Software-Defined Security Services Framework

Version 2.0

A white paper from the
ONUG Software-Defined
Security Services Working Group

May, 2016

SOFTWARE-DEFINED
SECURITY SERVICES
WORKING GROUP
2016
Open Networking
USER GROUP

Definition of Open Networking

Open networking is a suite of interoperable software and/or hardware that delivers choice and design options to IT business leaders, service and cloud providers. At its core, open networking is the separation or decoupling of specialized network hardware and software - all in an effort to give IT architects options in the way in which they choose to design, provision, and manage their networks. These technologies must be based on industry standards. The standards can be de-facto as adopted by a large consortium of the vendor community, open in the sense that they are community based, or defined as standards by the prevailing standards bodies. Open networking hopes to deliver on two promises:

1) Decoupling of network hardware and software which mitigates vendor lock-in and shifts network architecture structure options to users
2) Significant reduction of the total cost of ownership model, especially operational expense

SOFTWARE-DEFINED SECURITY SERVICES WORKING GROUP
2016
Open Networking
USER GROUP

Table of Contents

The Security Triad

Security as a discipline has extremely wide scope. It touches upon virtually every activity in business and deals proactively with the management of risk. Across the information technology (IT) landscape, security addresses specific risks in the physical, administrative and technical business realms, while also manifesting its own set of special challenges in dealing with the complexity and cost that attend these activities.

Generally, information security deals with the risks presented by threats to the confidentiality, integrity and availability (often abbreviated as the "CIA") of information assets - the "security triad." Information assets are represented by data and the information services that manage data. Information services are assembled, deployed and operated to enable data and information to be created, shared and used by the enterprise to pursue its business interests, execute its mission and deliver value to its stakeholders. The purpose of information security is to ensure that these assets are there to support the business. The security triad (CIA), is a well-known and commonly used term throughout the InfoSec industry globally. It was officially codified by the U.S. government under the Federal Information Security Management Act of 2002 (FISMA).

Within this context, every information service comprises one or more service elements, each of which is dependent on one or more systems to perform specific functions under stated conditions on behalf of service consumers. It is the combination of functions deployed across these systems that defines the composition of the service. Every system supporting an information service (or a service element thereof) is composed from some combination of *people*, *processes* and *technologies*, where each of these constituent elements brings with it its own specific set of security concerns. These concerns include technology vulnerabilities (defects in architecture, design, implementation, deployment, operation, support, retirement) that make a system susceptible to failure and threats that may trigger or exploit a vulnerability. Similarly, both people and processes possess vulnerabilities that can make those system elements susceptible to failure (e.g., via illness, human failure, procedural flaws and complexity). The potential of encountering any of these failures represents risk to the business. As a result, information security requires taking a holistic approach to how information systems are developed, deployed, operated, sustained and retired (the *system lifecycle*). This is applicable across all of the constituent elements that define the system.

A properly engineered information system considers early on in its lifecycle the need for specific security controls – *safeguards* and *countermeasures* – required to deal with vulnerabilities and protect information assets from threats, both naturally occurring and adversarial. These early considerations are driven by security principles and policies established by the enterprise, as well as regulations and laws imposed by oversight and governing bodies. These drivers, together with the assessed risks to the business and its assets, give rise to the security requirements for information systems.

The proper engineering of any system relies first upon understanding the problem at hand and then crafting an architecture and solution that solves the problem. Within the discipline of systems engineering, there is a clear distinction between the *problem space* and the *solution space* that results in a separation of activities, roles and responsibilities between the two. The problem space involves understanding what needs to be accomplished, while the solution space deals with how to accomplish it. While an obvious distinction, it is critical to keep this in mind, for the activities associated with developing requirements properly fall within the problem space. In our view, it is accordingly the responsibility of those consuming a product or service (the customers) to articulate their needs to the supplier(s) that provide that product or service. Conversely, the activities, roles and responsibilities associated with delivering a product or service sit squarely in the solution space – where product and service suppliers work to satisfy the needs expressed by their customers through requirements. The requirements definition therefore constitutes an agreement between the supplier and consumer that specifies what the supplier will provide to the consumer to be successful in the marketplace.

Open Networking User Group (ONUG)

ONUG is one of the largest industry user groups in the IT infrastructure market. Its board is comprised exclusively of IT business leaders, with representation from Fidelity Investments, FedEx, Bank of America, UBS, Cigna, Pfizer, GE, JPMorgan Chase, Morgan Stanley, Citigroup, Credit Suisse, Gap and Yahoo. The ONUG mission is to enable greater choice and options for IT business leaders by advocating open, interoperable hardware and software-defined infrastructure solutions that span the entire IT stack, all in an effort to create business value.

