SUN'S PATTERN-BASED DESIGN FRAMEWORK: THE SERVICE DELIVERY NETWORK

Mikael Lofstrand, Sun Client Solutions
Jason Carolan, Sun Client Solutions

Sun BluePrints™ OnLine — September 2005

SUN BLUEPRINTS
[illegible]
APPROVED

Part No 819-4148-10
Revision 1.0, 9/7/05
Edition: September 2005

TABLE OF CONTENTS

The Service Delivery Network

This Sun BluePrints™ Online article provides a high-level introduction to Sun's Service Delivery Network (SDN), Sun's approach for designing dynamic, service optimized network architectures.

This article consists of the following sections:

Network Infrastructure Challenges

Today's IT organizations grapple with numerous challenges when building, retooling, and managing their network infrastructures. Many struggle with inflexible legacy networks in the face of accelerating technological growth and increasing business and customer demands. Integrated network solutions are complex to design, support, and manage. IT organizations must deliver reliable results within severe resource constraints: tight budgets, short time lines, and specialized expertise. Network infrastructures must be quickly adapted to support new technologies (such as Grid Services, Web services, Internet applications, and application integration) that enable rapidly evolving business processes. Services and providers continue to converge and provide many different types of services, such as wireless, broadband media, and voice services. Mounting security and compliance pressures are forcing organizations to re-evaluate their traditional network architectures in order to enforce stronger access control and auditing policies without inhibiting the levels of flexibility and agility that they have come to require. Data centers are now distributed across the globe, as are the devices and clients that they support.

Data centers of the past usually hosted long-lived services: a single server ran the same application for its lifecycle of operation. Few changes were made, networks were static, often flat to ease administration, and network security was confined to just a few points within the implementation.

Designing networks and solutions for millions of users, transactions, and diverse content types requires an approach that differs from that of standard infrastructure designs. These networks must support convergent services (end-user services, supporting services, and infrastructure services), with high levels of reliability and performance, while maintaining manageability and, above all, security.

Network architectures and applications must be designed to support ubiquitous access. Flexibility is required to support new or evolving business and technical requirements as well as to more able to respond to business conditions and competitive pressures. Always-on access is has become a requirement whether services are accessed over the Internet, intranet or extranet.

While the network has always provided the foundation for the applications and servers within the data center, it is now a critical part of service delivery. As service-oriented architectures and web services become the new development norm, the network must become service-oriented as well.

To address these needs, Sun's Service Delivery Network (SDN) methodology provides an effective framework for designing, deploying, and managing network infrastructure solutions in support of key business goals and services.

What is Sun's Service Delivery Network?

The SDN is the approach that Sun uses to design service optimized network architectures for customer and in-house implementations. This approach consists of basic network building blocks, common network design patterns, integrated network components, and industry best practices that together are carefully blended in response to a customer's business and technical goals. The SDN provides a set of network connectivity, routing, load balancing, and advanced security mechanisms that, when applied in combination, result in flexible network infrastructure designs that provide high performance, scalability, availability, security, flexibility, and manageability.

The primary goal of the SDN is simple:

Service delivery at any time, from anywhere, to any device.

A service optimized network architecture focuses on the services provided over the network to the end user, rather than the enabling technologies or their related components. By virtualizing resources and understanding the core services offered directly to end users, as well as the other data center services that support these end user services, organizations can take advantage of a true service-driven architecture.

Concepts of Pattern-Based Solution Design

This section defines key concepts that provide a foundation for using the SDN methodology to design service-optimized network architectures. The SDN methodology allows for a common approach to designing network architectures, and provide a common set of tools that ensure proper architectural design decisions and trade-offs.

Services

In the context of service-oriented architecture design, a *service* is an implementation of well-defined business functionality. Once implemented, services can be consumed by clients via different applications or business processes. Services are software components with well-defined interfaces that are implementation-independent. An important aspect of a service-oriented architecture is the separation of the service interface (the what) from its implementation (the how). Such services are consumed by clients who are not concerned with how these services will execute their requests.

Because the SDN methodology defines an approach to network architecture and the “glue” within the data center data communication layer, SDN focuses on services as they appear within that context. These “network” services use standards, protocols, and other descriptive characteristics to propose how these services use the network. For example, a network service of HTTP on Port 80 may be used to describe the protocol a web service might use to deliver content.

Service Characteristics

Services have the following characteristics:

- self-contained (perform predetermined tasks)
- loosely coupled (for independence)
- can be dynamically discovered
- have well defined interfaces and functions (for modularity)
- can be aggregated to build composite services

Types of Services Delivered Over the Network

For SDN, services can be classified by the following types:

Service	Description
End-User Services (Presentation Tier)	Provided directly to end-users. Examples include an Internet web presence, e-commerce facilities, or even e-mail, news and collaboration services.
Supporting Services (Integration and Resource Tier)	Directly support end-user services but are typically not visible to end-users. Examples include an application server providing dynamic content for a Web site (such as shopping catalogs or a shopping cart), (such as JMS messaging or application container). However, the end-user service is the service access point, and users do not directly initiate connections into a supporting service.
Infrastructure Services (Management Tier)	Ease internal operation and support. Often infrastructure services include system and network management services. Examples include internal DHCP, DNS, security or directory services that are used to support the infrastructure but should not directly interact with end-user services.

Business, Application, and Network Services

The SDN methodology organizes services into the following categories:

- **Business Service**, for example, a book store and catalog system.
- **Network Service**, a component of the business service, providing a specific service viewed from the network, such as a HTTP web service over Port 80.
- **Application Service**, a specific application component, making up a business service. In SDN, these are realized by service instances, each running an instance of an application.

Building Blocks, Patterns, and Microarchitectures

The following figure shows the core design components used in the SDN methodology.

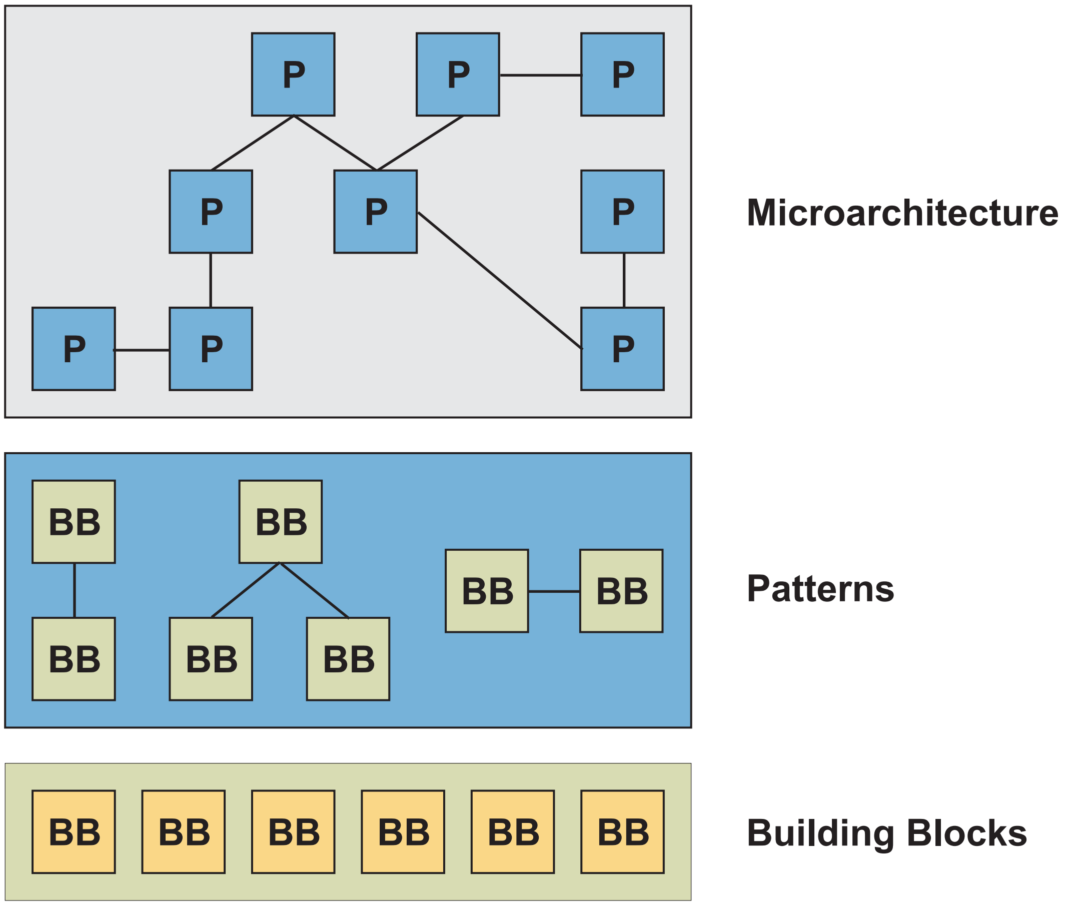

Figure 1. Core Design Components in the SDN Methodology

Building Blocks

Building blocks represent the smallest, irreducible component within the pattern or architecture. These building blocks are assembled to provide systemic functionality. In the SDN methodology, the building blocks include the modules, service instances, and logical execution containers. As a general rule, building blocks should provide clear features, clean interfaces, and have limited interdependencies to other building blocks. For more information, see See “SDN Building Blocks” on page 9.

Patterns

In this article, a *pattern* is defined as a generalized, reusable design solution to a recurrent network architecture design problem. Patterns derive from the experience of solving the same or similar design problem over and over. Once defined, the generalized design solution can be used heuristically, applied to customers with similar network design challenges, for the purpose of specifying a “best fit” solution that is shaped by the prevailing and relevant forces that are present in each implementation project. For more information, see See “Common SDN Patterns” on page 18.

Patterns by their nature are descriptive (not prescriptive), defined at a sufficiently high level to avoid working through and bogging down over specific details. While patterns are adequate for high-level discussions and direction, the devil is in the details. The concept of patterns provides an excellent basis for documenting and describing the various attributes that make up a solution to a problem.

