01 들어가기 — VSX란 무엇인가

Introduction

VSX는 **Virtual System eXtension**의 약자입니다. 한 줄로 말하면 하나의 하드웨어 위에서 여러 개의 가상 방화벽을 동시에 돌리는 기술입니다. 공식 문서의 첫 설명도 바로 이것입니다.

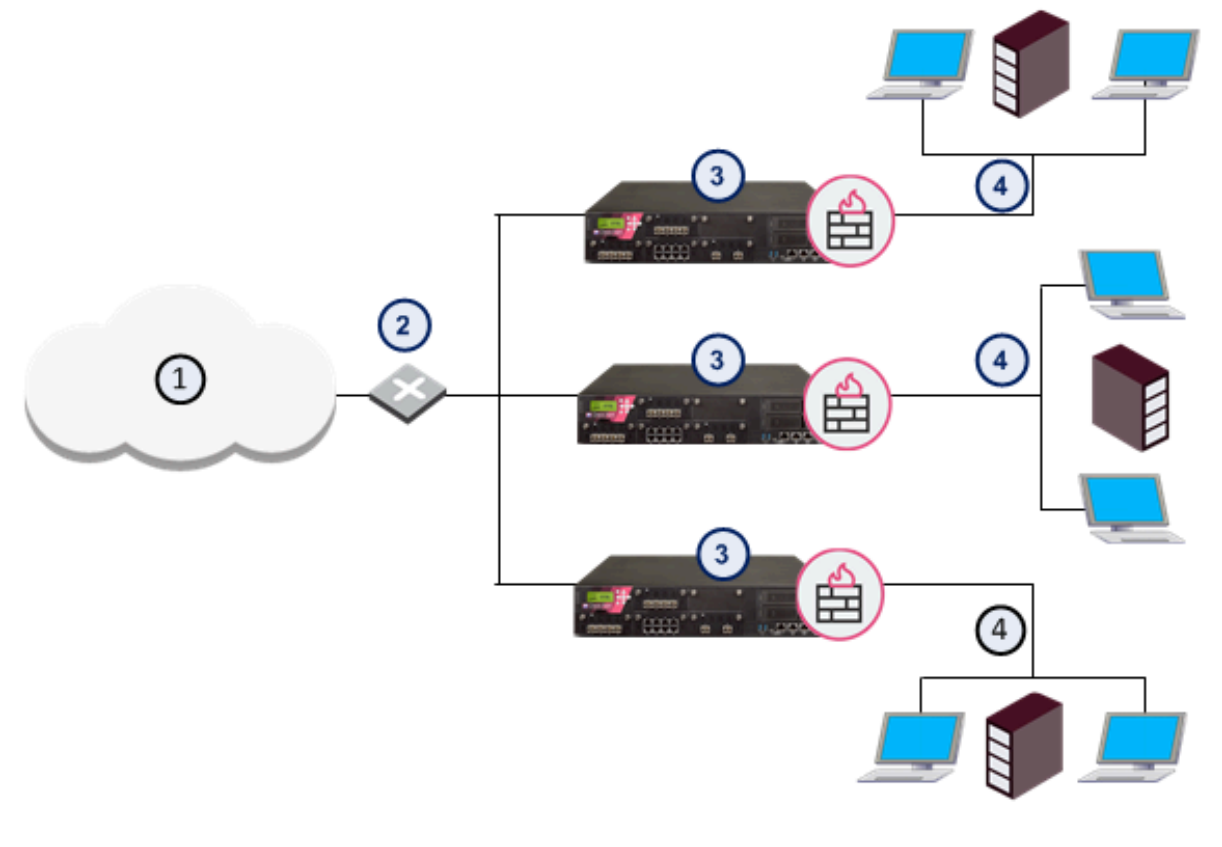
> *The VSX Administration Guide describes the Virtual System eXtension product that runs several virtual firewalls on the same hardware.*
>
> — AdminGuide, "Introduction" (p.16)

여기서 가상 방화벽 하나하나를 **Virtual System(VS)** 이라고 부르는데, 각 VS는 그 자체로 완전한 **Security Gateway**처럼 동작하면서 보통 하나의 네트워크를 보호합니다. 패킷이 **VSX Gateway**에 도착하면 게이트웨이는 그 트래픽을 목적지 네트워크를 담당하는 VS에게 넘기고, 그러면 그 VS가 트래픽을 검사해 자신의 보안 정책에 따라 허용하거나 거부합니다.

> *Each Virtual System works as a Security Gateway, typically protecting a specified network. When packets arrive at the VSX Gateway, it sends traffic to the Virtual System protecting the destination network. The Virtual System inspects all traffic and allows or rejects it according to rules defined in the security policy.*
>
> — AdminGuide, "Introduction" (p.16)

물리 네트워크와 무엇이 다른가

VSX를 이해하는 가장 쉬운 방법은 기존 물리 네트워크와 견주어 보는 것입니다. 전통적인 방식에서는 보호할 네트워크마다 물리 Security Gateway를 한 대씩 따로 둡니다. 각 게이트웨이는 바깥쪽 경계 라우터와 안쪽 보호 대상 네트워크 양쪽에 인터페이스를 꽂고 동작하며, 보호할 네트워크가 늘어나면 그만큼 장비도 늘어납니다. 결국 네트워크가 셋이면 방화벽도 세 대가 필요하고, **구매비와 랙 공간과 전력과 관리 포인트가 모두 세 배**가 됩니다.

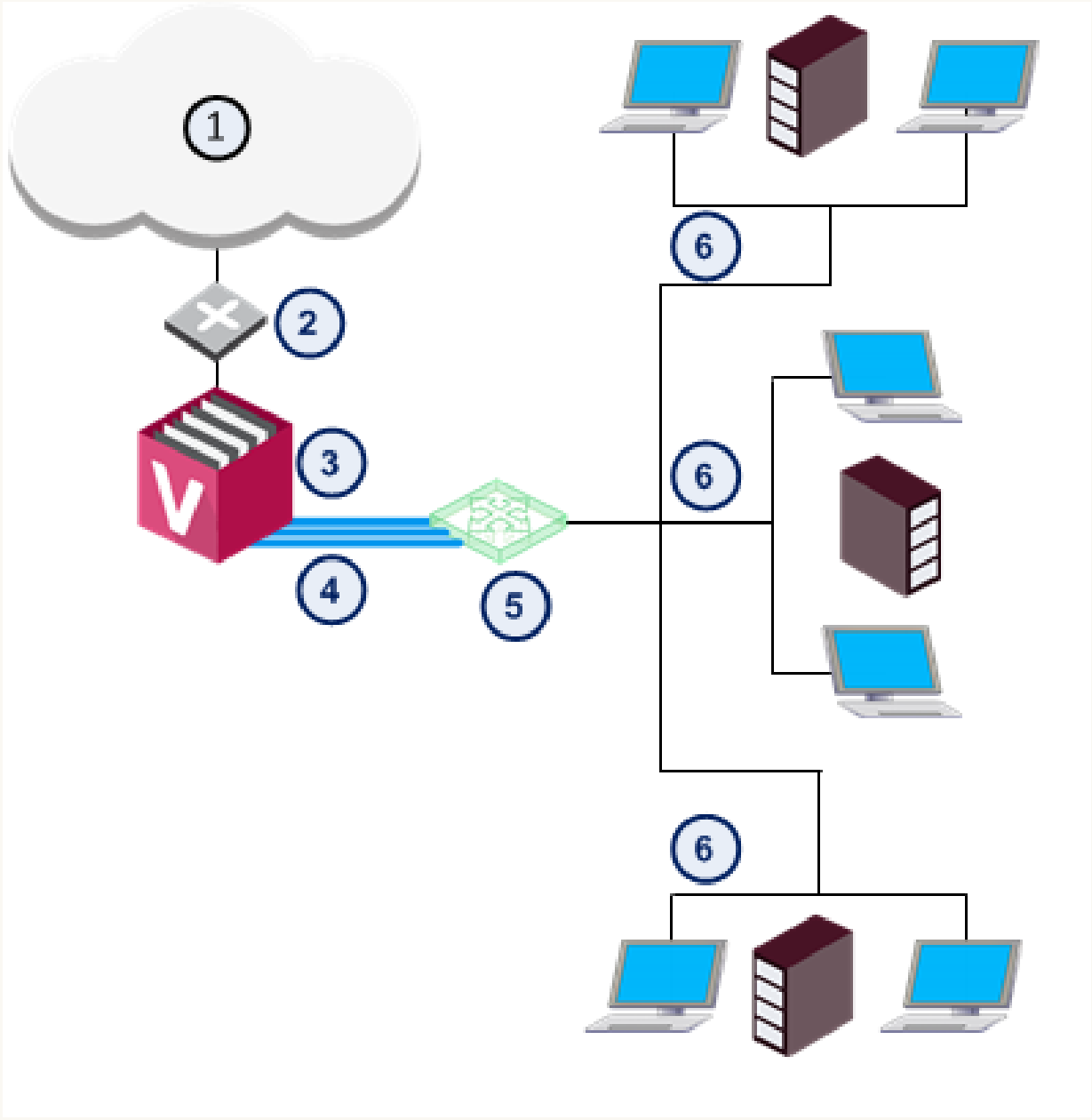


① 인터넷 ② 라우터 ③ Security Gateway ④ 네트워크 — 물리 네트워크에선 보호할 네트워크마다 게이트웨이가 따로 있다

VSX는 이 여러 대를 장비 한 대 안의 가상 시스템들로 합칩니다. 물리 네트워크가 여러 하드웨어로 이뤄지는 데 반해, VSX 가상 네트워크는 설정 가능한 하나의 VSX Gateway(또는 클러스터) 위에 존재하면서 그 안에서 여러 독립 네트워크와 가상 구성 요소를 함께 정의하고 보호합니다. 공식 문서는 이 대비를 다음과 같이 짚습니다.

While physical networks consist of many hardware components, VSX virtual networks reside on a single configurable VSX Gateway or cluster that defines and protects multiple independent networks, together with their virtual components.

— AdminGuide, "Introduction" (p.16)



① 인터넷 ② 라우터 ③ VSX Gateway ④ Warp Link ⑤ Virtual Switch ⑥ 네트워크 — 게이트웨이 한 대 안에 여러 VS가 들어간다

이 그림에는 앞으로 계속 등장할 요소들이 들어 있습니다. 물리 장비 본체에 해당하는 것이 **VSX Gateway**이고, 그 안의 각 VS가 물리 게이트웨이와 똑같은 보안·네트워킹 기능을 가진 채

자기가 맡은 네트워크의 트래픽을 처리합니다. 여러 VS를 인터넷 라우터 같은 공용 회선에 L2로 묶어 주는 것이 **Virtual Switch**이고, VS와 이 Virtual Switch(또는 **Virtual Router**)를 잇는 내부 가상 케이블이 **Warp Link**입니다. 각각이 정확히 무엇인지는 VSX 아키텍처와 핵심 개념 장에서 차근차근 풀어 설명합니다.

결국 VSX의 핵심은 한 대처럼 보이지만 안에서는 여러 대처럼 동작한다는 것입니다. 그리고 그 여러 대의 VS가 서로 완전히 분리되어 있다는 점이, 다음 챕터부터 이 문서 전체를 관통하는 VSX의 본질입니다.

02 VSNext

VSNext

R82에서 가장 큰 변화가 바로 VSNext입니다. R82부터 VSX에는 두 가지 모드가 있는데, 새로 도입된 **VSNext**와, R81.20 이하에서 그냥 "VSX"라 부르던 **Traditional VSX**입니다. 이 가이드의 다른 챕터들은 대부분 Traditional VSX를 설명하고, 이 챕터만 VSNext를 다룹니다.

> 먼저 읽으면 좋은 것
>
> VSNext는 VS·VR·VSW 같은 VSX의 기본 개념 위에 세워진 새 모드입니다. 아래에 나오는 Virtual Gateway·ElasticXL 같은 용어가 낯설다면, VSX 아키텍처와 핵심 개념 장을 먼저 보고 오면 훨씬 쉽게 읽힙니다.

VSNext란 무엇인가

VSNext는 R82에서 도입된 더 단순하게 설정하고 더 쉽게 프로비저닝하며, 물리 Security Gateway와 거의 같은 경험을 주는 향상된 VSX 모드입니다.

> *Introduced in the R82 version, VSNext is an enhanced VSX mode that allows simpler configuration, easier provisioning, and a similar experience to a physical Security Gateway.*
>
> — AdminGuide, "Introduction to VSNext" (p.19)

핵심 이점은 세 가지입니다. 첫째, 물리 Security Gateway와 가상 게이트웨이를 통합된 방식으로 관리하며, 각 가상 게이트웨이를 서로 다른 관리 서버로 관리할 수도 있습니다. 둘째, 가상 게이트웨이와 Virtual Switch를 Gaia Portal·Gaia Clish·Gaia REST API로 생성·수정·삭제할 수 있어 프로비저닝 성능과 경험이 개선됩니다. 셋째, 가상 게이트웨이(VGW)와 물리 Security Gateway 사이에 관리 기능과 API가 동등(parity) 해집니다. 용어도 바뀌어서, Traditional VSX의 Virtual System에 해당하는 것을 VSNext에서는 **Virtual Gateway(VGW)** 라고 부릅니다.

여기에 중요한 전제가 있습니다. VSNext 모드는 ElasticXL Cluster 또는 Maestro Security Group에서만 켤 수 있습니다. 단독 게이트웨이에서는 쓸 수 없습니다. 그리고 CPUSE 패키지 설치는 ElasticXL/Maestro Security Group 전체에 적용된다는 점도 기억해야 합니다.

설정하는 세 가지 방법

VSNext에서 가상 게이트웨이와 Virtual Switch는 Gaia Portal, Gaia Clish, Gaia REST API 중 어느 것으로도 설정할 수 있습니다. 물리 게이트웨이와 같은 경험을 준다는 말이 여기서 드러나는데, SmartConsole이 아니라 Gaia 쪽 도구로 객체를 만든다는 점이 Traditional VSX와의 큰 차이입니다.

Gaia Portal에서는 Virtual Systems 페이지에서 가상 게이트웨이와 Virtual Switch 객체를 만들고, 상단 필터에서 원하는 가상 게이트웨이를 고른 뒤 각 페이지에서 Gaia OS 설정을 합니다.

Gaia Clish에서는 `add vsnext ...` 계열 명령으로 객체를 만들고 `set vsnext ...`로 설정하며 `show vsnext ...`로 확인하고 `delete vsnext ...`로 지웁니다. 예를 들어 가상 게이트웨이를 추가할 때는 인터페이스·ID·SIC 활성화 키·CoreXL 인스턴스 수를 함께 지정하고, Virtual Switch를 만든 뒤 둘을 잇는 virtual-link를 만듭니다.

```
# 가상 게이트웨이 추가 (인터페이스·ID·SIC키·CoreXL 인스턴스 지정)
add vsnext virtual-gateway interfaces <인터페이스> id {auto | <1-511>} \
    one-time-password <SIC키> instances <IPv4 인스턴스 수> ...

# Virtual Switch 추가 후 가상 게이트웨이와 연결
add vsnext virtual-switch name <이름> id <ID>
add vsnext virtual-link virtual-gateway <VGW ID> virtual-switch <VSW ID> wrp-

# 확인 / 작업 추적
show vsnext overview virtual-systems
show vsnext state
show vsnext running-tasks
```

Gaia REST API에서는 `add-virtual-gateway`·`add-virtual-switch`로 만들고 `set-virtual-gateway`·`set-virtual-switch`로 설정하며 `show-...`로 조회하고 `delete-...`로 지웁니다. 자세한 명세는 Gaia API Reference의 "VSNext" 챕터에 있습니다.

Gaia Portal로 설정하는 흐름

설정에서 가장 중요한 제약부터 짚으면, VSNext는 깨끗한 새 설치에서만 가능하고 기존 플랫폼을 VSNext로 변환할 수는 없습니다.

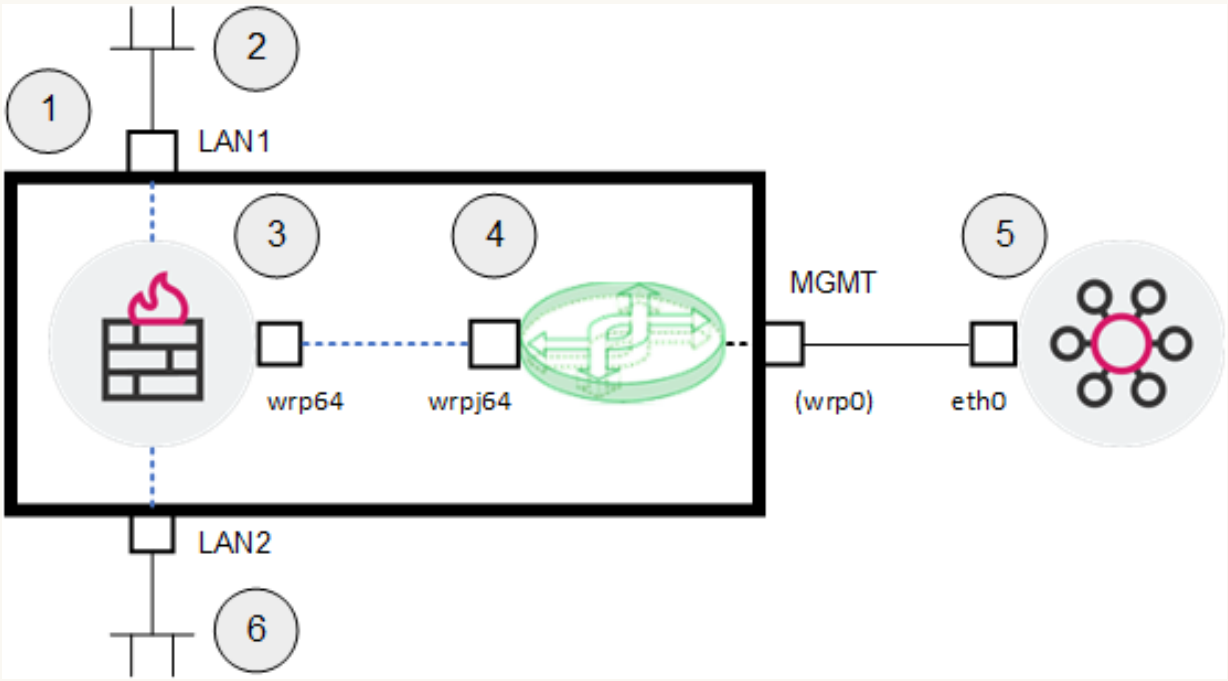
> *It is not supported to convert an existing platform to the VSNext mode.*
>
> — AdminGuide, "Procedure for Gaia Portal" (p.22)

그래서 첫 단계는 지원 플랫폼을 클린 설치하고 First Time Configuration Wizard에서 VSNext로 설정하는 것입니다. ElasticXL Cluster라면 마법사에서 Security Gateway를 고르고 클러스터 항목에서 ElasticXL을 선택한 뒤 Gateway Virtualization 항목에서 "Install as VSNext" 를 고르고 SIC 활성화 키를 입력합니다. Maestro Security Group이라면 Orchestrator에서 새 Security Group을 만들 때 마법사에서 "Install as VSNext / VSX"를 선택합니다. 이후 유효한 라이선스를 설치합니다.

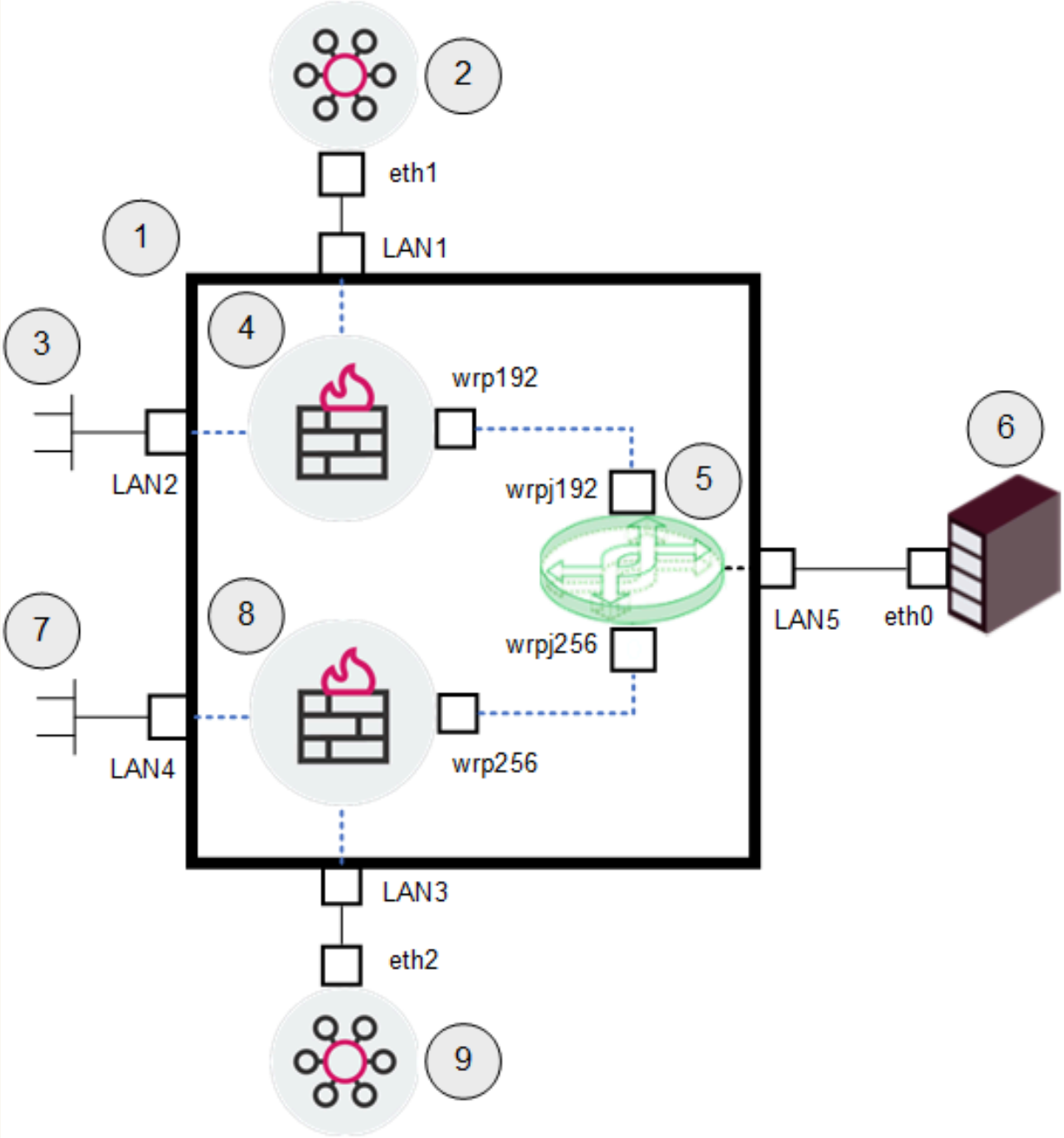
설치가 끝나면 마법사에서 지정한 메인 IP로 Gaia Portal에 접속합니다. IPv6 주소나 IPv6 CoreXL을 쓰려면 먼저 System Configuration에서 IPv6 Support를 켜고 재부팅해야 합니다. 상단 Virtual System 필터 옆에 자물쇠 아이콘이 보이면 클릭해 잠금을 풀어 연필 아이콘으로 바꾼 뒤, 좌측 Virtual Systems 페이지로 들어갑니다.

이 페이지에는 미리 정의된 기본 객체 두 개 가 보입니다. 하나는 ID 500의 기본 Virtual Switch(인터페이스 `magg1`)이고, 다른 하나는 ID 0의 기본 Virtual Gateway(인터페이스 `wrp0`)인데 이 기본 게이트웨이는 기본 Virtual Switch에 연결돼 있습니다. 이 기본 객체들은 건드리면 안 됩니다. ID 0 기본 Virtual Gateway는 다른 가상 게이트웨이를 설정하려고 접속하는 **관리 컨텍스트를 보호**하는 역할이라, Gaia OS에 안전하게 접근하기 위해 설정해 두는 것이 권장됩니다.

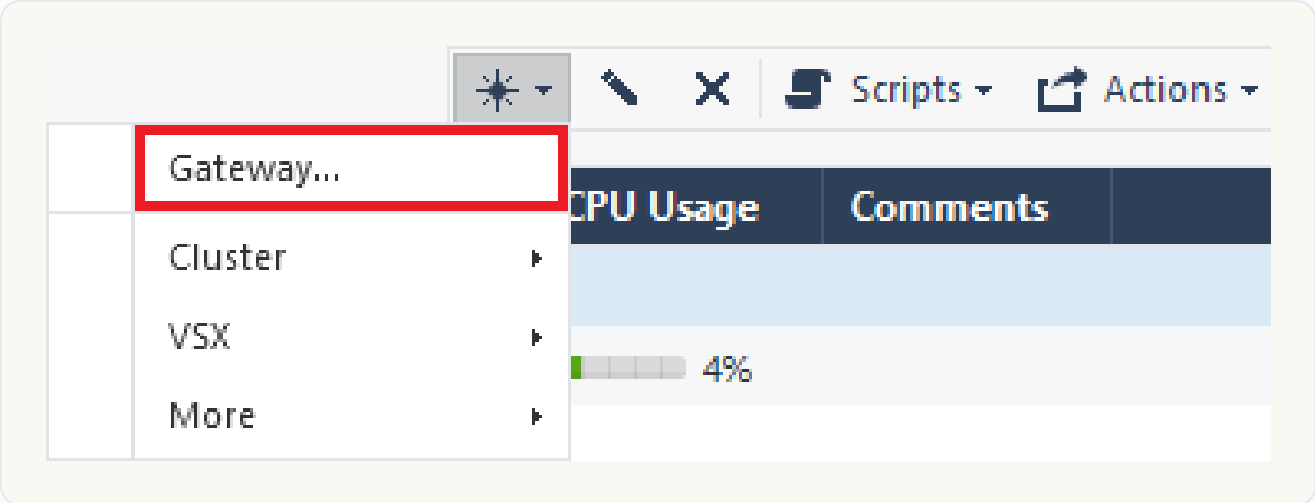
그다음 실제 업무용 가상 게이트웨이와 Virtual Switch를 만듭니다. 기본 Virtual Switch 하나로 여러 가상 게이트웨이를 관리할 수 있는데, 이때 그 가상 게이트웨이들의 IP는 모두 ElasticXL Cluster의 MGMT 포트(또는 Maestro Security Group에 배정된 관리 포트)와 같은 서브넷 이어야 합니다. 작업 진행 상황은 하단 Tasks 패널에서 Status가 Completed가 되는지로 확인하고, 자세한 로그는 게이트웨이의 `/var/log/vsxd.elg*` 에서 볼 수 있습니다.



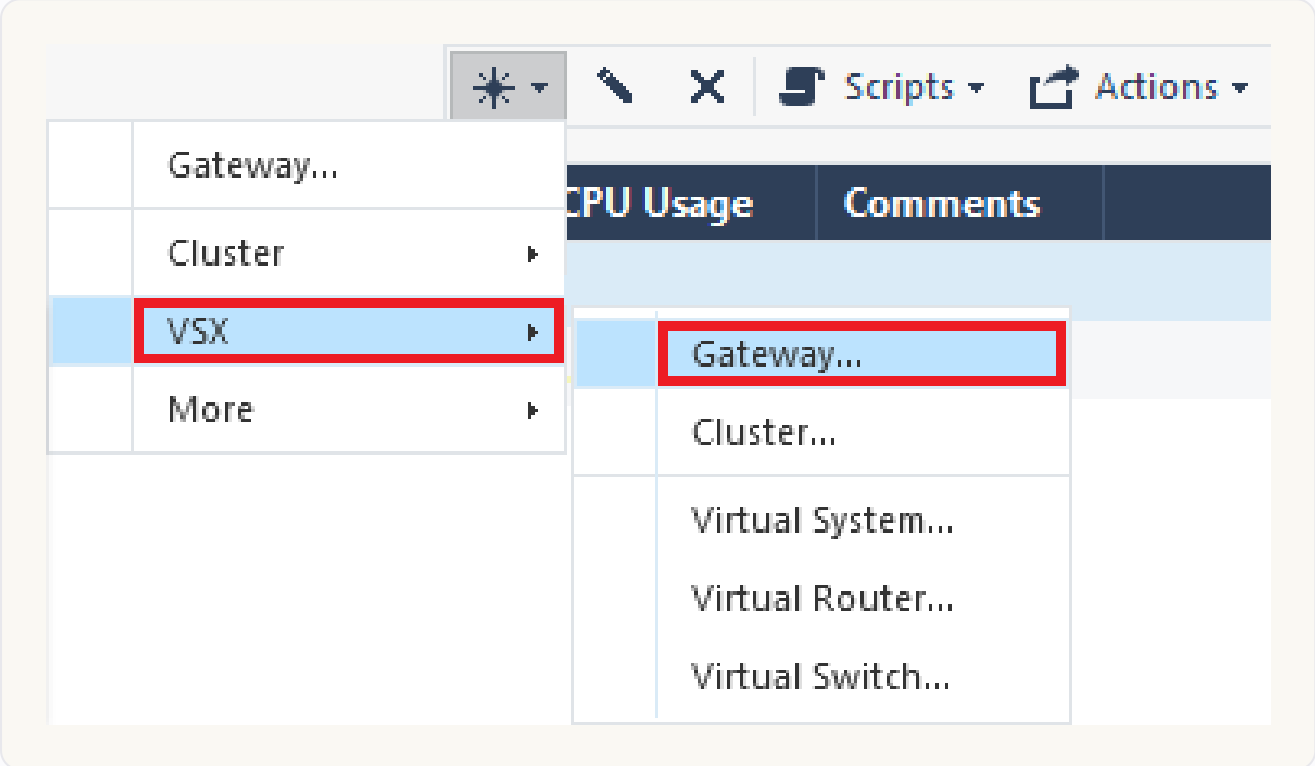
① VSNext 모드 ElasticXL Cluster ② 네트워크 #1 ③ Virtual Gateway ④ 기본 Virtual Switch ⑤ 관리 서버 ⑥ 네트워크 #2



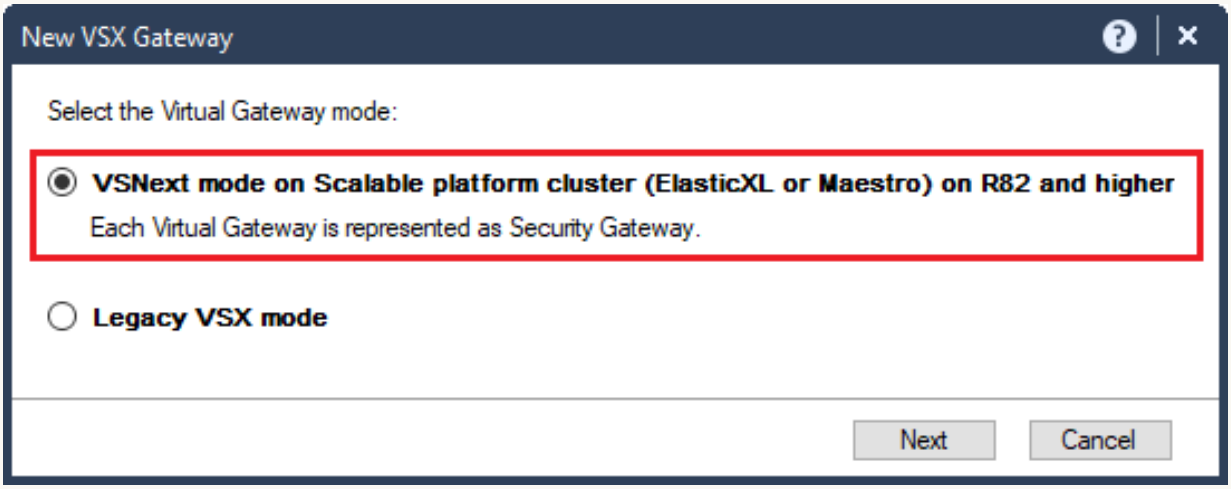
① ElasticXL Cluster(VSNext) ② 관리 서버 #1 ③ 네트워크 #1 ④ Virtual Gateway #1 ⑤ Virtual Switch
⑥ Backup 서버 ⑦ 네트워크 #2 ⑧ Virtual Gateway #2



VSNext 가상 게이트웨이 구성



VSNext 인터페이스·연결 구성


New VSX Gateway
Select the Virtual Gateway mode:
VSNext mode on Scalable platform cluster (ElasticXL or Maestro) on R82 and higher
Each Virtual Gateway is represented as Security Gateway.
Legacy VSX mode
Next
Cancel

VSNext 구성 상세

03 VSX 아키텍처와 핵심 개념

VSX Architecture and Concepts

이 챕터가 VSX의 뼈대입니다. VSX Gateway가 무엇을 하는 호스트인지, 그 안의 가상 장치들은 어떤 종류가 있는지, 패킷은 어떻게 흐르는지를 한자리에서 정리합니다. 뒤의 모든 챕터가 여기서 나온 개념 위에 쌓이므로, 다른 건 몰라도 이 챕터만큼은 천천히 읽어 두는 편이 좋습니다.