The ONUG community is led by IT business leaders and aims to drive industry dialogue to set the technology direction and agenda with vendors, standards and open source organizations. To that end, ONUG hosts two major conferences per year where use cases are developed and members vote to establish a prioritized list of early adopter, open interoperable hardware and software-defined infrastructure projects that communicate propensity to buy and budget development. The vendor community stages proof of concepts based upon ONUG Use Cases, while standards and open source organizations prioritize their initiatives and investments based upon them. ONUG organizes working groups to fully develop use cases and set industry initiatives. ONUG also hosts user summits and smaller, regional user-focused Fireside Chat Meet-Ups through the year.

SOFTWARE-DEFINED
SECURITY SERVICES
WORKING GROUP
2016
Open Networking
USER GROUP

Document Scope

The scope of this document is to provide a set of tactical and strategic requirements to help guide enterprises in their thinking and selection of Software-Defined Security Services (S-DSS). While the impacts discussed are commensurate with the Information Technology Infrastructure Library (ITIL) service delivery model, enterprises can leverage the information for a Request for Information (RFI) and adapt to scale and suit their current or planned organizational support delivery and maturity capabilities. The ONUG S-DSS working group seeks to define security requirements for software-defined infrastructure that provides protection equivalent to today's physically isolated networks and infrastructure.

Out of Scope

The working group focused its efforts on what is needed to deliver on three important S-DSS use cases to the enterprise market. The working group did not focus and does not offer the how; that is, there is no specific protocol(s) or Application Program Interface(s) (APIs) specified to deliver on the S-DSS use cases. The working group leaves that work to the vendor and standards communities. Open Networking User Group (ONUG) strongly encourages open interfaces and protocols in the construction of multivendor interoperable S-DSS solutions to deliver the greatest value and choice to enterprise IT executives. Further, the working group does not specify where various S-DSS functions should reside – that is, within physical or virtual products, as modules within existing products or standalone products. That is up to the vendor community.

Executive Summary

The ONUG S-DSS working group prioritized its efforts around three fundamental use cases, which stress that security policies should be bound to workloads independent of how those workloads are created; that is, whether the workload runs with a virtual machine (VM), container, application or applications based upon microservices or unikernels. Further, security policy should be portable, meaning: policy can be written in one place and deployed in multiple locales, where workload policy enforcement is distributed close to said workload. Verification and auditability of security policy must be enabled to measure the ability of network workloads to ensure the confidentiality, integrity and availability of the services they are delivering. Security policy bound to workloads, policy portability, distributed enforcement local to that workload and verification constitute the minimum set of requirements for cloud-based applications and hybrid-cloud deployments within a software-defined infrastructure. In short, these are the minimum security requirements for Software-Defined Security Services.

This document defines an open framework and a common set of functional solution requirements for one of the open networking use cases – Software-Defined Security Services (S-DSS) – identified by the ONUG Board. The content of this document is intended to provide general guidelines for:

- IT enterprise end users to compare vendor solutions and develop formal RFI specifications;
- IT vendors to develop and align product requirements;
- Standards organizations to align their initiatives and efforts to deliver an open approach to S-DSS.

Defining a common set of solution requirements aligns with ONUG's goal to drive the IT vendor and standards communities to deliver open interoperability solutions to provide IT business leaders maximum choice and flexibility in deploying open IT framework solutions. The expectation is that this document provides a common baseline, covering a set of enterprise deployment requirements for S-DSS solutions. The assumption being made is that this set of requirements will be completed by enterprise-specific requirements to meet specific deployment needs. As the working group developed the S-DSS use case, it found that there are, in fact, multiple use cases as the industry transitions from a hardware-based infrastructure to software-defined infrastructure, as discussed below. This version of the document provides but a few use cases in what could be the basis for a new computer network security market.

Introduction

Like building a house or any other structure, secure systems require a solid foundation. The foundations of secure information systems are built upon fundamental concepts that address specific needs spanning the Security Triad imperatives: confidentiality, integrity and availability. Within information system security engineering, these fundamentals form the framework by which higher-level security concepts such as identity, authentication, authorization, auditing and non-repudiation are expressed. These concepts, along with still higher-level constructs and with the principle of defense-in-depth, form the security profile for a system.

The security profile represents a set of requirements that the information system must satisfy to be considered "secure" within the business environment. In the context of a software-defined infrastructure (SDI), these requirements are aimed at the technology suppliers (OEMs & ISVs) whose products and services comprise the infrastructure. As such, it is incumbent upon those articulating needs to be clear about what is being required – both in terms of concepts and language. This includes ensuring that key terms used to construct statements of need (requirements) are unambiguous and well understood between the suppliers and consumers. To accomplish this, an agreed upon vocabulary and taxonomy is needed between those specifying the requirements and those acting upon them. The goal in developing a requirements definition is to define a meaningful set of needs statements that are considered concise, complete, relevant, understandable, non-conflicting, testable and attainable. In other words, such requirements must be actionable on the part of technology suppliers within their product development cycles so as to deliver products that satisfy the consumers they serve.

Software-Defined Security Services

The concept of Software-Defined Security Services (S-DSS) is intended to define security services needed as IT infrastructure transitions from hardware based to a software-defined market. Here, S-DSS describes the set of network-based security controls (safeguards and countermeasures) that overlay an administrative domain within a trusted boundary formed through confidentiality, integrity, identity, authentication and authorization that spans untrusted physical and logical boundaries to satisfy a specified security policy.