Microarchitectures

In this article, a *microarchitecture* represents is an abstract collection of design patterns (which are made up of building blocks) that work together for a common purpose (such as network design). An *instance* of a microarchitecture is the network design that results from applying the SDN methodology to a specific organization’s stated business and technical requirements. These design patterns and building blocks are uniquely configured to address the organization's needs.

The following figure shows the kinds of components that make up a network microarchitecture.

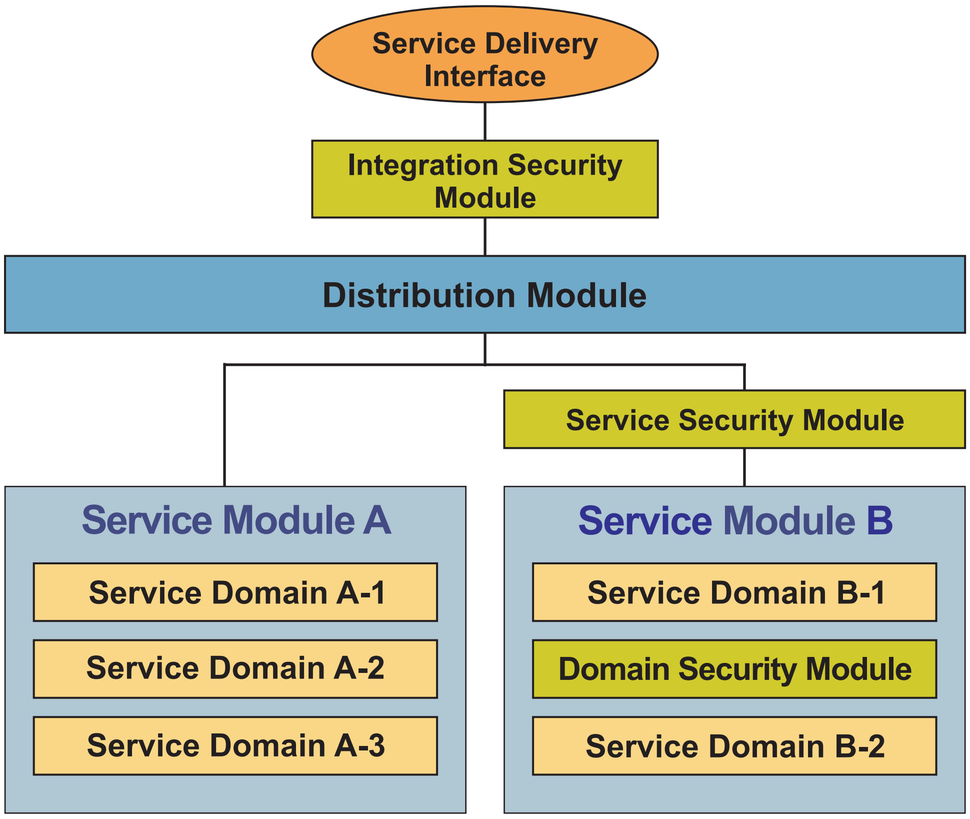

Figure 2. Components of a Service-Optimized Network Design

These components are described later in this article.

Forces

The SDN methodology tailors network infrastructure designs to meet the stated business and technical requirements of an organization. It is these requirements that clarify the most important factors to consider in a given network design. Requirements are translated into *forces* that will influence how the entire design is realized.

At an abstract level, the service-optimized pattern design framework uses the term *forces* to describe:

- the relevant *goals* to achieve (such as high security and availability), and
- the pertinent *constraints* to work with (such as cost or standards compliance)

when designing a service optimized architecture. The combination of goals and constraints reveal the intricacies of a given problem and how certain forces interact with or conflict with each other. Analyzing the inherent forces for a given solution exposes the concordance and dissonance among these forces, yielding an understanding of the kinds of trade-offs that must be considered in the design. Forces influence the final design, pushing and pulling it towards different solutions. Well-defined architecture and design patterns should fully encapsulate all of the forces that have impact on them and resolve them to achieve the “best fit” solution possible.

For example, suppose an organization wants to deploy a network infrastructure that provides high availability (a design goal) within a defined budget (a design constraint). A high availability architecture incurs higher costs because it requires deploying redundant components to eliminate single points of failure in the design, as well as deploying the software to handle automatic failover in the event of a component failure. In order to keep costs in line with the budget and achieve full component redundancy, an organization might choose to deploy lower cost components with lower throughput or less capacity. As a result, the trade-off for this high availability solution is performance and capacity.

The following system quality goals should be considered when designing a service optimized network architecture. These goals can be translated into service level objectives, supported by a service level agreement (SLA) or “contract” on how the entire system should perform. SLAs are measurable expectations and the desired results of pertinent system characteristics.

System Quality	Description
Security	Ability of a system to protect itself and its resources from unauthorized use – including access, disclosure, modification, and destruction.
Availability	Ability of a service to remain operational, with minimal interruption or degradation, based on a service level. In the event of a failure of one or more of its parts, a service should remain operational. The level of tolerance for faults will vary, so one or more parts has a bound that is influenced by other business conditions.
Reliability	Ability of a system component to repeatedly produce the same results in accordance with its specifications.
Performance	Refers to a combined measurement of response time and throughput that meets established, service-level agreements. This may include network performance or latency between two network end points or, more importantly, how it effects a client application and end user.
Scalability	Ability of an architecture to accommodate greater intermittent or sustained demands for service without reducing performance or availability. The network provides scalability by being able to handle additional traffic, services, and service instances (hosts, switches, and so on).
Manageability	Ease with which a system can be managed, monitored, and maintained.

Security and Compliance

Security is fundamentally about freedom from risk or danger. While risk cannot generally be completely eliminated, it can be managed to an acceptable level. Compliance refers to the ability of an organization to successfully demonstrate that it is operating in conformity with its policies and any regulations that may be imposed upon it.

Both security and compliance are critical areas of focus for organizations today wanting to protect their intellectual capital, physical and logical assets, as well as the confidentiality, integrity and availability of their customers', partners', and employees' information. While security can help to protect these assets, it is compliance that must be demonstrated (to a reasonable degree of assurance) in order to successfully navigate audits and engender trust.

The SDN methodology was designed with security and compliance in mind. SDN leverages a number of key security principles in the design of network architectures, including:

- self-preservation
- defense in depth
- compartmentalization
- least privilege

These security principles can be combined with other enterprise system design goals (such as performance, reliability, and availability) to construct architectures that meet customer requirements. Because the SDN methodology was developed from the start with security in mind, SDN-based architectures can support advanced security mechanisms while operating at extremely high speeds with low latency.

Self-Preservation

In the SDN methodology, each component used in a network architecture design is configured for self-preservation. Fundamentally, this means that each component is configured to reduce its attack surface and potential for exposure by:

- limiting access to management functions
- using secure protocols
- ensuring that access to services and functions is restricted to authorized parties

Self-preservation can be thought of as component level security (for example, hosts, switches, routers, firewalls, load balancers, and so on).

Defense in Depth

In the SDN methodology, components and assemblies used in an network architectures must support defense in depth through the use of multiple, complementary, and reinforcing security controls. The number and type of security controls used will vary based on the threat profile of the architecture, as well as organizational policies and preferences. The use of defense in depth helps to protect the environment against single points of security failure. Note that these security controls can be integrated into the components used to provide a service (such as hosts, switches, routers, load balancers, and so on) or they can be independent entities (such as firewalls, intrusion detection systems, and VPN concentrators).

Compartmentalization

The SDN service module pattern, described later in this article, implements the principle of compartmentalization by isolating application services into their own protected compartment (and network broadcast domain). Compartmentalization allows application services to be isolated from one another in order to limit the risk and potential for damage caused by a compromised service. Compartmentalization can be further extended to service instances within a service domain in order to prevent individual instances from communicating with one another.

Finally, compartmentalization can be applied to multiple instances of the same service in order to create effective security zones (by organizing each instance based on its threat model and security and communication requirements). This approach allows for multiple service domains (supporting the same service) to be isolated from one another based on an organization's security policy. As a general precaution, organizations should refrain from mixing multiple security zones within the same service domain.

Least Privilege

Service domains are configured to allow only specific service protocols (traffic types). In this regard, service domains implement the principle of least privilege in that they offer a service only to those who have been permitted to consume it (and only for permitted application services/protocols). By implementing the principle of least privilege, network architectures can be configured to provide access to services to only those who need it. Just as with compartmentalization, this design principle helps contain a security breach and limit the potential for damage by ensuring that service domains expose only well-defined services to specified groups of consumers – regardless of what may actually be running and listening on the systems supporting the application services.

Phases in the SDN Methodology

The process of applying the SDN methodology to design a service-optimized network architecture generally involves a series of phases, as shown in the following figure.

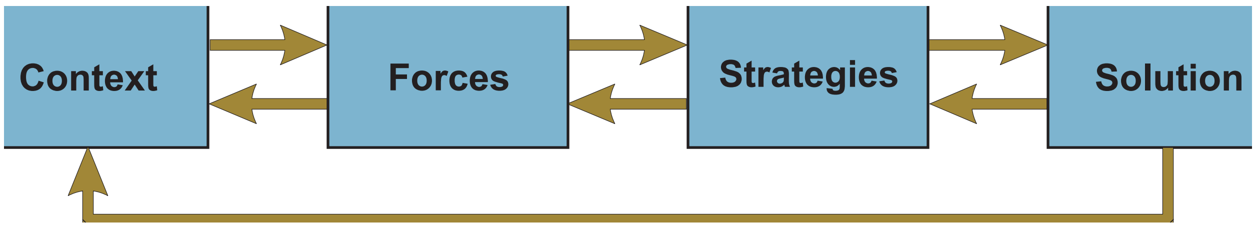

Figure 3. Phases for Applying SDN in Designing a Service-Optimized Network Architecture

Rather than a linear process, these phases are iterative and cumulative, progressing forward but also revisiting previous phrases and reevaluating decisions based on new understandings of how the network architecture should work. This process also follows a path of decomposition, in which a problem is broken down into its component parts, solutions are created for each component of the problem, and the design is constructed from the ground up using individual components and common design patterns.