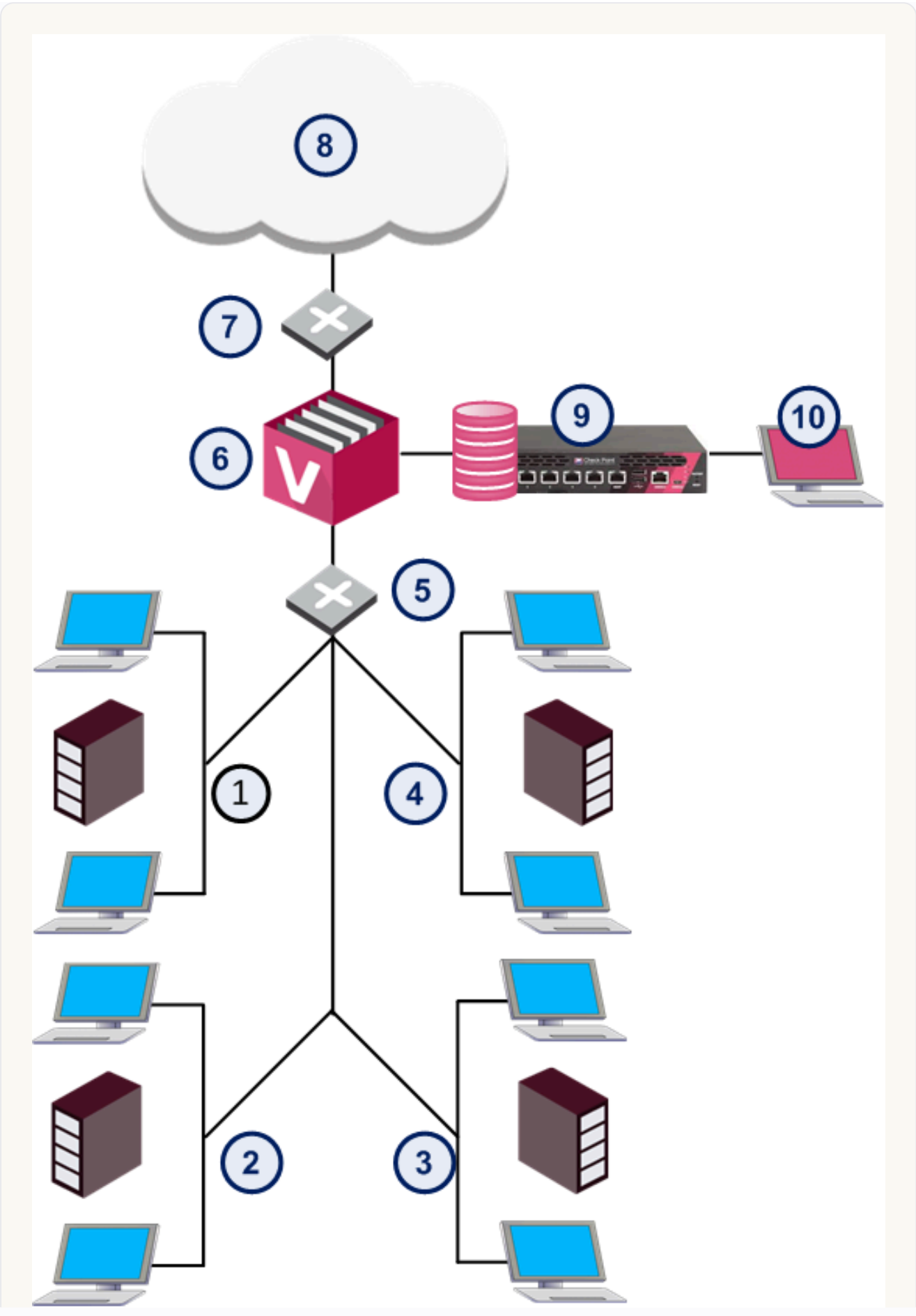
VSX Gateway — 가상 네트워크를 얹는 물리 호스트

VSX Gateway는 물리 장비 한 대입니다. 이 위에 Security Gateway·라우터·스위치 같은 물리 장비의 기능을 그대로 흉내 내는 **가상 장치(Virtual Device)**들을 올려 가상 네트워크를 구성합니다. VSX Gateway 자신이 직접 맡는 일은 두 가지인데, 하나는 관리 서버와 통신하며 모든 가상 장치를 배포하고 설정하고 관리하는 것이고, 다른 하나는 클러스터로 묶였을 때 High Availability와 Load Sharing을 위해 멤버 간 상태를 동기화하는 것입니다.

한 가지 용어상의 주의가 있습니다. Maestro나 Scalable Chassis 환경에서는 "VSX Gateway"라는 말이 VSX 모드로 동작하는 Security Group을 가리키며, 이 경우 일부 VSX 기능은 제한적이거나 아예 지원되지 않습니다.

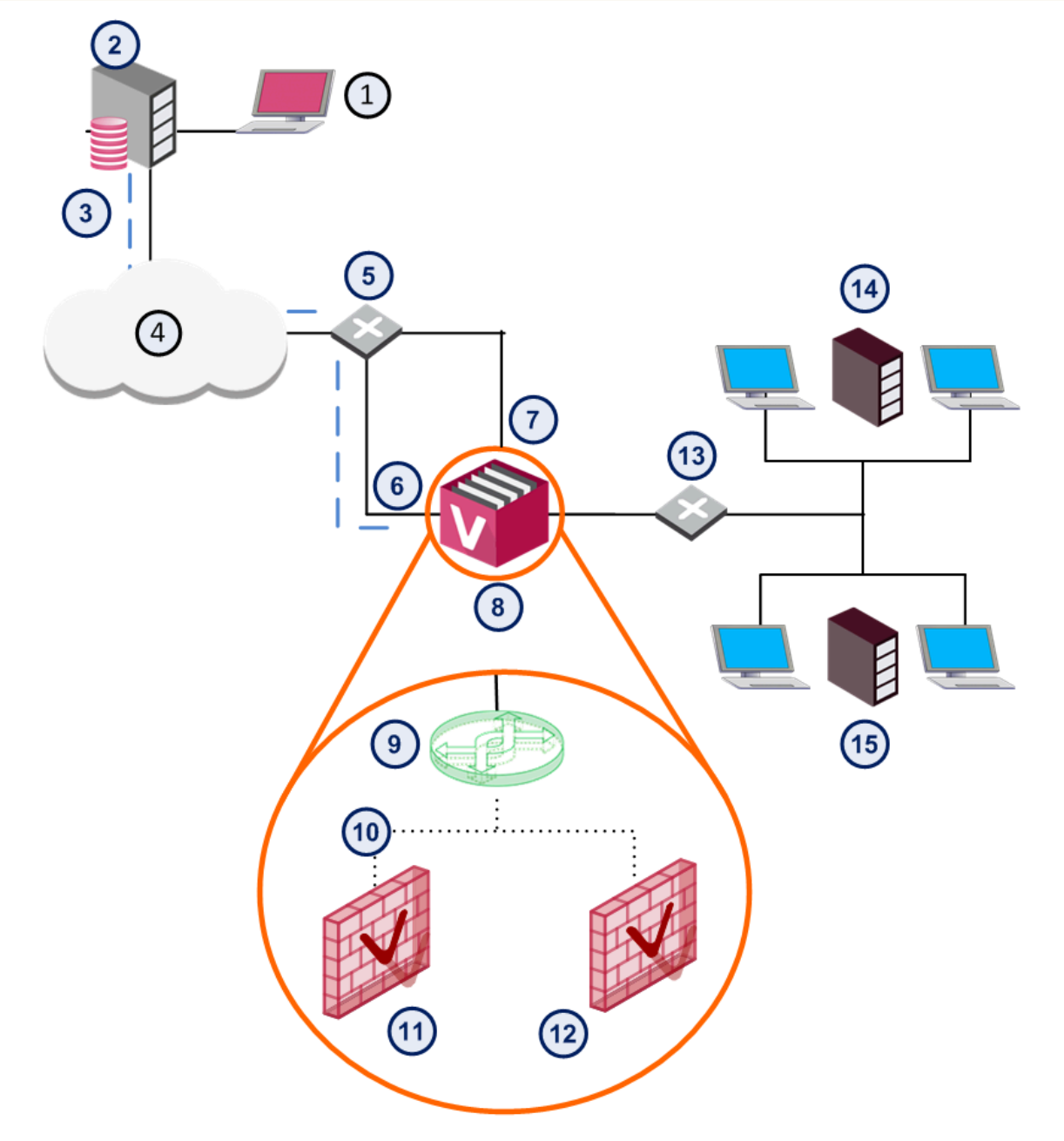
관리 서버는 어떻게 붙는가

관리 서버는 VSX Gateway에 연결해 가상 장치들을 프로비저닝하고 설정하는데, 연결 방식은 로컬과 원격 두 가지입니다. 로컬 관리 연결은 관리 서버가 전용 관리 인터페이스(DMI)를 통해 VSX Gateway에 직접 붙는 방식으로, 이때 관리 인터페이스의 IP는 사설이든 공인이든 상관없습니다.


8
7
6
9
10
5
1
4
2
3

① 네트워크 1 ② 네트워크 2 ③ 네트워크 3 ④ 네트워크 4 ⑤ 스위치 ⑥ VSX Gateway ⑦ 라우터 ⑧ 인터넷 ⑨ 관리 서버 ⑩ SmartConsole

원격 관리 연결은 라우터를 거쳐 내부망이나 외부망을 통해 관리 인터페이스에 도달하는 방식인데, 이렇게 하면 관리 트래픽이 일반 트래픽과 분리되는 이점이 있습니다. 다만 인터넷을 통해 들어오는 경우라면 라우팅 가능한 공인 IP를 배정해야 합니다.



① SmartConsole ② 관리 서버 ③ 관리 트래픽 ④ 인터넷 ⑤ 라우터 ⑥ 전용 관리 인터페이스(eth0) ⑦ 외부 인터페이스 ⑧ VSX Gateway ⑨ Virtual Switch ⑩ Warp Link ⑪ Virtual System 1 ⑫ Virtual System 2 ⑬ 스위치 ⑭ 네트워크 1 ⑮ 네트워크 2

여기에는 반드시 알아 둘 제약이 몇 가지 있습니다. 우선 관리 서버나 VSX Gateway가 NAT 뒤에 있으면 관리가 지원되지 않습니다. 또한 관리 트래픽이 같은 게이트웨이 위의 어떤 VS를 통과해서는 안 되며, **관리 서버와 직접 통신할 수 있는 것은 오직 VS0뿐**입니다. 그리고 R81부터는 비전용 관리 인터페이스(Non-DMI, 공유 인터페이스) 설정이 폐기되어 더 이상 지원되지 않으니, 관리에는 전용 인터페이스인 DMI를 쓰는 것이 정석입니다. 관리 트래픽을

운영 트래픽과 분리하면 성능에도 유리한데, 다만 VLAN 인터페이스가 설정된 물리 인터페이스로는 DMI 관리가 지원되지 않는다는 점만 기억하면 됩니다.

가상 장치 세 가지 — VS, VR, VSW

VSX의 주인공은 세 가지 가상 장치입니다. 이 셋의 차이만 또렷이 잡으면 VSX의 절반은 이해한 셈입니다.

첫째는 **Virtual System(VS)**, 곧 가상 방화벽입니다. 방화벽과 VPN 기능을 다 갖춘 가상 보안·라우팅 도메인으로, 한 VSX Gateway 위에서 여러 VS가 동시에, 그리고 저마다 완전히 독립적으로 돕니다.

> *A Virtual System is a virtual security and routing domain that provides the functionality of a Security Gateway with full Firewall and VPN facilities. Multiple Virtual Systems can run concurrently on a single VSX Gateway. Each Virtual System functions independently.*
>
> — AdminGuide, "Virtual Devices" (p.48)

여기서 각 VS가 모든 것을 자기 것으로 따로 갖는다는 점이 핵심입니다. 자기만의 Software Blade와 인터페이스와 IP를 갖는 것은 물론, **라우팅 테이블과 ARP 테이블과 동적 라우팅 설정**까지 따로 가진다는 뜻입니다. 더 나아가 활성 연결이나 IPsec 터널 정보를 담는 상태 테이블도, 관리 서버에서 받아 디스크와 커널에 저장하는 보안·VPN 정책(INSPECT 코드 포함)도, IPS 설정이나 TCP/UDP 타임아웃 같은 설정값도, 심지어 로그까지도 모두 VS마다 별개입니다. 그래서 어떤 VS는 L3 모드로, 다른 VS는 L2 모드로 같은 게이트웨이 위에서 섞여 공존할 수도 있습니다.

둘째는 **Virtual Router(VR)**, L3 가상 라우터입니다. VSX Gateway 안에서 물리 라우터 역할을 하는 독립 라우팅 도메인으로, 여러 VS가 인터넷으로 가는 회선처럼 하나의 공유 인터페이스를 함께 써야 할 때, 또는 VS끼리 트래픽을 라우팅해야 할 때 씁니다.

> *A Virtual Router is an independent routing domain within a VSX Gateway that performs the functionality of physical routers. Virtual Routers are useful for connecting multiple Virtual Systems to a shared interface, such as the interface leading to the Internet, and for routing traffic from one Virtual System to another.*
>
> — AdminGuide, "Virtual Devices" (p.49)

VR은 출발지나 목적지 IP를 보고 알맞은 VS로 패킷을 보내고 DMZ 같은 공유 자원과의 트래픽도 처리하며, 물리 라우터처럼 자기 라우팅 테이블을 유지합니다. 자신에게 직접 오고

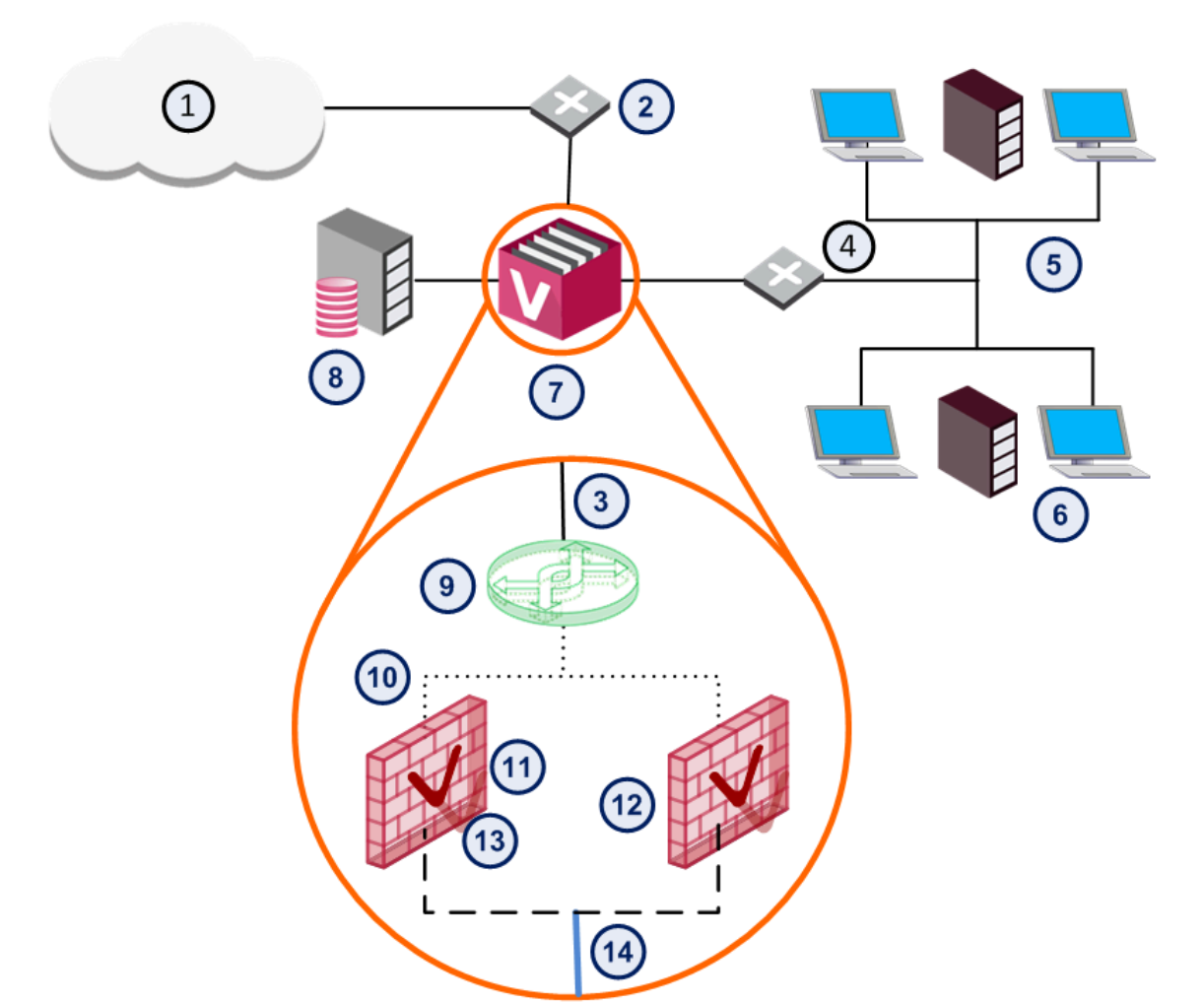
가는 트래픽, 예컨대 VR의 IP로 보내는 ping 같은 것은 정책으로 검사하지만, 자신을 거쳐 가기만 하는 트래픽은 검사 없이 목적지로 넘깁니다. 다만 여기에는 꼭 기억해야 할 제약이 하나 있습니다. Maestro와 Scalable Chassis의 Security Group은 Virtual Router를 지원하지 않으므로(Known Limitation 01413513), Maestro 위에서 VSX를 쓸 때는 VR 기반의 공유 업링크 설계를 쓸 수 없고 VLAN 직결이나 Virtual Switch로 설계해야 합니다.

셋째는 **Virtual Switch(VSW)**, L2 가상 스위치입니다. L2 연결성을 제공해 여러 VS를 잇고, 기존 IP 네트워크를 쪼개지 않으면서 공통 물리 인터페이스를 공유하게 해 줍니다. 물리 스위치가 그렇듯 MAC 주소와 포트를 담은 포워딩 테이블을 유지합니다.

> *By providing Layer 2 connectivity, a Virtual Switch connects Virtual Systems and facilitates sharing a common physical interface without segmenting the existing IP network. As with a physical switch, each Virtual Switch maintains a forwarding table with a list of MAC addresses and their associated ports.*
>
> — AdminGuide, "Virtual Devices" (p.50)

Virtual Router와 비교하면 차이가 분명한데, Virtual Switch를 쓰면 VS들을 위한 별도 서브넷을 따로 할당할 필요도 없고 인접 라우터의 라우팅을 손댈 필요도 없습니다. 결국 VR과 VSW 사이의 선택은 트래픽을 라우팅으로 나눌 것인지(L3, Virtual Router), 아니면 같은 네트워크에 그냥 붙일 것인지(L2, Virtual Switch)의 문제로 정리됩니다.

인터페이스의 종류



① 인터넷 ② 라우터 ③ 물리 인터페이스 ④ VLAN 스위치 ⑤ 네트워크 1 ⑥ 네트워크 2 ⑦ VSX Gateway ⑧ 관리 서버 ⑨ Virtual Switch ⑩ Warp 인터페이스 ⑪ Virtual System 1 ⑫ Virtual System 2 ⑬ VLAN 인터페이스 ⑭ VLAN 트렁크

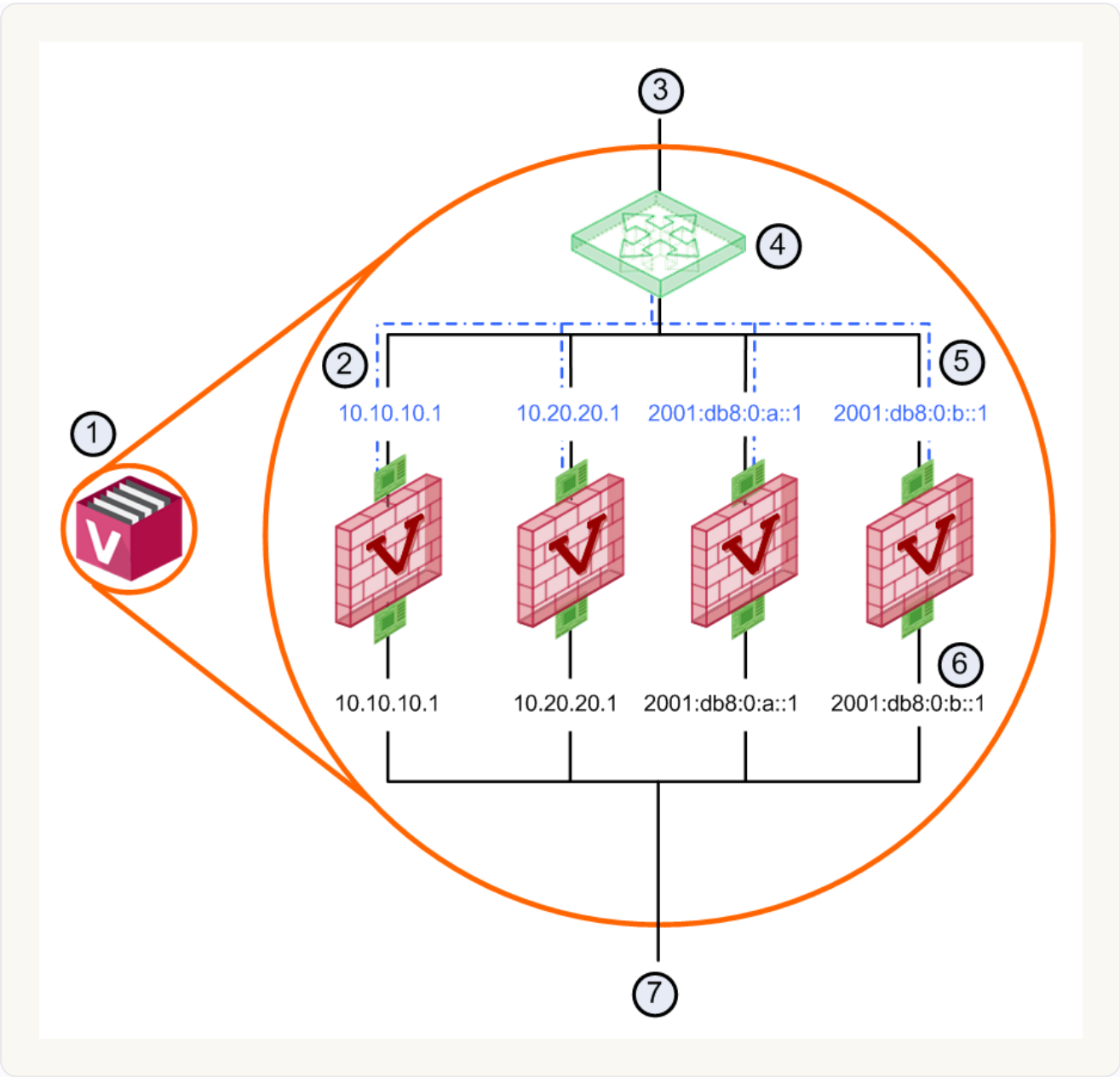
VSX에서 쓰는 인터페이스는 네 가지로 나뉩니다. 가장 기본이 되는 것은 **물리 인터페이스**로, VSX Gateway를 관리 서버와 내부망·외부망에 연결합니다. 용도에 따라 관리 서버에 붙는 전용 관리 인터페이스, 인터넷처럼 신뢰할 수 없는 망에 붙는 외부 인터페이스, 보호 대상 네트워크에 붙는 내부 인터페이스, 그리고 클러스터 멤버 간 상태 동기화를 담당하는 동기화 인터페이스가 있습니다.

실무의 핵심은 **VLAN 인터페이스**입니다. 보통 보호 대상 네트워크들을 802.1q 호환 스위치에 연결하면, 그 스위치가 모든 트래픽을 하나의 물리 인터페이스로 트렁킹해 VSX Gateway에

넘깁니다. 그러면 VSX가 프레임의 VLAN 태그를 보고 각 네트워크를 담당하는 VS로 분배하는데, 이때 물리 인터페이스의 VLAN 태그마다 가상 인터페이스가 하나씩 배정됩니다. 예를 들어 eth3의 VLAN 태그 100은 `eth3.100`이라는 이름의 가상 인터페이스가 됩니다.

세 번째는 **Warp Link**입니다. VS와 Virtual Router 또는 Virtual Switch 사이를 잇는 내부 가상 P2P 연결로, 양쪽 끝이 각각 가상 인터페이스가 됩니다. VS 쪽 인터페이스에는 `wrp` 접두어가, VR이나 VSW 쪽에는 `wrpj` 접두어가 붙고 그 뒤에 고유 번호가 따라옵니다. Virtual Switch에 연결될 때는 각 Warp Link에 고유 MAC 주소도 함께 배정됩니다.

마지막은 **Unnumbered Interface**로, IP 주소를 아끼기 위한 기법입니다. Virtual Router로 향하는 Warp Link에 전용 IP를 따로 주는 대신 다른 인터페이스의 IP를 빌려 쓰는 방식인데, 몇 가지 제약이 따릅니다.



① VSX Gateway ② Virtual Router의 next hop이 되는 외부 인터페이스 ③ 외부 ④ Virtual Router ⑤ 내부 인터페이스 IP를 빌린 Unnumbered 외부 인터페이스 ⑥ 고정 IP를 가진 내부 인터페이스 ⑦ 내부

빌려 쓰는 인터페이스는 반드시 Virtual Router에 연결돼 있어야 하고, 한 인터페이스의 IP는 한 번만 빌려줄 수 있으며, VPN이나 Hide NAT을 쓰려면 빌린 주소가 라우팅 가능해야 합니다.

관리 모델 — SMS와 MDS

VSX는 두 가지 관리 모델을 지원합니다. **Security Management Server(SMS)** 모델은 VS는 많지만 도메인은 하나인 기업 환경에 적합합니다. 이 경우 SmartConsole이 VS들을 담고 있는 VSX Gateway에 붙어 각 VS를 직접 관리하며, 권장 VS 수는 25개 정도입니다. 반면 **Multi-Domain Server(MDS)** 모델은 서로 다른 도메인이나 부서, 지사를 중앙에서 관리할 때 씁니다. 이때 MDS가 각 네트워크의 정책 데이터베이스를 통제하는 중앙 노드가 되고, 도메인마다 Domain Management Server가 자기 SmartConsole로 자신의 VS와 가상 장치, 정책을 관리합니다. 실용적으로 도메인을 최대 250개까지, VS도 250개까지 다룰 수 있습니다.

MDS 환경에서는 역할에 따라 이름이 나뉩니다. VSX Gateway나 Cluster를 관리하는 Domain을 Main Domain Management Server라 하고, 그 안의 개별 VS나 VR을 관리하는 Domain을 Target Domain Management Server라 합니다. 한 VSX Gateway 위의 VS들이 서로 다른 Domain의 관리를 받을 수도 있습니다. 한편 라이선스 측면에서 주의할 점이 있는데, EULA상 하나의 Security Management Server는 단일 법인에 속한 VS의 정책만 관리할 수 있습니다. 따라서 여러 법인의 VS를 관리하려면 법인마다 별도의 Domain Management Server를 둔 MDS를 써야 합니다.

관리 서버와 VSX Gateway 사이의 모든 통신은 인증서 기반 보안 채널인 SIC(Secure Internal Communication)로 이뤄집니다. 최초의 신뢰는 VSX Gateway를 설정할 때 일회용 비밀번호로 맺는데, 가상 장치는 물리 장비와 달리 만들어질 때 VSX가 관리 서버와 게이트웨이 사이의 채널을 통해 SIC 신뢰를 자동으로 설정해 줍니다.

패킷은 어떻게 흐르는가

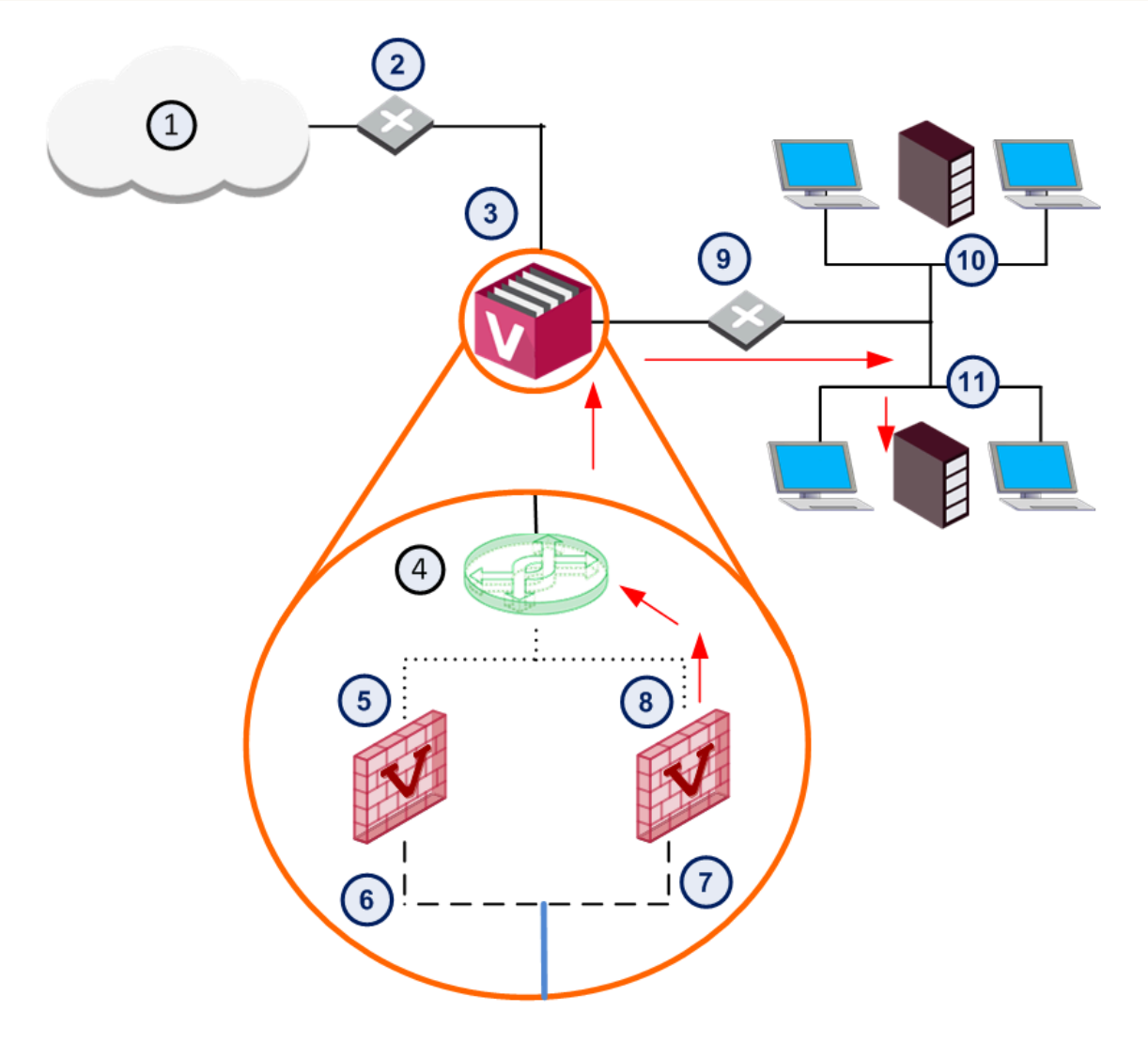
VSX Gateway는 트래픽을 세 단계로 처리합니다. 먼저 어느 컨텍스트로 보낼지 판별하고(Context Determination), 그 컨텍스트에서 보안 정책을 적용한 다음(Security Enforcement), 목적지로 전달합니다(Forwarding).

가장 중요한 첫 단계인 컨텍스트 판별을 이해하려면 VSX가 라우터의 VRF(Virtual Routing and Forwarding) 기술을 채택했다는 점을 알아야 합니다. VSX는 이 기술로 하나의 게이트웨이나 클러스터 위에 여러 독립 라우팅 도메인을 만드는데, 이 도메인들이 서로 독립적이기 때문에 IP가 겹치는 가상 장치도 쓸 수 있고, 바로 이 라우팅 도메인 하나하나를 컨텍스트(context)라고 부릅니다.