This definition is intended to cover an almost limitless combination of architectures that are able to comprise both physical and virtual systems, scale across wide geographical distances and over multiple protocols, while employing different deployment and operational models. This implies that the constituent physical or virtual elements comprising the SDI be resilient in their own right; that is, capable of ensuring that their own confidentiality, integrity and availability characteristics remain sufficiently intact under defined adverse conditions without reliance on external protections. This further suggests the existence of a set of foundational (or core) security requirements that must be satisfied by each system element supporting the S-DSS to achieve component resiliency.

Core Security Requirements

The definition of a "core requirement" is one that is generally applicable across the larger problem space being considered. In this case, a technology supplier whose product or service satisfies such needs would provide an offering with many of the characteristics necessary to deliver secure services. The initial list that follows is not intended to be exhaustive, but rather to provide a baseline starting point for a framework of requirements that can be extended to build higher-level requirements that would inherit the security protections indicated.

The assertion being made here – and explicitly so – is that products delivered to the marketplace (and thus by definition to the ONUG community) cannot address the security concerns of critical infrastructure and high impact business environments unless these foundational requirements are satisfied.

SOFTWARE-DEFINED
SECURITY SERVICES
WORKING GROUP
2016
Open Networking
USER GROUP

IETF RFC 2119 provides a lightweight, consistent and convenient method along with guidance for constructing requirements. In adopting RFC 2119 for defining requirements, the needs articulated below will incorporate the use of specific key words to establish the force of the requirements as stated. The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," "RECOMMENDED," "MAY" and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Core Security Requirement Term Definitions

The following definitions are provided to establish baseline meaning and context for important terms used within individual core security requirement statements appearing below. The terminology describes the operational relationship between subjects and objects in a system. This subject-object relationship is commonly used throughout security engineering practice and is being adopted here to express foundational security needs. As such, these terms are employed as generic synonyms intended to replace specific nouns and noun/verb pairs that would otherwise imply or prescribe a specific architecture, design or implementation of a requirement.

- The term "object" will be defined as an abstract representation of a tangible or intangible thing – a system, system element, system-of-systems, data or other resource – having a defined interaction or access relationship with a subject.
- The term "subject" will be defined as an abstract representation of an actor – a human, machine or other entity operating with intent – having a defined interaction or access relationship with an object.
- The term "security" or "secure" will be defined as maintaining the systemic properties of confidentiality, integrity and availability for an object or object resource within a stated operational context and access policy.
- The term "network-enabled object" will be defined as a physical or virtual object (e.g., switch, router, sensor, server, appliance, device) having presence on a network within an administrative domain and under governance of an administrative policy and actions.
- The term "data element" will be defined as a named identifier with associated value and data type (e.g., integer, float, string, enumeration, data structure).
- The term "control function" will be defined as any method authorized to invoke an action that alters the operational state of an object on behalf of an authorized subject.

Core Security Requirement Statements

The definition of core security requirements is intended to be generally applicable to individual products or services offered by technology suppliers to the ONUG community (and ultimately, beyond). These requirements are applicable to any and all products or services, regardless of their role in the network and are to be evaluated collectively and in total. As the following list of requirements is indeed "core," we forgo the priority assignments here, but apply them for individual use cases below.

1. A network-enabled object must provide for a logically-isolated (from other object functions), secure, logically or physically out-of-band network interface, purpose-built to facilitate the secure monitoring and administration of the object throughout each stage of its in-service lifecycle (i.e., deployment, operation, maintenance, retirement).
2. A network-enabled object must provide secure methods and interfaces for implementing mandatory access controls to facilitate policy-based subject access to the object and its resources.
3. A network-enabled object must be capable of exposing the full set of available operational state reporting and administrative control function elements using a secure method and interface to an authorized subject.
4. A network-enabled object must limit exposure of elements pertaining to its operational state and administrative control functions to only those elements granted to a subject based on the subject's authorization profile and use context (e.g., a polymorphic interface rendering a view/projection of available state and control resources available to the subject).
5. A network-enabled object must provide secure methods and interfaces for querying the object's state by facilitating the retrieval of a specified set of operational data elements on behalf of an authorized subject.
6. A network-enabled object must provide secure methods and interfaces for establishing clipping levels, or thresholds, as mutable values associated with operational data elements to detect state changes within the object. The object must provide a related method, expressed through the same interface, to define action events that may be invoked within the object and/or notifications to be sent to cooperating entities (subjects or other objects), when the object's operational state change results in a transition across (e.g., below to above, or above to below) a specified level.
7. A network-enabled object invoking an action event or issuing a notification in response to a state transition across an established clipping level or threshold must result in output of one or more log entries sent to one or more specified logging destinations, and include as metadata/header information in each message:
   - A precision timestamp derived from a time source with known accuracy, reliability and integrity, having resolution of not less than 1ms and in a format consistent with Internet date/time formats described by RFC 3339 and ISO 8601;
   - A unique (deterministically resolvable) name or identifier of the source object issuing the log entry (e.g., system, device, process);
   - An event priority level indicating the importance of the event to the administrative domain's operational context and policy;
   - A human readable event description issued using the preferred language of the administrative domain;
   - An object-specific payload and format containing encoded data elements/information relevant to the event. The payload may be null;
   - In the absence of a secure communications channel, a message authentication code (MAC) and associated trust chain attesting to the integrity of the message payload and its metadata/header information.
8. A network-enabled object must validate inputs for each mutable administrative control data element modified by an authorized subject and enforce input values and ranges based on the access policy governing the subject.
9. A network-enabled object must provide secure methods and interfaces to facilitate in-band confidentiality of data in transit (encryption on the wire and/or over the air) as the default operational mode.