The following table describes the activities in each phase:

Phase	Activities
Context	• Define Business Requirements • Define Functional Requirements • Define Service-Level Objectives
Forces	• Define Design Goals • Define Design Constraints (Current Environment, Interoperability) • Assess Trade-offs
Strategy	• Select Building Blocks • Select Patterns • Design the Logical Architecture
Solution	• Design the Physical / Implementation Architecture

SDN Building Blocks

The SDN methodology uses the following core building block components to construct service-optimized network design solutions.

- service instances
- service domains
- service and distribution modules
- optional specialty modules, such as security modules, caching, and so on
- network components, such as switches, routers, and load balancers

The following figure shows an overview of these building blocks.

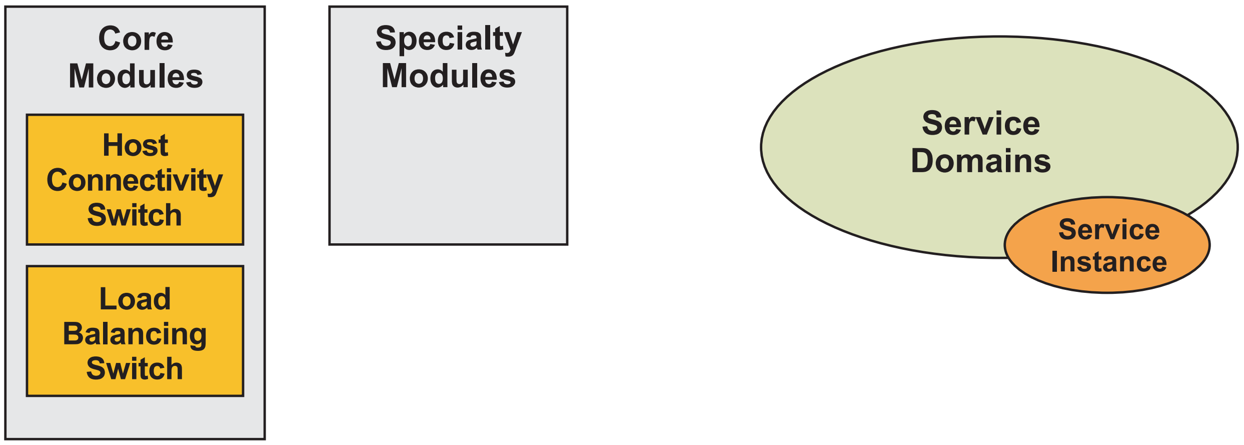

Figure 4. SDN Building Blocks

Service Instances

In SDN, a *service instance* is a running application that provides part of a service. Generally there are multiple service instances, each providing the same service component but on different physical hardware or infrastructure. The following figure shows an example of a service instance.

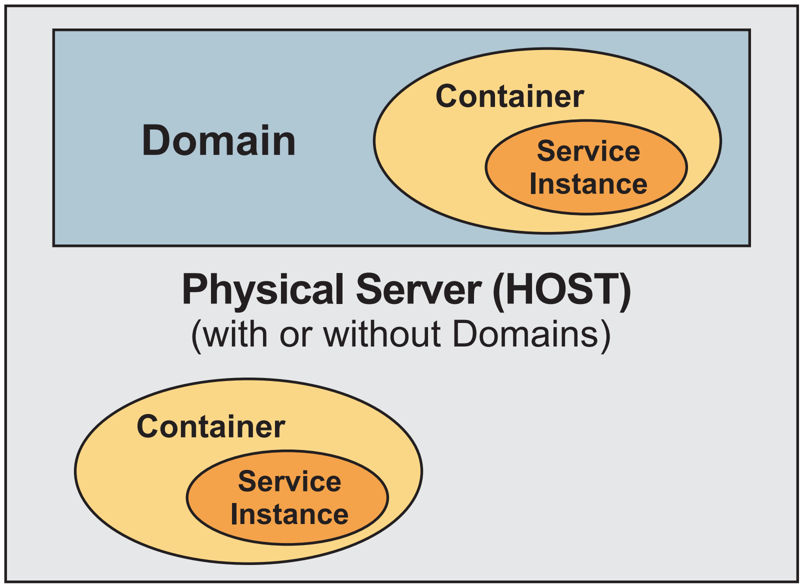

Figure 5. Service Instance

In the above figure, a service instance is shown instantiated within a Solaris Container. This Solaris Container requires various services, not just the network—it requires storage and execution resources, including memory, CPU, and so on, which are provided by the container (running on a physical server) and its operating system. This can also be implemented within domains, a Sun feature on larger enterprise systems.

Service Domains

Services reside in *service domains*, which are logical groupings of related services and the service instance s that provide these services. A service domain consists of several service instances (and ultimately containers running on physical servers) that provide the same (or very similar) logically grouped services. As defined in the SDN methodology, a service domain is a repeatable pattern that serves as the core building block in network architecture design.

Service domains organize similar services (such as security zones, functions, and so on) into logical groupings of services that can be managed individually. Separating services into logical domains provides an additional scalability mechanism and enables services to take advantage of other service domains (that use all of the same features and capabilities). Service domains provide network services for Solaris containers.

This approach offers significant security benefits based on the principle of compartmentalization and least privilege (applied to the offering of services based on policy). From a security perspective, elements in the domain can be (optionally) isolated from each other where such communication is not required (for example, with Web servers). This approach also mitigates the effects of undiscovered flaws, because the execution containers are not directly reachable except through the specific service they provide for the domain. Any other services running on the host are effectively out of reach. Finally, this approach of applying policies to interfaces (in, out, management) allows organizations to specify which services may talk with which other (thereby greatly simplifying IDS/FW policies).

Service domains provide:

- network services to OS instances (such as Solaris Containers) and their dependent application services.
- the physical access layer and communication path to each service instance. Each service domain provides network services for a single application. These applications (and their service domains) aggregate to provide a business service or services. Each service domain is a logical collection of similar service containers, each providing the same application service.
- physical network access to containers and underlying servers. Each server machine has at least two physical connections to the service domain host connectivity switch (HCS) to ensure that the service instance will still respond in the event that a failure occurs within a network switch or cable.
- security features, such as access control lists (ACLs) and physical port isolation

Service Modules

The basic SDN building block is the *service module*, which consists of service domains and associated hardware. A service module provides the services, the physical access, the routing, the distribution within the service module, the availability features, and the integration to other networks (for a single service module) and network services. The service module consists of physical network hardware, server connections for the service networks, and the software applications that make up the services to be delivered. The following figure shows an overview of a service module.

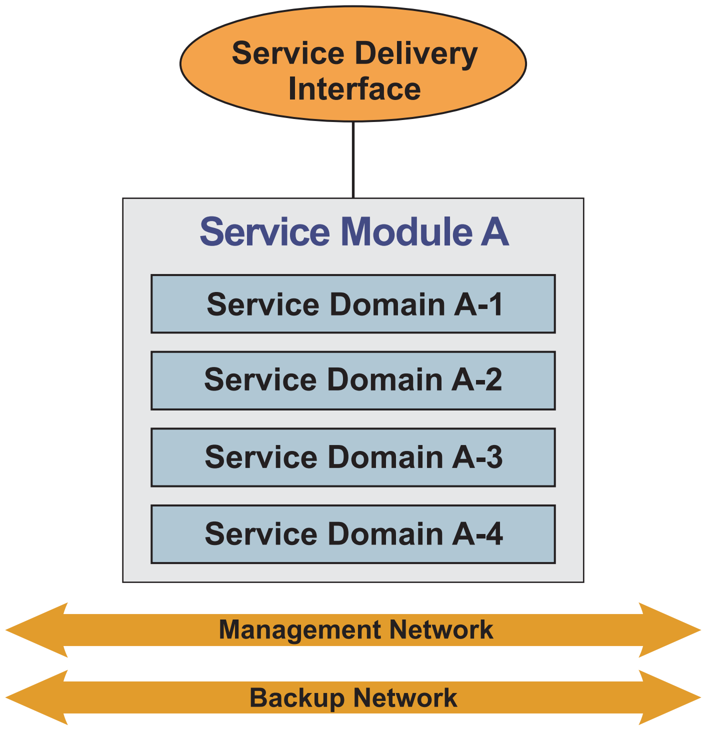

Figure 6. Service Module

A service module can exist as a standalone component, or it can be linked together and scaled to provide virtually unlimited services. A service module is a collection of several service domains that, in combination, provides certain services that can be grouped. A service module may consist of several service domains that provide a specific business service, a specific set of services for a particular layer (such as a presentation layer). It might be a mixture of services, depending on several factors (such as the size of the deployment). An example may include a service module for messaging, and another for identity services.

The *service delivery interface* is the primary service interface. It provides the integration point to upstream access providers, LAN or WAN access, and is the primary connection point for clients and end users. Typically, the service delivery interface connects to a router, pair of routers, or other distribution network that connects to the data center. This allows the network design to be self-contained behind this interface, and limits the amount of configuration necessary above this point for service routing. The requirements for integration are based on the physical connections of the load-balancing switch (LBS) and the basic requirements for network access, including IP routing.

Distribution Modules

Distribution modules allow the integration of several service modules and provide intelligent service routing, presentation services, and other features in a highly scalable, flexible design. For example, after capacity has been reached within a single service module, a distribution module and additional service modules can be added. Distribution modules also allow a better logical layout of services and reduce complexity while maintaining required service relationships.

Distribution modules are required when multiple service modules are used in a single environment. The following figure shows how a distribution module can be used to integrate multiple service modules.

