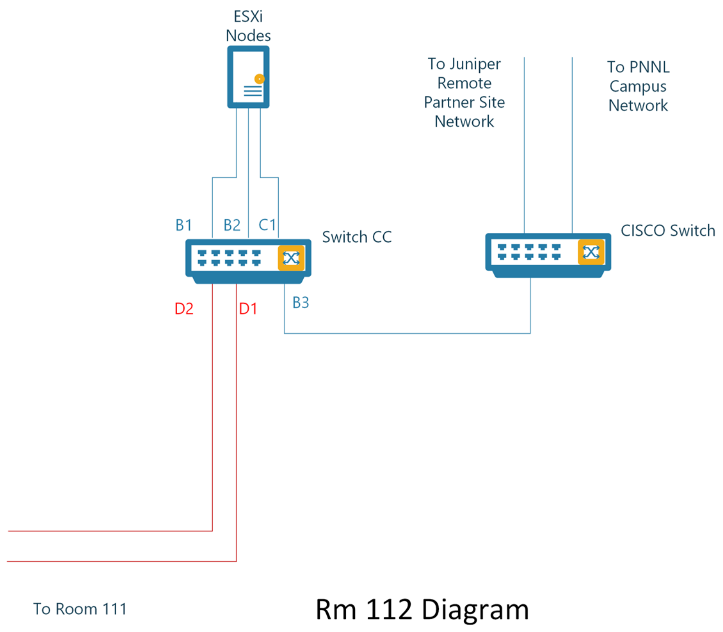


Figure 12-12. SDN4EDS Laboratory Environment - 3

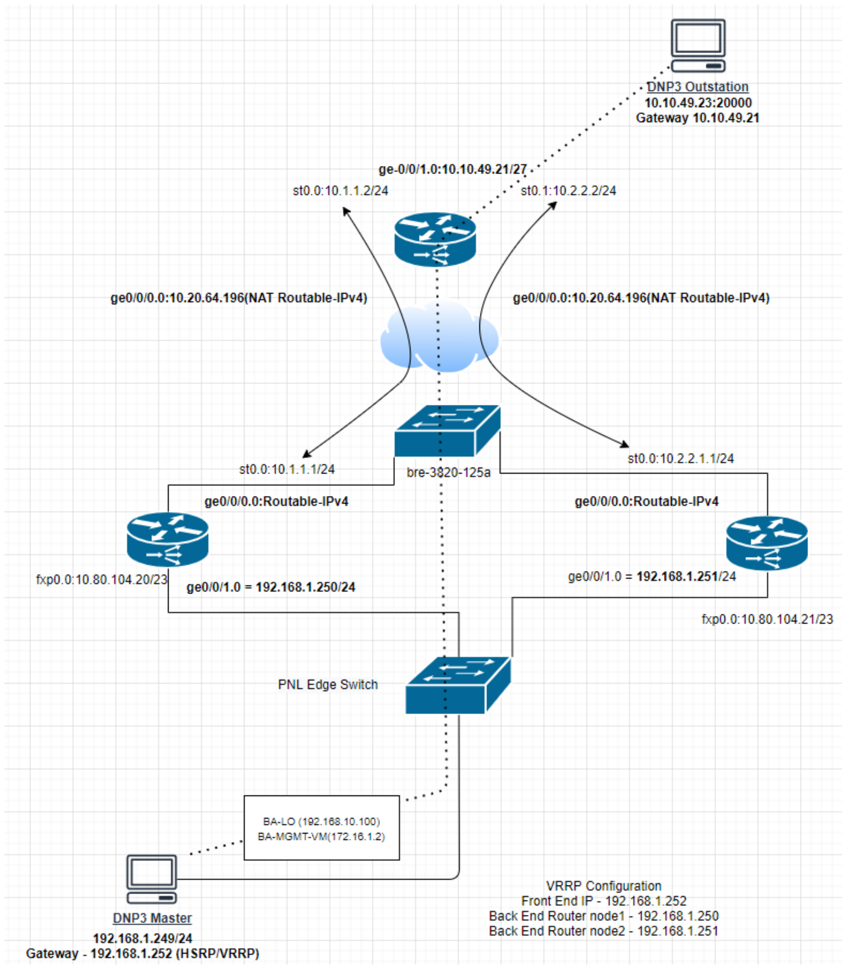


Figure 12-13. WAN connection to NREL

13.0 References

[Amiri 2019] E. Amiri, E. Alizadeh and K. Raeisi, "An Efficient Hierarchical Distributed SDN Controller Model," *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)"*, Tehran, Iran, 2019, pp. 553-557.