SOFTWARE-DEFINED SECURITY SERVICES WORKING GROUP 2016
Open Networking USER GROUP

10. A network-enabled object must provide secure methods and interfaces to facilitate in-band confidentiality of data at rest (encrypted in storage) as the default operational mode.
11. A network-enabled object must be capable of secure, unattended startup (boot) and shutdown (halt) without exposing confidential data or administrative credentials/keys over insecure channels or protocols.
12. A network-enabled object must provide secure programmatic methods and interfaces to facilitate object monitoring and administrative control by authorized, cooperating objects. Programmatic interfaces should include language bindings for current (2016) as well as available prior versions dating back two years of: ANSI C/C++, Java, Ruby, Perl, Python, PHP and W3C compliant HTML5 with requisite extensions and/or supporting markup languages.
13. A network-enabled object should incorporate quantum-resistant protocols (e.g., lattice-based cryptography) to mitigate long-term risks associated with deferred cryptanalysis of intercepted data, facilitated through emerging quantum technologies (i.e., intercept now, decrypt later).

Software-Defined Security Services – Use Cases

During the winter months of 2016, the ONUG S-DSS working group met twice a month exclusively with IT executives to capture common security use cases associated with SDI. Two design principles emerged that are fundamental to the following use cases:

1) Software-defined infrastructure must possess a common language to define declaratives and policies that can be consumed up and down the IT stack, physically and virtually.
2) Software-defined infrastructure must be able to instantiate an “Internet-facing workload” in a software-defined data center that provides protection equivalent to today's physically isolated networks.

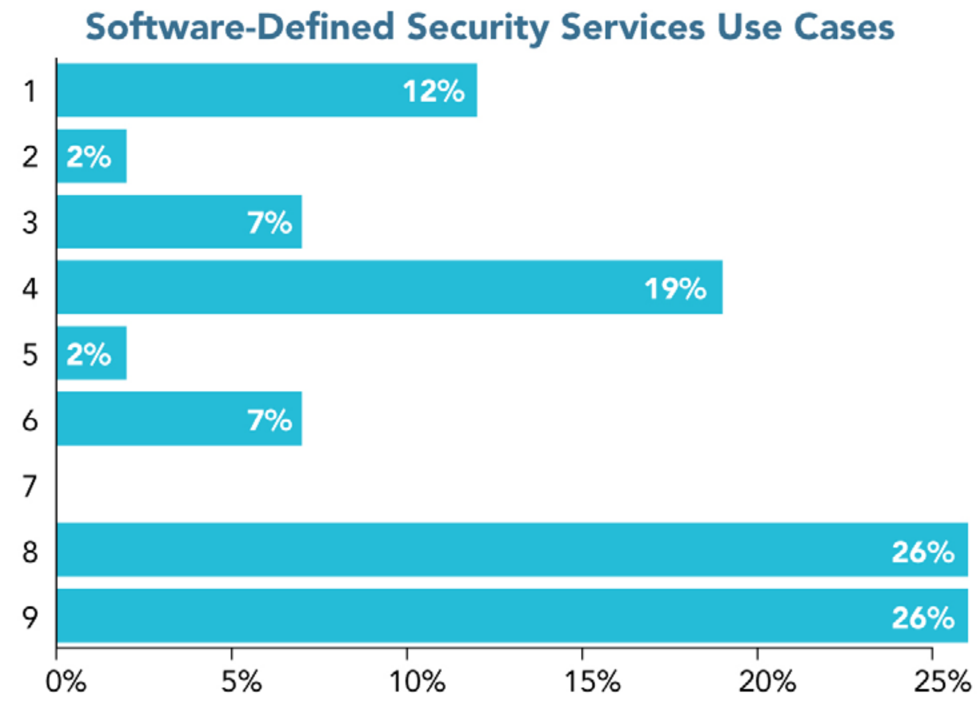
It is with these design principles in mind that the following nine use cases emerged:

1) Dynamically capture packets from both bare metal switches and vSwitches
2) Ensure networks possess "data awareness" as well as "application awareness"
3) Latest encryption capabilities remain within the SDN security services framework
4) Must be able to measure the ability of network workloads to ensure the confidentiality, integrity and availability (the Security Triad) of the services they are delivering
5) Must be able to provide automated security alerts
6) Must be able to provide predictive capabilities
7) Messaging bus implementation
8) Policies should be bound to workloads: whether virtual machines, containers, applications, services or microservices
9) Write security policy in one place and deploy in multiple places, where workload policy would then be enforced.