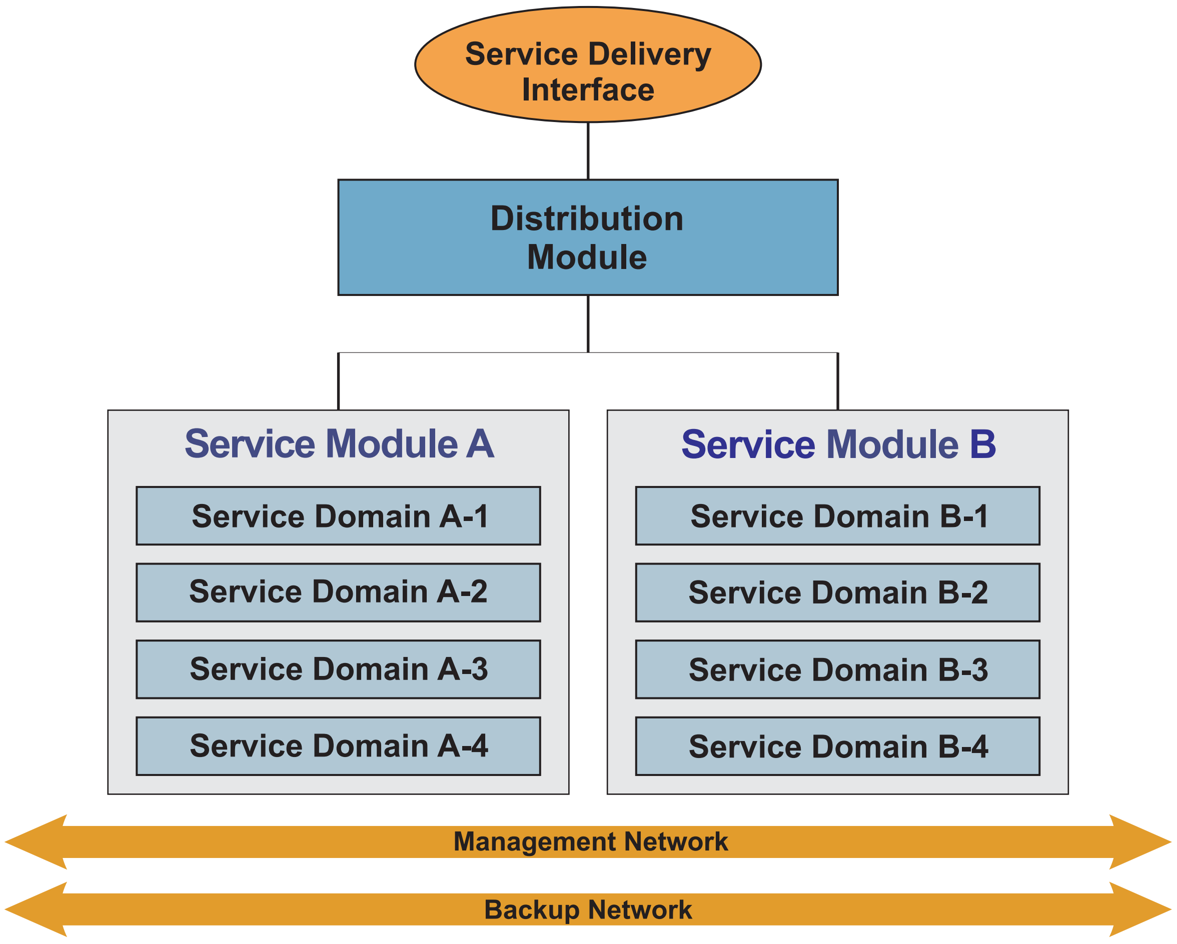

Figure 7. Distribution Module

Distribution modules consist of network components that are similar to the service modules. They enable several service modules to work together and aggregate services to the service delivery interface (where clients connect). Because distribution modules tie several service modules together, service modules may be isolated from one another, use different implementation hardware, or be managed by different organizations. The distribution module then provides central access to the delivery of those services and can be managed by a centralized network team. They may also manage common specialty modules, such as proxies or caches that provide these services to the entire network.

Specialty Modules

Specialty modules provide specific services in the SDN methodology. Depending on the deployment location within the network, a specialty module may provide services to the entire network architecture, to individual service modules, and to individual service domains. Specialty modules are optional or required based on business and technical requirements. Common specialty modules include:

- Security
  - Firewalls
  - Intrusion Detection Systems
  - VPN Gateways/SSL Gateways
- Performance
  - Caching
  - Application Acceleration
  - Network Shaping/QOS

Security Modules

SDN supports pluggable security modules that can be used to augment the native security of individual platforms and services, as well as security controls implemented in the network infrastructure itself. This allows for the use of security appliances and devices (such as firewalls, intrusion detection and prevention systems, and virtual private network concentrators) and other security controls to be integrated into the architecture. The decision to use these modules must be made based on an analysis of organizational policies and requirements, in conjunction with the threat profile for the network architecture.

This article defines three types of specialty security modules:

- integration security module
- service security module
- domain security module

SDN's in-depth security goal distributes many of the security-related functions across many of the systems within the environment. Among its many security features, the Solaris OS provides two security features that are particularly relevant to the SDN: the Solaris Security Toolkit, and support for IP-based filtering on each host.

SDN also recommends specific configurations for network hardware, including:

- Port-based VLANs or no VLANs, based on requirements
- ACLs/Filter applied to load balancing rules
- Specific rules for management access, and disabling as much management access as possible
- Specific configurations to reduce errors and complexity, increasing the ability to successfully manage and mitigate risk

Integration Security Modules

An *integration security module* sits between the network and the primary client or integration network, or between the service delivery interface. An integration security module protects the leading line of network hardware and switches, providing logging and access control. An integration security module is usually provided by a scalable firewall complex, such as a firewall cluster. For high-bandwidth networks, firewall appliances can provide highly-available and scalable firewall services. Depending on the implementation, the integration security module can also provide VPN or other services. Another option is the use of an intrusion detection system within the integration security module. In this case, the intrusion detection system provides specific features to detect anomalies and attacks.

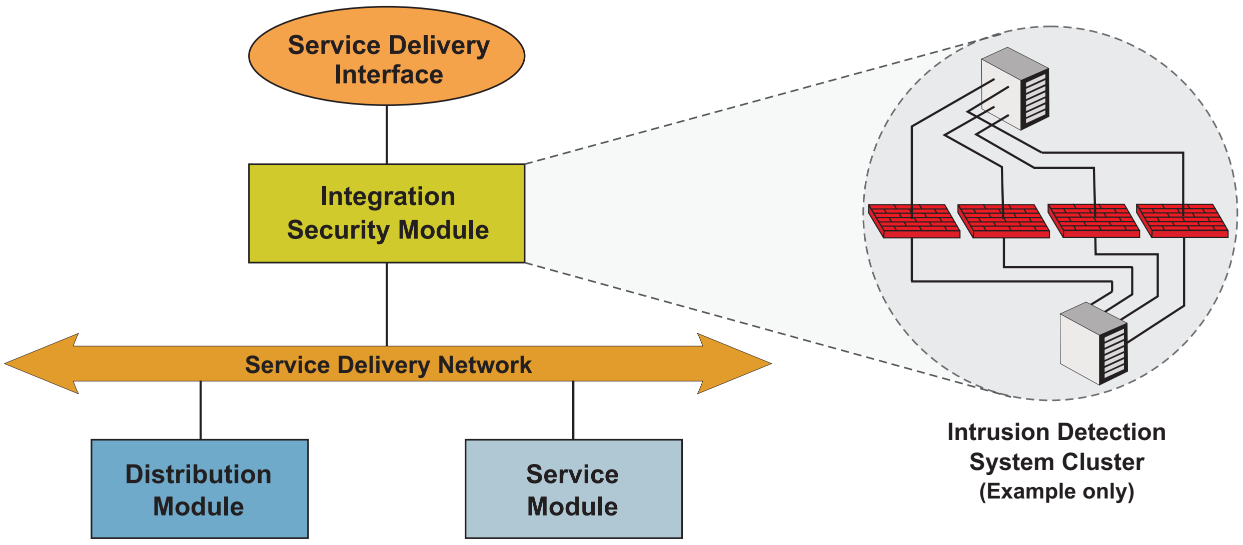
Service Delivery Interface
Integration Security Module
Service Delivery Network
Distribution Module
Service Module
Intrusion Detection System Cluster (Example only)

Figure 8. Integration Security Module

Service Security Modules

A *service security module* can be used to secure a sensitive service module and to allow for several different firewall technologies to be used in the network design. The following figure shows an example service security module:

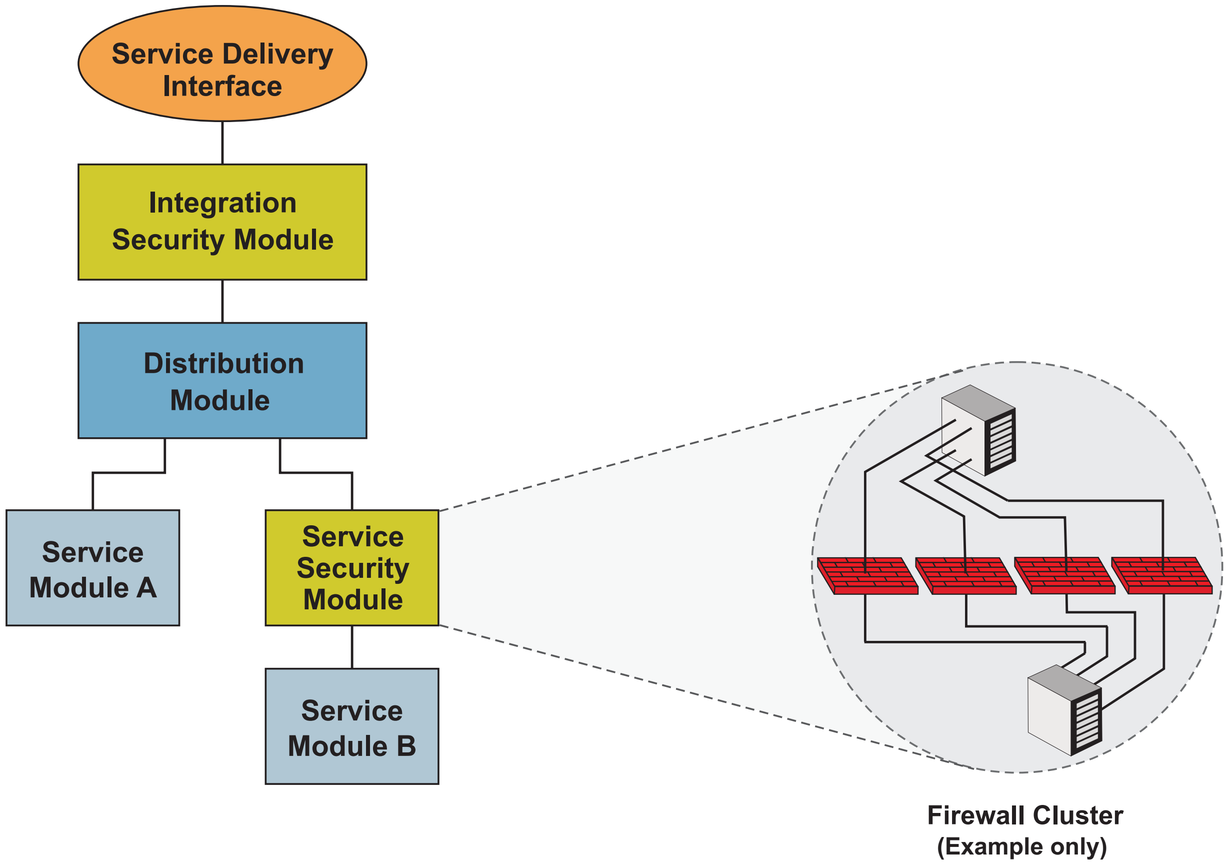

Figure 9. Service Security Module

Domain Security Modules

To protect a particular service that handles highly sensitive data, organizations can use a *domain security module*, which can be implemented between a load balancing switch and a host connecting switch in a given broadcast domain. A domain security module is focused on a single service or service domain and should include a transparent/bridging firewall that does not require a change in the destination or source network addresses between a load balancing switch and service instances.

Caching Integration Modules

Caching, another type of specialty module used in the SDN methodology, can be inserted between a distribution module and a service module, or at a service delivery interface, to provide caching services that improve the performance of commonly accessed static content. Caching modules are typically made up of cache software and hardware, such as Sun servers or other appliances that provide this type of capability.

Network Equipment

In the SDN methodology, each service module and distribution module is made up of common network equipment—load-balancing switches, host connection switches, and cabling—as shown in the following figure.

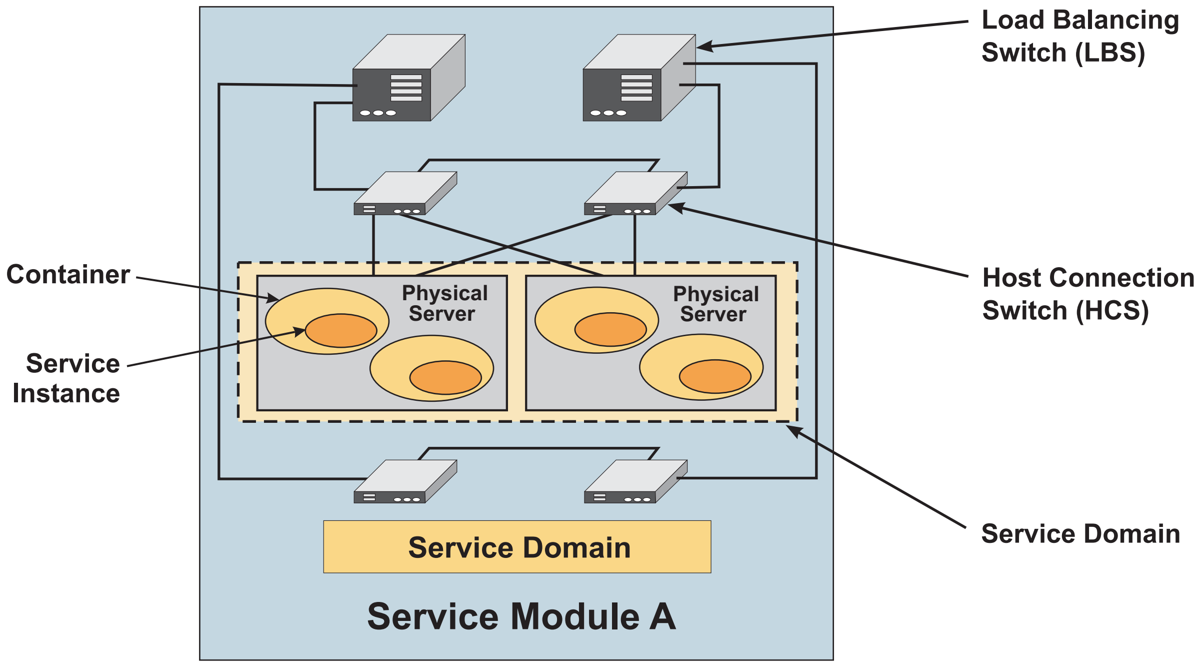

Figure 10. Network Equipment in the SDN

Note that a typical network design using the SDN methodology does not include a connection directly between the two load balancing switches, but rather utilizes the layer 2 network established by the host connection switches to pass data back and forth between the pair of load balancing switches. This approach allows the load balancing to maintain the link status of the layer 2 network to detect switch failures.

Load Balancing Switches

Load balancing switches provide high-performance routing, health checks, and availability features. Load balancing switches provide the interface for client access via the service domain interface, and aggregate services within service domains, together to form a service module. Load balancing switches are also a key component of domain modules, providing services that are similar to scalable service modules.
In routing-only service domains, load balancing switches play a role in ensuring that the correct traffic is allowed to the service domain, and that a secure VIP is provided for service access.

Load balancing switches are scalable, feature-rich, high-speed, high-performance switches that provide:

- Sufficient performance and physical capacity
- Security capabilities around DOS, ACLs/filtering, some content inspection
- Wire-speed (or as close as possible) permits and non-blocking
- Flexible management interface, with out-of-band and in-band capabilities
- Other features based on the requirements of the design

Generally, load balancing switches are implemented in pairs, and are the same model within a service module. For more information about Sun load balancing switches, see the Sun Secure Application Switch product page at http://www.sun.com/n2000.

Host Connection Switches

Host connection switches are basic, high-speed, second generation, non-blocking, layer 2 switches whose primary purpose is to attach hosts and servers to the network. When using a distribution module, Host connection switches can also be used to link service modules together.

Common SDN Patterns

This section briefly describes the most common patterns used in the SDN methodology to design network architectures. For each pattern, it briefly highlights which key forces differentiate that pattern from the others. These patterns are based on the building blocks described in “SDN Building Blocks” on page 9.

Single Service Module Pattern

The following figure shows the single service module pattern.

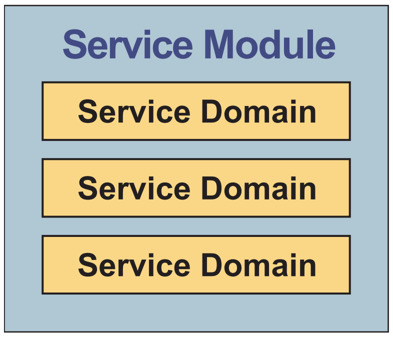

Figure 11. Single Service Module Pattern (With Multiple Service Domains)

This pattern provides the core hardware, software, and configuration components, consisting of network devices (such as load balancing switches and host connection switches) that are configured in a specific way to support service instances configured within service domains. Use this design pattern when a limited number of services are being offered with low growth expectations. Network security is built into the implementation using the Systemically Secure Architecture (SSA) approach. This core design pattern becomes the basis for other design patterns that are described later in this section. For more information about SSA, see the Sun document titled *Toward Systemically Secure Architectures* (by Glenn M. Brunette, Jr. and Christoph L. Schuba) at:

http://www.sun.com/security/docs/systemic-security-wp-1.pdf

Multi-Service Module Pattern

The multi-service module pattern includes two or more service modules and a distribution module. The following figure shows the multi-service module pattern.

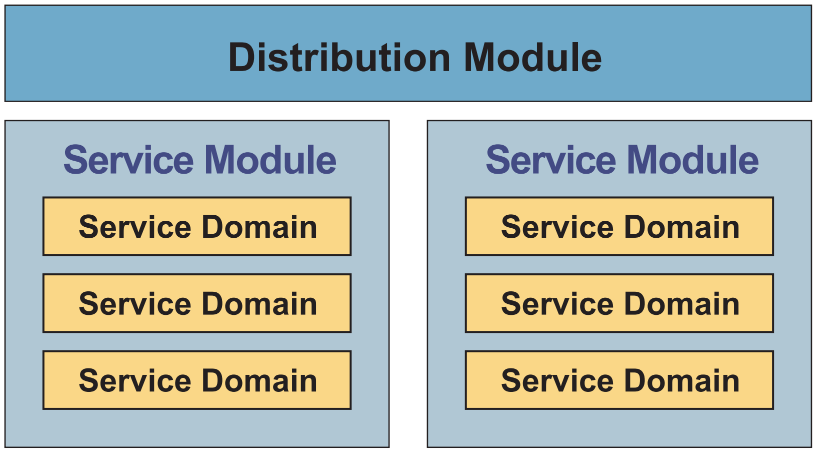

Figure 12. Multi-Service Module Pattern

Adding a distribution module allows an implementation to scale beyond a single service module. Use this design pattern when growth potential is high or when many services need to be organized together to reduce complexity or increase security (for example, when security requirements include front-end firewalls or intrusion detection devices). This core design pattern becomes the basis for other design patterns that are described later in this section.