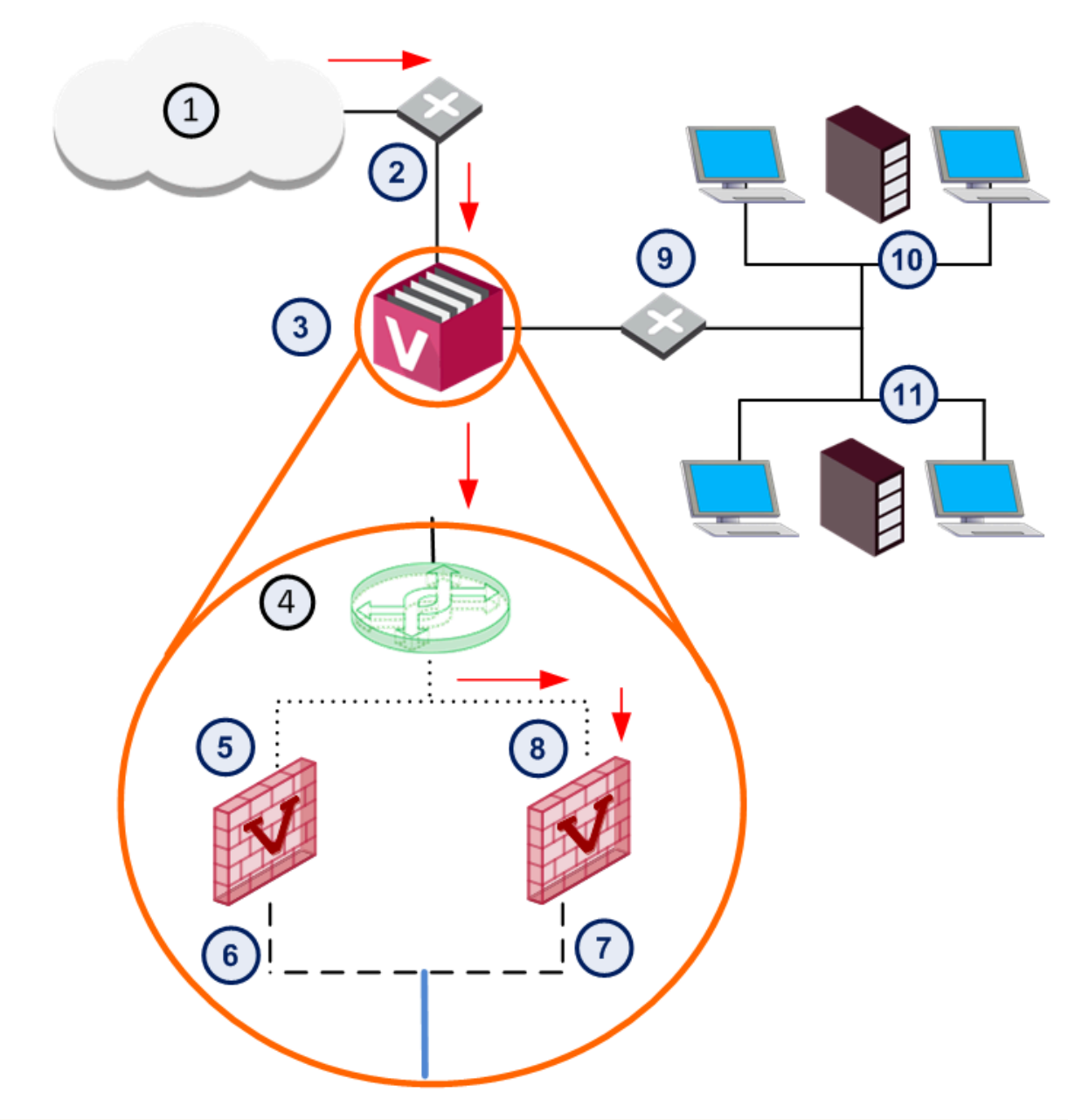
> *VSX incorporates VRF (Virtual Routing and Forwarding) technology that allows creation of multiple, independent routing domains on a single VSX Gateway or VSX Cluster. The independence of these routing domains makes possible the use of Virtual Devices with overlapping IP addresses. Each routing domain is known as a context.*
>
> — AdminGuide, "Context Determination" (p.62)

트래픽이 도착하면 컨텍스트 판별 과정이 그 패킷을 알맞은 VS나 VR, VSW로 보내는데, 어떻게 판별하는지는 연결 방식에 따라 셋으로 갈립니다.

VS가 물리 인터페이스나 VLAN 인터페이스에 직접 연결돼 있으면, 그 인터페이스 자체가 컨텍스트를 결정합니다. 아래 그림으로 따라가 보면, 인터넷(①)에서 라우터(②)를 지나 VSX Gateway(③)에 도착한 트래픽이 `eth1.200` (⑦)으로 들어오면, 그 VLAN ID가 정의하는 컨텍스트에 따라 자동으로 Virtual System 2(⑧)로 향합니다. 같은 트렁크라도 `eth1.100` (⑥)으로 들어온 것은 Virtual System 1(⑤)로 갈립니다.



① 인터넷 ② 라우터 ③ VSX Gateway ④ Virtual Switch ⑤ Virtual System 1 ⑥ eth1.100(VLAN 인터페이스)
⑦ eth1.200 ⑧ Virtual System 2 ⑨ VLAN 스위치 ⑩ VLAN 100 ⑪ VLAN 200



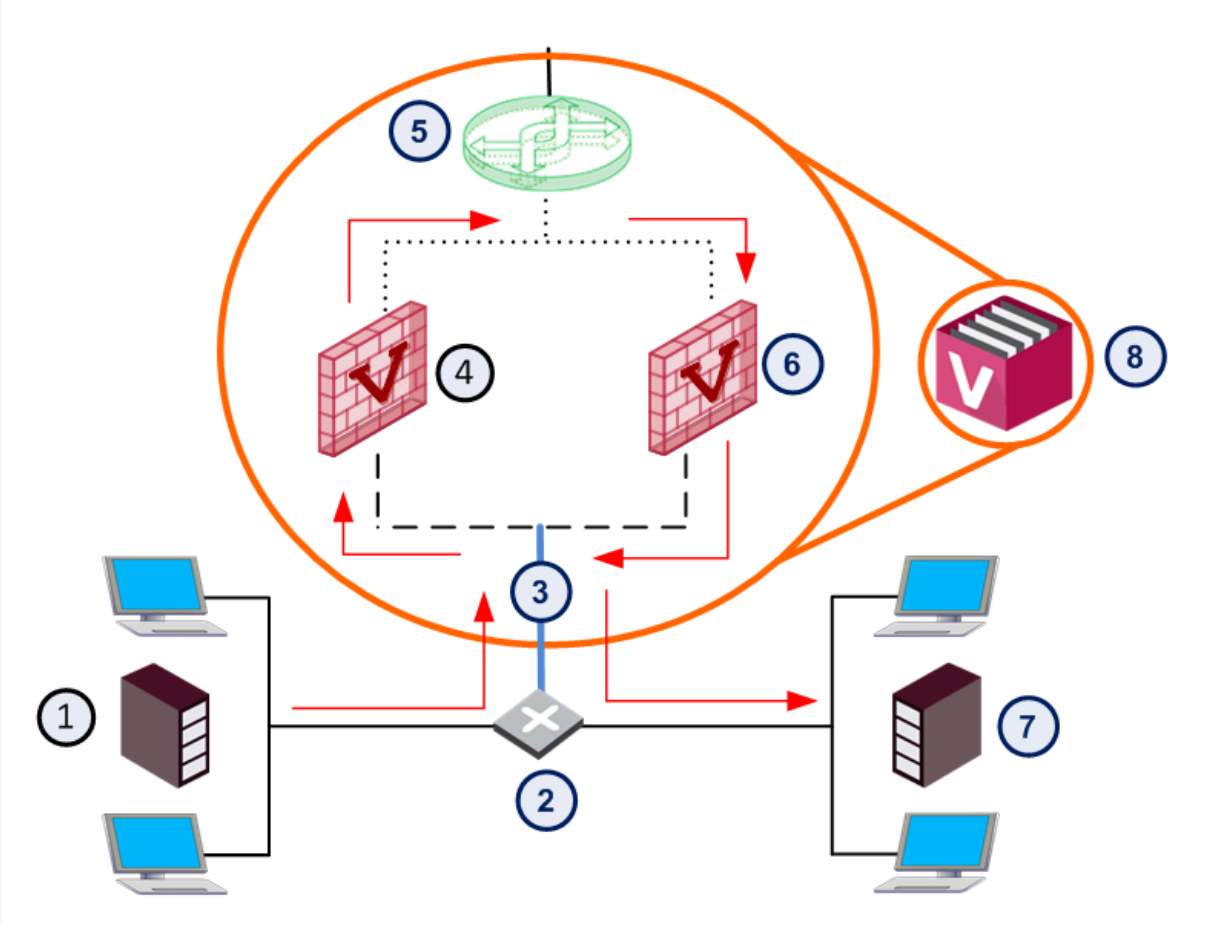
① 인터넷 ② 라우터 ③ VSX Gateway ④ Virtual Switch ⑤ MAC …:01 ⑥ Virtual System 1 ⑦ Virtual System 2 ⑧ MAC …:03 ⑨ VLAN 스위치 ⑩ VLAN 100 ⑪ VLAN 200

VS가 Virtual Switch를 거쳐 연결돼 있으면 목적지 MAC 주소로 판별하는데, VSW의 포워딩 테이블에서 MAC을 찾아 해당 VS의 Warp Link로 보내고, 만약 테이블에 없는 MAC이면 모든 Warp Link로 브로드캐스트합니다. 이 방식은 외부나 인터넷에서 들어오는 트래픽에서 흔히 볼 수 있습니다. 마지막으로 VS가 Virtual Router를 거쳐 연결돼 있으면 VR의 라우팅 테이블 항목으로 판별합니다. 이때 라우팅은 목적지 기반일 수도, 출발지 기반일 수도, 둘 다일 수도 있으며, 결정된 VS의 Warp Link로 트래픽이 도착합니다.

컨텍스트가 정해지면 그다음은 단순합니다. 각 VS는 독립된 Security Gateway이므로 자기만의 보안 정책으로 뒤의 네트워크를 보호하며, 지정된 VS가 모든 트래픽을 검사해 정책 규칙대로 허용하거나 차단합니다. 그렇게 통과된 트래픽은 각 VS가 가진 자기만의 설정과 규칙, 곧 NAT나 VPN 같은 처리를 거쳐 최종 목적지로 전달됩니다.

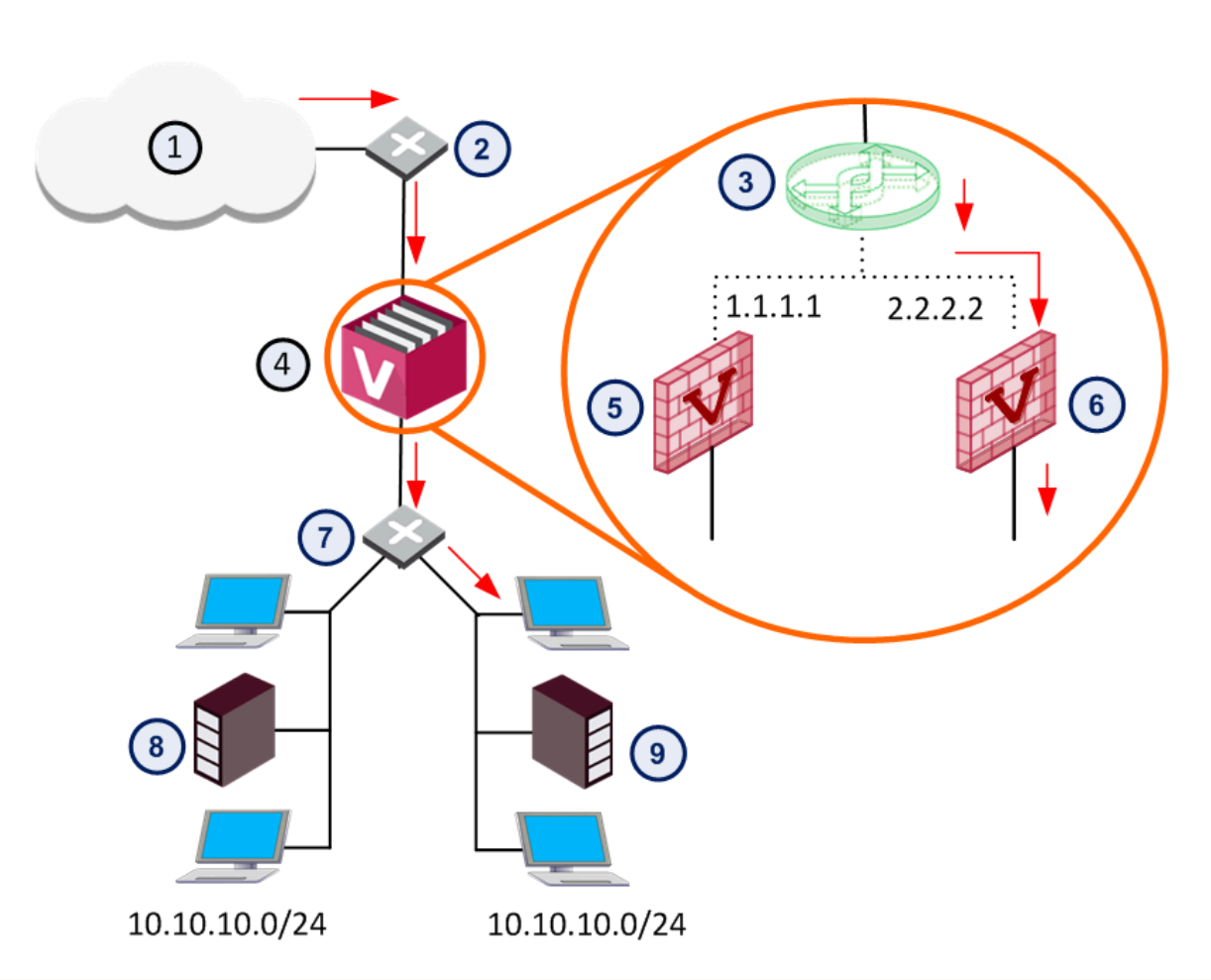
VSX의 라우팅

VSX의 라우팅은 물리 네트워크의 라우팅과 거의 똑같이 동작합니다. Virtual Router나 Virtual Switch를 통해 VS 뒤의 네트워크끼리 통신시킬 수 있는데, 가령 VLAN 100의 호스트가 VLAN 200의 서버로 데이터를 보내는 식입니다.



① VLAN 100 ② VLAN 스위치 ③ VLAN 트렁크 ④ Virtual System 1 ⑤ Virtual Switch ⑥ Virtual System 2 ⑦ VLAN 200 ⑧ VSX Gateway

여기서 VSX의 가장 독특한 성질이 나옵니다. 각 VS가 자기만의 상태 테이블과 라우팅 테이블을 갖기 때문에, 서로 다른 VS에 같은 IP 대역이 들어 있어도 충돌하지 않습니다. 같은 `10.0.0.0/24` 라도 VS1이 보는 것과 VS2가 보는 것이 전혀 다른 인터페이스와 연결을 가리키므로, 마치 물리 방화벽 두 대를 따로 둔 것과 똑같은 격리가 이뤄집니다.



① 인터넷 ② 라우터 ③ Virtual Switch ④ VSX Gateway ⑤ Virtual System 1 ⑥ Virtual System 2 ⑦ 스위치 ⑧ 네트워크 1 ⑨ 네트워크 2

VS는 NAT을 써서 이런 내부 IP를 외부 IP로 매핑해 바깥과 통신하며, Virtual Router는 동적 라우팅 프로토콜도 지원합니다.

이처럼 모든 것이 컨텍스트 단위로 분리돼 있기 때문에, 막상 운영에 들어가면 "지금 내가 어느 VS를 보고 있는가"가 늘 중요해집니다. 셸의 컨텍스트를 특정 VS로 바꾸는 `vsenv` 명령과, 컨텍스트를 혼동했을 때 생기는 운영상의 함정은 VSX 진단·트러블슈팅 챕터에서 자세히 다룹니다.

클러스터로 이중화하기

VSX를 이중화하면 두 가지 모드 중 하나를 쓰게 됩니다. High Availability는 Active 멤버가 죽으면 모든 세션이 중단 없이 Standby 멤버로 넘어가는 방식으로, 모든 VS가 한 멤버에서 Active이고 나머지 멤버는 대기합니다. 반면 VSLS(Virtual System Load Sharing)는 VS들을 여러 멤버에 나눠 배치해 부하를 분산하는 방식으로, 트래픽이 늘어나면 멤버를 추가해 확장할 수 있어 Maestro 환경에서 주로 씁니다. 두 모드의 자세한 설정은 VSX 클러스터 구성 챕터에서 다룹니다.

04 VSX 시작하기 — 전체 작업 순서

Getting Started with VSX

VSX를 처음 구축할 때 어떤 순서로 진행하는지 큰 그림을 잡는 챕터입니다. 세부 설정은 각각 해당 챕터에서 다루므로, 여기서는 무엇을 먼저 하고 무엇을 나중에 하는지 흐름만 짚겠습니다.

가장 먼저, 토폴로지를 설계한다

무엇보다 설치와 설정을 시작하기 전에 전체 VSX 토폴로지를 먼저 계획해야 합니다. **한 번 설정하고 나면 되돌리기 어려운 항목들**이 있기 때문입니다.

> *Plan the complete VSX topology before you start the installation and configuration, because it is not possible to change some settings after you configure them.*
>
> — AdminGuide, "Getting Started with VSX" (p.77)

설계 단계에서 미리 정해 둘 것은 크게 두 가지인데, 하나는 **단독 장비인 VSX Gateway로 갈지 이중화된 VSX Cluster로 갈지**이고, 다른 하나는 **Virtual Router를 쓸지** 여부입니다. VS들이 하나의 회선을 공유하거나 VS끼리 라우팅으로 통신해야 한다면 Virtual Router가 필요한데, 이 판단은 VSX 아키텍처와 핵심 개념 챕터를 참고해 내리면 됩니다.

Multi-Domain 환경이라면 여기에 한 가지를 더 정해야 합니다. 어느 Domain Management Server가 VSX Gateway를 관리하고, 어느 Domain이 그 안의 VS·VR·VSW를 관리할지를 미리 나눠 두어야 한다는 점입니다. 자세한 내용은 Multi-Domain Server와 함께 쓰기 챕터에서 다룹니다.

구축은 바깥에서 안으로

설계가 끝나면 **바깥 틀부터 세우고 안쪽을 채우는 순서**로 진행합니다. 먼저 관리 서버(SMS 또는 MDS)를 설치하고, 그다음 물리 장비에 VSX Gateway나 Cluster 멤버를 설치합니다. 이렇게 토대가 마련되면 이후 작업은 대부분 **SmartConsole에서 객체를 만들고 설정하는 일**입니다.

SmartConsole에서는 먼저 **VSX Gateway 또는 Cluster 객체**를 설정해 게이트웨이 본체를 정의합니다. 설계 단계에서 Virtual Router나 Virtual Switch를 쓰기로 했다면 이 공용 장치들의 객체를 이어서 만듭니다. 그런 다음 가상 방화벽인 **Virtual System 객체**들을 생성하고, 각 VS마다 방화벽·VPN·IPS 같은 **Software Blade**를 켜고 설정합니다. 마지막으로 VS별 **Access Control 정책**과 **Threat Prevention 정책**을 만들어 설치하면 구성이 완성되며, 이후에는 SmartConsole이나 SmartView에서 각 VS의 로그를 확인하며 운영합니다.

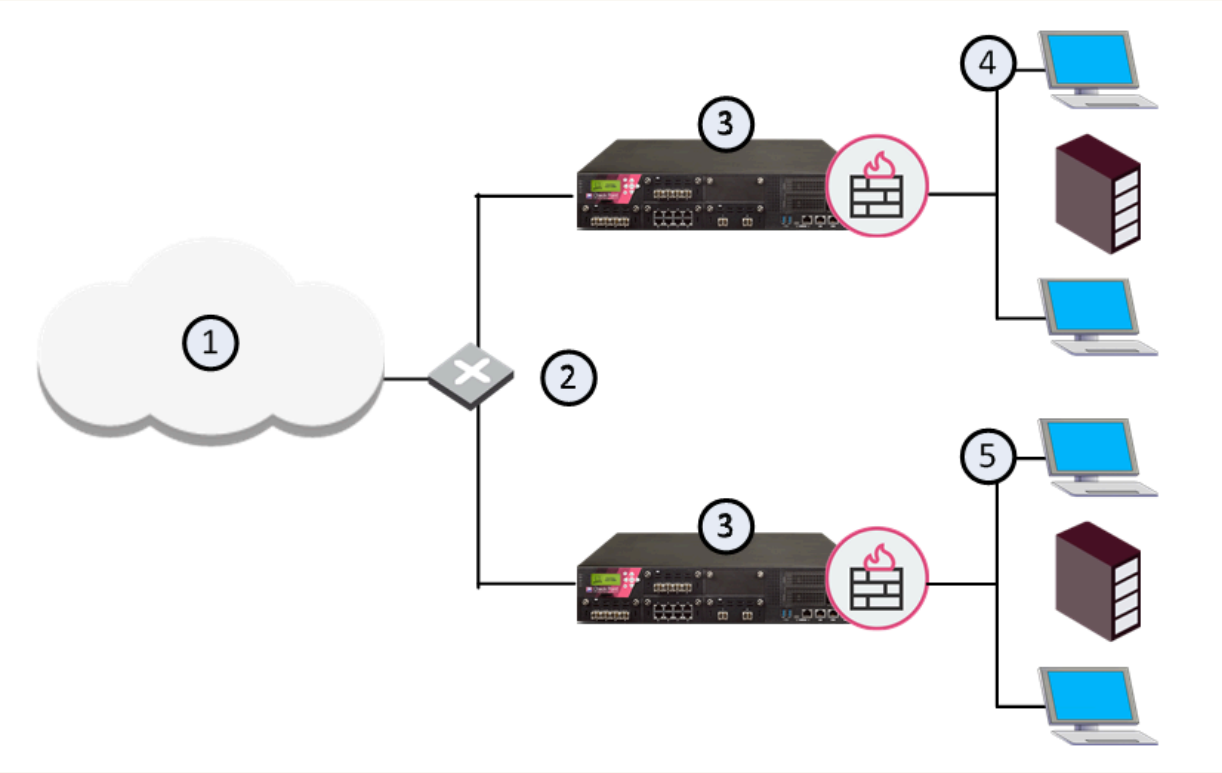
정리하면 관리 서버 → 게이트웨이 본체 → 공용 장치(VR/VSW) → 가상 방화벽(VS) → 정책 → 로그의 흐름입니다. 공용 장치인 Virtual Router와 Virtual Switch를 만드는 단계는 설계에서 쓰기로 한 경우에만 거치면 됩니다.

05 내부 네트워크 배포 전략

Deploying VSX - Internal Network Deployment Strategies

VSX 환경에서 Virtual System은 내부 네트워크를 보호합니다. 이 챕터는 그 내부망을 보호하는 네 가지 대표적인 배포 방식을 보여 주는데, 각 방식이 VSX의 서로 다른 특징을 부각합니다. 실제 현장에서는 이 방식들을 조합해 복잡한 기업 환경에 맞는 보안 구조를 만듭니다.

비교의 출발점으로 물리 네트워크를 떠올려 보면, 큰 물리 배포에서는 여러 Security Gateway가 각자 자기 내부망과 라우터에 물리적으로 연결되어 망 구간들을 보호합니다.

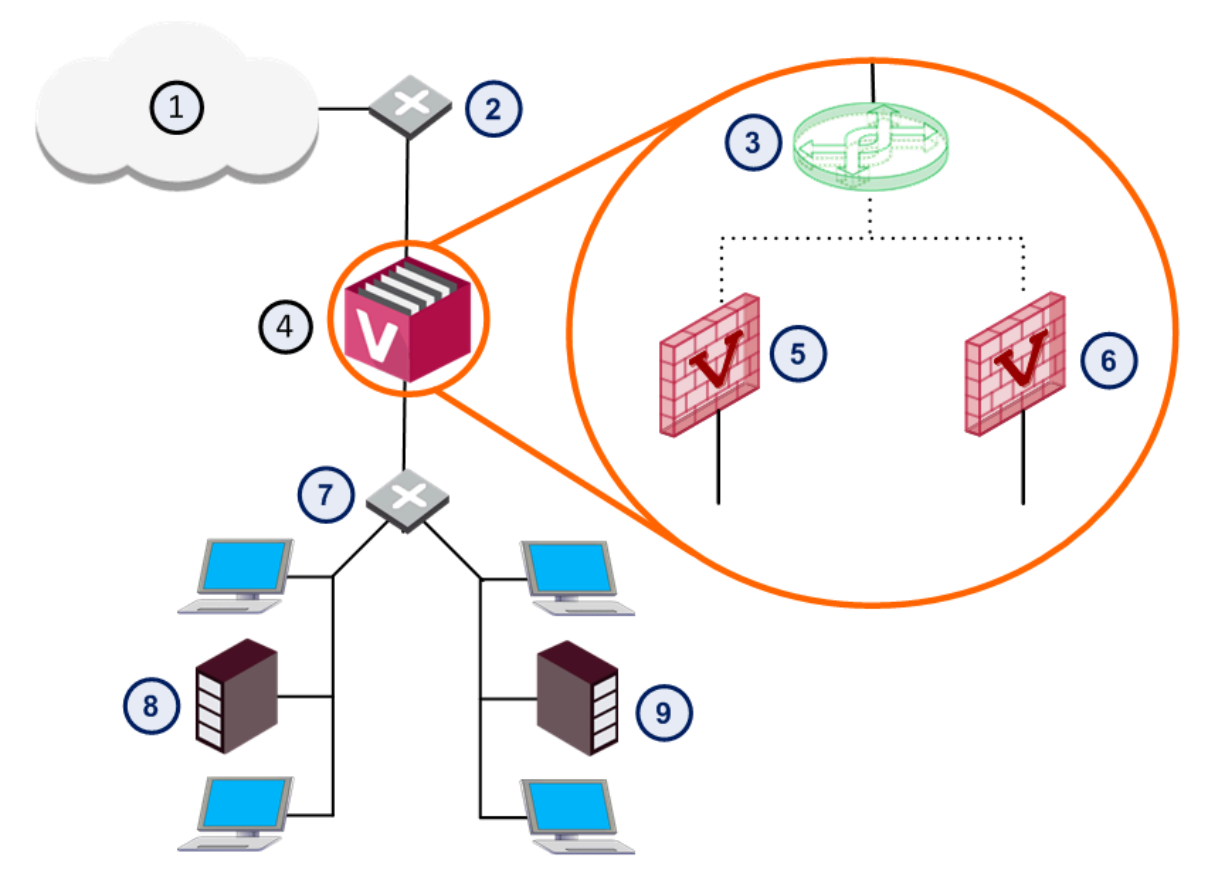


① 인터넷 ② 라우터 ③ Security Gateway들 ④ 네트워크 1 ⑤ 네트워크 2

VSX는 이 그림을 게이트웨이 한 대 안으로 옮기는 것이고, 옮기는 방법이 아래 네 가지입니다.

방식 1 — VS마다 전용 물리 인터페이스

가장 기본적인 구성입니다. 각 Virtual System이 VSX Gateway의 **물리 인터페이스를 통해 보호 대상 내부망에 직접 연결**되고, 내부망들 사이와 인터넷 쪽은 Virtual Switch가 이어 줍니다. 구조가 단순해서 적고 고정된 수의 내부망을 보호할 때 적합합니다.

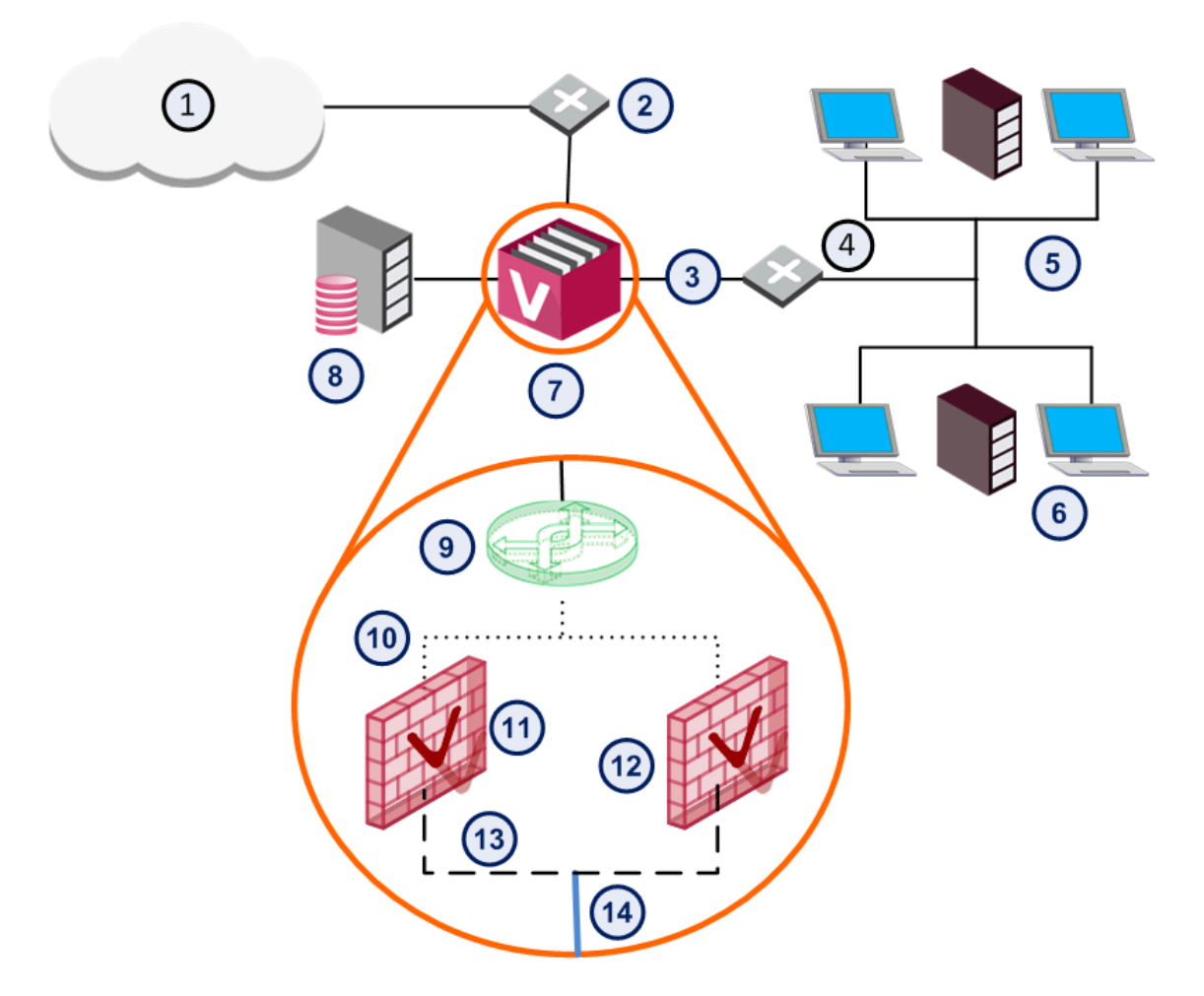


① 인터넷 ② 라우터 ③ Virtual Switch ④ VSX Gateway ⑤ Virtual System 1 ⑥ Virtual System 2 ⑦ 스위치 ⑧ 네트워크 1 ⑨ 네트워크 2

다만 분명한 단점이 있습니다. **보호할 네트워크마다 VSX Gateway에 전용 물리 인터페이스가 하나씩 필요**하다는 점입니다. 물리 포트 수는 한정돼 있으니, 많은 Virtual System이 필요한 환경에는 맞지 않습니다.

방식 2 — 내부 VLAN 인터페이스 (가장 일반적)

이 방식에서는 Virtual System들이 **VLAN 인터페이스를 통해** 내부 보호망에 연결됩니다. VSX Gateway는 802.1q VLAN 트렁크 하나로 VLAN 스위치에 연결되는데, 이 트렁크가 그 위를 지나는 모든 VLAN을 한데 묶어 나릅니다.

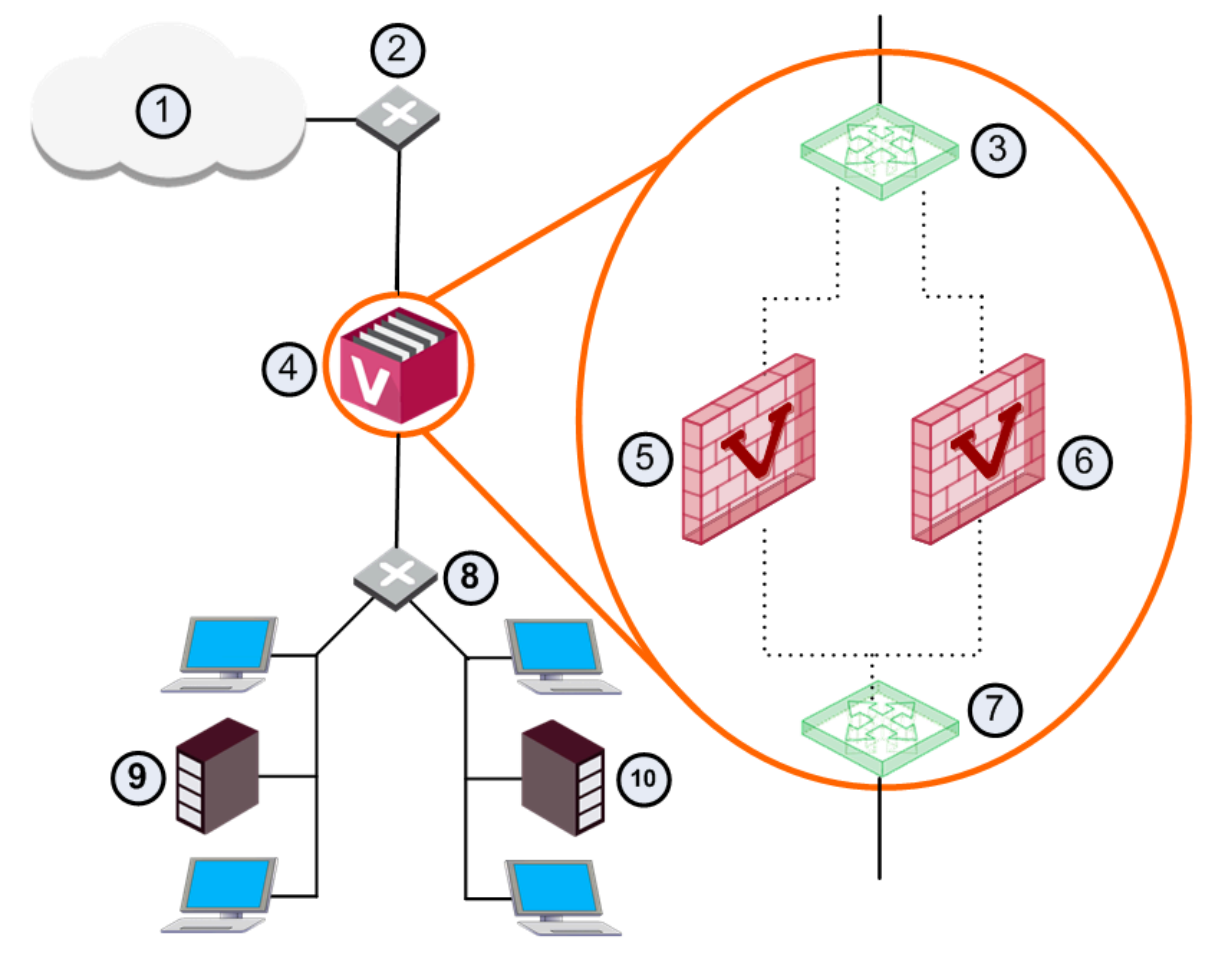


① 인터넷 ② 라우터 ③ 물리 인터페이스 ④ VLAN 스위치 ⑤ 네트워크 1 ⑥ 네트워크 2 ⑦ VSX Gateway ⑧ 관리 서버 ⑨ Virtual Switch ⑩ Warp 인터페이스 ⑪ Virtual System 1 ⑫ Virtual System 2 ⑬ VLAN 인터페이스 ⑭ VLAN 트렁크

이 구성은 하나의 VSX Gateway나 클러스터로 많은 Virtual System이 많은 내부망을 보호하는 환경에 알맞습니다. VLAN이 확장성과 세분성을 주기 때문에, **기존 IP 주소 구조를 바꾸지 않고도** Virtual System과 보호망을 빠르게 늘릴 수 있습니다. 실무에서 가장 흔히 쓰는 방식이 바로 이것입니다.