Software-Defined Security Services – Use Case Voting

The above nine S-DSS use cases were voted upon by IT executives participating in the S-DSS working group to prioritize their development and focus the working group's activities. The following graphic illustrates the aggregated voting results.

From the above voting, the following three use cases were decided to become the focus of the working group's activities. These three use cases have been prioritized based upon the common needs expressed by S-DSS IT executive working group members:



8) Policies should be bound to workloads: whether virtual machines, containers, applications, services or microservices

9) Write security policy in one place and deploy in multiple places, where workload policy would then be enforced

4) Must be able to measure the ability of network workloads to ensure the confidentiality, integrity and availability (the Security Triad) of the services they are delivering

Software-Defined Security Services – Top Three Use Cases and Requirements

The following section provides descriptions and requirements for the above three use cases. We keep their original use case numbers so as to provide consistency.

Use Case #8

Policies should be bound to workloads, such as virtual machines, containers, applications, services or microservices

Problem Statement:

In today's data centers, virtualized servers have provided significant benefits driven by innovation in the more effective design and use of hardware technology, and by the increasingly dynamic nature of implementing and modifying resources (quick start/stop of VMs, memory allocation, processing power, storage allocation, the snapshot of VMs for backups, live VM migration capabilities and more). In comparison, a similar transformation has been taking place with network services and associated resources (routers, load balancers, firewalls, etc.) as they become more virtualized and software-defined.

Network security policies have been traditionally bound to traditional network topologies and attributes. While, in the past, these have been aligned with physical network infrastructure, improvements have been realized of late with the virtualization of network security appliances, which has in turn improved security policy enforcement functionality and efficacy. SDN capabilities have also brought the ability to better orchestrate policies, yet these are still very much tied to traditional constructs that have been responsible for enforcing those policies. These constructs have varied depending on specific network infrastructure deployment architectures, which are typically driven by application needs.

While application architectures have evolved to drive different infrastructure deployment architectures, much of the security policy architecture framework is still constrained by how the network and infrastructure systems are put together (physical or virtual). As application workloads have become more fluid and less dependent on the network and infrastructure, the current approach to define and enforce security policies has lagged behind. This may drive inconsistency in the technical implementations of those policies, hence increasing operational constraints and opening a window of opportunity to improve how security policies are defined, managed and enforced from a software-defined security perspective.

One key paradigm shift to improve the status quo is to pursue an approach by which network security policies are less dependent on the network – and bound instead to the workload.

That said, it is, of course, important to recognize that not all policies and workloads are "created equal," and so while policies may include such activities as monitoring/DPI/IDS, etc., it may indeed not be feasible to instantiate every one of those capabilities directly at every workload (e.g., as a result of licensing costs or performance issues); it is thus further recognized that this use case should allow for service chaining to bind a workload and its policies across an arbitrary topology, which might include forwarding packets between physical nodes to receive the service.

At the same time, it is important to keep in mind that one of the key, necessary characteristics of a properly formed requirement is that it be achievable given current technology limits.

Requirements:

The requirements for Use Case #8 have been combined with those of Use Case #9 and are listed below.

Use Case #9

Write security policy in one place and deploy in multiple places, where workload policy would then be enforced

Problem Statement

As the same time that the SDI market is developing, so too is the hybrid cloud market. The first trend leverages commoditized hardware across compute, storage and networking environments, and de-couples features and services into software that may run on a range of platforms – be they controllers, x86 compute platforms and/or within or on top of commoditized hardware. The second trend offers a workload placement or IT service delivery option to host applications in private, public or hybrid cloud facilities. These two trends are creating new workload deployment options and newfound corporate agility. However, from a security perspective, the ability to define and write security policy for a workload that spans an application and its data in one place and deploy it in multiple places, where the workload policy would then be enforced, represents a major feature gap in the marketplace.

There are many new orchestration tools available for SDI, such as VMware's vCloud, Kubernetes, Mesos, Scalar, Cirba, Ansible, OpenStack, Docker's Docker Machine and others. However, while some orchestration tools may include security features such as micro-segmentation orchestration, there is no broad policy engine framework that pushes policy once centrally defined through orchestrators to elements for distributed enforcement. In addition, policy engines are typically vendor-specific with most engineered for a hardware-defined infrastructure deployment model.

This use case calls for a policy engine that communicates with a wide range of orchestrators through open and industry-standard interfaces so that centrally defined policies have their enforcement distributed to workloads independent of physical locale and dependency map. The workload's dependency map is more often than not a collection of services, which is instantiated within physical and virtual form factors such as firewalls, load balancers, intrusion protection systems, etc.