[Bailey 2016] Josh Bailey, , and Stephen Stuart. "FAUCET: Deploying SDN in the Enterprise". ACM Queue 14 Issue 5 (2016): 54-68. Available at https://research.google.com/pubs/pub45641.html. (Accessed July 23, 2021)

[Bobba 2014] R. Bobba, D. R. Borries, R. Hilburn, J. Sanders, M. Hadley, and R. Smith, "Software-Defined Networking Addresses Control System Requirements," April 2014. Available: https://www.selinc.com

[Brewer 2000] Brewer, E. A. Towards robust distributed systems. (Invited Talk). Principles of Distributed Computing, Portland, Oregon, July 2000.
[Canini 2012] Canini, Marco, Daniele, Venzano, Peter, Perešíni, Dejan, Kostić, and Jennifer, Rexford. "A NICE Way to Test Openflow Applications." . In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation* (pp. 10). USENIX Association, 2012. Available at https://www2.cs.duke.edu/courses/fall14/compsci590.4/Papers/NICE12.pdf. (Accessed July 27, 2021)

[Case 1990] J. Case, M. Fedor, M. Schoffstall, and J. Davin. *A Simple Network Management Protocol (SNMP).* RFC 1157, May 1990. Available at https://tools.ietf.org/pdf/rfc1157.pdf. (Accessed July 27, 2021)

[Cheshire 2013] Cheshire, S., and M. Krochmal. *Special-Use Domain Names*. RFC 6761, February 2013. Available at https://www.rfc-editor.org/rfc/pdfrfc/rfc6761.txt.pdf. (Accessed August 10, 2021)

[Comer 2019] D. Comer and A. Rastegarnia, "Externalization of Packet Processing in Software Defined Networking," in IEEE Networking Letters, vol. 1, no. 3, pp. 124-127, Sept. 2019.

[Fernandez 2013] M. P. Fernandez, ‘‘Comparing OpenFlow controller paradigms scalability: Reactive and proactive,’’ in Proc. Int. Conf. Adv. Inf. Netw. Appl. (AINA), Mar. 2013, pp. 1009–1016

[Fuller 1993] Fuller, Vince, Tony Li, Jessica Yu, and Kannan Varadhan. *Classless inter-domain routing (CIDR): an address assignment and aggregation strategy*. RFC 1519, September 1993. Available at https://www.rfc-editor.org/rfc/pdfrfc/rfc1519.txt.pdf. (Accessed August 6, 2021)

[Fuller 2006] Fuller, Vince, and Tony Li. *Classless inter-domain routing (CIDR): The Internet address assignment and aggregation plan*. BCP 122, RFC 4632, August 2006. Available at https://www.rfc-editor.org/rfc/pdfrfc/rfc4632.txt.pdf. (Accessed August 10, 2021)

[Giatsios 2019] D. Giatsios, K. Choumas, P. Flegkas, T. Korakis, J. J. A. Cruelles and D. C. Mur, "Design and Evaluation of a Hierarchical SDN Control Plane for 5G Transport Networks," *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, 2019, pp. 1-6.

[Gilbert 2002] S. Gilbert and N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services", ACM SIGACT News, Volume 33 Issue 2 (2002), pg. 51–59. doi:10.1145/564585.564601.

[Haleplidis, 2015] E. Haleplidis, K. Pentikousis, S. Denazis, J. Hadi Salim, D. Meyer, and O. Koufopavlou. *Software-Defined Networking (SDN): Layers and Architecture Terminology*. RFC 7426, January 2015. Available at https://tools.ietf.org/pdf/rfc7426.pdf (Accessed July 23, 2021)

[Hinden 2006] Hinden, Robert, and Stephen Deering. *IP version 6 addressing architecture*. RFC 4291, February 2006. Available at https://www.rfc-editor.org/rfc/pdfrfc/rfc4291.txt.pdf. (Accessed August 10,2021)

[IEEE 802.1X] IEEE Std 802.1X, "IEEE Standard for Local and Metropolitan Area Networks--Port-Based Network Access Control," in IEEE Std 802.1X-2020 (Revision of IEEE Std 802.1X-2010 Incorporating IEEE Std 802.1Xbx-2014 and IEEE Std 802.1Xck-2018) , vol., no., pp.1-289, 28 Feb. 2020, doi: 10.1109/IEEESTD.2020.9018454. https://ieeexplore.ieee.org/document/9018454. (Accessed November 10, 2021).