방식 3 — 내부 Virtual Router + 출발지 기반 라우팅

이 방식은 **VLAN 기술 없이 단일 물리 인터페이스만으로** Virtual System들을 보호망에 연결합니다. 핵심은 Virtual Router가 출발지 IP 주소를 보고 알맞은 Virtual System으로 트래픽을 보내는 출발지 기반 라우팅(source-based routing) 규칙을 쓴다는 점입니다. 각 Virtual System이 하나의 Virtual Router에 연결된 구성에서, VR에 출발지 기반 라우팅 규칙을 설정해 트래픽을 분배합니다.



① 인터넷 ② 라우터 ③ Virtual Switch ④ VSX Gateway ⑤ VS 1 ⑥ VS 2 ⑦ Virtual Router ⑧ 라우터 ⑨ 네트워크 1 ⑩ 네트워크 2

이 시나리오에는 몇 가지 전제가 있습니다. 각 Virtual System은 공인 IP로 Virtual Switch에 연결되고, Virtual Router에 연결된 각 로컬 네트워크는 사설 IP를 씁니다. 이 구성은 **IP 주소 중복을 지원하지 않으며**, 공유 내부 인터페이스에서 나온 패킷에 대해서는 Anti-Spoofing 보호가 동작합니다. 체크포인트는 내부 물리 라우터에서도 Anti-Spoofing 보호를 설정할 것을

권합니다. 그리고 앞서 여러 번 언급했듯 **Maestro·Scalable Chassis Security Group은 Virtual Router를 지원하지 않으므로**(Known Limitation 01413513) 이 방식 자체를 쓸 수 없습니다.

방식 4 — Bridge 모드의 Virtual System

마지막은 Bridge 모드입니다. Bridge 모드의 Virtual System은 IP 라우팅 대신 L2 브리징을 그대로 수행하며, 같은 VSX Gateway 위에서 L3 Virtual System과 공존할 수 있습니다.

> *A Virtual System in bridge mode implements native Layer 2 bridging instead of IP routing and can co-exist with Layer 3 Virtual Systems on the same VSX Gateway.*
>
> — AdminGuide, "Virtual Systems in Bridge Mode" (p.85)

이 방식의 가장 큰 장점은 **기존 IP 라우팅 구조를 건드리지 않고도** 기존 네트워크 토폴로지에 Virtual System을 투명하게 끼워 넣을 수 있다는 것입니다. 특히 대규모 클러스터 환경에 잘 맞습니다. Bridge 모드의 자세한 동작과 설정은 Bridge 모드 챕터에서 다룹니다.

06 조직 유형별 배포 전략

Deploying VSX - Organizational Deployment Strategies

앞 챕터가 내부망을 보호하는 기술적 방식이었다면, 이 챕터는 조직의 성격에 따라 VSX를 어떻게 배치하는지를 큰 그림으로 보여 줍니다. 다루는 조직은 세 가지인데, 대기업과 매니지드 서비스 제공업체(MSP), 그리고 데이터센터입니다. 각 경우 VSX가 내부와 경계 양쪽에서 어떻게 보안을 제공하는지가 핵심입니다.

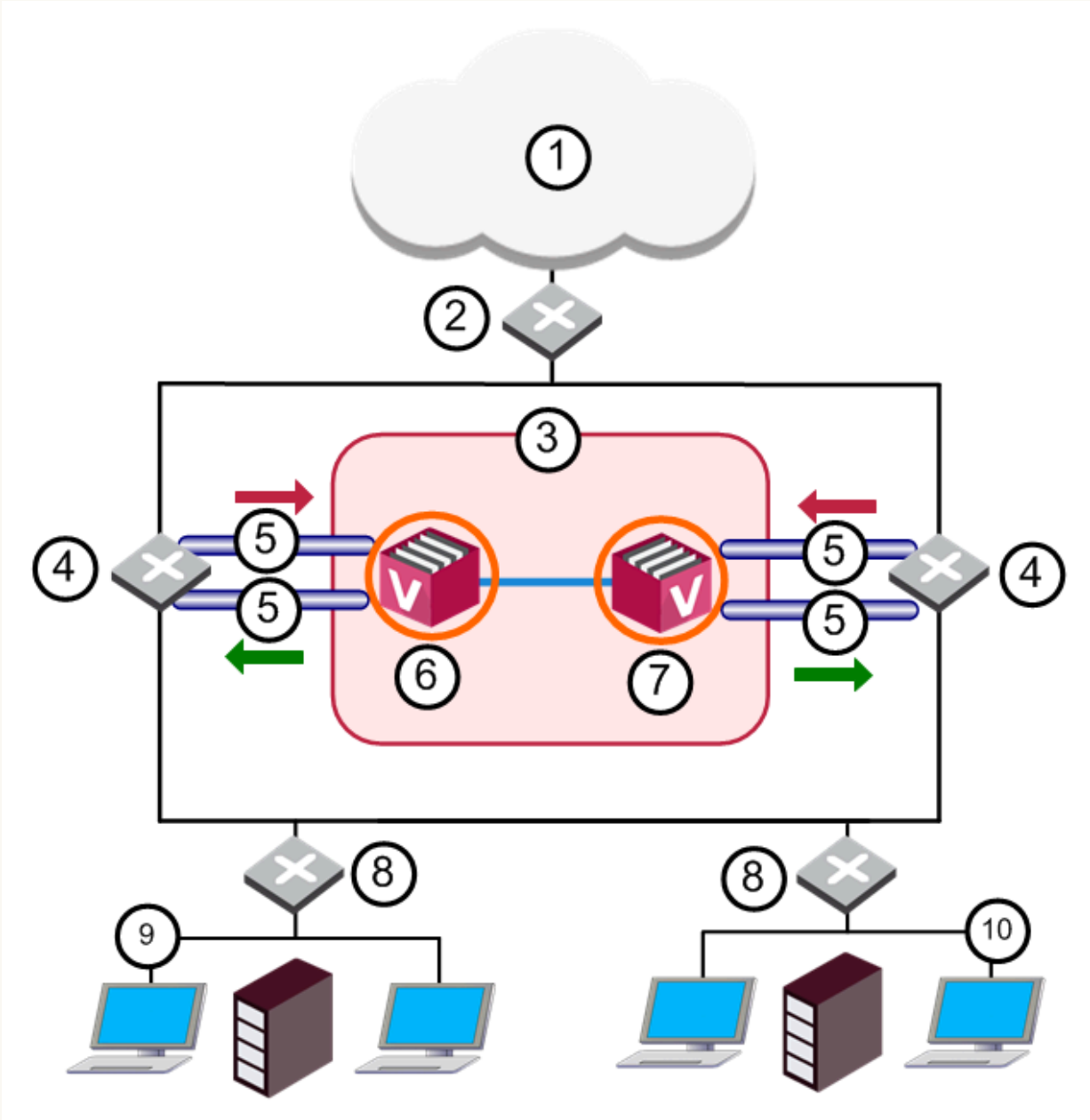
대기업 — 코어 네트워크 보안

대기업은 보통 전 세계 여러 곳에 흩어진 다양한 네트워크를 가지며, 부서와 지사마다 보안·접근 요구가 다릅니다. 그래서 보안을 중앙에서 관리하면서도 처리 성능을 유지하는 것이 결정적인 요구사항이 됩니다.

많은 대기업이 코어 네트워크를 중심으로 구성되는데, VSX는 **코어 백본 스위치 옆에 자리 잡아** L2나 L3, 또는 둘 다로 내부망을 보호합니다.

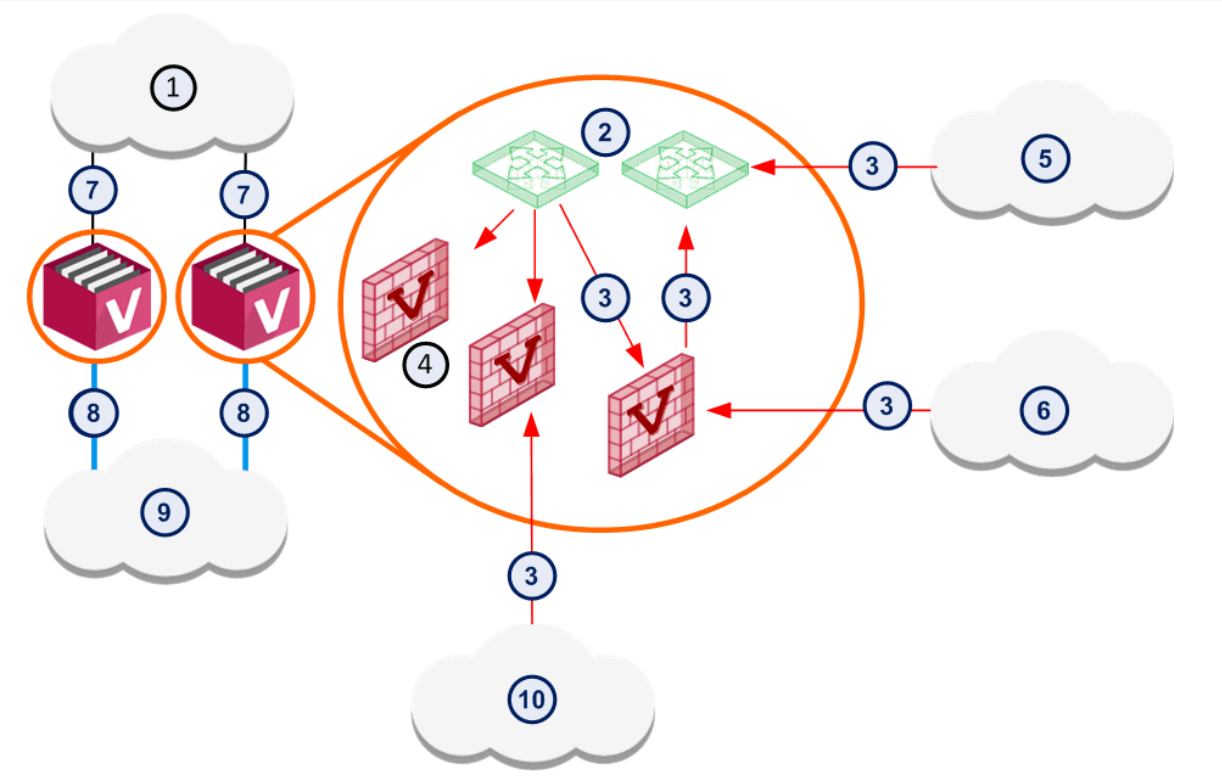
> *Situated adjacent to core network backbone switches, VSX protects the internal network by providing security at Layer 2, Layer 3 or both.*
>
> — AdminGuide, "Core Network Security" (p.86)

특히 **Bridge 모드의 Virtual System**을 쓰면 네트워크를 분할하지 않으면서도 부서별 네트워크를 보호할 수 있어, 각 부서 입구에 스위치를 두는 구성이 가능합니다. 이렇게 VSX가 코어 네트워크와 인터넷·외부망 사이의 연결을 보장하면서 경계 보안을 제공하며, 보안은 VLAN 단위로 설정할 수 있습니다.



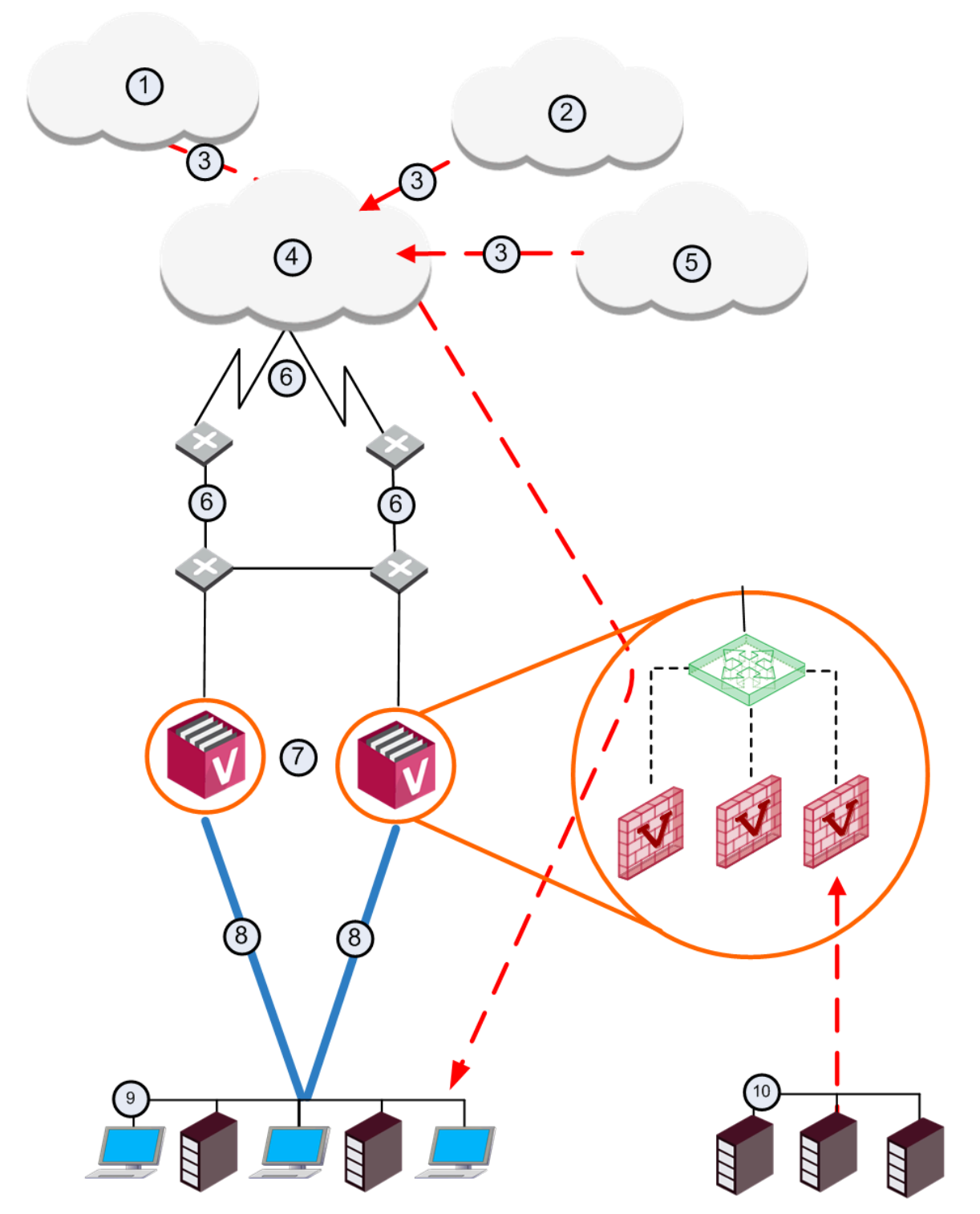
① 인터넷 ② 코어 백본 스위치 ③ VSX Cluster ④ 라우터 ⑤ VLAN ⑥ 멤버 1 ⑦ 멤버 2 ⑧ LAN 스위치 ⑨ 영업부 ⑩ 재무부

대기업이 OSPF나 BGP 같은 **동적 라우팅 프로토콜**을 쓰는 환경이라면, VSX는 DMZ 서비스와 VPN 피어, 도메인, 파트너 네트워크를 보호합니다. 예를 들어 라우팅되는 코어에서 일어나는 BGP 이웃 업데이트를 애플리케이션 네트워크로 선택적으로 재분배하고, OSPF로 Virtual Router·Virtual System·코어·애플리케이션 네트워크 사이의 연결을 제공하는 식입니다.



① 인터넷 ② Virtual Router들 ③ OSPF ④ Virtual System들 ⑤ Extranet ⑥ 파트너 네트워크 ⑦ BGP ⑧ OSPF 802.1q ⑨ 라우티드 코어의 BGP ⑩ DMZ

경계 보안 측면에서는 각 VLAN마다 보안을 적용하고, OSPF와 BGP가 경계를 따라 여러 보안 구역으로의 연결을 제공합니다. 이 시나리오에서는 파트너가 Virtual System을 통해 원격으로 네트워크 자원에 접근하는데, 각 Virtual System이 자기 요구에 맞는 보안 정책을 갖고, 파트너별 로그와 감사 정보가 따로 수집되어 별도 DB에 저장됩니다. 애플리케이션과 서비스는 전용 Virtual System으로 분리되고, 여러 Virtual Router와 Virtual Switch가 접근 경로를 통제합니다.

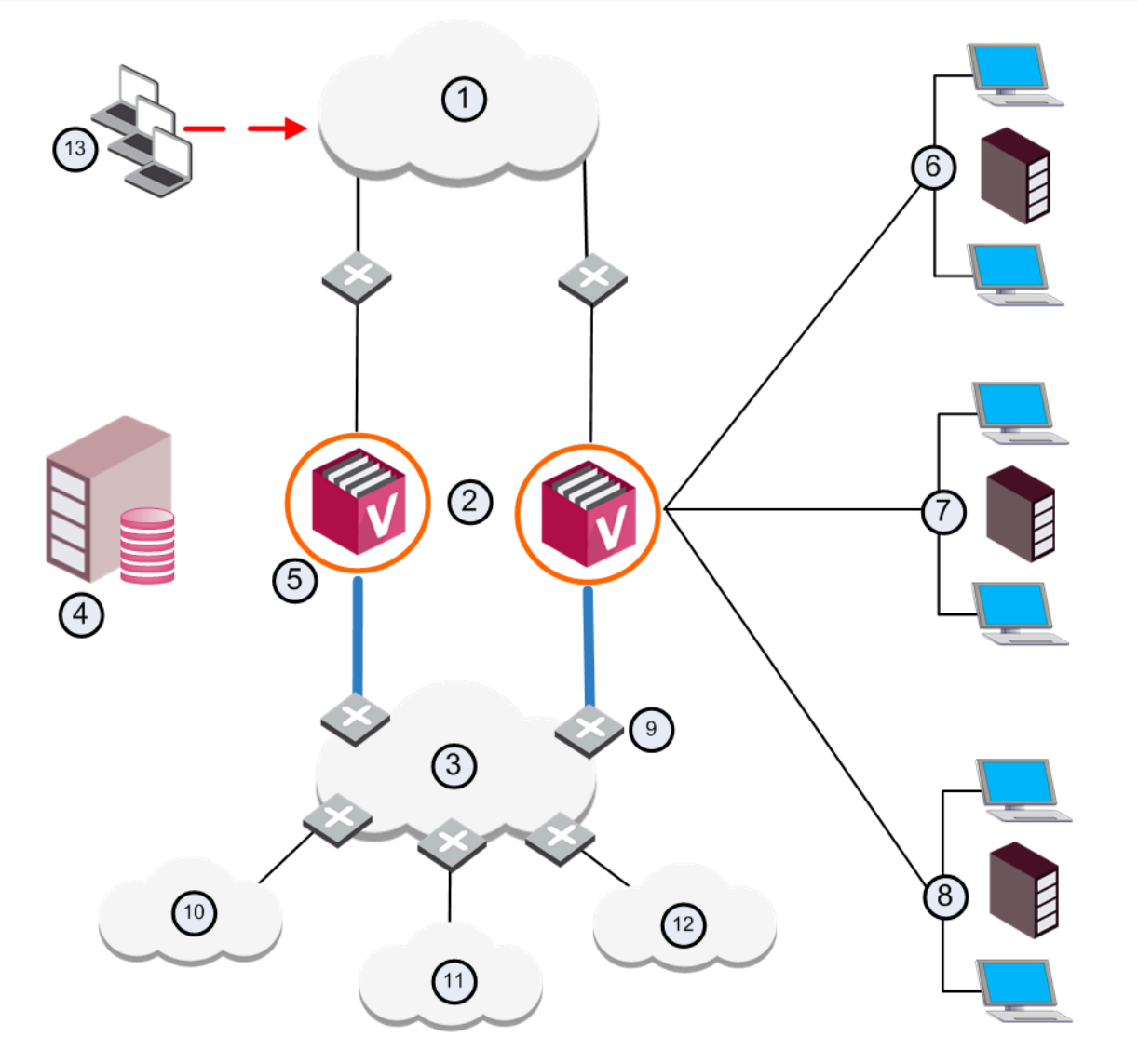


① 파트너 접속 ② 고객 ③ IPsec 터널 ④ 인터넷 ⑤ 파트너 ⑥ BGP ⑦ VSX Cluster ⑧ 802.1q VLAN 트렁크 ⑨ 내부 네트워크 ⑩ DMZ

매니지드 서비스 제공업체 (MSP)

MSP는 여러 도메인 네트워크에 연결성과 보안 서비스를 제공하며, 일부 도메인은 원격 접근까지 필요로 합니다. 이런 서비스 중심 환경에서 VSX와 Multi-Domain Server는 기존 IP 토폴로지를 건드리지 않으면서 중앙 관리와 손쉬운 연결·보안을 제공합니다.

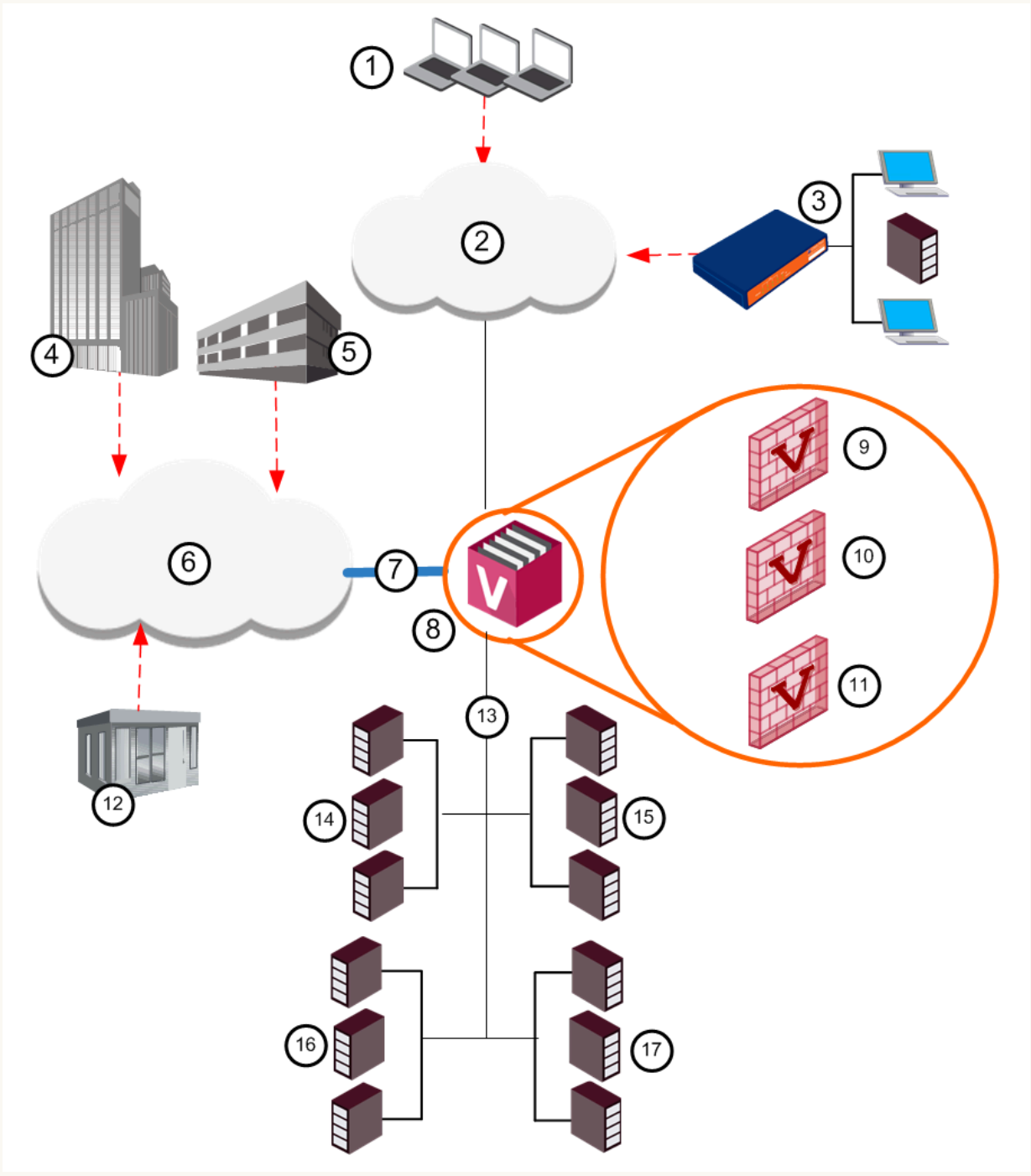
대표적인 시나리오는 서비스 제공업체의 POP(Point of Presence)에 VSX Cluster를 두는 것입니다. VSX가 서비스 제공업체의 하드웨어를 통합하고 사용자에게 VPN으로 프라이버시와 보안 연결을 보장하며, 이 구성은 High Availability와 VSLS 클러스터 모드 모두에 적합합니다. **애플리케이션과 서비스는 개별 Virtual System으로 분리**되고 접근은 필요에 따라 허용됩니다. 운영센터(NOC)의 Multi-Domain Server가 POP를 모니터링하며 VSX Gateway에 연결되고, Multi-Domain Log Server가 도메인마다 데이터를 모아 각각 별도의 비공개 DB에 로그를 저장한다는 점이 이 모델의 특징입니다.



① 인터넷(라우터가 클러스터 멤버와 인터넷 사이) ② VSX Cluster ③ Core IP VPN 네트워크 ④ NOC의 Multi-Domain Server ⑤ NOC의 MDS와 VSX Gateway가 만드는 Local Exchange ⑥ Domain A 웹 서버 ⑦ Domain B DMZ ⑧ Domain C 메일 서버 ⑨ PE 라우터 ⑩ 원격 접속

데이터센터

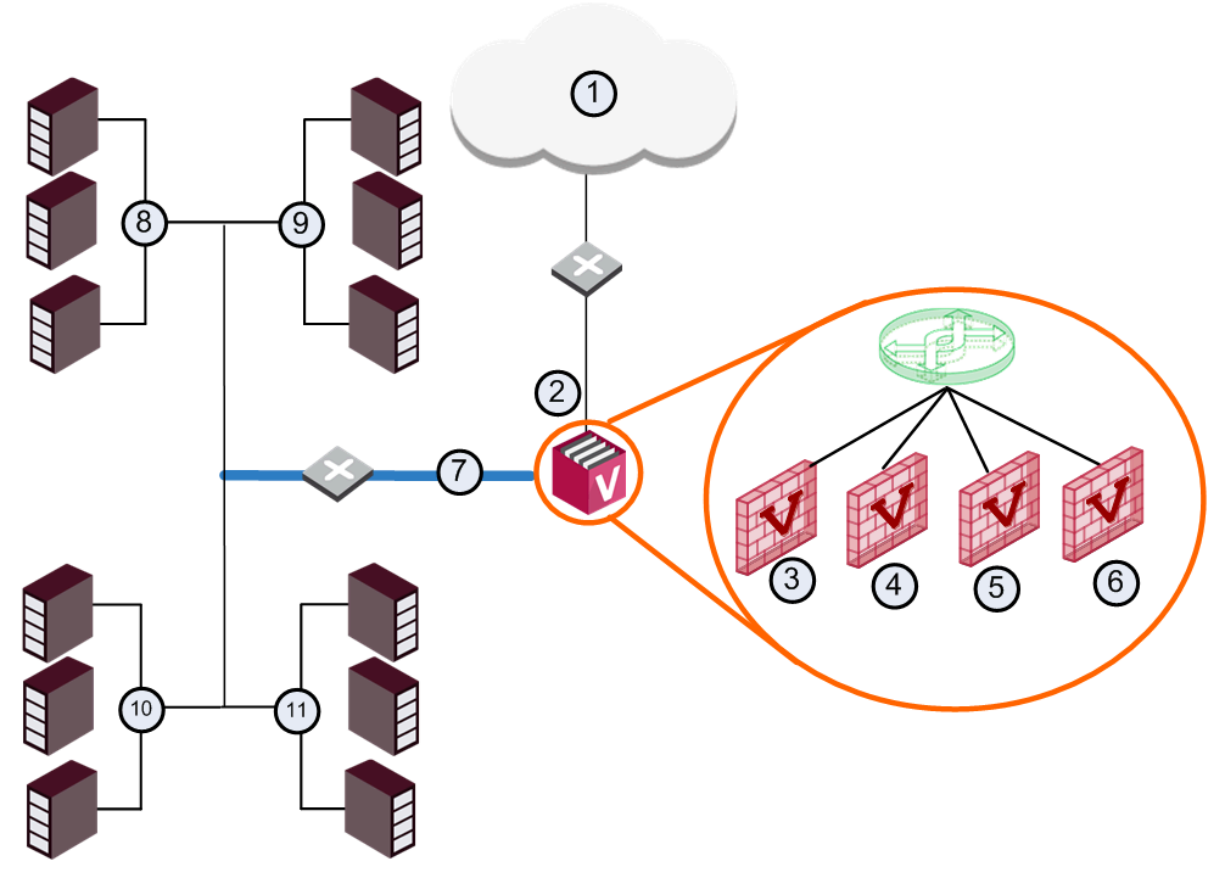
데이터센터 사업자는 여러 도메인의 서버와 데이터베이스에 호스팅 서비스를 제공하며, 보통 인프라·연결·보안을 함께 묶어 줍니다. 예컨대 여러 도메인 네트워크가 공통 물리 인프라를 공유하고 백본이 각 도메인과 데이터센터를 잇는 가운데, A 도메인은 웹 호스팅 서버에, B 도메인은 메일 서버에, C 도메인은 DB 서버에 연결되는 식입니다. 이때 데이터센터 사업자는 도메인마다 Virtual System을 하나씩 둔 VSX Gateway를 배치해 보안과 관리를 해결합니다.



① 원격 사용자 ② 인터넷 ③ VPN-1 Edge 뒤 원격 네트워크 ④ 고객 B ⑤ 고객 A ⑥ MPLS 백본 ⑦ 802.1q ⑧ VSX Gateway ⑨·⑩ 고객 A용 Virtual System ⑪ 고객 C용 Virtual System ⑫ 고객 C ⑬ 데이터센터 ⑭ 고객 A 웹 서버(각각 별도 VLAN ID) ⑮ 고객 B 메일 서버 ⑯ 고객 A Extranet ⑰ 고객 C 데이터베이스

기업 내부의 데이터센터에서도 마찬가지입니다. VSX는 Virtual System에 L2 연결을 배정해 물리적으로 관리해야 하는 장비 수를 줄이면서도 동일한 수준의 보안을 제공합니다. 핵심 목표는 접근 권한과 보안 요구가 서로 다른 공유 자원을, 망을 세분화하면서 보호하는 것입니다. 예를 들어 한 Virtual System은 SQL 취약점으로부터 데이터베이스를 보호하고, 다른

Virtual System은 IPS로 웹 서버를 보호합니다. 그리고 새 애플리케이션이나 서비스가 추가될 때마다 그 요구에 맞는 새 Virtual System을 손쉽게 만들어 보호할 수 있다는 점이 이 방식의 확장성입니다.