In short, this use case addressed how best to deploy security effectively in a way that transcends physical and virtual infrastructure in private, public or hybrid cloud deployment models. Stated another way, this use case is focused upon "choreographing" a set of workload policies that are defined centrally, with enforcement distributed local to that workload, and which moves with the workload as it moves independently from the form factor of its dependency map.

Requirements (Shared by Use Cases #8 and #9):

R-10: Network reachability for workloads must be provided to ensure that network segmentation security policies are enforced at the policy enforcement mechanism that's nearest to the workload.

R-20: Service chaining must be supported to ensure that, as per policy (for a workload), traffic may be steered to specialized security services (deep packet inspection based policies, etc.).

R-30: Policy enforcement mechanism should be common across all different deployment infrastructure models (physical or virtual).

R-40: Policy enforcement mechanism should be integrated with the infrastructure in which the workload resides.

R-50: Policy enforcement mechanism should be at the nearest location to the workload, with the ability to understand and represent the context in which a workload resides (i.e., virtual machine tied to a hypervisor, container tied to an OS).

R-60: Policy enforcement mechanism should be one from which security policies can be portable.

R-70: Policy enforcement mechanism must be programmable, and it must leverage an open interface for management and control.

R-80: Policy enforcement mechanism must scale to enable filtering (ACLs) stateful policies as well as path isolation (whether or not via overlay) for workload-bound traffic, as per policy defined in a single, authorized control plane.

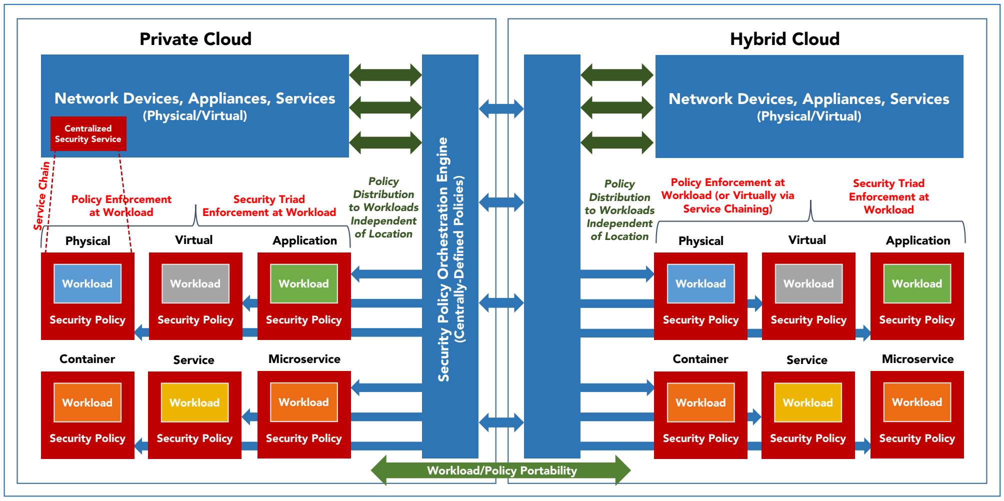
R-90: Policy enforcement mechanism must be locked down so that remote access to it can only come from a single, authorized control plane.

R-100: Policy enforcement mechanism must be protected against local privilege escalation so that it can only be changed from a single and remote control plane (i.e., local admin of a system cannot change local policy configuration, even if 'root' to local system).

R-110: Policy enforcement mechanism must be implemented so that its local configuration and policies constructs are protected from tampering and compromise (confidentiality and integrity at the local enforcement).

R-120: Policy enforcement mechanism must enable backend integration with policy management/orchestration engine via open and secure APIs (authentication, coarse and fine grain authorization, data-in-transit protection).

R-130: Key management that enables secure communication between policy enforcement mechanism and security policy control plane should follow key management guidelines, such as those offered by SANS Top 20 Critical Controls, NIST in the U.S. etc., and various European country-specific specifications embedded in ISO 27002:2013, officially entitled "Information technology — Security techniques — Information security management systems — Requirements."


Private Cloud
Network Devices, Appliances, Services
(Physical/Virtual)
Centralized Security Service
Service Chain
Policy Enforcement at Workload
Security Triad Enforcement at Workload
Policy Distribution to Workloads Independent of Location
Physical
Virtual
Application
Workload
Security Policy
Container
Service
Microservice
Security Policy Orchestration Engine (Centrally-Defined Policies)
Hybrid Cloud
Network Devices, Appliances, Services
(Physical/Virtual)
Policy Distribution to Workloads Independent of Location
Policy Enforcement at Workload (or Virtually via Service Chaining)
Security Triad Enforcement at Workload
Workload/Policy Portability

Use Case #4

Must be able to measure the ability of network workloads to ensure the confidentiality, integrity and availability (the Security Triad) of the services they are delivering

Problem Statement

A network workload represents a set of related and coordinated (managed) data transformation processes addressable somewhere on a network. The workload may consist of any class of monolithic or distributed processing, including network node functions (e.g., switching, routing, load balancing, accelerators, security). Collectively, these processes deliver some form of service to a consumer through the network. As these processes transform and move data about the network, they consume resources: compute cycles (data transformations), storage (persistent and non-persistent data at rest) and communications bandwidth (data in transit).