[IEEE 1588] IEEE Std 1588, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," in IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008) , vol., no., pp.1-499, 16 June 2020, doi: 10.1109/IEEESTD.2020.9120376. https://ieeexplore.ieee.org/document/9120376. (Accessed November 10, 2021)

[IEEE 1815] IEEE Std 1815, "IEEE Standard for Exchanging Information Between Networks Implementing IEC 61850 and IEEE Std 1815(TM) [Distributed Network Protocol (DNP3)]," in IEEE Std 1815.1-2015 (Incorporates IEEE Std 1815.1-2015/Cor 1-2016) , vol., no., pp.1-358, 16 Dec. 2016, doi: 10.1109/IEEESTD.2016.7786998. https://ieeexplore.ieee.org/document/7786998 (Accessed December 3, 2021).

[Johnson 2008] Implementing the Pure Digital High-Speed Substation, Augustus Johnson IV, Substation Communications Engineer, Dominion Virginia Power Electricity Today, November/December 2008

[Koshibe 2014] A. Koshibe, A. Baid, and I. Seskar, "Towards distributed hierarchical SDN control plane," *2014 International Science and Technology Conference (Modern Networking Technologies) (MoNeTeC)*, Moscow, 2014, pp. 1-5.

[McCloghrie 1991] K. McCloghrie and M. Rose. *Management Information Base for Network Management of TCP/IP-based internets: MIB-II*. RFC 1213, March 1991. Available at https://tools.ietf.org/pdf/rfc1213.pdf. (Accessed July 27, 2021)

[Rekhter 1998] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear. *Address Allocation for Private Internets*. RFC 1918, February 1996. Available at https://tools.ietf.org/pdf/rfc1918.pdf. (Accessed July 27, 2021)

[McCloghrie 2000] K. McCloghrie and F. Kastenholz. *The Interfaces Group MIB*. RFC 2863, June 2000. Available at https://tools.ietf.org/pdf/rfc2863.pdf. (Accessed July 27, 2021)

[O'Raw 2017] J. O'Raw, D. M. Laverty and D. J. Morrow, "IEC 61850 substation configuration language as a basis for automated security and SDN configuration," 2017 IEEE Power & Energy Society General Meeting, Chicago, IL, 2017, pp. 1-5.

[OpenFlow 2012] Open Network Foundation *OpenFlow Switch Specification Version 1.3 (Wire Protocol 0x04)*, June 25, 2012, document ONF TS-006, available at https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.0.pdf. (Accessed April 1, 2021)

[OpenFlow 2014] Open Network Foundation *OpenFlow Switch Specification Version 1.3.4 (Wire Protocol 0x04)*, March 27, 2014, document ONF TS-019, available at https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.3.4.pdf. (Accessed April 1, 2021)

[Rekhter 1998] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear. *Address Allocation for Private Internets*. RFC 1918, February 1996. Available at https://tools.ietf.org/pdf/rfc1918.pdf. (Accessed July 27, 2021)

[Rose 1990] M. Rose and K. McCloghrie. *Policy Requirements for Inter Administrative Domain Routing*. RFC 1155, May 1990. Available https://tools.ietf.org/pdf/rfc1155.pdf. (Accessed July 27, 2021)

[SEL 2017] Schweitzer Engineering Laboratories, Inc. (SEL) *Implementing Packet Sniffing (IDS, DPI, Port Mirroring, etc.) in an SDN Network, Application Guide AG2017-18* available from selinc.com

[SEL 2019] Schweitzer Engineering Laboratories, Inc. (SEL) *SEL-2740S Software-Defined Network (SDN) Switch and SEL-5056 SDN Flow Controller Instruction Manual* available at https://selinc.com/api/download/117185/ (registration required). (Accessed April 1, 2021)

[SEL 2020] Schweitzer Engineering Laboratories, Inc. (SEL), *REST API Preliminary Draft for v2.1.0.0 Draft Only*, March 2020 (unpublished)

[Shah 2018] R. Shah, M. Vutukuru and P. Kulkarni, "Cuttlefish: Hierarchical SDN Controllers with Adaptive Offload," *2018 IEEE 26th International Conference on Network Protocols (ICNP)*, Cambridge, 2018, pp. 198-208.