① 인터넷 ② VSX Gateway ③ VLAN 02 ④ VLAN 03 ⑤ VLAN 04 ⑥ VLAN 05 ⑦ VLAN 트렁크 ⑧ 웹 서버 VLAN 02 ⑨ 데이터베이스 VLAN 03 ⑩ 애플리케이션 A VLAN 04 ⑪ 애플리케이션 B VLAN 05

07 VSX 구성하기

Configuring VSX

이 챕터는 SmartConsole로 VSX의 가상 장치들을 만들고 설정하고 관리하는, 구성 작업의 중심입니다. 분량이 많은 만큼 주제별로 나눠 풀어 설명합니다. 한 가지 전제는, 가상 장치와 정책 작업의 상당수가 물리 Security Gateway와 동일하다는 점입니다. 그래서 표준 게이트웨이 객체나 정책을 만드는 일반 절차는 이 가이드에서 다루지 않고, VSX에 특화된 부분만 설명합니다. MDS 환경이라면 가상 장치를 관리하는 Domain Management Server에 SmartConsole로 접속해 작업합니다 (Multi-Domain Server와 함께 쓰기 참고).

VSX Gateway 만들기 — 5단계 마법사

새 VSX Gateway는 VSX Gateway Wizard로 만듭니다. 먼저 알아둘 제약은 기존 Security Gateway를 VSX Gateway로, 또는 그 반대로 변환할 수 없다는 것입니다. SmartConsole에서 New > VSX > Gateway로 마법사를 시작하면 다섯 단계를 거칩니다.

1단계 **General Properties**에서는 VSX Gateway 이름(공백·특수문자 불가, 밑줄만 허용)과 관리 인터페이스 주소, 설치된 VSX 버전을 정합니다(IPv6를 주면 IPv4도 함께 줘야 합니다). 2단계 **SIC 신뢰 설정**에서는 Gaia 최초 설정 때 입력한 활성화 키를 똑같이 입력하고 Initialize를 누르는데, 키가 맞으면 Trust State가 "Trust established"로 바뀝니다. 신뢰가 안 맺어지면 Check SIC Status로 원인을 보는데, 대개 활성화 키 오류나 관리 서버-게이트웨이 간 연결 문제입니다.

3단계 **물리 인터페이스 정의**에서는 어느 인터페이스를 VLAN 트렁크로 쓸지 지정하고, 4단계 **가상 네트워크 장치 구성**에서는 게이트웨이와 인터페이스를 공유하는 가상 장치를 정의할 수 있습니다. 여기서 DMI냐 Non-DMI냐가 결정되는데, 이 설정은 마법사를 마친 뒤에는 바꿀 수 없습니다. 공유 가상 장치를 정의하지 않으면 기본적으로 DMI VSX Gateway가 만들어지며(Non-DMI는 폐기됨), 5단계 **VSX Gateway Management**에서는 게이트웨이 자신을 보호하는 정책 규칙을 정합니다. 이 정책은 오직 게이트웨이로 향하는 트래픽에만 적용되고 SNMP·SSH·ICMP(ping)·HTTPS 서비스에 대한 사전 정의 규칙으로 구성되는데, 기본적으로 모두 차단이라 예컨대 관리 서버에서 ping하려면 ICMP를 허용해야 합니다. 마법사를 마치면 완료까지 몇 분 걸릴 수 있습니다.

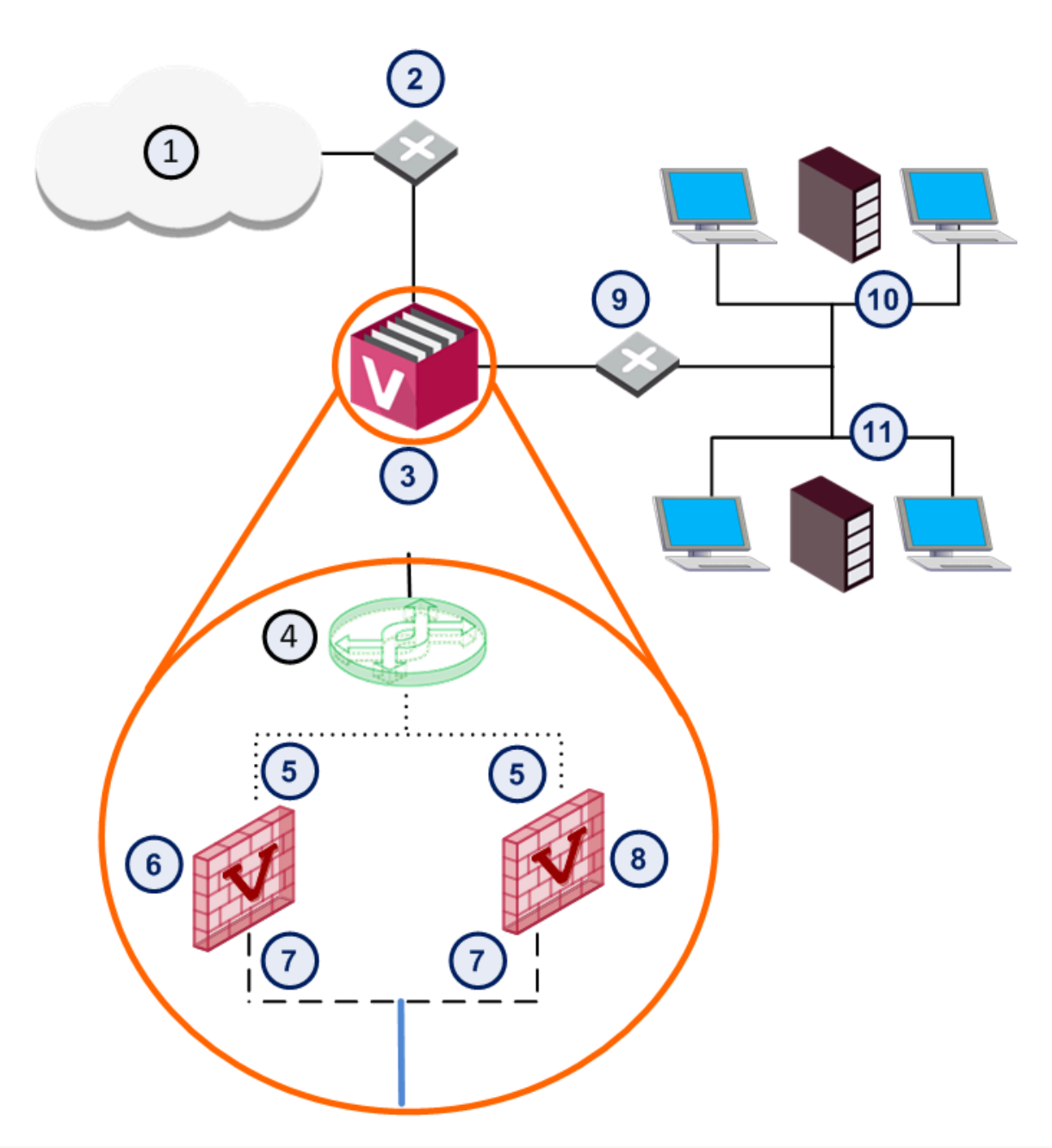
VSX Gateway 관리하기

만든 뒤에는 객체를 더블클릭해 VSX Gateway Properties에서 토폴로지와 여러 속성을 바꿉니다. General Properties에서 SIC를 확인·재설정하고 이 게이트웨이에 설치할 Software Blade를 고릅니다. SIC를 리셋할 때는 게이트웨이 CLI에서 `cpconfig`로 SIC를 재초기화하고 Communication 창에서 Reset 후 다시 Initialize한 다음, `<게이트웨이명>_VSX` 정책을 설치하고 게이트웨이에서 `cpstop;cpstart`를 실행합니다.

Physical Interfaces 페이지에서 물리 인터페이스를 추가·제거하거나 VLAN 트렁크를 지정하고, Topology 페이지에서 인터페이스와 라우트를 정의합니다. 라우트를 추가할 때 Propagate route to adjacent Virtual Devices를 켜면 이웃 가상 장치에 경로를 알려 연결을 활성화하며, 토폴로지 자동 계산 옵션은 기본 켜짐이지만 동적 라우팅을 쓸 때는 끄는 것이 권장되고 Bridge 모드에서는 쓸 수 없습니다. VSX Gateway 객체를 삭제하면 그에 딸린 모든 Virtual System과 가상 장치가 관리 DB에서 함께 삭제됩니다. 장애에 대비한 백업·복원은 sk100395를 따릅니다.

Virtual System 만들고 수정하기

Virtual System은 Virtual Systems Wizard로 만듭니다. 전형적인 VS는 외부망·DMZ·인터넷으로 가는 외부 인터페이스와, 보통 VLAN 트렁크로 내부망에 닿는 내부 인터페이스, 이렇게 두 개를 가집니다.



① 인터넷 ② 라우터 ③ VSX Gateway ④ Virtual Switch ⑤ 외부 인터페이스 ⑥ Virtual System 1 ⑦ 내부 인터페이스 ⑧ Virtual System 2 ⑨ VLAN 스위치 ⑩ 네트워크 1 ⑪ 네트워크 2

마법사의 General Properties에서 이름과 호스팅 VSX Gateway를 고르고 필요하면 Bridge Mode를 선택합니다. Network Configuration에서는 내·외부 인터페이스와 내부 인터페이스 뒤의 IP 토폴로지를 정의하는데, Main IP Address는 보통 외부 인터페이스에 배정되며 NAT나 VPN 연결에 쓰이는 VS 주소가 됩니다. 만든 뒤에는 Access Control 정책을 설치해야 합니다.

수정은 객체를 더블클릭해 합니다(이름만은 바꿀 수 없습니다). Topology 페이지에서 인터페이스를 추가할 때 Regular(물리), Leads to Virtual Router, Leads to Virtual Switch, Bridge 중에서 고르며, 이 인터페이스 설정을 바탕으로 VSX가 가상 장치와 게이트웨이로 가는 라우트를 자동 생성합니다. 클러스터 환경에서 특정 VS의 토폴로지를 바꾸면 그 VS에 정책을 설치해야 클러스터 토폴로지가 갱신된다는 점에 주의합니다. VPN을 켰다면 VPN Domain에서 그 VS 뒤의 어떤 호스트들이 VPN 터널로 통신할지 정하고, NAT > Advanced에서 그 VS에서 나오는 패킷에 Hide NAT이나 Static NAT을 설정합니다.

R81.10부터는 일반(L3) Virtual System에도 브리지 인터페이스를 추가할 수 있고, IP 없이 브리지 인터페이스만 가진 VS도 만들 수 있습니다. Topology에서 New > Bridge를 골라 두 종속 인터페이스(예: eth2, eth3)를 차례로 지정하면 됩니다. Bridge 모드의 동작 원리와 Multi Bridge는 Bridge 모드 장에서 자세히 다룹니다.

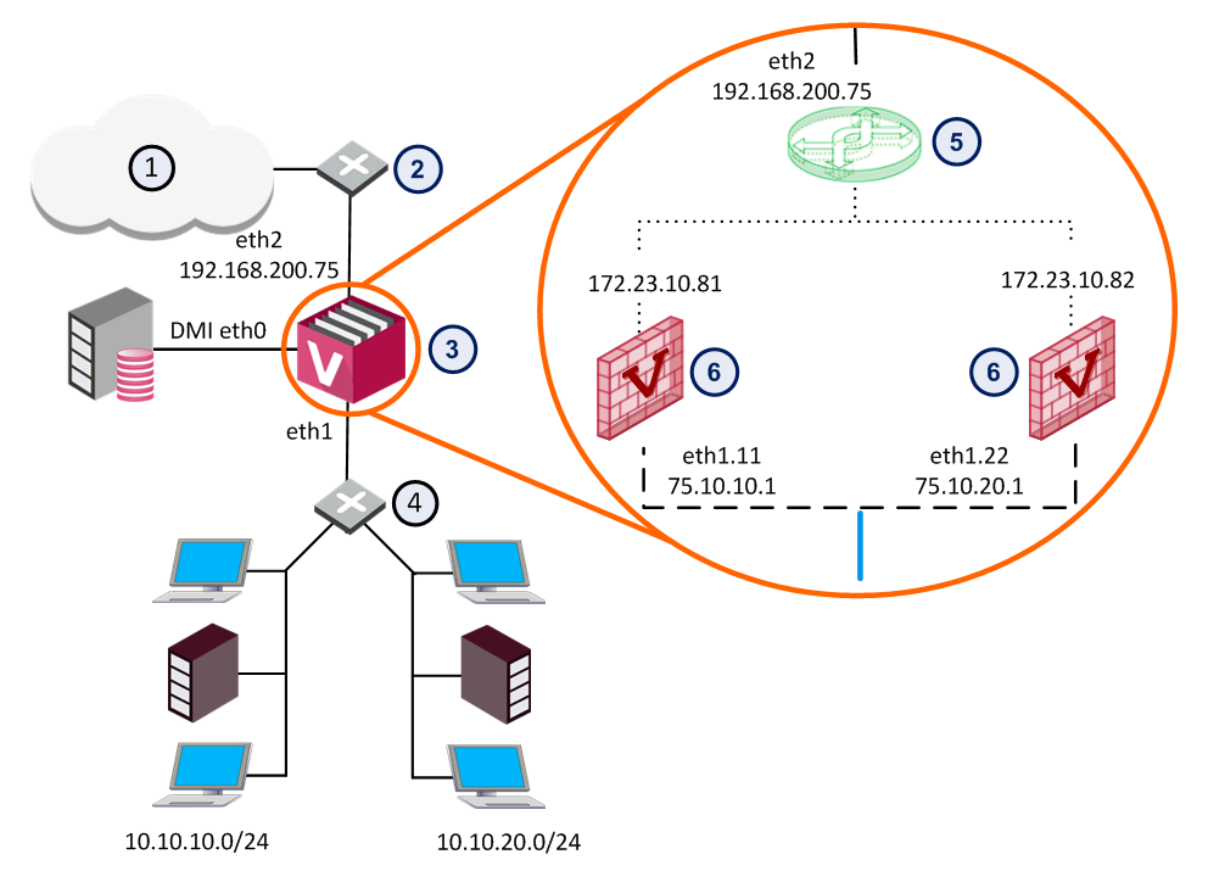
Virtual System의 DNS와 DHCP

R82에서는 Virtual System마다 별도의 DNS 설정을 줄 수 있습니다. 게이트웨이 CLI(Gaia Clish, Scalable Platform은 gClish)에서 먼저 `set dns mode per-vs`로 기능을 켜고, `set virtual-system <VSID>`로 대상 VS 컨텍스트에 들어간 뒤 DNS 서버와 suffix를 설정합니다. 특정 도메인만 다른 DNS로 보내는 forwarding domain이나, VS를 DNS Relay로 동작시키는 listening-interface 설정도 같은 방식으로 추가합니다. DHCP 서버도 `set virtual-system <VSID>`로 컨텍스트를 바꾼 뒤 R82 Gaia Administration Guide의 DHCP 절차대로 설정합니다. 이처럼 CLI 작업은 늘 "지금 어느 VS 컨텍스트인가"가 중요한데, Expert 모드에서는 `set virtual-system` 대신 `vsenv`로 컨텍스트를 전환합니다.

```
set dns mode per-vs                 # VS별 DNS 기능 켜기
set virtual-system 1                # 대상 VS 컨텍스트
set dns primary 192.168.10.21
set dns suffix mycompany.com
add dns proxy forwarding-domain anothercompany.com   # 특정 도메인만 다른 DNS
add dns proxy listening-interface wrp128             # DNS Relay로 동작
save config
```

Virtual Switch와 Virtual Router

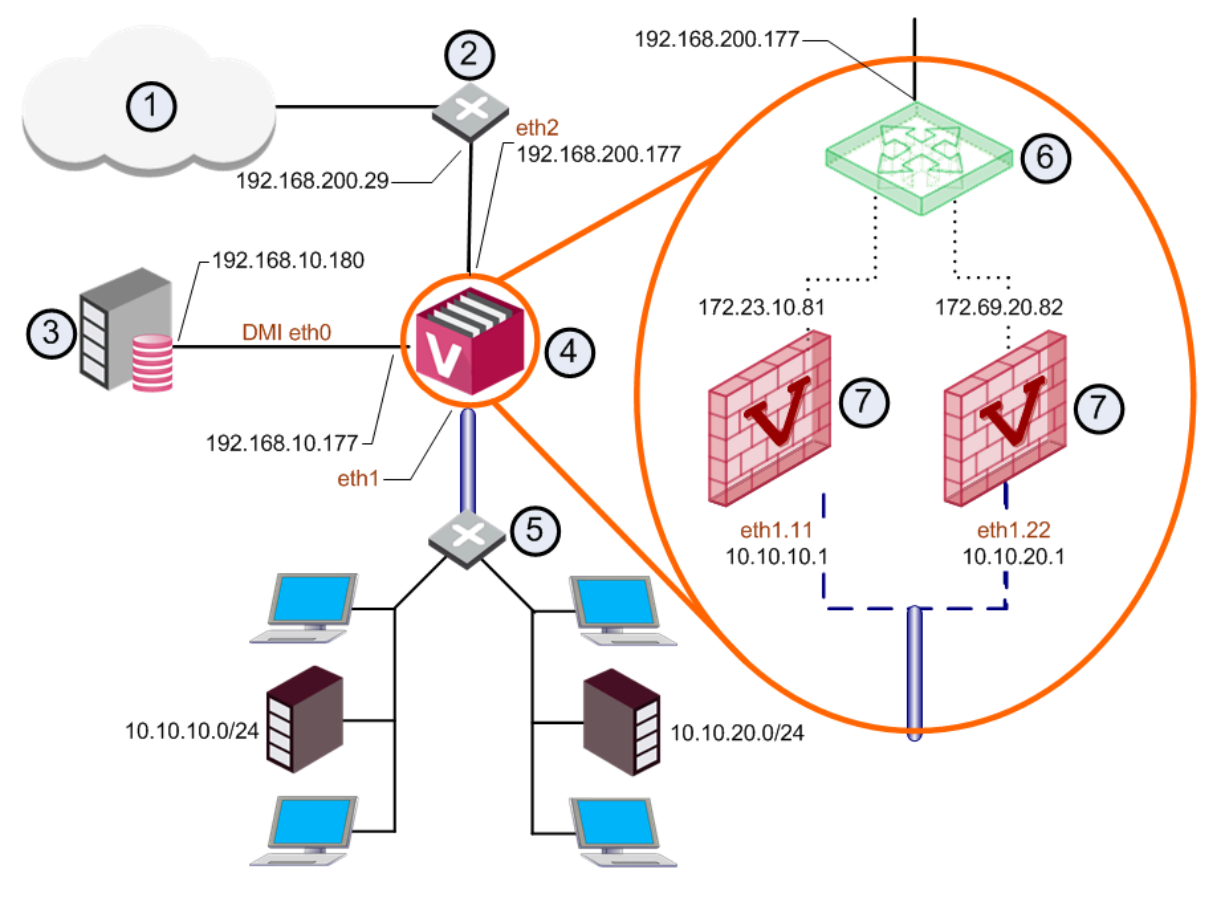
Virtual Switch는 VS들과 내·외부 네트워크 사이에 L2 연결을 제공하며 물리 스위치처럼 포워딩 테이블을 유지합니다. Virtual Switch Wizard로 이름과 호스팅 게이트웨이를 정하고 연결할 인터페이스를 추가하면 만들어집니다.



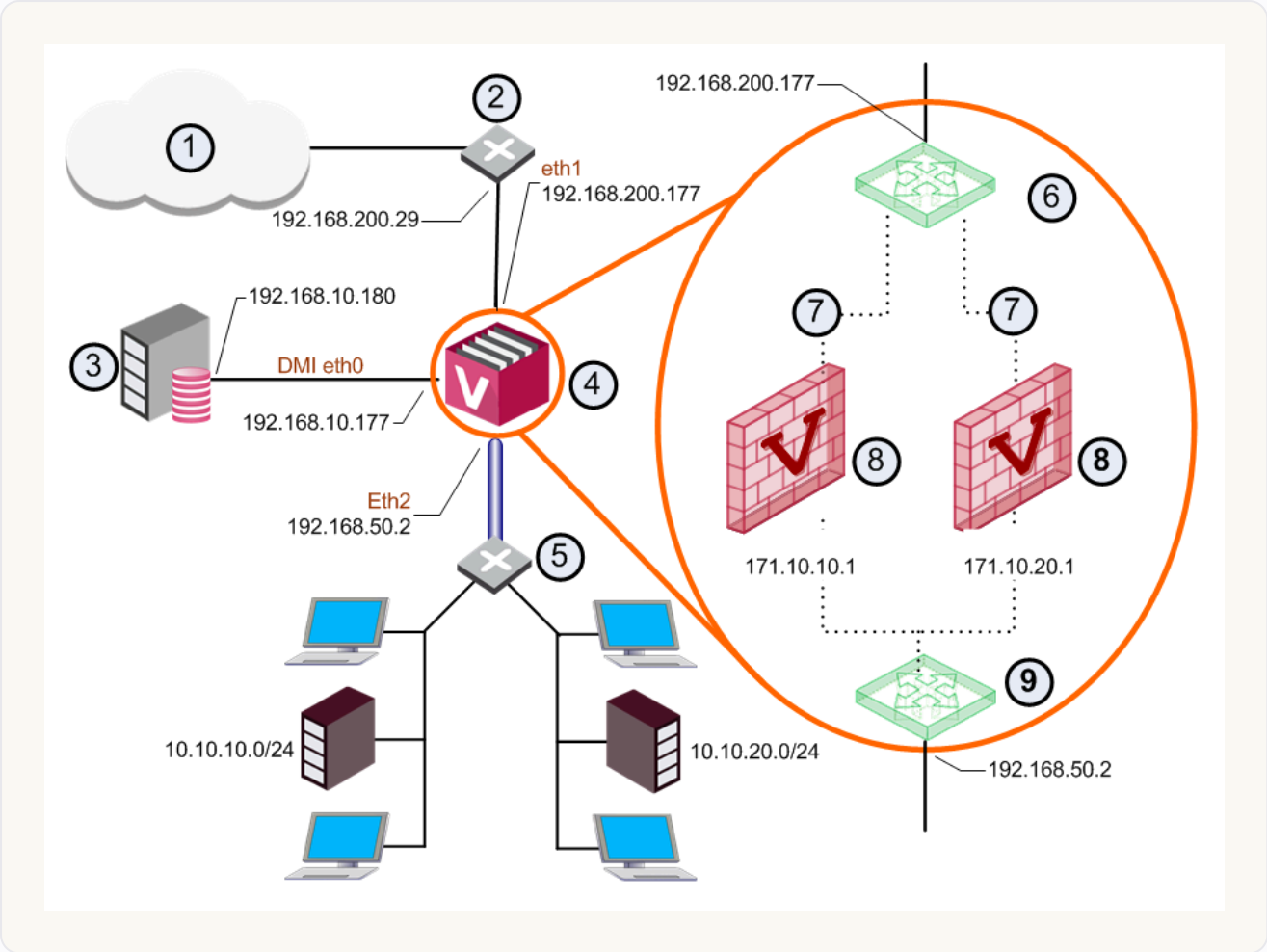
① 인터넷 ② 라우터 ③ VSX Gateway ④ VLAN 스위치 ⑤ Virtual Switch ⑥ Virtual System들

Topology에서는 정의된 단일 인터페이스만 수정할 수 있고 Warp 인터페이스 설정은 바꿀 수 없습니다. 삭제할 때는 먼저 모든 인터페이스를 제거한 뒤 객체를 삭제합니다.

Virtual Router는 물리 라우터처럼 동작하며 여러 VS의 공유 회선이나 VS 간 라우팅에 씁니다(앞서 여러 번 언급한 대로 Maestro·Scalable Chassis에서는 지원되지 않습니다). 외부망·인터넷으로 향하면 외부 Virtual Router, 내부 보호망으로 향하면 내부 Virtual Router라 부르는데, 외부 Virtual Router는 여러 VS가 하나의 보안 물리 인터페이스를 공유해 인터넷으로 나가는 공용 게이트웨이 역할을 합니다.



① 인터넷 ② 라우터 ③ Security Management Server ④ VSX Gateway ⑤ VLAN 스위치 ⑥ External Virtual Router ⑦ Virtual System들

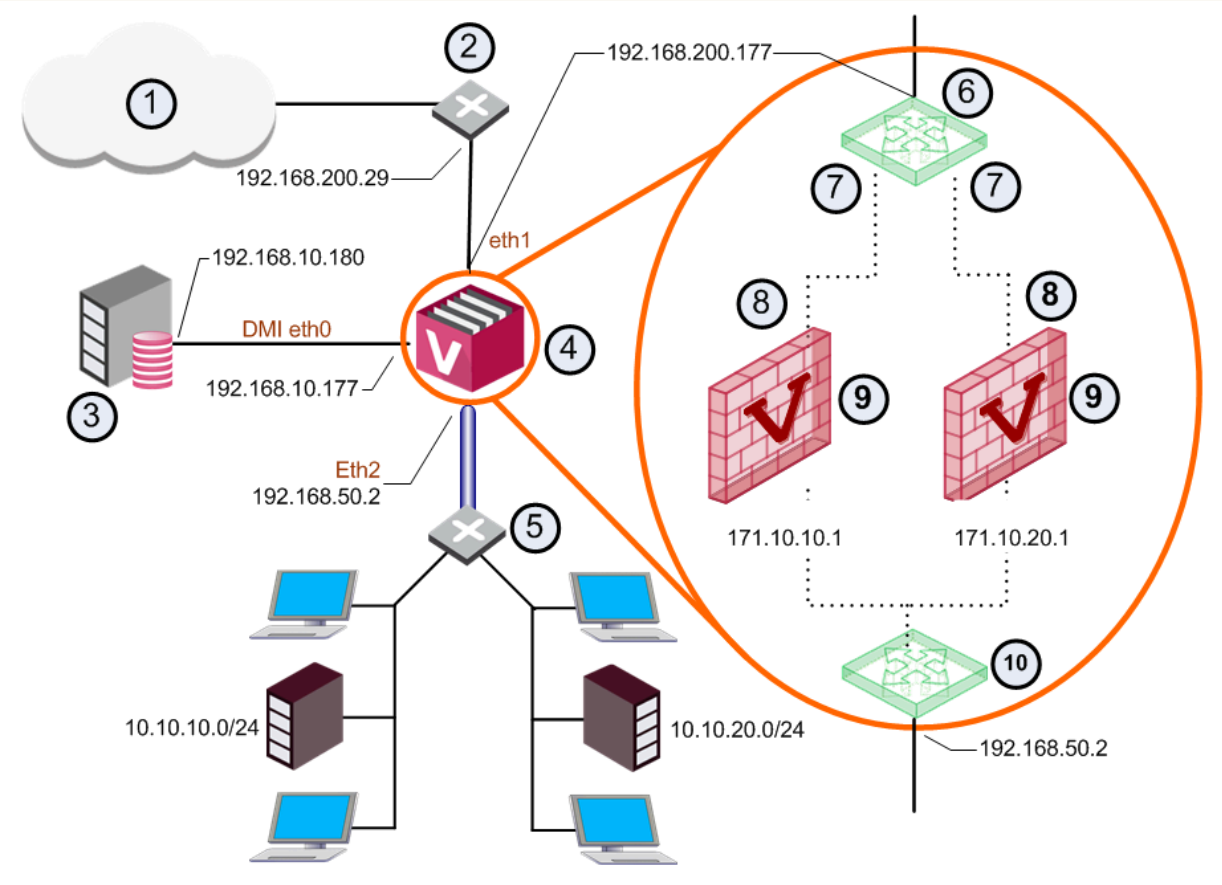


① 인터넷 ② 라우터 ③ Security Management Server ④ VSX Gateway ⑤ 스위치 ⑥ External Virtual Router ⑦ Unnumbered ⑧ Virtual System들 ⑨ Internal Virtual Router

Virtual Router Wizard로 이름·호스팅 게이트웨이를 정하고 인터페이스와 라우트(필요하면 기본 라우트)를 추가해 만들며, Topology의 Routes에서 Warp Link 라우트는 자동 생성되어 수정·삭제할 수 없고 그 외 라우트만 손댑니다. 만든 뒤에는 각 VS에 Virtual Router로 가는 인터페이스를 추가해 연결합니다.

출발지 기반 라우팅 (Source-Based Routing)

보통의 라우팅은 목적지 IP로 경로를 정하지만, **출발지 기반 라우팅**은 출발지 IP(또는 출발지와 목적지의 조합)로 경로를 정하며, 일반 목적지 기반 규칙보다 우선 합니다.



① 인터넷 ② 라우터 ③ Security Management Server ④ VSX Gateway ⑤ 스위치 ⑥ External Virtual Router
⑦ wrpj ⑧ Wrp Unnumbered 인터페이스 ⑨ Virtual System들 ⑩ Internal Virtual Router

덕분에 VLAN 없이 단일 물리 인터페이스만으로도 출발지에 따라 알맞은 VS로 트래픽을 보낼 수 있습니다. 설정은 Virtual Router 객체의 Topology 페이지에서 **Advanced Routing** 을 열어, 규칙마다 출발지 IP·넷마스크, 목적지 IP·넷마스크, next hop 게이트웨이를 지정하는 식입니다. 이 절차는 VSX Gateway든 VSX Cluster든 동일합니다.

VS를 위한 CoreXL

CoreXL은 여러 방화벽 인스턴스를 만들어 여러 CPU 코어에서 병렬로 돌려 성능을 높이는 기술입니다.

> *CoreXL creates multiple Firewall instances that are, in reality, independent firewalls. You can use CoreXL to increase the performance of the VSX Gateway with multiple CPU cores.*
>
> — AdminGuide, "CoreXL for Virtual Systems" (p.140)

VSX에서 중요한 점은 VS0(게이트웨이)과 나머지 Virtual System의 설정 방법이 다르다는 것입니다. VS0은 CLI(`cpconfig` 의 Configure Check Point CoreXL)로, 개별 Virtual System은 SmartConsole(객체 > CoreXL)로 인스턴스 수를 정합니다. Scalable Platform Security Group은 gClish의 `cpconfig` 나 Expert 모드의 `g_all cp_conf corexl` 로 설정합니다.

여기에는 반드시 알아야 할 비용이 있습니다. CoreXL 인스턴스 하나하나가 추가 메모리를 쓰므로, 인스턴스 5개짜리 VS는 별도 VS 5개와 비슷한 메모리를 먹습니다(VS별 CPU·메모리 사용량을 보는 방법은 VSX 성능 최적화 장 참고). 그래서 인스턴스 수는 그 VS의 예상 트래픽에 맞춰 정하고 물리 코어 수를 넘기지 않는 것이 권장됩니다. 또 IPv6 인스턴스 수는 IPv4 인스턴스 수를 넘을 수 없고, VS의 인스턴스 수를 바꾸면 그 VS에 약간의 다운타임이 생깁니다. 한편 VS0에 CoreXL을 켜는 것은 권장되지 않는데, VS0의 주 역할이 게이트웨이 관리라 인스턴스 하나로 충분하고 메모리 부담만 늘기 때문입니다. VS0의 CoreXL 변경은 재부팅이 필요 없지만, 상위 버전으로 업그레이드한 뒤에는 VS0 컨텍스트에서 `fw ctl affinity -vsx_factory_defaults` 로 어피니티를 기본값으로 되돌리고 재부팅하는 것이 좋습니다.

가상 장치의 동적 라우팅

Virtual System과 Virtual Router는 OSPF·BGP 같은 동적 라우팅 프로토콜로 서로, 그리고 외부 라우터와 경로를 주고받을 수 있습니다. VSX는 이를 위해 Gaia 라우팅 데몬(routed)을 쓰며, 각 가상 장치가 자기만의 동적 라우팅 인스턴스와 설정 파일을 가집니다. 설정은 게이트웨이 CLI(Gaia Clish, Scalable Platform은 gClish)에서 `set virtual-system <VSID>` 로 대상 장치 컨텍스트에 들어간 뒤 라우팅 데몬 명령을 실행하고 `save config` 로 저장합니다. 단 정적 라우트는 반드시 SmartConsole의 객체에서만 설정해야 하고, 클러스터라면 모든 멤버를 동일하게 맞춰야 합니다.

```
set virtual-system <VSID>    # 대상 가상 장치 컨텍스트
# (OSPF/BGP 등 라우팅 데몬 명령 — 구문은 Gaia Advanced Routing Guide)
save config
```

인터페이스 정의 다루기

VSX Gateway·Virtual Router·Virtual Switch는 적어도 하나의 인터페이스 정의를 가지며, 보통 토폴로지를 설정할 때 함께 정의합니다. Warp 인터페이스는 가상 장치 정의에 따라 자동 생성되며 수정·삭제할 수 없습니다. 인터페이스를 추가할 때는 객체의 Topology에서 New로 Regular(물리)·Leads to Virtual Router·Leads to Virtual Switch 중 하나를 고릅니다. General 탭에서 물리 인터페이스·VLAN 태그·IP·넷마스크·MTU(기본 1500)를 정하고, Virtual Router/Switch로 가는 인터페이스는 넷마스크가 항상 IPv4 `255.255.255.255` (IPv6 `/128`) 로 고정됩니다. 여기서도 Propagate route 옵션으로 이웃 장치에 경로를 알릴 수 있습니다.