Single Service Module With Integration Security Module Pattern

The following figure shows the single service module with integration security module pattern.

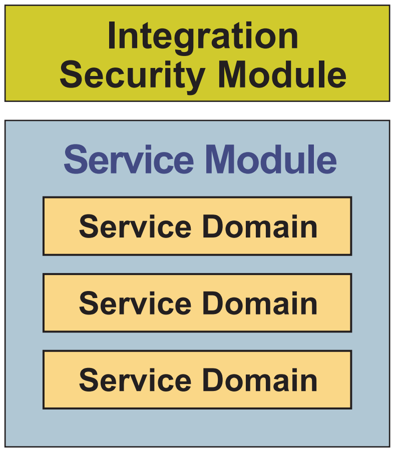

Figure 13. Single Service Module With Integration Security Module Pattern

This pattern extends the single service module pattern by adding and integration security module, which provides additional network-layer inspection before traffic enters the service module. An example of an integration security module is a firewall or IDS cluster. Use this pattern when front end firewalls or a intrusion detection system is required.

Single Service Module With Domain Security Module Pattern

The following figure shows the single service module with domain security module pattern.

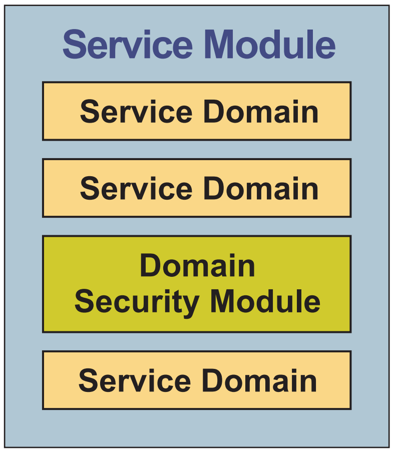

Figure 14. Single Service Module With Domain Security Module Pattern

This pattern extends the single service module pattern by adding a domain security module, which provides additional network-layer inspection before traffic enters one or more service domains. An example of an DSM is a firewall cluster. Use this pattern when only a single service domain requires network inspection (based on security requirements).

Single Service Module With Integration Security Module and Domain Security Module Pattern

The following figure shows the single service module with integration security module and domain security module pattern.

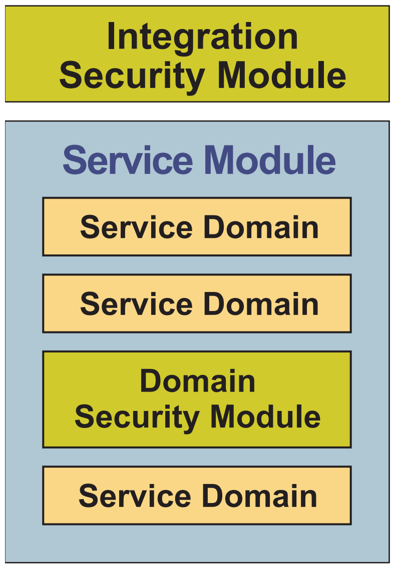

Figure 15. Single Service Module with Integration Security Module and Domain Security Module Pattern

This pattern extends the single service module pattern by adding a domain security module (which provides additional network-layer inspection before traffic enters one or more service domains), and an integration security module, which provides inspection for the entire service module. This pattern allows for multiple inspection points above and beyond the integrated security of the core SDN network devices. Use this pattern when both network inspection and an intrusion detection system are required for a single service domain.

Multi-Service Module With Integration Security Module
The following figure shows the multi-service module with integration security module pattern.

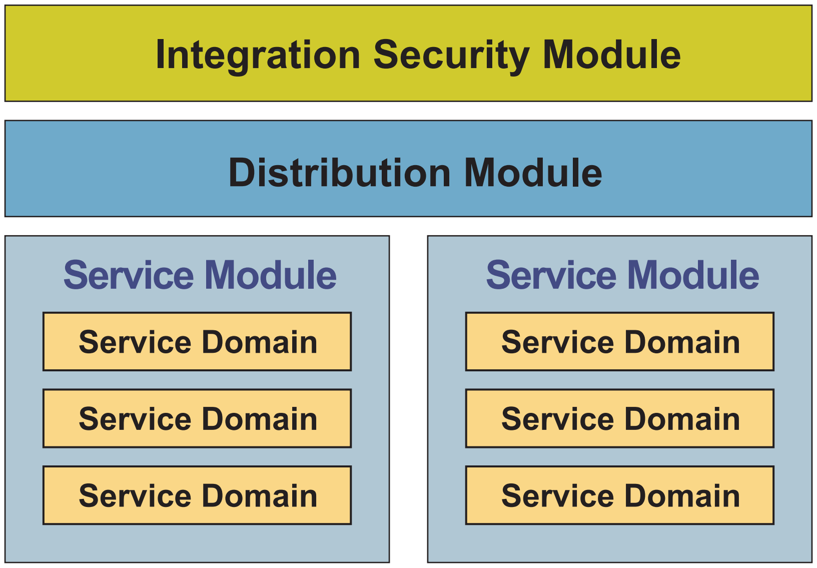

Figure 16. Multi-Service Module With Integration Security Module Pattern

This pattern extends the multi-service module pattern by adding an integration security module, which provides additional network-layer inspection before traffic enters the domain module. An example of an integration security module is a firewall or IDS cluster. Use this pattern when both network inspection and an intrusion detection system are required for a single service domain.

Multi-Service Module With Service Security Module
The following figure shows the multi-service module with service security module pattern.

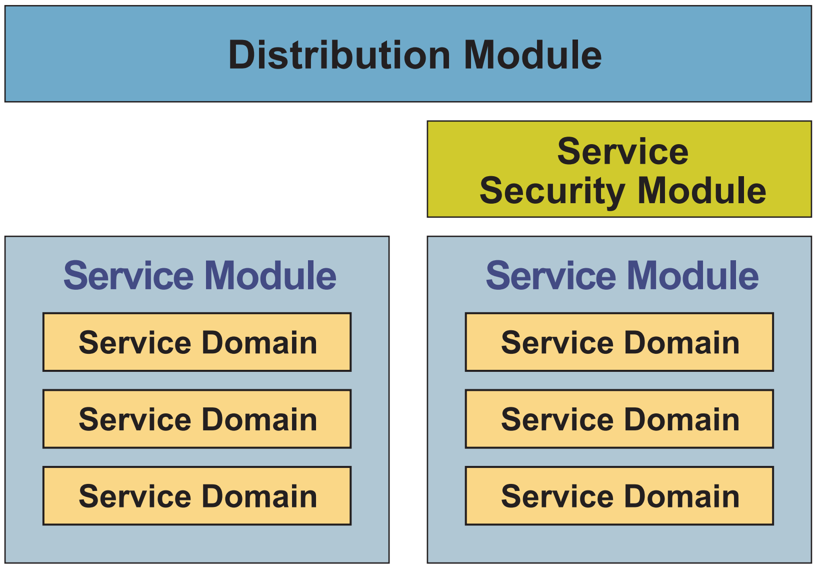

Figure 17. Multi-Service Module With Service Security Module Pattern

This pattern extends the multi-service module pattern by adding a system security module, which provides additional network-layer inspection before traffic enters a service module. An example of a security service module is a firewall or intrusion detection system cluster.

Use this pattern when only one service module requires an intrusion detection system or network inspection. Usually, similar services are grouped by their security zone values. This pattern might be used, for example, to allow high-performance services (such as the presentation tier) to be provided by the service module on the left side of the figure, with higher security risk connections permitted through the service module on the right.

Multi-Service Module With Integration Security Module and Service Security Module Pattern
The following figure shows the multi-service module with integration security module and service security module pattern.

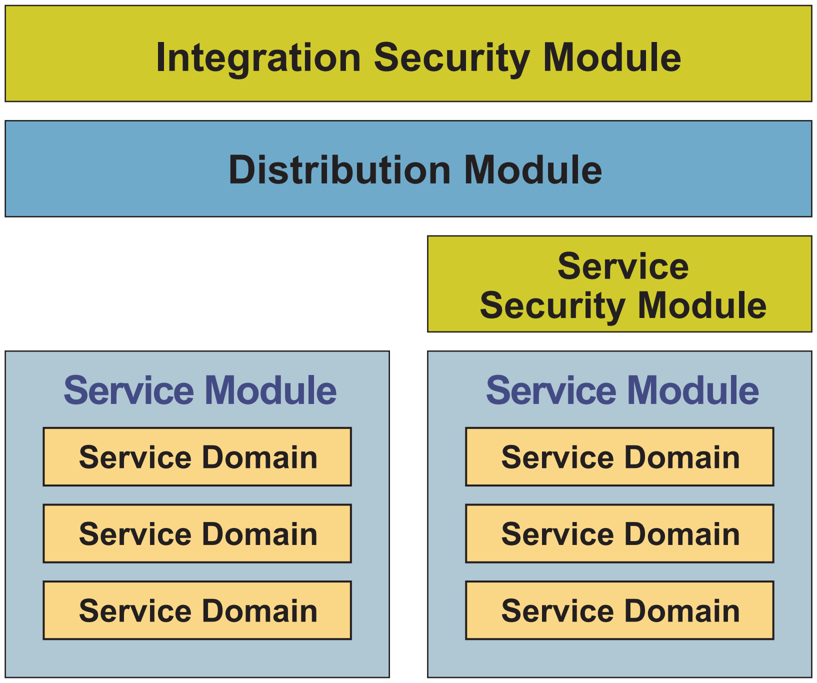

Figure 18. Multi-Service Module With Integration Security Module and Service Security Module Pattern

This pattern extends the multi-service module pattern by adding a system security module (which provides additional network-layer inspection before traffic enters a service module), and an integration security module (which provides additional network-layer inspection before traffic enters the distribution module).