During operation, workload processes are hosted or contained within one or more physical or virtual systems (network enabled objects). The processes may share these systems and their resources with other unrelated and/or untrusted processes (e.g., applications, OS, VMM). These systems manage process execution and consumption of resources, as well as provide needed protections (e.g., isolation, redundancy, fault recovery, fail-secure). At the most basic level, protecting a workload means protecting the process instructions (code) and data that the workload is operating on by ensuring that the confidentiality, integrity and availability (the aforementioned CIA – the Security Triad) of these objects is preserved. Within information security communities, these protections are referred to as security controls, or controls.

The ability to protect the workload from threats (both operating hazards and malicious actions) within an operational environment speaks to the resiliency of the system. Resiliency is a desirable systemic quality that describes a system's ability to withstand adverse conditions or failure modes induced by stress, wearout, latent defects, misuse or intentional attack. The resiliency of a system contributes to the notion of trust. Trust is the confidence that the system can and will ensure that the workload process(es) it's managing execute as intended without compromise of the CIA elements of security.

To establish trust, it is necessary to measure the effectiveness of each protection contributing to the resiliency of a system (or a system-of-systems such as a network). It is also necessary to prove that each protection is operating as intended within the system (operational performance). Such evidence allows stakeholders to gauge the trustworthiness of a system in supporting the business. In order to attest to a system's resiliency, a testing or benchmarking capability is required to measure, score and substantiate the applied protections used to satisfy the workload's stated protection needs (security profile). The requirements for this capability have been articulated below.

Requirements:

R-140: The set of measures selected as the basis for assessing system protection effectiveness and operational performance must be rooted and derived from a source deemed to be both trusted and acceptable by the named authority having oversight of the operational domain (e.g., the business risk owner of the network).

R-150: A system operating on the network must be able to guarantee the operational performance of each protection it claims to provide for maintaining the confidentiality, integrity and availability of a workload object (process or data). If the operational performance of a claimed protection cannot be guaranteed, then the system must be able to report/alert such a condition upon detection to a designated set of receivers (e.g., monitoring system(s)). The confidentiality and integrity of the report/alert must be guaranteed. The measure of each protection's effectiveness and operational performance must be in accordance with the set of measures described in R-140 above.

R-160: An authorized subject (e.g., administrator, external system) must be able to issue a protection profile request query to a system operating on the network. The system must respond to the query by reporting each protection it claims to provide for maintaining the confidentiality, integrity and availability of a workload object (process or data). The response content must enumerate each claimed protection and provide sufficient measurement detail to describe each protection's features, capabilities, implementation methods and certifications (if any). The confidentiality and integrity of the response delivered to the authorized subject must be guaranteed.

R-170: An authorized subject (e.g., administrator, external system) must be able to issue a protection state request query to a system operating on the network. The system must respond by reporting the current operational performance state (e.g., nominal/available, degraded/failing, disabled/unavailable, unknown/indeterminate) of each protection it claims to provide for maintaining the confidentiality, integrity and availability of a workload object (process or data). For a claimed protection reported as functional (i.e., operating in a functioning state), the report must provide proof of protection effectiveness in accordance with the set of measures described in R-140 above. The confidentiality and integrity of the response delivered to the authorized subject must be guaranteed.

R-180: An administratively independent assessor system under the control of an authorized subject (e.g., security assessor) must have the ability to confirm the presence, effectiveness and operational performance of each protection claimed by the target system (the system on the network under assessment) for maintaining the confidentiality, integrity and availability of a workload object (process or data). The measure of each protection's effectiveness and operational performance must be in accordance with the set of measures described in R-140 above. The confidentiality and integrity of the assessment results provided to the authorized subject must be guaranteed.

Software-Defined Security Systems – WG’s Top 10 Requirements

The S-DSS working group members voted on which of the above requirements constituted the “Top 10” from the above three use cases. The following are the top 10 S-DSS requirements:

Requirement	Description
1	Policy enforcement mechanism must be common across physical or virtual infrastructure.
2	Service chaining must be supported.
3	Policy enforcement mechanism should be at the nearest location to the workload.
4	Policy enforcement mechanism should be integrated with the infrastructure the workload resides.
5	An authorized subject (e.g., administrator, external system) must be able to issue protection profile and protection state request queries to a system operating on the network.
6	Policy enforcement mechanism must be programmable via open interface for management/control.
7	Policy enforcement mechanism must scale to enable filtering (ACLs) stateful policies and path isolation (overlay/underlay) for workload-bound traffic.
8	Policy enforcement mechanism must enable backend integration with policy management/orchestration engine via open and secure APIs.
9	A system operating on the network must be able to guarantee the operational performance of each protection it claims to provide.
10	Policy enforcement mechanism should be one from which security policies can be portable.