인터페이스에는 보안 설정도 붙습니다. **Anti-Spoofing(스푸핑 방지)** 은 신뢰된 출발지 IP를 위조해 들어오는 공격을 막는 것으로, 토폴로지가 제대로 정의돼 있으면 인터페이스의 Topology 탭에서 "Perform Anti-Spoofing based on interface topology"로 켭니다(동적 라우팅을 쓴다면 토폴로지 자동 계산을 끄고 수동으로 정의해야 합니다). **멀티캐스트 제한** 은 특정 멀티캐스트 그룹(IPv4 224.0.0.0~239.255.255.255 등)으로 나가는 패킷을 차단하는 규칙으로, 인터페이스의 Multicast Restrictions 탭에서 허용/차단 목록을 정의합니다.

인증 — RADIUS, TACACS, RSA SecurID

VSX도 일반 게이트웨이처럼 여러 인증 스킴을 씁니다. Check Point 비밀번호와 OS 비밀번호는 로컬에 저장되고, RADIUS·TACACS는 외부 서버에 인증을 맡기며, SecurID는 토큰이 만드는 일회용 비밀번호를 RSA Authentication Manager로 검증합니다.

RADIUS·TACACS 설정에는 두 가지 방식이 있습니다. **Shared(공유, 기본값)** 는 모든 VS가 VSX Gateway를 통해 인증 서버에 연결하고, **Private(개별)** 은 각 VS가 자기 클러스터 IP를 출발지로 직접 연결합니다. SmartConsole에서 VS 객체의 Other > Authentication(개별은 Legacy Authentication)에서 방식을 고른 뒤 정책을 설치하는데, 클러스터에서는 Hide NAT 설정이 핵심입니다. Shared는 Hide NAT을 꺼야 하고, `table.def` 의 `no_hide_services_ports` 에 RADIUS(UDP 1645)나 TACACS(49) 포트를 넣어 줍니다(sk98339). Private은 반대로 Hide NAT을 켜고 그 포트들을 빼 둡니다. MDS라면 이 설정을 해당 VS를 관리하는 Target Domain Management Server에서 해야 합니다.

RSA SecurID도 Shared와 Private이 있습니다. Shared는 멤버 단위로 `sdconf.rec` (암호화 키)와 `securid` (노드 시크릿) 파일을 쓰고, Private은 VS마다 고유한 `sdconf.rec` 와 `securid` 를 씁니다. 큰 흐름은 RSA Authentication Manager에서 해당 MIP(Member IP)로 `sdconf.rec` 를 생성해 각 VS 컨텍스트의 `$FWDIR/conf/` 에 복사하고, SmartConsole에서 VS 객체에 SecurID를 선택해 정책을 설치한 뒤, 최초 인증 때 서버가 보내 준 노드 시크릿 파일 `securid` 를 모든 VS 컨텍스트로 복사하는 것입니다. 클러스터에서는 UDP 5500 포트의 Hide NAT 처리(`no_hide_services_ports`)를 서버 기대값에 맞춰 설정합니다.

VSX 객체 상태 추적

SmartConsole의 Logs & Events에서 모든 게이트웨이·가상 장치의 상태를 볼 수 있는데, 가상 장치의 전체 상태는 그 Software Blade들 중 가장 심각한 상태로 결정됩니다. 예컨대 다른 Blade가 다 OK라도 하나가 Problem이면 전체가 Problem입니다. 상태는 정상인 **OK**, 사소한 문제가 있지만 동작하는 **Attention**, Blade 오작동이나 미설치를 뜻하는 **Problem**, 데이터를 기다리는 **Waiting**, 도달 불가인 **Disconnected**, 그리고 SIC가 안 맺어진 **Untrusted** 로 나뉩니다.

NAT

VSX는 물리 게이트웨이와 거의 같은 방식으로 VS에 NAT을 지원합니다. NAT을 켠 VS가 Virtual Router에 연결되면 변환된 라우트가 자동으로 그 Virtual Router로 전달됩니다.

> *VSX supports NAT for Virtual Systems much in the same manner as a physical Security Gateway. When a NAT enabled (Static or Hide) Virtual System connects to a Virtual Router, the translated routes are automatically forwarded to the appropriate Virtual Router.*
>
> — AdminGuide, "Working with Network Address Translation (NAT)" (p.169)

단일 게이트웨이에서는 VS 객체의 NAT > Advanced에서 Add Automatic Address Translation을 켜고 Hide(게이트웨이 뒤로 숨기기 또는 라우팅 가능한 가상 IP 뒤로 숨기기)나 Static(사설↔공인 1:1 변환) 을 고른 뒤 정책을 설치하면 됩니다.

클러스터에서는 한 가지 특수한 경우가 있습니다. VS 자신이 만들어 내는 트래픽(예: Anti-Bot 시그니처 업데이트)을 Hide NAT 하려면, 먼저 그 VS 컨텍스트에서 `fw getifs` 로 **Funny IP**(클러스터 내부 통신망에 속한 IP) 를 알아낸 다음, SmartConsole에서 Funny IP를 가진 Node Host와 NATed IP를 가진 Node Host 객체를 만들고, NAT 정책에 그 VS 트래픽을 NATed IP 뒤로 숨기는 규칙을 추가해 정책을 설치합니다.

App Control, URL Filtering, Threat Prevention

VSX에서도 Application Control·URL Filtering·Threat Prevention 같은 상위 보안 Blade를 VS별로 켤 수 있습니다(sk106496, sk79700). 기본 동작과 정책 구성은 물리 게이트웨이와 같지만, 두 가지 전제가 중요합니다. 하나는 VS0(VSX Gateway/클러스터 멤버)의 라우팅·DNS·프록시 설정이 올바라야 한다는 것이고, 다른 하나는 이 Blade들을 VSX Gateway나 클러스터 객체 자체에는 켜지 말아야 한다는 것입니다. 켜는 것은 어디까지나 개별 Virtual System입니다. 각 VS가 독립 정책을 갖는다는 VSX의 원리 덕분에, 테넌트나 부서마다 서로 다른 위협 방어 정책을 줄 수 있습니다.

VSX 라이선스

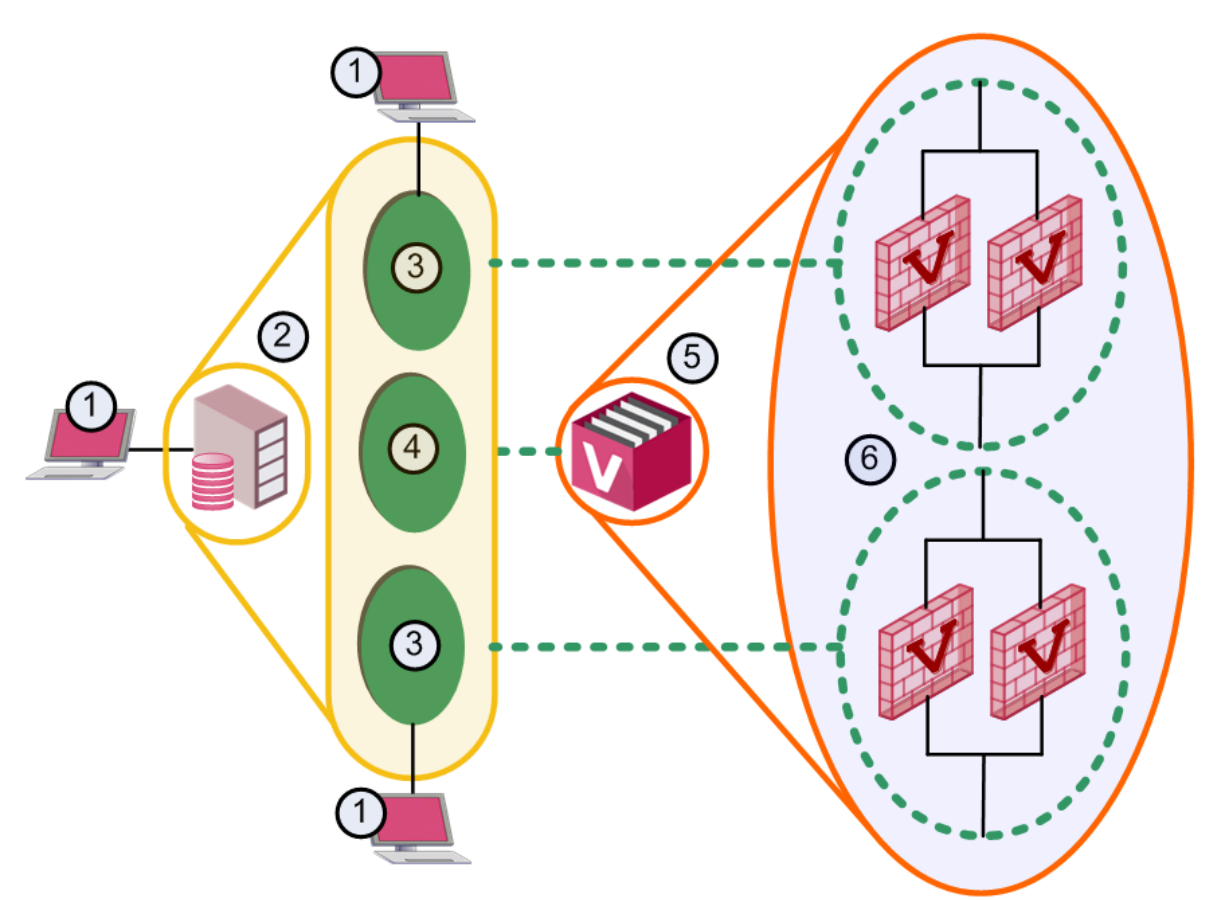
마지막은 라이선스입니다. VSX는 게이트웨이가 허용하는 Virtual System 수가 라이선스로 카운팅되며, 이를 `vsx stat`의 "Number of Virtual Systems allowed by license" 값으로 확인할 수 있습니다(0보다 커야 정책 설치가 됩니다). 라이선스가 없거나 유효하지 않으면 VSX 생성 마법사에서 정책 설치 오류가 나므로, 관리 서버에서 `cplic check`로 필요한 라이선스를 확인하고 각 VSX Gateway·클러스터 멤버마다 유효한 VSX 라이선스를 설치해야 합니다. VSX/VSNext의 NGSM 라이선스 카운팅 방식은 sk119075를 참고합니다.

08 Multi-Domain Server와 함께 쓰기

Using VSX with Multi-Domain Server

VSX는 **Multi-Domain Server(MDS)** 로도 관리할 수 있습니다. 이 챕터는 VSX에만 해당하는 부분만 다루고, MDS 제품 자체에 대한 이해는 이미 있다고 전제합니다. MDS의 일반적인 개념과 절차는 R82 Multi-Domain Security Management Administration Guide를 참고하면 됩니다.

MDS는 서비스 제공업체나 대기업처럼 서로 다른 도메인·부서·지사의 여러 독립 네트워크를 한곳에서 중앙 관리하기 위한 솔루션입니다. MDS 자신이 이 네트워크들의 관리·정책 데이터베이스를 호스팅하는 중앙 관리 서버가 되고, 독립된 각 도메인은 하나의 **Domain**으로 표현됩니다. 그리고 각 도메인을 실제로 관리하는 것이 **Domain Management Server**인데, 이 Domain Management Server는 Virtual System·Virtual Router·Virtual Switch는 물론 물리 Security Gateway까지 호스팅할 수 있습니다.



① SmartConsole ② Multi-Domain Server ③ Domain Management Server ④ Main Domain Management Server ⑤ VSX Gateway ⑥ 각 Domain Management Server의 Virtual System들

여기서 역할에 따라 이름이 갈립니다. VSX Gateway나 클러스터를 관리하는 Domain Management Server를 **Main Domain Management Server** 라고 부릅니다. 하나의 MDS 위에 여러 게이트웨이와 클러스터를 올릴 수 있고, 한 도메인에 속한 Virtual System들이 여러 VSX Gateway와 클러스터에 흩어져 있어도 됩니다. SmartConsole로 MDS에 연결하면 도메인과 Domain Management Server, MDS 환경 전체를 중앙에서

관리할 수 있는데, 각 Domain Management Server는 MDS를 통해서만 접근되는 자기만의 SmartConsole 인스턴스로 자신의 가상 장치와 물리 게이트웨이, 정책을 다룹니다.

무엇이 다른가 — 도메인을 먼저 만든다

VSX Gateway와 클러스터, 가상 장치를 프로비저닝하고 설정하는 절차 자체는 앞서 본 Security Gateway 관리 모델과 본질적으로 같습니다. 가장 큰 차이는, MDS에 연결한 SmartConsole에서 먼저 각 Domain과 그에 딸린 Domain Management Server 객체부터 만들고 설정해야 한다는 점입니다. 바꿔 말하면 **하나의 Domain Management Server가 기능적으로는 하나의 VSX Gateway에 대응**하며, 그 VSX Gateway의 네트워크 객체·보안 정책을 다루려면 해당 Domain Management Server에 SmartConsole로 접속하면 됩니다.

> *The most important difference is that you must first create and configure each Domain and its associated Domain Management Server objects using the SmartConsole connected to a Multi-Domain Server.*
>
> — AdminGuide, "VSX Provisioning" (p.182)

전체 흐름은 이렇습니다. 먼저 배포 규모에 맞게 MDS와 (필요하면) Multi-Domain Log Server를 정의하고 설정합니다. 그다음 VSX Gateway나 클러스터마다 Domain과 Domain Management Server를 만들고, SmartConsole로 그 Domain Management Server에 접속해 VSX Gateway·클러스터 객체를 생성·설정한 뒤 기본 보안 정책을 잡아 줍니다.
이어서 배포에 필요한 개별 Domain과 Domain Management Server를 정의하고, 각 Domain에 연결한 SmartConsole에서 그 도메인의 Virtual System과 나머지 가상 장치를 만들어 설정합니다.

관리 이중화와 부하 분산

MDS 머신을 설치·배포하는 자세한 절차는 R82 Multi-Domain Security Management Administration Guide에 있지만, VSX 관점에서 알아둘 점이 둘 있습니다. **관리 High Availability**를 쓰려면 MDS 머신을 최소 두 대 정의해야 합니다. 또한 여러 MDS 머신을 두고 도메인마다 Domain Management Server를 둘 이상 두면 **관리 트래픽을 분산(Load Sharing)** 할 수 있는데, 이 경우 각 MDS마다 Domain Management Server를 하나씩 정의합니다.

MDS에서 가상 장치 다루기

MDS에서 가상 장치를 다룰 때는 반드시 해당 가상 장치를 관리하는 Domain Management Server의 SmartConsole을 써야 합니다. 그 점만 빼면 설정 절차는 Security Management Server와 똑같고, MDS는 가상 장치를 물리 장치와 동일하게 취급합니다. 라이선스가 허용하는 한 얼마든지 Virtual System을 Domain Management Server에 추가할 수 있으며, 한 Domain Management Server에 속한 Virtual System들이 같은 VSX Gateway나 클러스터에 있을 필요도 없습니다.

새 Virtual System을 추가하려면 해당 Domain Management Server에서 SmartConsole을 띄워 Virtual System을 만들고 설정한 뒤 보안 정책을 정의해 설치하면 됩니다. Virtual Router나 Virtual Switch를 추가할 때도 같은 방식으로 해당 Domain Management Server의 SmartConsole에서 만들면 되는데, 다만 **Maestro·Scalable Chassis Security Group은 Virtual Router를 지원하지 않는다**(Known Limitation 01413513)는 제약은 여기서도 그대로 적용됩니다.

09 VSX 클러스터 구성

Configuring VSX Clusters

VSX를 이중화하는 방법을 다루는 챕터입니다. 먼저 짚어둘 점은, 이 챕터는 Scalable Platform(Maestro·Chassis)에는 적용되지 않는다는 것입니다. 또한 VSX 클러스터는 ClusterXL 개념 위에 세워지므로, ClusterXL에 어느 정도 익숙하다는 전제로 설명합니다.

클러스터가 주는 것

VSX 클러스터는 Virtual System과 가상 장치들에 이중화와 부하 분산을 제공합니다. 둘 이상의 동일한 VSX Gateway를 연결해 데이터를 끊임없이 동기화하는 구성으로, 멤버나 Virtual System이 죽어도 투명하게 페일오버되고, 상태 동기화 덕분에 핵심 업무에 무중단을 보장하며, 부하 분산으로 피크 시간에도 처리량을 유지하고 향후 트래픽 증가에도 확장할 수 있습니다.

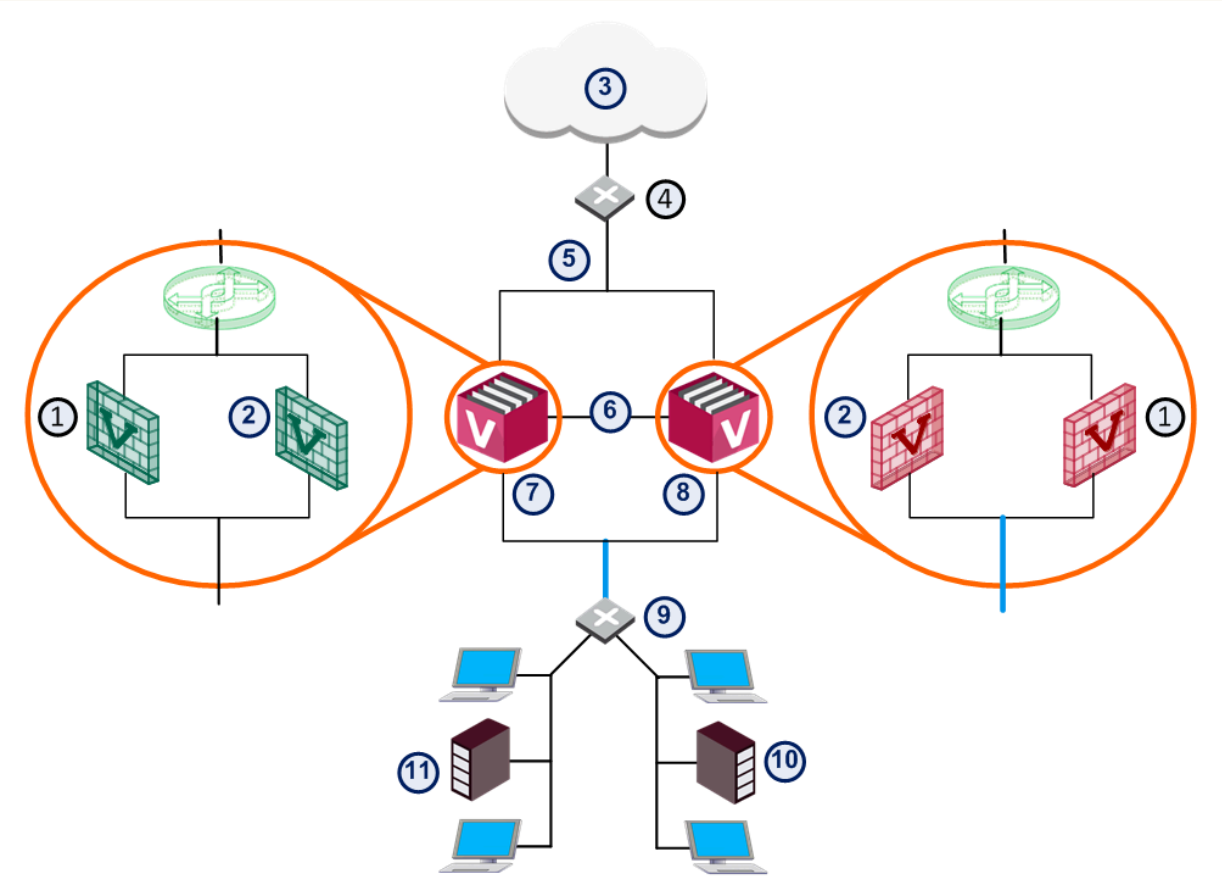
> *A VSX Cluster consists of two or more identical, interconnected VSX Gateways that ensure continuous data synchronization.*
>
> — AdminGuide, "VSX Cluster Overview" (p.185)

모드는 두 가지인데, R81.10 이상으로 새로 설치한 클러스터에서는 VSLS만 쓸 수 있고, **High Availability는 R81 이하에서 R82로 업그레이드한 경우에만** 가능합니다(VSLS를 HA로 변환할 수도 있습니다). VSLS(Virtual System Load Sharing)는 Active Virtual System들을 여러 멤버에 분산해 성능을 높이면서 투명한 페일오버를 제공하고, High Availability는 모든 멤버와 Virtual System이 Active/Standby로 동작하며 계속 동기화됩니다.

물리 클러스터의 개념부터

VSX 클러스터를 이해하려면 물리 클러스터를 먼저 떠올리는 게 좋습니다. 일반적인 클러스터는 둘 이상의 동일한 물리 게이트웨이를 묶어 하나의 게이트웨이처럼 동작하게 하고, **Cluster IP(Virtual IP)** 라는 자체 IP를 가집니다. 외부에서 내부로 가는 트래픽은 이 외부 Cluster IP로 도착하고, 모드에 따라 지정된 멤버가 받아 검사한 뒤 목적지로 보내거나 버립니다. 반대로 내부에서 외부로 가는 트래픽도 Cluster IP로 향합니다. 각 멤버의 인터페이스에는 고유한 물리 IP가 있지만 이 IP는 외부 네트워크에 보이지 않으며, 멤버끼리 그리고 관리 서버와 정책 다운로드·로그·상태 점검 같은 내부 통신에만 쓰입니다.

VSX 클러스터도 똑같이 동기화된 게이트웨이들을 묶어 하나가 죽으면 다른 하나가 즉시 자리를 대신합니다. 다만 VSX에서는 Virtual System·Router·Switch와 그 인터페이스가 모든 멤버에 동일하게 프로비저닝·설정된다는 점이 추가됩니다. 각 멤버에 있는 동일한 가상 장치 인스턴스를 서로 **peer(피어)** 라고 부릅니다. VSX는 가상 장치 인터페이스마다 가상 IP를 배정하고, 가상 장치의 상태 테이블을 다른 멤버의 피어로 동기화합니다.



① Virtual System 2 ② Virtual System 1 ③ 인터넷 ④ 라우터 ⑤ 외부 Cluster 인터페이스 ⑥ Sync ⑦ VSX Cluster Member 1 ⑧ VSX Cluster Member 2 ⑨ VLAN 스위치 ⑩ 네트워크 2 ⑪ 네트워크 1

배포 계획 — IP 주소 할당

물리 네트워크처럼 사전 계획이 성공의 열쇠이며, 특히 IP 주소 할당에 주의해야 합니다. VSX 클러스터에는 실제 IP와 가상 IP가 모두 필요하지만, VSX가 가상 장치 사이의 Warp Link IP는 자동으로 배정해 주므로 부담이 줄어듭니다. 클러스터 네트워크는 세 가지 요소로 이뤄집니다.

동기화 네트워크(Synchronization Network) 는 멤버 간 상태 동기화 데이터를 나르는 물리 네트워크입니다. 클러스터를 처음 정의할 때 설정하며, 멤버를 추가·제거할 때 바꿀 수 있습니다. 여기에는 기업 네트워크 어디에서도 쓰지 않는 고유한 IP 를 써야 합니다.

내부 통신 네트워크(Internal Communication Network) 는 ClusterXL 환경에서 동기화 네트워크에 더해 필요한 가상 네트워크로, 외부에는 보이지 않으며 멤버들이 환경의 상태를 인식하고 통신하게 해 줍니다. VSX가 클러스터 생성 때 IP를 자동 배정하는데, 기본값은 IPv4 `192.168.196.0/255.255.252.0` (클래스 C 4개 범위)이고 IPv6는 `FD9A::1FFE:0:0:0/80` 입니다. 이 IP는 Virtual System을 만들기 전에만 바꿀 수 있고, 일단 Virtual System을 만들면 변경할 수 없습니다. 그래서 가상 장치를 만들기 전에 이 기본 범위가 외부 네트워크 어디에서도 쓰이지 않는지 반드시 확인해야 IP 충돌을 피할 수 있습니다.

가상 IP 주소(Virtual IP) 는 외부 네트워크에 보이는 유일한 IP 입니다. 직접 연결된 서브넷에 대응해야 하고 유효한 next hop 역할을 해야 합니다.

클러스터 만들기

기본 설정은 대부분 SmartConsole에서 하지만, 클러스터 정의를 바꾸는 것 같은 많은 관리 작업은 CLI(`vsx_util`)가 필요합니다. 멤버에서 상태를 들여다보는 `cphaprob state` 같은 명령은 현재 VS 컨텍스트 기준으로 동작하므로, 진단할 때는 `vsenv` 로 대상 VS에 들어간 상태인지 늘 확인하세요. 새 클러스터는 SmartConsole의 VSX Cluster Wizard로 만드는데, 관리 서버나 Main Domain Management Server에 접속해 New Cluster를 시작하면 마법사가 단계별로 안내합니다. General Properties에서 클러스터 이름(공백·특수문자 불가, 밑줄만 허용)과 IPv4·IPv6 Cluster IP를 정하고, 이후 멤버·인터페이스·토폴로지를 설정한 뒤 마법사를 마치면 대부분의 속성은 SmartConsole에서 바로 수정할 수 있습니다.

설정에서 알아둘 항목 몇 가지를 짚으면, ClusterXL 창에서는 상태 동기화를 켜고 끄거나 멤버 상태 변화의 추적 옵션을 고를 수 있고 나머지 ClusterXL 속성은 비활성화됩니다. Topology 페이지에서는 인터페이스와 라우트를 정의하는데, 라우트를 추가할 때 **Propagate route to adjacent Virtual Devices** 를 켜면 이웃 가상 장치에 그 경로를 알려 장치 간 연결을 활성화합니다. 토폴로지를 라우팅 정보로 자동 계산하는 옵션은 기본으로 켜져 있지만, 동적 라우팅을 쓸 때는 끄는 것이 권장되고 Bridge 모드에서는 쓸 수 없습니다.

NAT 페이지에서는 Virtual System에서 나오는 패킷에 Hide NAT이나 Static NAT을 설정할 수 있습니다. Hide NAT은 내부에서 시작된 연결만 허용하며 게이트웨이 뒤로 숨기거나 라우팅 가능한 가상 IP 뒤로 숨길 수 있고, Static NAT은 사설 주소를 대응하는 공인 주소로 1:1 변환합니다. Bridge 모드를 쓴다면 VSX Bridge Configuration에서 ClusterXL 루프 감지(Active/Standby)나 표준 L2 루프 감지(STP/PVST+)를 고릅니다(자세한 내용은 Bridge 모드 장 참고).

클러스터 관리 IP나 서브넷을 바꾸려면 관리 서버에서 `vsx_util change_mgmt_ip` · `vsx_util change_mgmt_subnet` 을, 내부 통신 네트워크 IP를 바꾸려면 `vsx_util change_private_net` 을 씁니다.

멤버 추가와 제거

멤버를 추가·제거하기 전에는 반드시 다른 관리자가 관리 서버에 접속해 있지 않아야 합니다. DB가 잠겨 있으면 `vsx_util`이 관리 DB를 수정하지 못하기 때문입니다.

멤버를 추가할 때는 새 멤버를 설치한 뒤 모든 SmartConsole을 닫고 환경을 백업(sk100395)하고서, 관리 서버에서 `vsx_util add_member`를 실행합니다. 그다음 `vsx_util reconfigure`로 새 멤버를 구성하고 재부팅한 뒤 각 멤버에서 `cphaprob state`로 상태를 확인합니다. VSLS 모드라면 마지막에 `vsx_util vsls`로 Virtual System들을 새 멤버에 재분배해야 합니다.

멤버를 제거할 때는 SmartConsole을 닫고 백업한 다음, 먼저 그 멤버에서 라이선스를 분리해야 합니다(분리하지 않으면 제거할 수 없습니다). 그리고 관리 서버에서 `vsx_util remove_member`를 실행해 안내에 따라 클러스터와 멤버를 고르면 됩니다. 끝나면 SmartConsole에서 클러스터 객체의 Cluster Members에 그 멤버가 사라졌는지 확인합니다.

```
vsx_util add_member         # 새 멤버 추가
vsx_util reconfigure        # 새 멤버 구성 (이후 재부팅)
cphaprob state              # 각 멤버에서 상태 확인
vsx_util vsls               # VSLS 모드: VS 재분배
vsx_util remove_member      # 멤버 제거 (먼저 라이선스 분리)
```

클러스터 타입 바꾸기 (VSLS ↔ HA)

VSLS와 HA 사이의 변환에는 관리 서버의 `vsx_util convert_cluster` 를 씁니다 (`vsx_util` 의 모든 하위 명령은 명령어 레퍼런스 장에 정리돼 있습니다). 앞서 말했듯 R81.10 이상 신규 설치는 VSLS만 가능하고 HA는 업그레이드한 경우에만 쓸 수 있는데, VSLS를 HA로 변환하는 것은 가능합니다.

VSLS를 HA로 바꾸는 흐름은 이렇습니다. 먼저 `vsx_util vsls` 의 "Set all VSs active on one member" 옵션으로 모든 Active Virtual System을 한 멤버에 몰아넣습니다. 그다음 각 멤버에서 `cpconfig` 로 **Per Virtual System State**와 Bridge용 ClusterXL Active/Standby를 끄고 `cpstop ; cpstart` 로 서비스를 재시작합니다(이때 페일오버가 일어날 수 있습니다). 마지막으로 관리 서버에서 `vsx_util convert_cluster` 로 클러스터를 High Availability로 변환합니다.

```
vsx_util vsls               # 모든 VS를 한 멤버에 Active로 모음
cpconfig                    # 각 멤버: Per Virtual System State 비활성화
cpstop ; cpstart            # 서비스 재시작 (페일오버 유발 가능)
vsx_util convert_cluster    # VSLS → HA 변환 (메뉴에서 HA 입력)
```

반대로 HA를 VSLS로 바꿀 때 는 순서가 거의 뒤집힙니다. 각 멤버에서 `cpconfig` 로 **Per Virtual System State와 ClusterXL Bridge Active/Standby를 켜고** `cpstop ; cpstart` 로 재시작한 뒤, 관리 서버에서 `vsx_util convert_cluster` 를 실행해 메뉴에서 `LS` 를 입력하고 변환을 확인합니다. 이때 VS를 멤버에 균등 분배할지, 한 멤버에 모두 Active로 둘지 를 고른 뒤 각 멤버를 재부팅합니다. 다만 Active/Active Bridge 모드의 VS나 Virtual Router가 있으면 VSLS로 변환할 수 없습니다.

VSLS 자세히 보기

여기서부터는 VSLS의 동작 원리를 좀 더 깊이 들여다봅니다(이 내용은 Scalable Platform에는 적용되지 않으며, Maestro·Chassis에서는 gClish `set cluster configuration high-availability mode 3` 을 씁니다).

VSLS는 Active Virtual System들을 여러 멤버에 나눠 트래픽을 분산하는 기술로, 용량·이중화·확장성·비용 효율을 모두 줍니다.

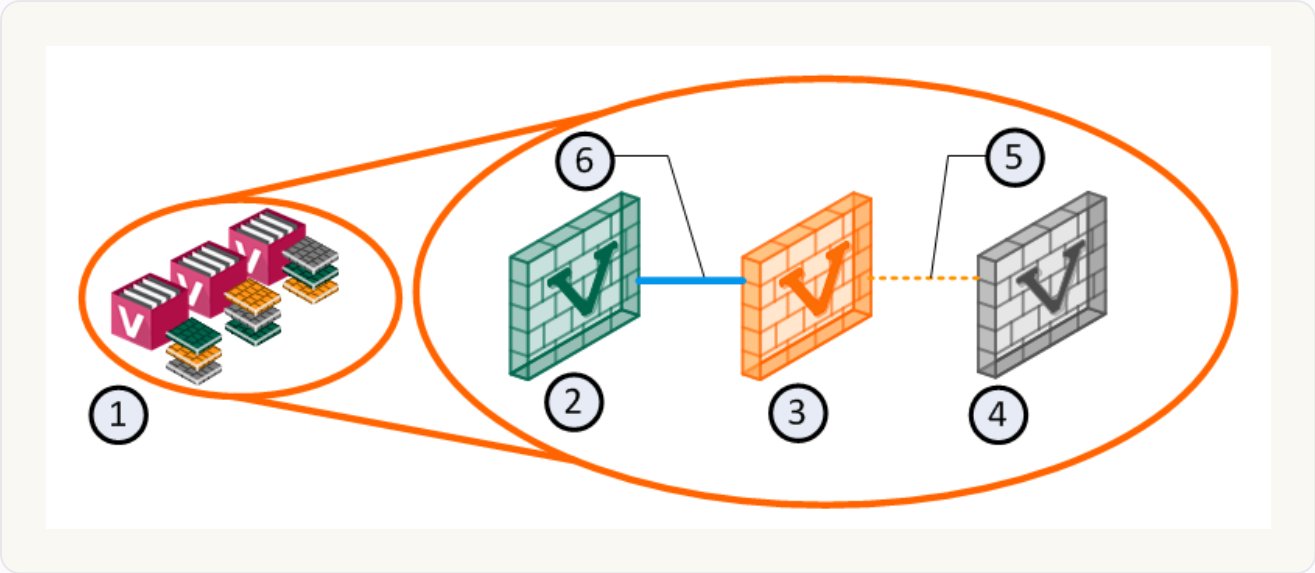
> *Virtual System Load Sharing is a cluster technology that assigns traffic for Virtual Systems to different Active VSX Cluster Members.*
>
> — AdminGuide, "Configuring Virtual System Load Sharing (VSLS)" (p.213)

특히 VS가 멤버들 사이에 자동으로 분배되어 별도 설정이 거의 필요 없고, 표준 스위치만으로 부하 분산이 되며, 멤버를 추가할 때마다 전체 용량과 이중화가 함께 늘어납니다. 한 가지 제약은, Per Virtual System State가 켜진 상태에서는 Virtual Router와, 물리·VLAN 인터페이스가 없는 Virtual Switch는 지원되지 않는다는 것입니다. 또한 VSLS가 상태를 감지·배정하려면 모든 멤버의 모든 VS가 스위치나 VLAN으로 서로 직접 연결돼 있어야 합니다.