Use this pattern when a perimeter intrusion detection system or inspection service is required, as well as additional inspection for a service module. This design can often replicate a typical three-tier design network. This pattern might be used, for example, to allow high-performance services (such as the presentation tier) to be provided by the service module in the left side of the figure, with higher security risk connections permitted through the service module on the right. These services could all be behind a distribution manager--wide integration security module, providing inspection or other services for the entire collection of components.

Security and Management

This section discusses the security frameworks and management networks used in the SDN methodology.

Security Framework

Throughout its design, a high security architecture must include multiple, independent, mutually reinforcing, and different security technologies. A decentralized approach to security in the SDN architecture allows all of the components to protect themselves. This approach enables the ability to provide a high level of security with few, if any, inspection bottlenecks. Differing, independent, and rationally configured technologies can be brought to bear at every level of the network stack (and in the host installation) to reinforce the security of the individual components.

When designing a network solution architecture, network architects must consider performance by minimizing latency without degrading the overall security framework. As such, network architects must examine the high-bandwidth segments of any proposed network architecture and reduce the volume of complex (frequent inspection-based) security checks that occur.

Integrated Security Framework

The SDN methodology uses an integrated security model that considers the security features in applications, containers, servers, network, security modules, and other security features in all of the different interfaces in the service delivery. The following figure provides an overview of this integrated security framework.

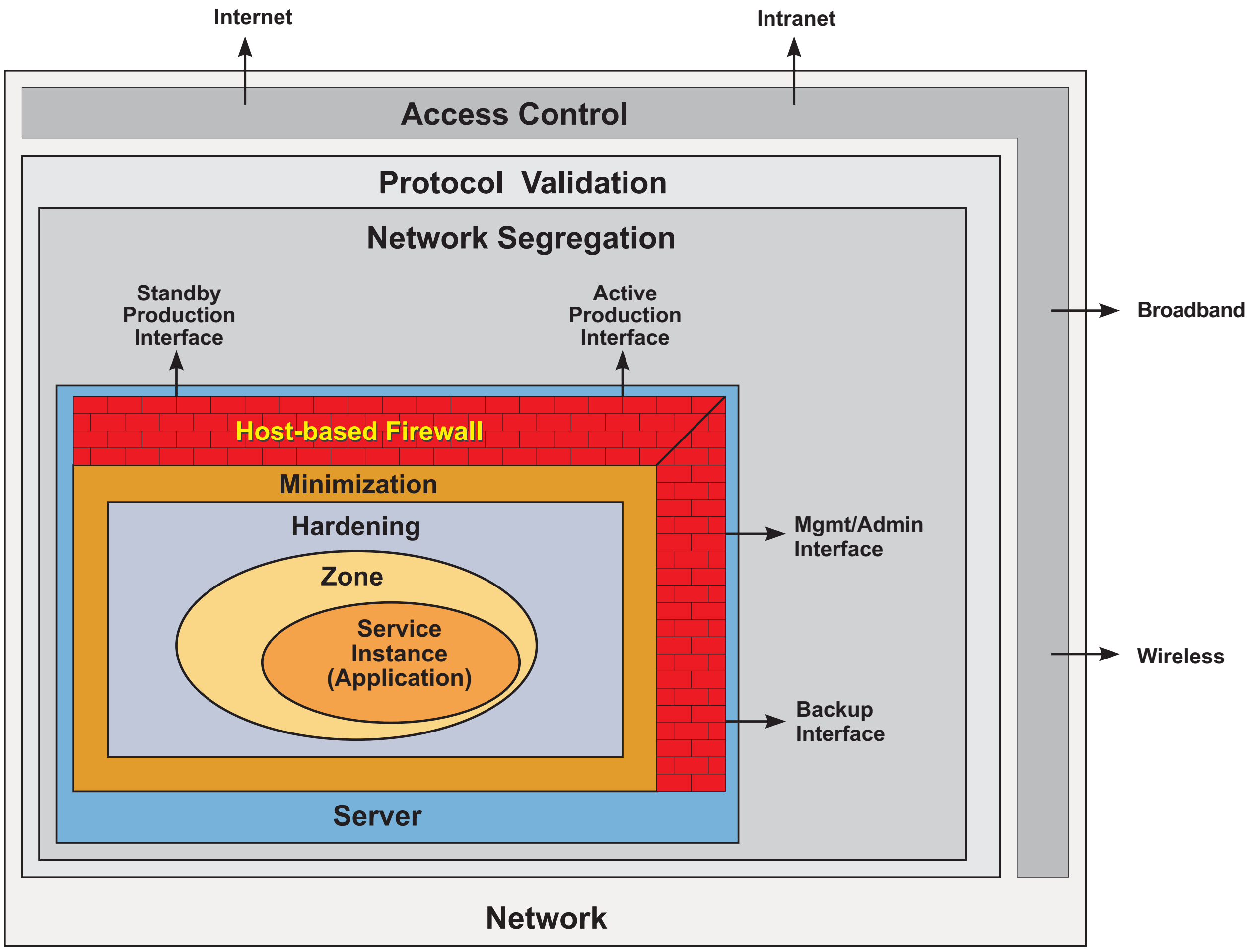

Figure 19. Integrated Security Framework

High security depends on securing each aspect of the path by which traffic flows through the network, its components, and applications, all the while maximizing performance and reducing latency.

Security Zones and Compartmentalization

Security zones enhance security and flexibility by enforcing compartmentalization. Security zones are implemented via virtual switching technology within a single load balancing switch. Each security zone in a single unit involves a virtual switch and includes a group of service domains that have similar security requirements. In addition, each security zone can be administered individually, which means that the administrator for security zone 1 cannot administer security zone 3 without explicit authorization.

Management Networks

Network administrators must be able to manage and monitor services and applications in a network, as well as any related logical and physical components. The SDN methodology incorporates a management network into a network architecture in a secure and reliable way, without risking adverse impact to the production side of the network. The following figure shows the operational, client-delivery side of a sample network.

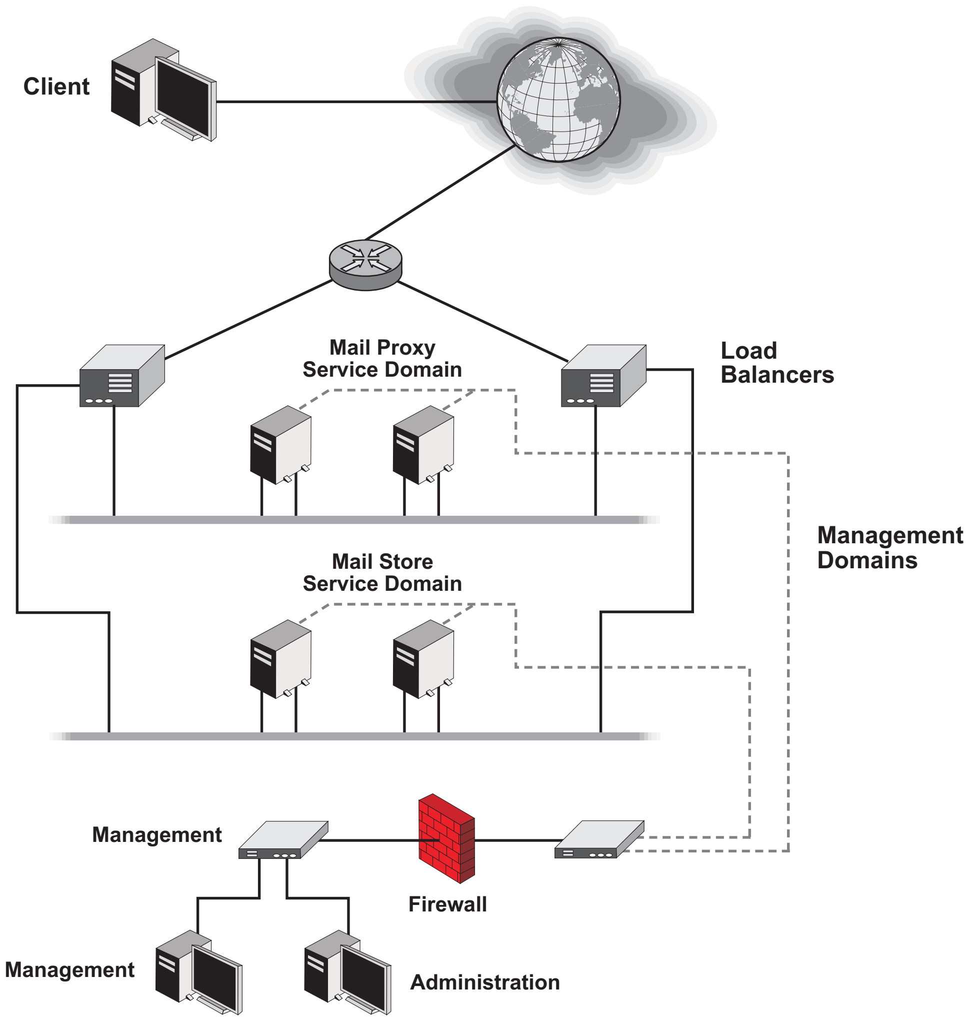

Figure 20. Service Domains and Management Domains

Inbound client requests are distributed by the load balancers. The management network is attached to a switch in front of the firewall. The management network infrastructure follows the security requirements of each service, and is separated into management domains.

Management Domains

Another design pattern, *management domains*, can be applied to network management requirements. Incorporating management domains into a network architecture design provides access to common management features, such as:

Feature	Description
NMS Integration	Includes network and system monitoring, usually performed using SNMP v2 usec, v3, or v3 usec, Sun Management Console or Customer Network Systems (CNS) agents, or third party products. Support for fault monitoring, performance monitoring, and security monitoring.
Access to Common Management Services	Includes DNS, NTP, SYSLOG, RADIUS, and so on.
Automated Provisioning	Includes the ability to automatically manage the configuration of the network, especially for common functions such as VLAN creation, modification, and load balancer updates. This coordination can be kicked off by N1, and integrated with third party network management products.
Administration Access	Usually provided through a command line interface, such as SSH, which is recommended, or (worse-case) telnet. Access through the NMS can provide another channel (usually via SNMP) for additional management options, in addition to the required serial connections for out of band management and access.
Network Equipment	Vendor-specific management capabilities, such as SNMP, HTTPS, or XML.
Hosts and Servers	A management network can provide access to agents such as SNMP, Sun Management Console, Teamquest, or BMC Patrol.
Staging Network Access	Enables support for operating system loads, initial setup, and provisioning. Using N1 and automated network provisioning (see above), systems can be automatically configured with an OS, then later configured for specific applications or containers.