Software-Defined Security Systems – Conclusion

As noted above, the task set forth by the ONUG community for the Software-Defined Security Systems working group was to provide a set of functional requirements to guide industry vendors in their development of open architecture framework security-related products and solutions. As the working group proceeded to explore the plethora of considerations needed to provide meaningful direction in this area, it became clear that the creation of a robust software-defined infrastructure would also require the development of a new security framework – not a framework that discards “traditional” security concerns or conceptualizations (such as the Security Triad, to pick one key example) – but rather one that builds upon, extends and fundamentally enhances those legacy frameworks.

As the history of earlier forays into open, virtualized architectural frameworks (such as server virtualization) demonstrates, there are many ways of potentially addressing the various problems and issues presented by such a complex challenge as the working group has now before it, and one can quickly end up deep in the quagmire of trying to sort out fundamental needs, priorities and possibilities. Hence the working group’s conviction, stated early on in this document, that it was critical to separate “problem space” from “solution space,” and to recognize explicitly that the proper purview and focus needed to be on exploring and defining the problem space via the development of specific use cases and attendant requirements. Hence too the working group’s explicit goal, shared by ONUG in general, that the solution space was properly within the purview of solutions vendors. The working group’s job here is accordingly to enable the follow-on development work needed from that community through the clear and sensible articulation of requirements.

As evidenced above, the working group proceeded with this task by first declaring a set of broadly-accepted industry standards and acknowledging the work of standards-setting bodies. This allowed it to establish “core requirements,” along with a common language and taxonomy upon which the working group could build an initial, open architecture-specific group of use cases and an SDI-specific set of security requirements.

From both the original superset of nine use cases to the three cases agreed upon as the areas for immediate focus, it is clear that the working group came to see the fundamental unit of concern as the integrity of the network or infrastructure “workload” itself. Understanding this led to the recognition that it is clearly incumbent upon the SDI security framework to ensure that security policies are fully bound to the workload itself, independent of their particular technology genesis (VM, containers, et al.), that those workloads will be protected both in transit and at rest, that policies should be both centrally created and distributed throughout and across the network, that policy enforcement efficacy is also maintained in a “close to the workload but distributed” manner, and that the confidentiality, integrity and availability of all workload services can be measured and verified. Workload security compliance can thus be ascertained at any juncture. Furthermore, solution vendors are expected to understand prerequisites that are indeed necessary to enable the top 10 requirements, which includes knowing the risk appetite of their client base as that may fundamentally influence what's acceptable for them as roots-of-trust for the solution.

The three use cases thus selected led to a set of requirements that focus on ensuring the ability to instantiate and enforce security policy. This is quickly ascertained by perusing the requirements presented above and numbered R-10 through R-180. At the most basic level, for instance, security policy enforcement must be possible across multiple workload deployment scenarios, whether physical or virtual, and whether carried out in on-premise data centers or in cloud- or hybrid-cloud-hosted environments. Beyond that, policy enforcement mechanisms should not only be able to understand and address the specific context in which a workload resides, but they must be programmable and leverage an open interface for management and control. Further, workload confidentiality, integrity and availability must be discoverable, reportable, verifiable – in short, the Security Triad of a workload must be guaranteed within the boundaries of the requirements laid out by the S-DSS working group.

ONUG Software-Defined Security Systems Working Group Members

Christopher Hoff, Co-Chair	Bank of America	Mike Elmore, Co-Chair	Cigna.	Adam Forch, Co-Chair	FedEx.
Mike Clark, Author	E/L/I Exceptional Leaders International				

Sanjai Narain	Applied Communication Sciences	Snehal Patel	Gap Inc.	Mahesh Desai	TESLA
Terry McGibbon	BARCLAYS	Thom Schoeffling	GENERAL DYNAMICS Information Technology	Brighten Godfrey	ILLINOIS
Chris Christou	Booz | Allen | Hamilton	Ted Turner	intuit.	Phil Hattwick	WELLINGTON MANAGEMENT
Michael Lundberg	Booz | Allen | Hamilton	Nick Lippis	Open Networking USER GROUP	John Burns	WF WELLS FARGO
Dave Larkin	Cigna.	Steve Lafrentz	Principal Financial Group	Fred Lima	VISA
James Noble	GAIKAI	David Lucey	salesforce		

Ken Cheng	BROCADE	Monier Jalal	Cybera.	David Anderson	vARMOUR
Goran Saradzic	CISCO	Anita Pandey	Cybera.	Colin Ross	vARMOUR
Gopinath Durairaj	CITRIX	Je Chiu	hp	Marc Woolward	vARMOUR
Marco Murgia	CITRIX	Sam Mita	NEC	Anusha Vaidyanathan	VERSA NETWORKS
Cliff Duffey	Cybera.				

SOFTWARE-DEFINED
SECURITY SERVICES
WORKING GROUP
2016
Open Networking
USER GROUP