우선순위, 가중치, 그리고 세 가지 상태

우선순위(Priority) 는 어느 멤버가 그 VS의 Active·Standby·Backup을 맡을지에 대한 선호를 정수로 나타냅니다. 0이 가장 높아 Active를 맡을 멤버, 1이 Standby를 맡을 멤버, 2 이상은 Backup을 맡되 우선순위 2가 장애 시 가장 먼저 Standby로 전환되고 3이 그다음 차례입니다. **가중치(Weight)** 는 VS마다 트래픽·부하가 다르다는 점을 반영하는 값으로, 무거운 가중치를 준 VS는 특정 멤버에서 더 많은 자원을 차지하고 나머지 VS는 다른 멤버로 분산됩니다. 기본 가중치는 모두 10이며, 우선순위와 가중치 모두 관리 서버의 `vsx_util vsls` 로 바꿉니다.

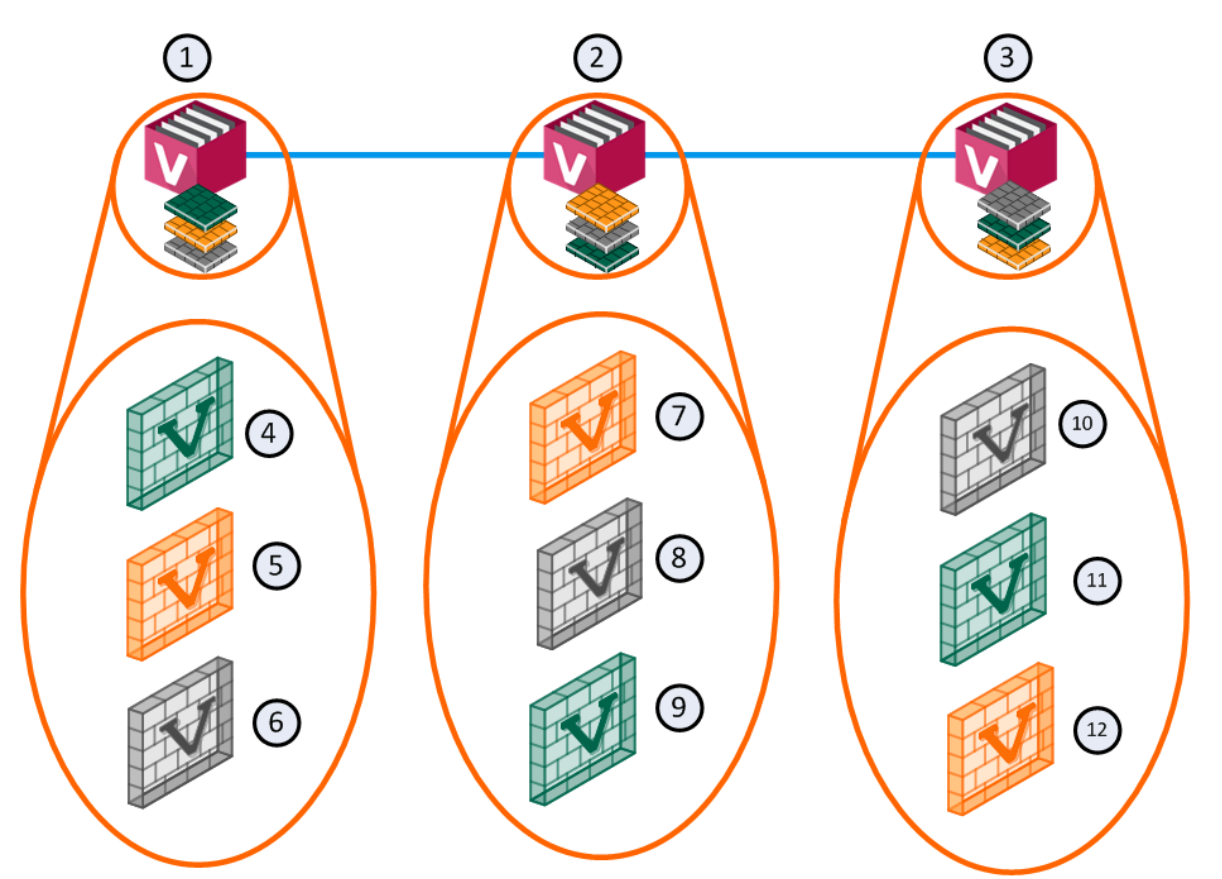
VSLS는 기존의 Active·Standby에 **Backup** 상태를 더합니다. 핵심은 Backup은 각 VS의 최신 설정은 갖지만 상태 테이블 동기화는 받지 않는다 는 점입니다. 즉 Active와 Standby만 상태 테이블을 동기화하고, Backup은 정책 업데이트만 받습니다. 이렇게 해서 동기화 네트워크의 부하를 줄입니다.



① 멤버 3대·각 3 VS인 VSX Cluster ② 멤버1의 VS1 = Active ③ 멤버2의 VS1 = Standby ④ 멤버3의 VS1 = Backup ⑤ 정책 업데이트만 ⑥ 상태 테이블 동기화

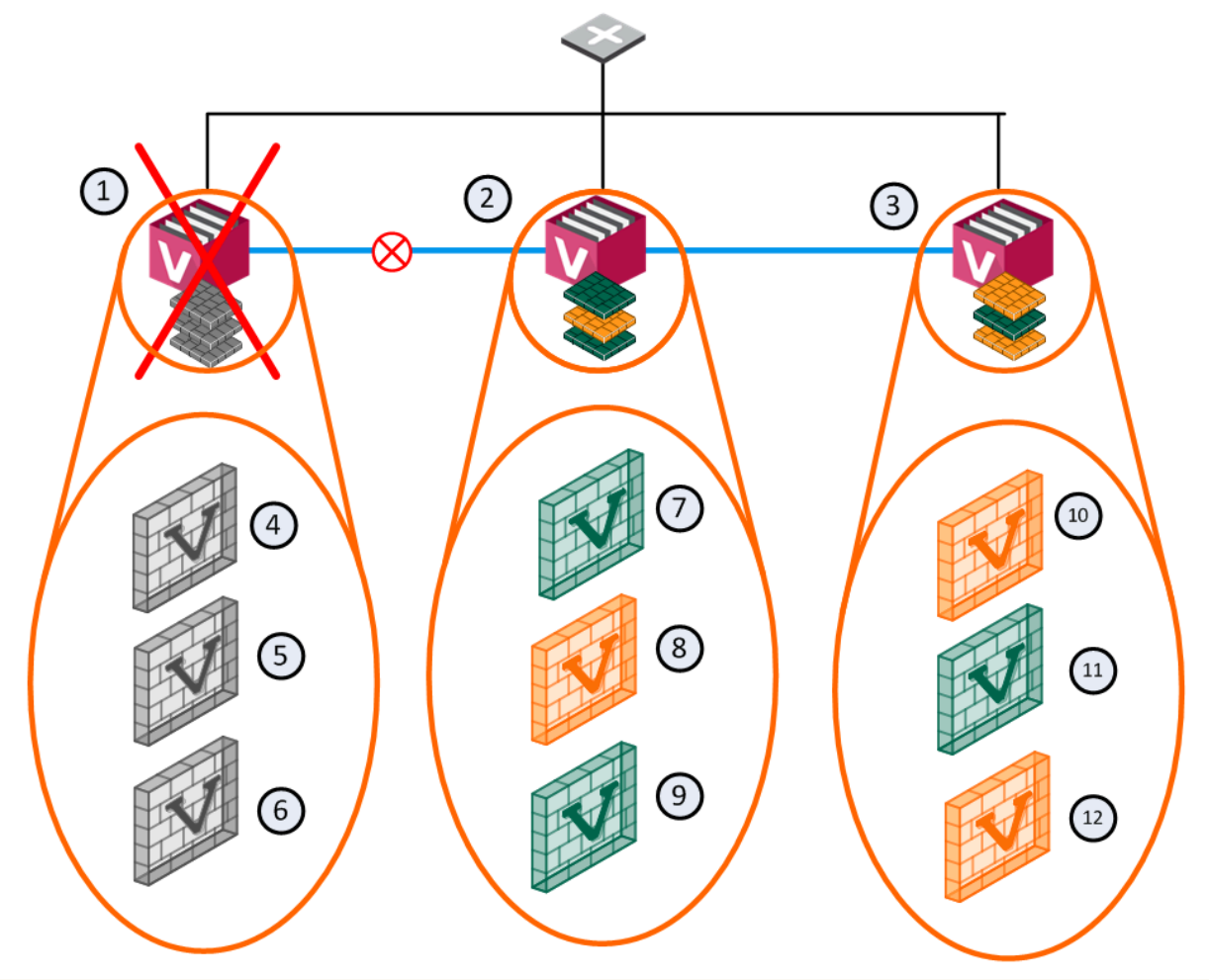
부하 분산과 장애 시나리오

정상 상태에서는, 멤버 3대에 VS 3개가 있다면 각 멤버가 서로 다른 VS의 Active를 하나씩 맡아 부하가 고르게 퍼집니다. VS를 만들면 VSX가 자동으로 Standby와 Backup 상태를 만들어 다른 멤버들에 분배합니다.



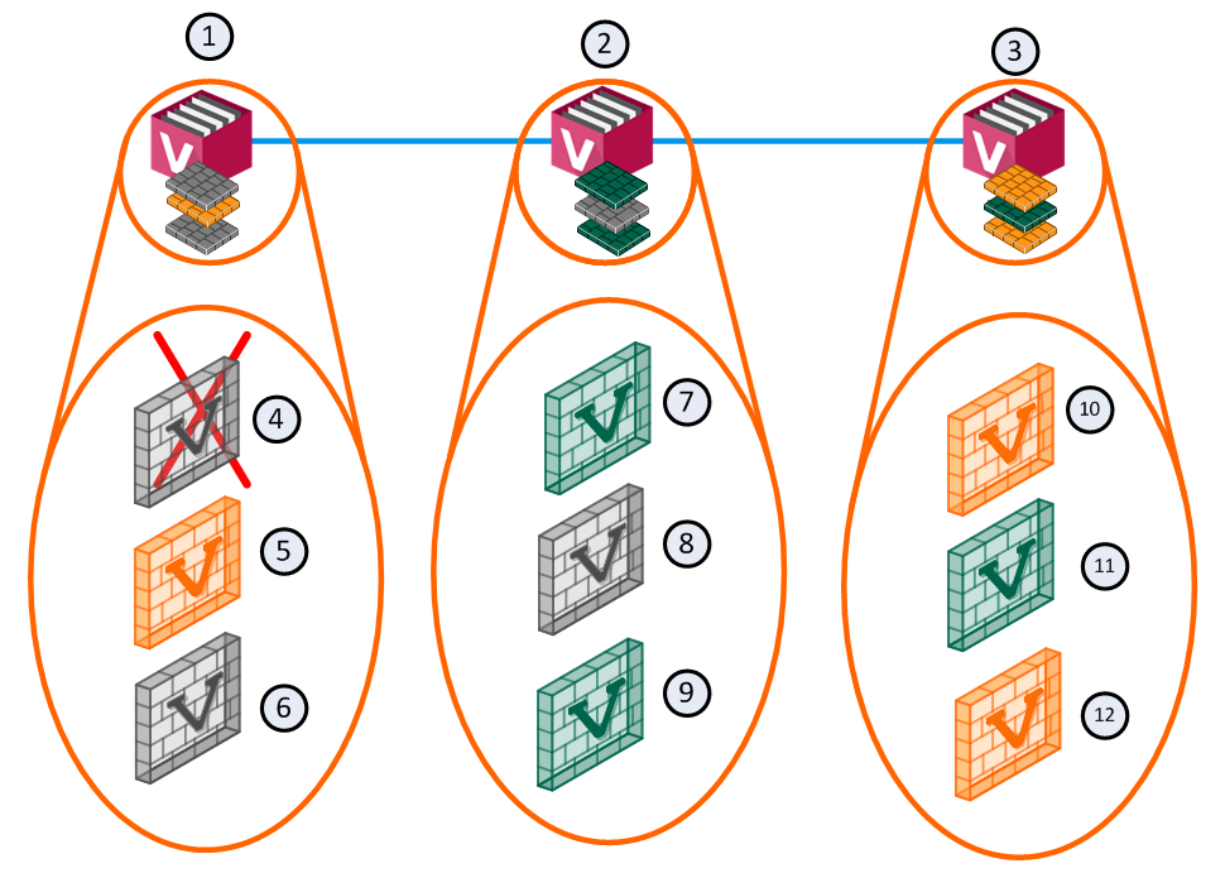
① 멤버 1 ② 멤버 2 ③ 멤버 3 ④ VS1 Active ⑤ VS2 Standby ⑥ VS3 Backup ⑦ VS1 Standby ⑧ VS2 Backup ⑨ VS3 Active ⑩ VS1 Backup ⑪ VS2 Active ⑫ VS3 Standby — 멤버마다 다른 VS가 Active

멤버 하나가 죽으면, VSLS가 이를 감지해 그 멤버가 맡던 VS들의 트래픽을 각자의 Standby로 보냅니다. 완전히 동기화돼 있던 Standby가 즉시 Active로 바뀌어 기존 연결을 끊김 없이 유지하고, 동시에 Backup이 Standby로 올라와 새 Active와 완전히 동기화됩니다.



① 멤버 1 ② 멤버 2 ③ 멤버 3 ④ VS1 Down ⑤ VS2 Down ⑥ VS3 Down ⑦ VS1 Active ⑧ VS2 Standby ⑨ VS3 Active ⑩ VS1 Standby ⑪ VS2 Active ⑫ VS3 Standby — 멤버1 장애 시 Standby가 Active로 승격

개별 VS 하나만 죽는 경우 도 마찬가지로, 그 VS는 자기 Standby로 페일오버하고 Backup이 새 Standby가 되어 동기화하며, 나머지 VS들은 영향 없이 계속 동작합니다.



① 멤버 1 ② 멤버 2 ③ 멤버 3 ④ VS1 Down ⑤ VS2 Standby ⑥ VS3 Backup ⑦ VS1 Active ⑧ VS2 Backup ⑨ VS3 Active ⑩ VS1 Standby ⑪ VS2 Active ⑫ VS3 Standby — 개별 VS 장애 시 그 VS만 페일오버

죽었던 멤버나 VS가 다시 살아나면 시스템은 원래의 부하 분산 구성으로 되돌아갑니다.

VSLS 모드로 설정하기

먼저 클러스터 멤버를 설치한 뒤, 각 멤버에서 `cpconfig` 로 **Per Virtual System State** 를 켜야 합니다. 이 설정이 Active VS를 여러 멤버에 두고 VS별 페일오버를 가능하게 하는, VSLS의 필수 조건입니다(켠 뒤 재부팅). 그다음 SmartConsole에서 새 VSX 클러스터를 만들 때 General Properties의 VSX Cluster Platform에서 **ClusterXL Virtual System Load Sharing** 을 선택하고 마법사를 완료합니다. 이어서 필요한 Virtual System(과 Virtual Router)을 만든 뒤, 관리 서버에서 `vsx_util vsls` 로 분배를 조정합니다.

```
cpconfig                    # 각 멤버: Enable Per Virtual System State → 재부팅
# SmartConsole: New Cluster → VSX Cluster Platform = ClusterXL Virtual System
vsx_util vsls               # 관리 서버에서 분배·우선순위·가중치 조정
```

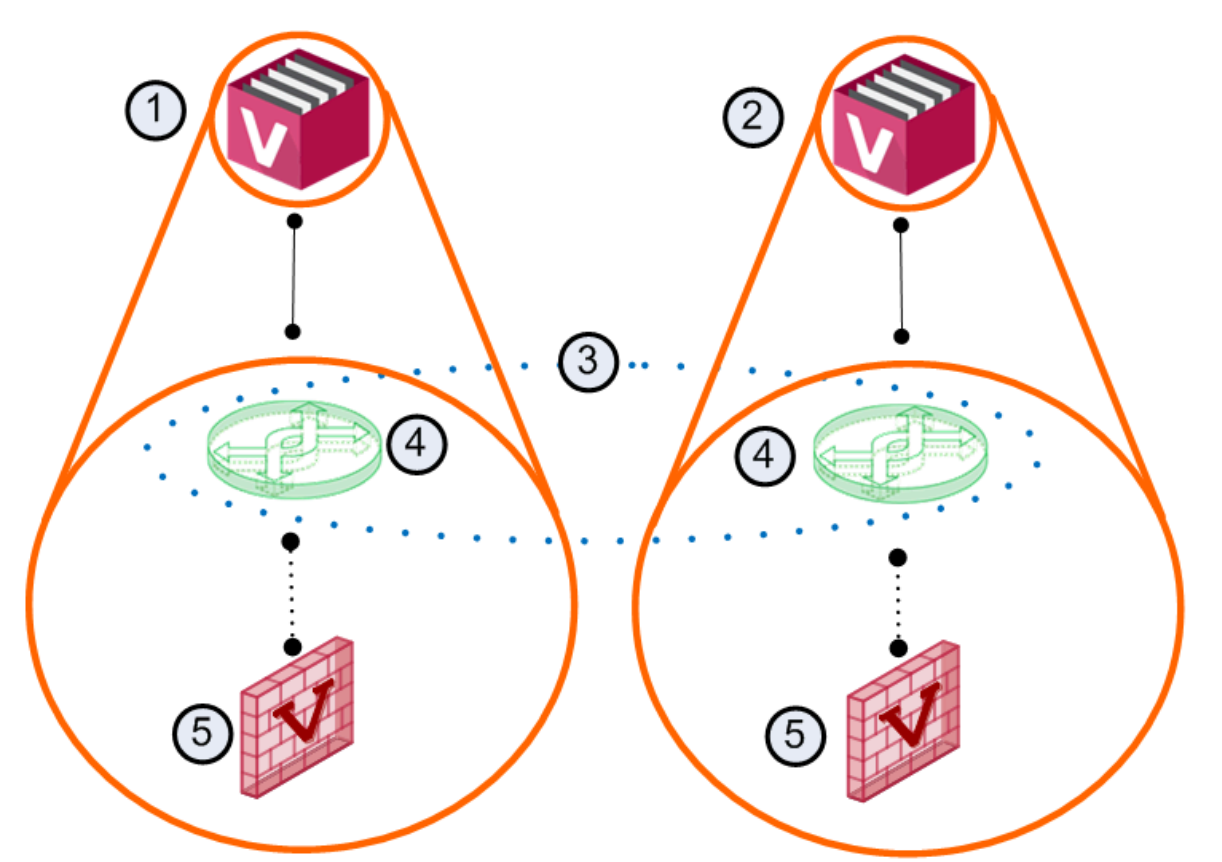
`vsx_util vsls` 는 VSLS 설정의 만능 도구입니다. 현재 VSLS 구성 보기, VS를 멤버에 균등 분배, 모든 VS를 한 멤버에 Active로 모으기, 개별 VS의 우선순위·가중치 수동 지정, Active Up/Primary Up 모드 전환, 그리고 VSLS 설정을 CSV로 import/export 하는 작업을 모두 이 명령으로 합니다. CSV로 내보내고 들여오는 기능 덕분에 복잡한 분배 설정을 백업하거나 다른 클러스터에 옮길 수 있습니다.

VSLS에서의 Virtual Router

VSLS 모드에서도 Virtual Router를 다루는 별도의 고려사항이 있어, 같은 부하를 받는 VS들이 하나의 Virtual Router 그룹으로 묶여 함께 분배됩니다. 그룹의 가중치는 그에 속한 VS들에 영향을 주며, 그룹 단위로 멤버 간 분배가 이뤄집니다.

HA 모드와 고급 클러스터링

High Availability 모드에서는 Virtual Switch도 클러스터의 일원으로 동작해, Active 멤버의 Virtual Switch가 트래픽을 처리하고 페일오버 시 Standby 멤버의 Virtual Switch가 이어받습니다.



① Active VSX Cluster Member ② Standby VSX Cluster Member ③ Virtual Switch Cluster ④ Active Virtual Switch ⑤ Virtual System 1

고급 설정으로는 두 가지가 있습니다. 하나는 모든 VLAN 감시(Monitoring all VLANs) 로, 기본적으로 최고·최저 VLAN ID만 감시하는 동작을 모든 VLAN으로 확장해 더 촘촘하게 연결 장애를 잡아내는 것입니다. 다른 하나는 VSLS 클러스터에서의 CoreXL 설정 으로, 부하 분산 환경에서 코어 자원을 VS들에 맞게 조정하는 것입니다.

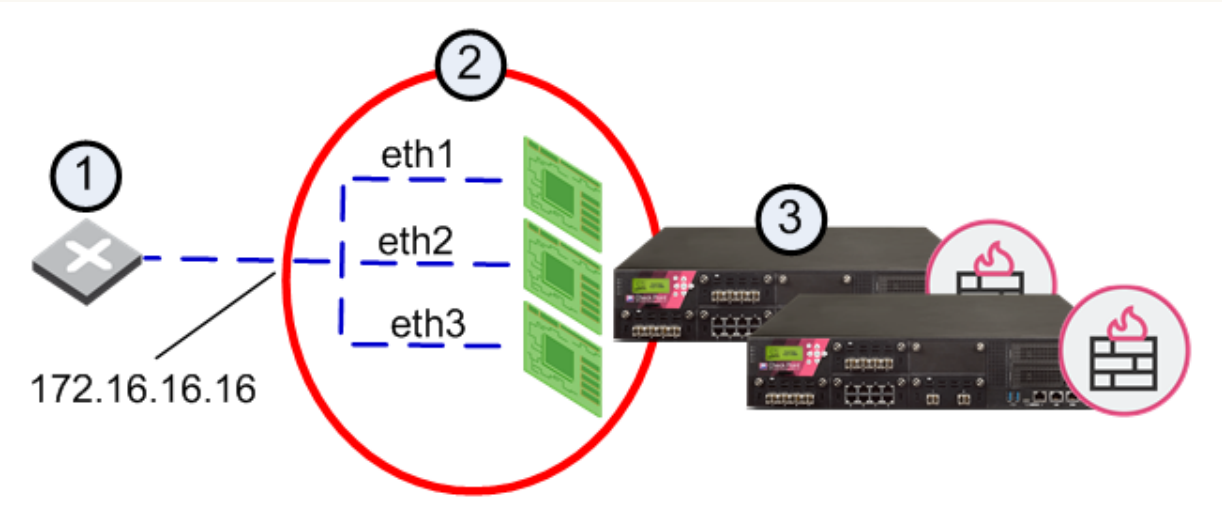
10 링크 집계 (Link Aggregation)

Working with Link Aggregation

링크 집계는 **인터페이스 본딩(bonding)** 이라고도 부르며, 여러 개의 물리 인터페이스를 하나의 가상 인터페이스(bond)로 묶는 기술입니다. 이렇게 묶으면 인터페이스나 링크 장애에 대비한 이중화를 얻거나, 한 물리 인터페이스로는 불가능한 수준으로 처리량을 끌어올릴 수 있습니다.

> *Link aggregation, also known as interface bonding, joins multiple physical interfaces together into a virtual interface, known as a bond interface. A bond interface can be configured for High Availability redundancy or for load sharing, which increases connection throughput above that which is possible using one physical interface.*
>
> — AdminGuide, "Link Aggregation Overview" (p.240)

용어를 먼저 정리하면, **bond**는 하나의 가상 인터페이스처럼 함께 동작하면서 IP와 MAC을 공유하는 물리 인터페이스 묶음이고 클러스터에서 `bond0` 같은 Bond ID로 식별됩니다. 그 묶음을 논리적으로 표현한 것이 **bond 인터페이스**이고, 묶음에 속한 각 물리 인터페이스를 **종속 인터페이스(subordinate interface)** 라 부르는데 종속 인터페이스는 자기 IP가 없고 경우에 따라 같은 MAC을 공유합니다. 하나의 bond에는 종속 인터페이스를 최소 1개에서 최대 8개까지 넣을 수 있으며, 클러스터 멤버마다 동일한 종속 인터페이스를 같은 수로 두는 것이 권장됩니다.

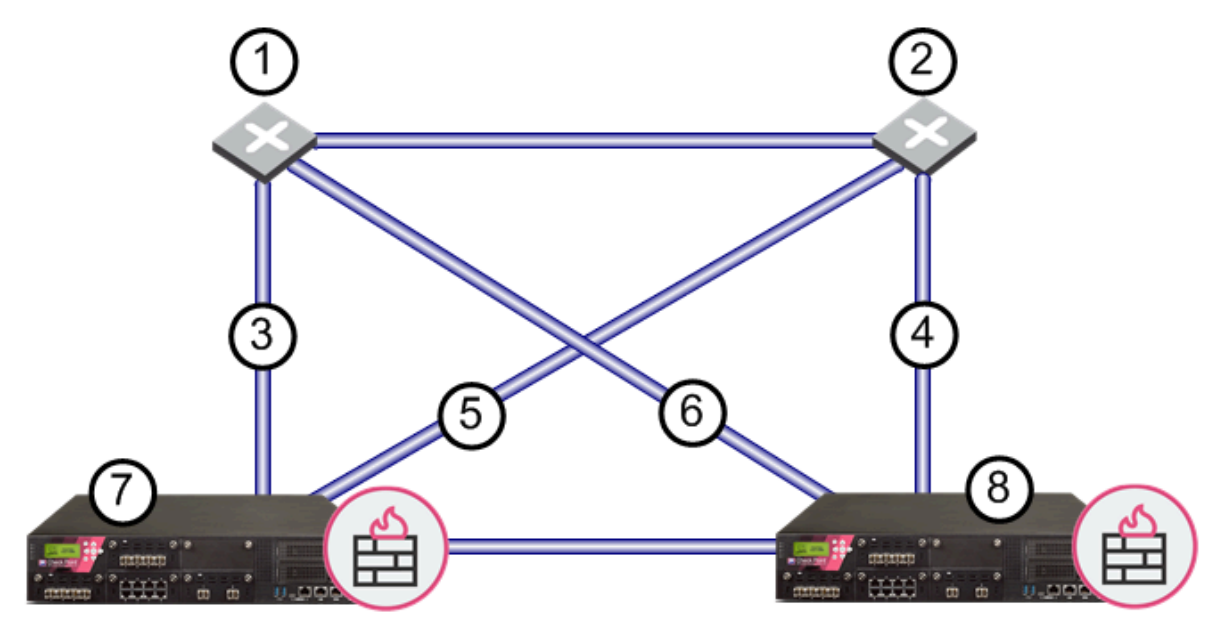


① 스위치 ② bond 0 ③ 클러스터

본딩에는 두 가지 전략이 있습니다. **High Availability(Active/Backup)** 는 인터페이스나 링크가 죽었을 때 이중화를 보장하며 스위치 이중화까지 제공합니다. 반면 **Load Sharing(Active/Active)** 은 모든 인터페이스가 동시에 활성 상태로 서로 다른 연결을 처리해 처리량을 극대화하는데, 스위치 이중화는 지원하지 않습니다.

High Availability 본딩

클러스터는 그 자체로 VSX Gateway 수준의 이중화를 제공하지만, 링크 집계는 여기에 같은 게이트웨이 안의 예비 인터페이스 카드로 자동 페일오버하는 인터페이스·스위치 이중화를 더합니다. HA 본딩에서는 한 번에 하나의 인터페이스만 활성이며, **활성 인터페이스가 감시 중인 인터페이스에서 링크 장애를 감지**하거나 **ClusterXL이 CCP(Cluster Control Protocol) keep-alive 패킷의 송수신 실패를 감지**하면 예비 종속 인터페이스로 페일오버합니다. 이 모드를 ClusterXL과 함께 풀 메시(fully meshed) 토폴로지로 배포하면 NIC와 스위치 양쪽에 독립적인 백업을 두는 세밀한 이중화를 얻습니다.



① 스위치 1 ② 스위치 2 ③ 네트워크 연결 1 ④ 네트워크 연결 4 ⑤ 네트워크 연결 2 ⑥ 네트워크 연결 3 ⑦ VSX Cluster Member 1 ⑧ VSX Cluster Member 2

설정은 VSX Gateway(VS0) 컨텍스트에서 CLI로 하며, 클러스터라면 각 멤버에서 실행합니다. HA를 위해서는 `mode` 파라미터에 `active-backup` 값을 씁니다. 큰 흐름은 본딩 그룹을 만들고, 거기에 종속 인터페이스를 넣고(이때 그 인터페이스들이 다른 설정에 쓰이지 않고 IP도 배정돼 있지 않아야 합니다), 본드가 제대로 구성됐는지 확인한 뒤, SmartConsole에서 클러스터 객체를 설정하는 것입니다. 정확한 명령은 R82 Gaia Administration Guide를 참고합니다.

기존 구성을 본딩으로 바꿀 때는 새로 만든 bond에 연결되도록 Virtual System·Virtual Router·Virtual Switch 객체의 토폴로지를 다시 잡아야 합니다. 객체가 적으면 SmartConsole에서 직접 하면 되지만, Domain과 가상 장치가 많은 대규모 배포에서는 관리 서버에서 `vsx_util change_interfaces` 명령을 쓰는 편이 훨씬 빠릅니다. 예컨대 200개 Domain에 가상 장치가 잔뜩 있는 MDS 환경이라면 이 명령이 모든 해당 객체의 인터페이스를 새 bond로 자동 교체해 줍니다. 다만 MDS에서는 이 작업이 성공하려면 모든 Domain Management Server의 잠금이 풀려 있어야 하므로, 모든 SmartConsole 클라이언트를 먼저 닫아야 합니다.

Load Sharing 본딩

Load Sharing은 인터페이스 이중화에 더해 여러 종속 인터페이스로 트래픽을 분산합니다. 모든 인터페이스가 항상 활성이고, 트래픽은 IEEE 802.3ad 또는 XOR 표준에 따라 분배됩니다. 개별 연결마다 특정 종속 인터페이스가 지정되어 그 연결은 지정된 인터페이스로만 처리되며, 그 인터페이스가 죽으면 트래픽이 다른 인터페이스로 넘어가되 그 인터페이스도 기존 트래픽을 계속 처리합니다.

Load Sharing 모드는 세 가지입니다. **Round Robin**은 활성 종속 인터페이스를 순서대로 선택하고, **802.3ad**는 LACP 프로토콜로 게이트웨이와 스위치 간 링크를 완전히 감시하면서 활성 인터페이스를 동적으로 활용하며, **XOR**은 UP 상태의 모든 종속 인터페이스를 활성으로 두고 전송 해시 정책(L2의 MAC XOR 또는 L3+4의 IP·포트)에 따라 트래픽을 배정합니다. 설정 흐름은 HA와 비슷하되, 본딩 그룹과 종속 인터페이스를 더한 뒤 **임계(critical) 인터페이스 수를 정의**하고 Performance Pack을 쓰면 코어 어피니티를 설정하는 단계가 추가됩니다.

여기서 **임계 필수 인터페이스(Critical Required Interfaces)** 개념이 중요합니다 (ClusterXL에서만 지원). Load Sharing 본드는 살아 있는 종속 인터페이스 수가 임계 최솟값 아래로 떨어지면 다운으로 간주됩니다. 명시하지 않으면 n개 인터페이스 본드의 임계 최솟값은 n-1이라, 인터페이스 두 개만 더 죽어도(n-2가 남으면) 본드 전체가 다운으로 처리됩니다. 더 적은 인터페이스로도 트래픽을 감당할 수 있다면 임계값을 직접 정의해 이중화를 높일 수 있는데, 최대 예상 트래픽 속도를 종속 인터페이스 속도로 나눠 올림한 값이 적절합니다. 이 값은 `$FWDIR/conf/cpha_bond_ls_config.conf` 파일에 `<본드명> <임계 최솟값>` 형식으로 적습니다.

```
# $FWDIR/conf/cpha_bond_ls_config.conf 예시
bond0 5      # 종속 7개 중 3개가 죽으면 다운
bond1 3      # 종속 6개 중 4개가 죽으면 다운
```

최적 성능을 위해서는 **종속 인터페이스를 CPU 코어에 정적으로 고정(affinity)** 하는 것이 좋습니다. 가능하면 인터페이스마다 코어 하나를 전담시키되, 물리 인터페이스가 코어보다 많으면 일부 코어가 둘 이상을 맡습니다. 이때 내부 본드와 외부 본드에서 같은 위치에 있는 종속 인터페이스를 한 쌍으로 묶어 같은 코어가 처리하게 하는 것이 요령입니다. 본드 내 종속

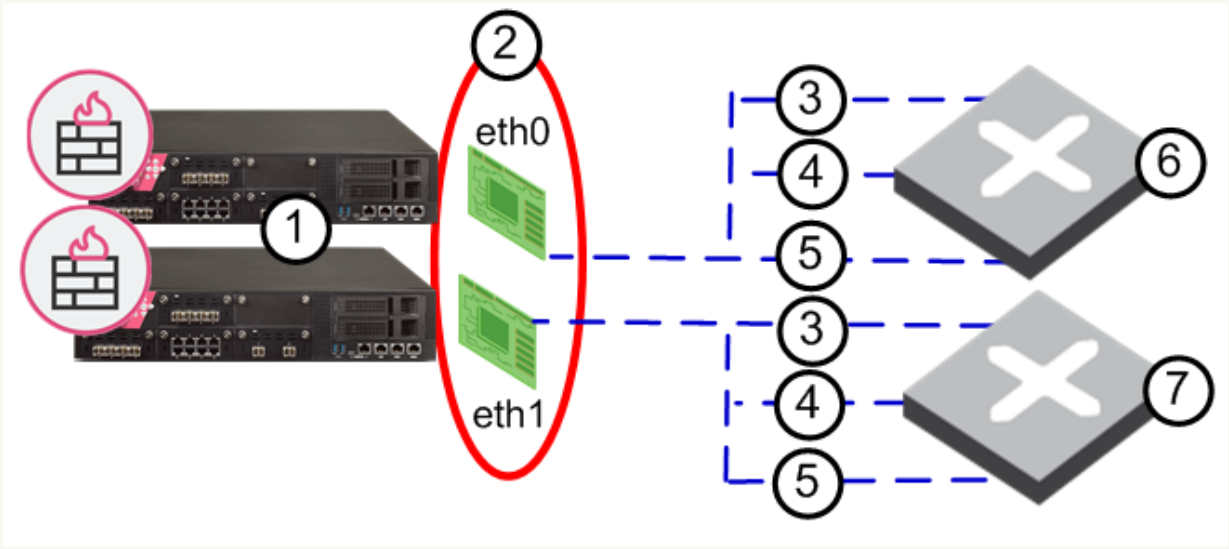
인터페이스의 순서는 `cat /proc/net/bonding/<본드명>`으로 확인할 수 있고, 두 본드에서 같은 위치에 나타나는 인터페이스가 짝입니다.