NMS Server Network and Network Operations Center Access

The management network can optionally use a firewall to provide network access s to the various management domains, the network operation center segments, and (if applicable), the network management system server networks. For example, the network management system software could be configured to allow both polling and push management traffic. SNMP traps could be enabled and could be configured to forward the traps to specific event servers located in an internal-only management network segment.

The following figure shows a logical representation of management domains in relation to a network operation center segment.

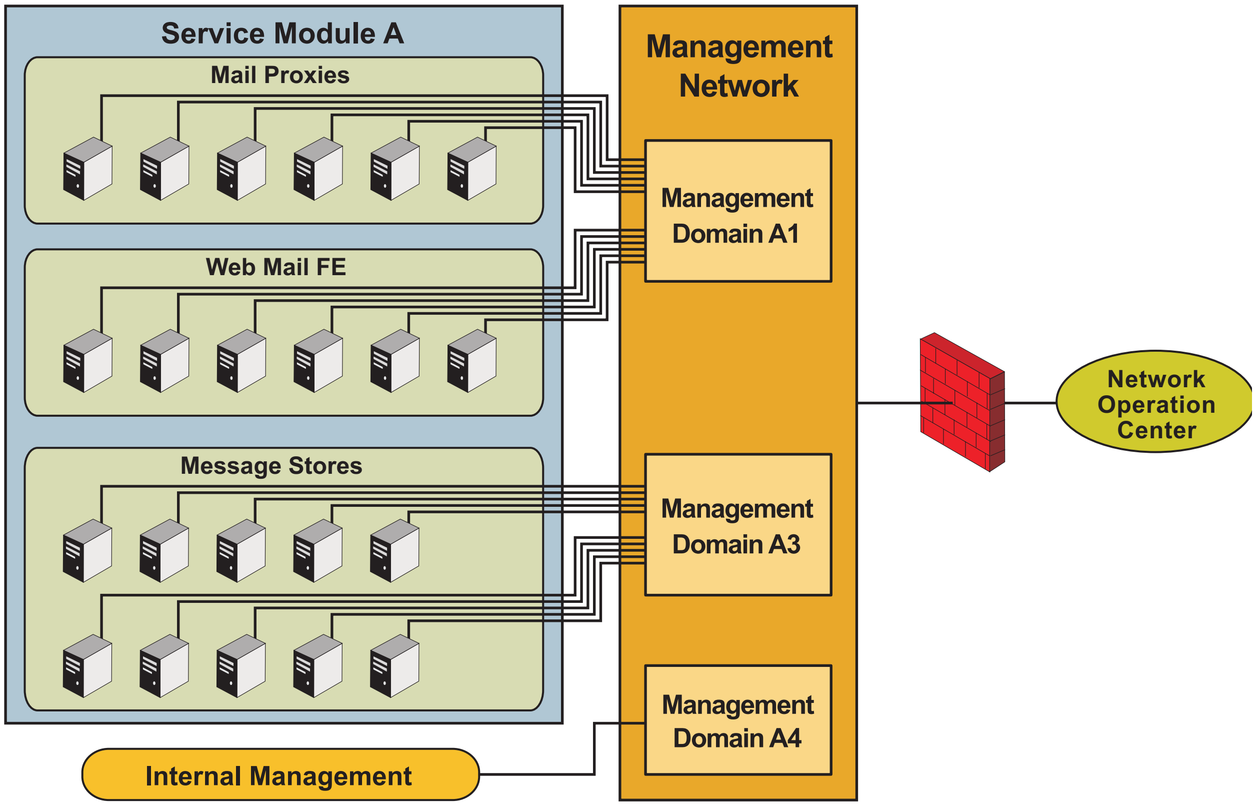

Figure 21. Management Network

In this example, management domains are shown in a logical architecture. In an actual implementation, management domains could be on separate switches or chassis-based switches using VLANs, depending on security and cost requirements.

A network management system can store data polled from the network on servers located in an internal-only network management system server network segment. Because the management network is a standard IP network, it can be used to integrate a variety of commonly used network management system platforms, which can in turn be integrated into an network operation center LAN or other network (including customer networks) as needed. The network operation center can access these servers to retrieve and view network and system events and other management data through a secure channel to the management server segment. This design can be further enhanced via user profile-based VPN technology.

Conclusion

Over the last year, several factors have affected future computing platforms and services delivered by service providers and enterprises:

- increasing requirements for availability, reliability, and compliance
- increasing pressure to reduce operational costs
- need to implement new services at a faster pace

These factors greatly enhance the need for scalable, secure, and high-performance distributed data center network topologies.

Today’s data centers face daunting challenges to satisfy huge growth and needs for continuous availability. The data center is now a dynamic, almost living organism, hosting new applications and services nearly every day. Network infrastructures must be designed with flexibility in mind—without compromising security, performance, and availability. Servers now reside in utility-like resource pools that require a dynamic network to provide the fluid ether to link systems together—wireless, wirebound, local and remote. Because applications require network services that are dynamically provisioned into service in a coordinated fashion, new technologies (such as Sun's N1 System Manager and N1 Grid Service Provisioning System) are needed to automate complex service deployments.

Network design has become too complex for the traditional, top-down approach of adapting a preconceived, cookie-cutter architecture to fit a given organization’s stated requirements. One-size-fits-all network architectures are a thing of the past. The SDN methodology is unique and important because it provides a framework for designing service-optimized network architectures from the bottom up—using core building blocks and common design patterns that are intrinsically secure, reliable, and fast. When used in conjunction with a holographic understanding of an organization’s unique business, technical, and service requirements, the SDN methodology can provide network designs and implementations that satisfy its core goal: service delivery at any time, from anywhere, to any device.

References and Related Sources

Sun's Service-Optimized Data Center

The SDN is part of Sun's more comprehensive the Sun^{SM} Service Optimized Data Center program (SODC). The SODC emphasizes dynamic data center service architectures, with increasing capabilities in virtualization and automation. The SODC employs optimization design patterns across a broad range of IT infrastructure elements, including servers and storage, operating systems, grids, middleware and shared services such as directory and J2EE, as well as application execution and messaging. The SODC is based on core Sun technologies such as the Solaris 10 OS, Sun Java Enterprise System (JES), Sun N1 Grid, and the Sun Secure application switches. For more information about the SODC, see:

http://www.sun.com/products-n-solutions/sodc/index.html

Sun's Service-Oriented Architecture

Orthogonal to the SDN and the SODC is Sun's Service-Oriented Architecture (SOA), an integrated software infrastructure and design approach based on industry best practices. SOA is an architectural style for building and composing software applications that use services available in a network. The SDN helps provide the communication fabric upon which the SOA is realized. It provides a foundation to build cost-effective, easy to manage services. For more information about SOA, see:

http://www.sun.com/products/soa/

Publications

- Alexander, Christopher. *The Timeless Way of Building*. (Oxford University Press, 1979).
- Dyson, Paul and Longshaw, Andy. *Architecting Enterprise Solutions: Patterns for High-Capability Internet-Based Systems* (John Wiley & Sons, Ltd., 2004)
- Brunette, Glenn M., and Schuba, Christoph, *Toward Systemically Secure IT Architectures*, which is available at the following URL:

  http://www.sun.com/software/security/docs/systemic-security-wp-1.pdf
- Carolan, Jason; Radeztsky, Scott; Strong, Paul; and Turner, Ed. *Building N1 Grid Solutions: Preparing, Architecting, and Implementing Service-Centric Data Centers*, which is available at the following URL:

  http://www.sun.com/blueprints/pubs.html

Web Sites

- Service Delivery Network Architecture Roadmap Services

  http://www.sun.com/service/sunps/architect/delivery/roadmap.html
- Architecture Services for Service Delivery Networks

  http://www.sun.com/service/sunps/architect/delivery/
- Service Delivery Network Architecture Implementation Service

  http://www.sun.com/service/sunps/architect/delivery/implementation.html
- Sun^{SM} Service Optimized Data Center

  http://www.sun.com/products-n-solutions/sodc/index.html
- Sun Secure Application Switch

  http://www.sun.com/n2000
- Sun Security

  http://www.sun.com/security

About the Authors

Mikael Lofstrand

Mikael Lofstrand is a Principal Architect at Sun Microsystems. Mikael has spent most of his time at Sun developing solutions for various customers improving systemic qualities through virtualization technologies. Mikael has recently spent two years in Sun's Strategy and Architecture team working on improving efficiency for globalized datacenters using various networking technologies. He is also one of the original authors of the Service Delivery Network Architecture, Sun's modular, data network architecture standard. Mikael often speaks at conferences world wide about the importance of network designs for a service driven architecture.

Jason Carolan
Jason Carolan is Client Solutions Chief Architect and Principal Engineer at Sun Microsystems. He has spent over six years in Sun Professional Services and Solutions organizations, focused on developing solutions for Sun's customers, systems architectures, and improving architectural quality through patterns. Jason contributed to many of the early internal and external documents of Sun's N1 software. He has also been responsible for the design of the Service Delivery Network Architecture, CSO's data center standard—a key example of modular architecture and reuse. He also speaks regularly at conferences throughout the world about network design, security, and the N1 Grid architecture.
Jason lives in Tucson, Arizona where he enjoys the amazing sun.

Acknowledgments
The authors would like to thank Glenn Brunette and Roy Pillay for their contributions to this article.