페일오버는 어떻게 일어나는가

본드 페일오버는 두 경우에 일어납니다. 활성 인터페이스가 감시 중인 다른 인터페이스의 링크 장애를 감지하거나, ClusterXL이 CCP keep-alive 패킷 송수신 실패를 감지하는 경우입니다. 상황에 따라 본드 안의 다른 종속 인터페이스로 넘어가거나 클러스터 멤버 간 페일오버가 일어납니다(이 동작에는 MII 표준을 지원하는 NIC가 필요합니다).

링크 상태로 시작되는 페일오버는 이렇게 진행됩니다. 활성 종속 인터페이스가 다운 링크를 감지하면 본드가 예비 인터페이스로 넘기는데, 이는 본드 내부의 페일오버라 다른 멤버 상태에는 영향이 없습니다. 다만 살아 있는 종속 인터페이스 수가 임계 최솟값 아래로 떨어지면 그때 다른 클러스터 멤버로 페일오버하고, 예비 인터페이스마저 계속 링크 장애를 보이면 멤버 간 페일오버가 일어납니다. CCP로 시작되는 페일오버는 다른 멤버가 죽지 않았을 때만 일어나는데, ClusterXL이 CCP 패킷 문제를 감지해 내부 본드 페일오버를 한 뒤, 3분 안에 추가 문제가 감지되면 다른 멤버로 페일오버합니다.

VLAN에 대해서도 페일오버가 지원됩니다. ClusterXL은 VLAN ID의 연결 실패나 통신 오류를 감시해 필요할 때 페일오버하는데, 기본적으로는 가장 높은 VLAN ID와 가장 낮은 VLAN ID만 감시합니다. 대부분의 배포에서는 트래픽 부담이 가벼워 이 설정이 바람직하지만, 이 경우 최고·최저 VLAN ID에 대한 스위치 설정 문제만 감지된다는 한계가 있습니다. 필요하면 모든 VLAN을 감시하도록 설정할 수 있습니다.



① 클러스터 ② bond 0 ③ VLAN 1 ④ VLAN 2 ⑤ VLAN 3 ⑥ S-1 ⑦ S-2

Cisco 스위치 설정

Load Sharing 모드를 쓰려면 스위치 쪽도 맞춰야 합니다. 아래는 Cisco 스위치 예시이며 (정확한 명령은 모델·OS 버전별 Cisco 문서 참고), 802.3ad(LACP)와 XOR의 차이는 채널 그룹 모드가 `active` 냐 `on` 이냐입니다.

```
# 802.3ad (LACP)
Switch(config)# port-channel load-balance src-dst-ip
Switch(config-if)# channel-group 1 mode active
Switch(config-if)# channel-protocol lacp

# XOR
Switch(config)# port-channel load-balance src-dst-ip
Switch(config-if)# channel-group 1 mode on
```

문제 해결

본드에 문제가 의심되면 Expert 모드에서 `cat /proc/net/bonding/<bond id>` 로 상태를 봅니다. 문제가 있으면 물리 링크가 죽었는지 확인하는데, Gaia Clish에서는 `show cluster bond name <본드명>` , Expert 모드에서는 `cphaprob show_bond <본드명>` 으로 링크 상태가 no인 종속 인터페이스를 찾아 케이블·하드웨어·스위치 포트 설정을 점검합니다. 클러스터 멤버가 죽었는지는 `show cluster state` (Clish)나 `cphaprob state` (Expert)로 확인하고, Active가 아닌 멤버가 있으면 R82 ClusterXL Administration Guide를 참고합니다.

운영상 기억할 점은, 본드 모드를 바꾸거나 종속 인터페이스를 본드에 추가하면 멤버 재부팅이 필요하다는 것입니다(반대로 종속 인터페이스를 제거할 때는 재부팅이 필요 없습니다). 모든 본드 상태 변화와 페일오버는 Logs & Events에 기록됩니다.

```
cat /proc/net/bonding/<bond id>     # 본드 상태
cphaprob show_bond <bond_name>      # 종속 인터페이스 링크 상태 (Expert)
cphaprob state                      # 클러스터 멤버 상태 (Expert)
```

마지막으로 스위치 쪽 주의가 하나 있습니다. 일부 스위치에서는 내부 본드 페일오버 시 연결 지연이 생기는데, 해당 인터페이스의 auto-negotiation을 끄거나 일부 Cisco 스위치에서 **PortFast**를 켜면 시작 시간을 줄일 수 있습니다. 다만 PortFast는 다른 스위치나 허브로 연결되는 포트에는 절대 쓰면 안 됩니다. 그런 경우 STP 초기화가 완료되어야 하는데, PortFast를 쓰면 물리 루프가 생겨 패킷이 계속 돌거나 증식해 결국 네트워크가 마비될 수 있기 때문입니다(본드가 스위치에서 Trunk로 설정된 경우엔 PortFast가 적용되지 않습니다).

11 VSX 성능 최적화

Optimizing VSX

이 챕터는 VSX의 성능을 모니터링하고 최적화하는 기능들을 다룹니다. 메모리와 CPU 자원을 들여다보는 방법, SNMP로 가상 장치를 감시하는 방법, 그리고 Jumbo Frame 설정이 핵심입니다.

메모리 자원 보기 — vsx mstat

`vsx mstat` 명령은 VSX Gateway가 쓰는 메모리를 전체적으로 보여 줍니다(Expert 모드에서 실행). 시스템과 각 가상 장치가 얼마나 메모리를 쓰는지 한눈에 볼 수 있는데, 전역 자원으로는 전체 물리 메모리(Memory Total)와 가용 메모리(Memory Free), 전체·가용 스왑 메모리, 그리고 초당 스왑 횟수(Swap-in rate)가 나옵니다. 가상 장치는 VSID 순으로 나열되며, 어떤 VSID가 어떤 장치인지는 `vsx stat -v`로 확인합니다.

같은 정보를 `cpview`로도 볼 수 있습니다. VS0 컨텍스트에서 `cpview`를 실행한 뒤 Advanced > VSX > VSs > Physical-Resources 로 들어가면 Virtual System과 Virtual Router의 메모리 소비가 표시됩니다.

```
vsx mstat          # VSX Gateway 메모리 개요 (Expert 모드)
vsx stat -v        # 어떤 VSID가 어떤 장치인지 확인
cpview             # VS0 컨텍스트 → Advanced > VSX > VSs > Physical-Resources
```

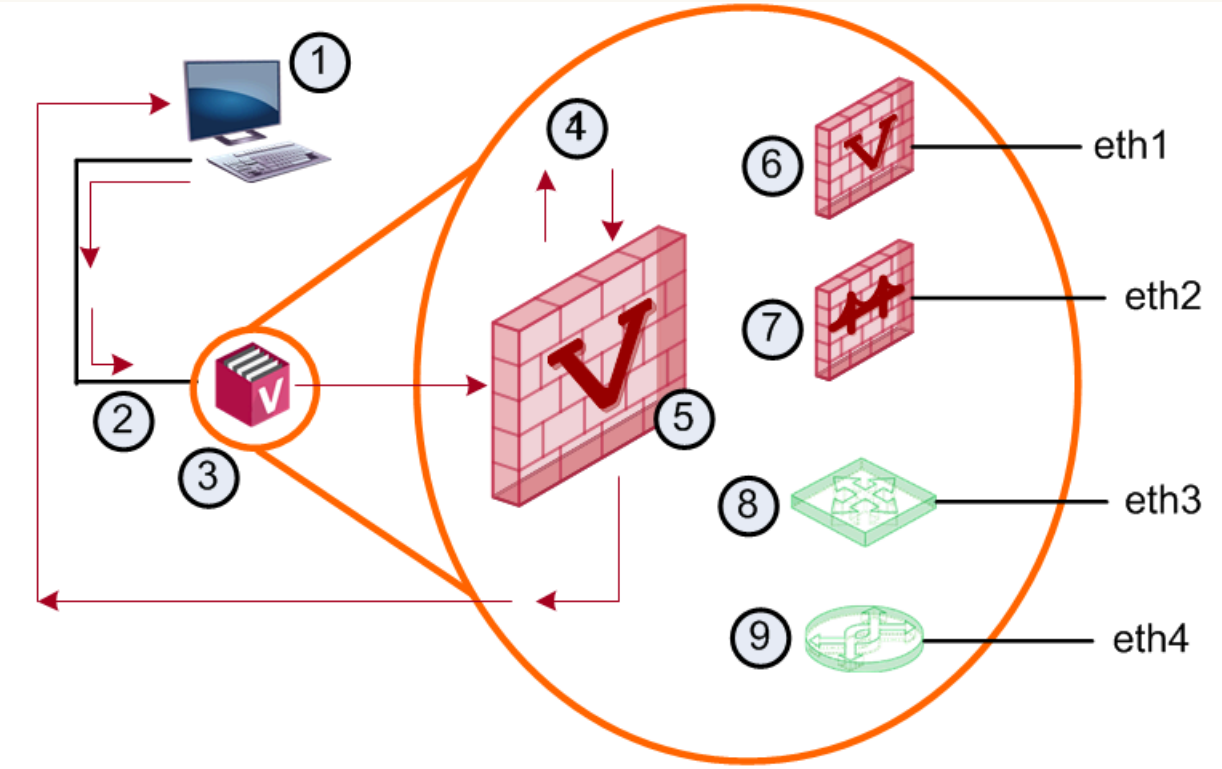
CPU 자원 보기

R80.40부터 **CPU Resource Control이 CPView에 통합**되었습니다. 먼저 VS0 컨텍스트로 가는데, Expert 모드에서는 `vsenv` , Gaia Clish에서는 `set virtual-system 0` 을 쓴 뒤 `cpview` 를 실행하고 같은 Physical-Resources 탭으로 들어가면 됩니다. 여기서 한 가지 CPU % 컬럼의 읽는 법을 알아야 합니다. 이 값은 `top` 명령처럼 **단일 CPU 기준의 백분율**이라, 여러 코어에 걸친 사용량이 합산됩니다. 예를 들어 코어가 4개인데 VS1이 CPU0의 30%, CPU1의 40%, CPU2의 50%, CPU3의 10%를 쓰면 CPU % 컬럼에는 130%로 표시됩니다. 따라서 게이트웨이 전체의 CPU 사용률을 보려면 Total Resource Consumption의 CPU % 값을 코어 수로 나눠야 합니다.

SNMP 모니터링

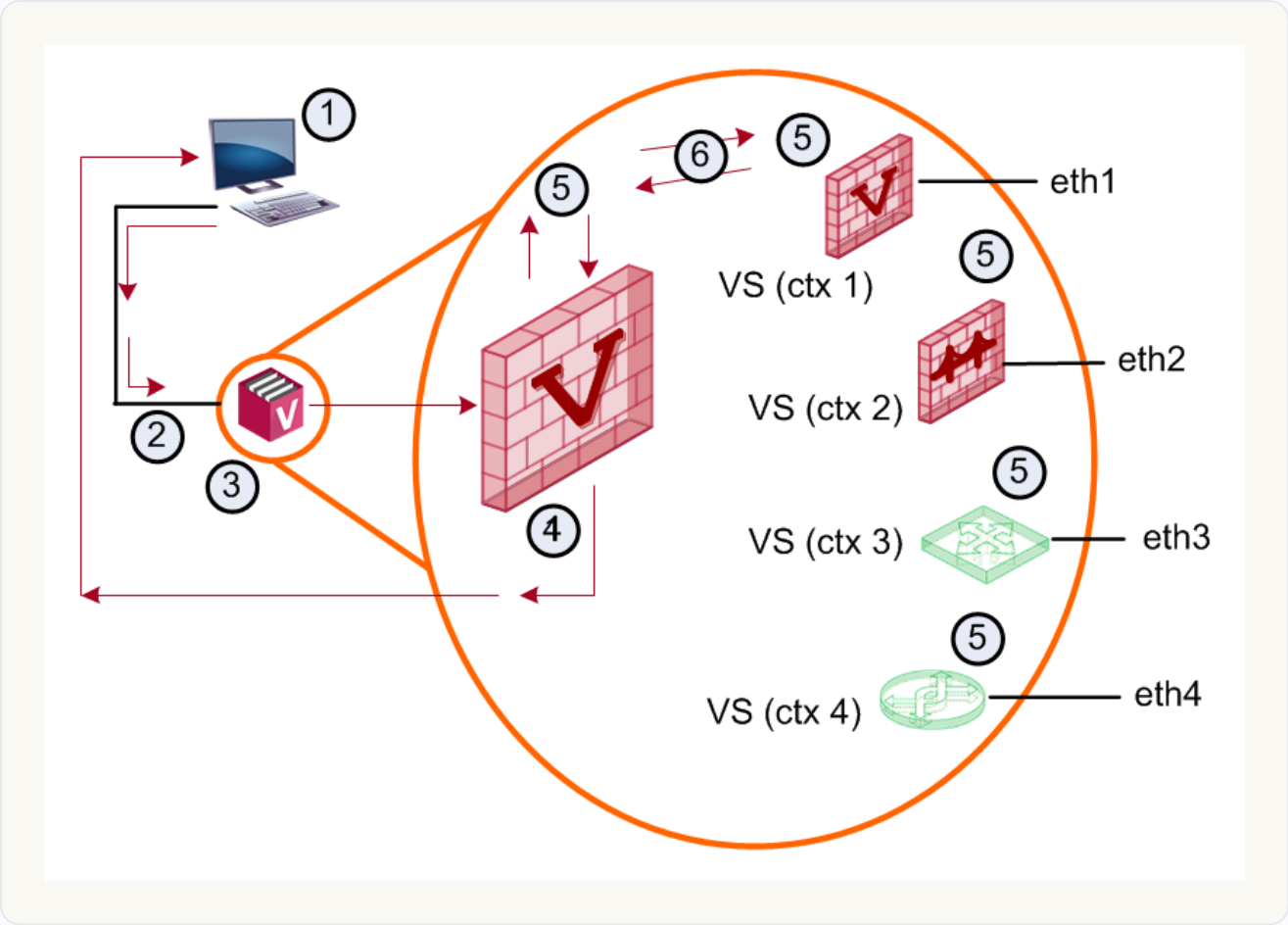
VSX는 SNMP v1·v2c·v3를 모든 모드에서 지원하며, 모드는 세 가지입니다. 모드의 차이는 SNMP 데몬이 어디서 도느냐, 그리고 어느 IP로 질의하느냐에 있습니다.

Default 모드에서는 SNMP 데몬이 VS0(VSX Gateway)에서만 돕니다. VS0의 데몬이 각 가상 장치별 카운터 테이블(VSX SNMP 트리)을 갖고 있고, 모든 질의는 VSX Gateway IP로 보냅니다.



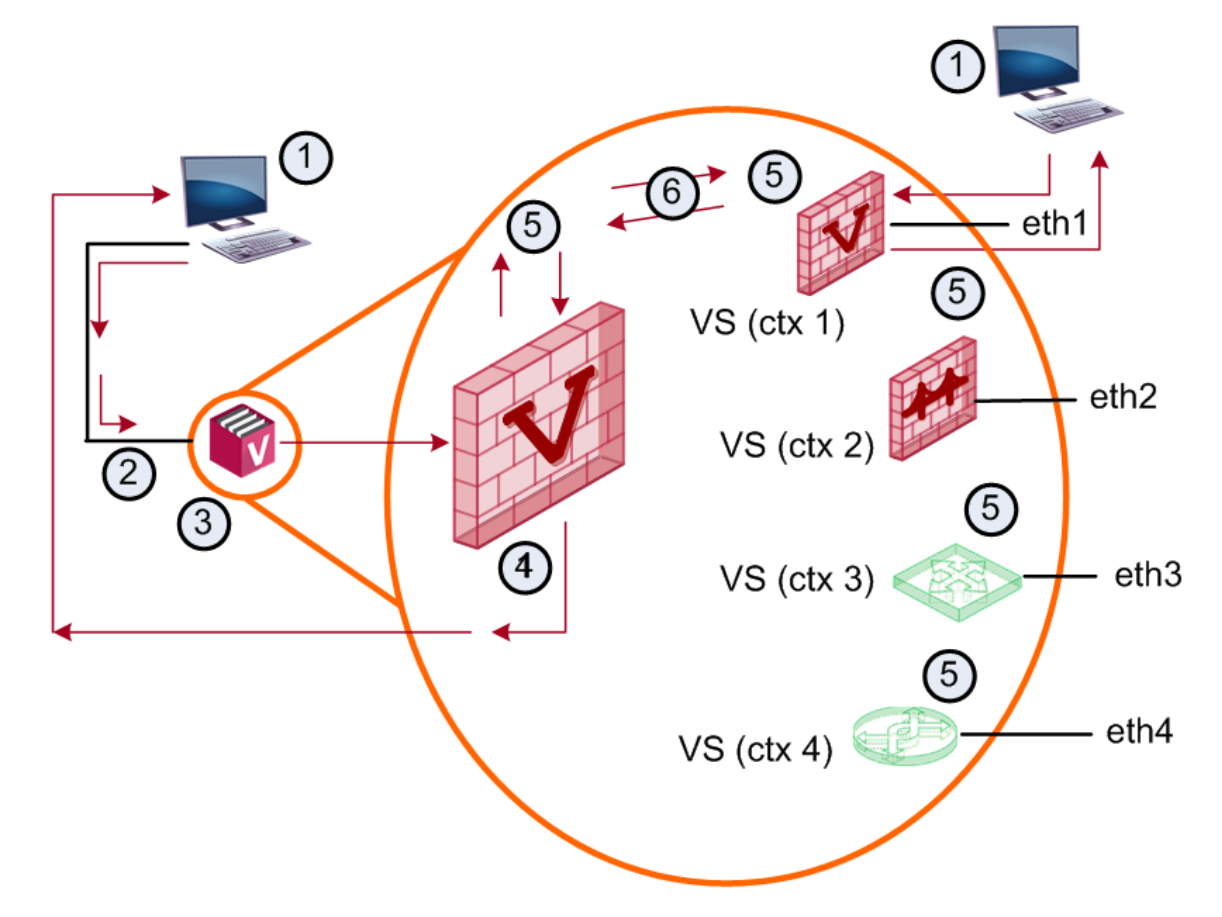
① VSX Gateway로 SNMP 요청을 보내는 SNMP 서버 ② eth0 ③ VSX Gateway ④ SNMP 데몬 ⑤ Virtual System 0 ⑥ Virtual System 1(ctx 1) ⑦ Bridge 모드 Virtual System(ctx 2) ⑧ Virtual Router(ctx 3) ⑨ Virtual Switch(ctx 4)

VS 모드에서는 각 가상 장치마다 자기 컨텍스트에서 별도의 SNMP 데몬이 돕니다. 질의는 여전히 VS0 IP(게이트웨이 인터페이스)로 보내되 VS ID를 함께 지정하면, 질의가 해당 장치로 전달되고 응답도 같은 게이트웨이 인터페이스로 돌아옵니다. VS 모드를 켜도 Default 모드 질의 기능은 줄어들지 않습니다.



① 질의 호스트 ② eth0 ③ VSX Gateway ④ VS 0 ⑤ SNMP 데몬 ⑥ UDS

vs-direct-access 모드는 VS 모드가 켜져 있을 때만 쓸 수 있으며, VS0뿐 아니라 Virtual System이나 Virtual Router 자신의 IP로 직접 질의할 수 있게 합니다.



① 질의 호스트 ② eth0 ③ VSX Gateway ④ VS 0 ⑤ SNMP 데몬 ⑥ UDS

다만 클러스터에서는 Virtual IP로만 질의할 수 있고, IP를 가진 가상 장치만 질의 가능하므로 Virtual Switch나 Virtual Bridge는 대상이 아닙니다.

설정은 Gaia Clish에서 SNMP 에이전트를 켠 뒤 모드를 고르는 식입니다. Default는 `set snmp mode default`, VS 모드는 `set snmp mode vs`, 직접 접근은 거기에 `set snmp vs-direct-access on`을 더합니다. SNMP v3 질의를 쓰려면 USM 사용자를 정의해야 하는데, VSX에서는 USM 사용자에게 허용할 가상 장치를 `set snmp usm user <이름> vsid <VSID>`로 지정해야 합니다(기본적으로 허용 장치가 없습니다). vs-direct-access를 켜면 Virtual System이 모든 인터페이스에서 SNMP 질의를 받으므로, 특정 인터페이스를 막고 싶으면 정책에 SNMP 차단 규칙을 추가합니다.

```
# SNMP 모드 설정 (Gaia Clish)
set snmp agent on
set snmp mode vs                  # 또는 default
set snmp vs-direct-access on      # 직접 접근까지 쓸 때
set snmp usm user admin vsid 2,15   # v3 USM 사용자에 허용 VS 지정
```

질의 예시를 보면 모드와 버전에 따라 형태가 다릅니다. VS0 IP로 v3 질의를 할 때는 `snmpwalk -n vsid2 -v 3 ...` 처럼 SNMP 컨텍스트 이름으로 `vsidN` 을 지정하고, v1/v2c 질의를 할 때는 `snmpwalk -v 1 -c public_2 ...` 처럼 커뮤니티 이름 뒤에 VSID나 VS 이름을 접미사로 붙입니다(이런 접미사 커뮤니티는 자동 생성됩니다). 가상 장치 IP로 직접 질의할 때는 그 장치의 Virtual IP를 대상으로 보내며, 활성 장치만 응답합니다.

```
# VS0 IP로 v3 질의 (vsidN = SNMP 컨텍스트)
snmpwalk -n vsid2 -v 3 -l authNoPriv -u admin -A abcd1234 192.0.2.5 ifDesc

# VS0 IP로 v1 질의 (커뮤니티에 _VSID 접미사)
snmpwalk -v 1 -c public_2 192.0.2.5 ifDescr

# 가상 장치 IP로 직접 질의 (vs-direct-access)
snmpwalk -v 2c -c public 192.0.2.81 ifDescr
```

v1/v2c에서는 커뮤니티 접미사 충돌에 주의해야 합니다. 예를 들어 읽기 전용 커뮤니티가 `private` 이고 읽기-쓰기 커뮤니티가 `private_1` 이면, VSID 1용으로 `private_1` 과 `private_1_1` 이 자동 생성되어 `private_1` 이 모호해지고 예기치 않게 동작할 수 있습니다.

클러스터에서 질의할 때는 응답 주체에 주의합니다. VSLS 클러스터를 VS0 Virtual IP로 질의하면 Active 멤버의 VS0가 응답하고, Standby 멤버에서 Active인 Virtual System은 응답하지 않습니다. Standby 멤버의 Active VS를 질의하려면 그 멤버의 실제 IP를 써야 합니다. 또한 **SNMP 트랩은 VS0에서만** 제공된다는 점도 기억해야 합니다.

가상 장치에서 정보를 얻으려면 SNMP 브라우저에 체크포인트 MIB 파일을 불러와야 합니다. MIB 파일은 VS0 컨텍스트의 `$CPDIR/lib/snmp/chkpnt.mib` 에 있고, VSX의 OID는 `.1.3.6.1.4.1.2620.1.16` 입니다. 예컨대 VSX 상태 테이블은 `vsxStatusTable` , 메모리 사용 테이블은 `vsxStatusMemoryUsageTable` 로 질의합니다. 카운터 테이블(`vsxCountersTable`)

의 갱신 주기는 `$FWDIR/conf/amon_vsx_refresh_interval` 파일에서 정하며 기본값은 30초입니다.

Jumbo Frame 설정

VSX는 Jumbo Frame을 지원하며 NIC 드라이버의 최대 MTU까지 설정할 수 있습니다. 설정은 어느 장치의 MTU를 바꾸느냐에 따라 다릅니다.

> *VSX supports Jumbo Frames and lets you configure up to the maximum MTU of the NIC driver.*
>
> — AdminGuide, "Configuring Jumbo Frames" (p.268)

Virtual Switch의 경우, Virtual Switch의 MTU를 바꾸면 거기에 연결된 Warp Link와 인터페이스가 모두 자동으로 새 값으로 변경됩니다. 그래서 Virtual System 쪽에서는 Warp Link의 MTU를 따로 설정할 수 없습니다. SmartConsole에서 Virtual Switch 객체를 열어 Topology에서 인터페이스를 편집하고 General 탭의 MTU를 바꾼 뒤 세션을 publish하면 됩니다.

Virtual Router나 Bridge 모드 Virtual System도 같은 방식으로 객체를 열어 Topology에서 인터페이스의 MTU를 설정합니다. 다만 이 둘은 설정 후 해당 객체에 Access Control 정책을 설치해야 적용됩니다(Virtual Switch는 publish만으로 충분). Virtual Router의 Warp Link MTU를 바꾸려면 연결된 Virtual System 쪽 설정에서 변경합니다.

12 Bridge 모드

Bridge Mode

Bridge 모드는 IP 라우팅 대신 L2 브리징을 그대로 수행하는 방식입니다. 가장 큰 장점은 **기존 IP 구조를 건드리지 않고 Virtual System을 끼워 넣을 수 있다**는 것입니다.

> *By implementing native Layer 2 bridging instead of IP routing, you can add Virtual Systems without adversely affecting the existing IP structure.*
>
> — AdminGuide, "Bridge Mode" (p.270)

Bridge 모드에서는 Virtual System의 인터페이스에 IP 주소가 필요 없습니다. 다만 인터페이스가 아니라 Virtual System 자체에는 선택적으로 IP를 줄 수 있는데, 이렇게 하면 네트워크 장애를 감지하는 L3 모니터링이 가능해집니다.

VSX는 두 가지 Bridge 모드를 지원합니다. **Active/Active Bridge 모드**는 중복 스위치 사이의 원치 않는 루프를 막으면서 이중화를 제공하고, **Active/Standby Bridge 모드**는 경로 이중화와 루프 방지를 제공하면서 VSLS를 매끄럽게 지원하고 STP의 여러 한계를 넘어섭니다.

Active/Active 모드 — STP 기반

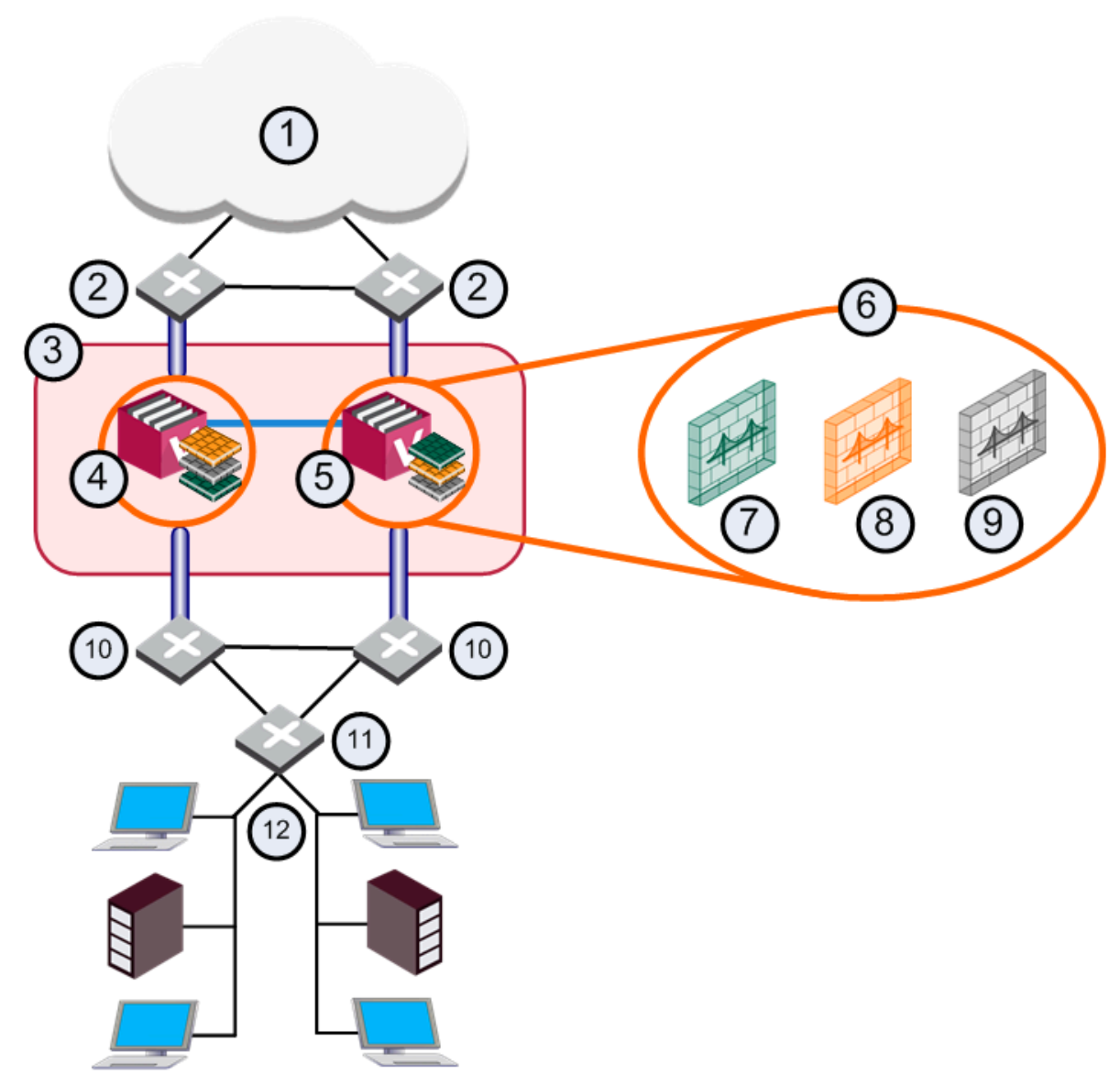
Active/Active 모드는 고속 스위치 네트워크에서 루프를 막는 업계 표준인 **STP(Spanning Tree Protocol)** 에 기댑니다. 따라서 이 모드를 쓰려면 네트워크에 STP가 배포되고 제대로 설정되어 있어야 합니다. 지원하는 L2 프로토콜은 802.1d, 802.1q, 802.1s, 802.1w, 그리고 PVST+입니다. STP를 네트워크 하드웨어에 어떻게 배포·설정하는지는 해당 벤더 문서를 참고하면 됩니다.

Active/Standby 모드

Active/Standby 모드는 **High Availability를 크게 개선**하고, VSX Cluster 환경에서 **VSLS의 처리량을 Virtual System들에 분산**시킵니다. 구체적인 이점은 즉각적인 페일오버, 관리자가 브리지 페일오버를 더 잘 통제할 수 있다는 점, VSLS 지원, 그리고 VLAN 변환입니다. 반면 가장 큰 한계는 **STP 트리 구조를 깬다**는 점입니다. 그래서 이 모드로 Virtual System을 설정할 때는 스위치의 STP 데이터베이스에서 해당 Virtual System의 VLAN을 빼두어야 트렁크 인터페이스 복구 지연을 막을 수 있습니다.

이 모드는 대규모 고트래픽 환경의 **3계층 계층 모델**에 자연스럽게 들어맞습니다. 인터넷·외부망과 트래픽을 주고받는 고속 백본 스위치의 코어 계층, 코어와 액세스 계층을 잇는 라우터들의 분배 계층, 내부망과 트래픽을 주고받는 이중화 LAN 스위치의 액세스 계층으로 나뉘는데, Active/Standby Bridge 모드의 VSX는 분배 계층에 들어가 보안 정책을 적용합니다. 라우터가 분리된 VLAN으로 외부 트래픽을 알맞은 Virtual System에 보내면, 검사된 트래픽은 또 다른 분리된 VLAN으로 빠져나가 라우터를 거쳐 내부 목적지로 향합니다.

표준 STP 설정으로는 동작하지 않는 **VLAN 공유 인터페이스 배포**도 이 모드에서 가능합니다. 각 VSX Cluster 멤버가 VLAN 트렁크로 한 쌍의 이중화 스위치에 연결되고, 한 멤버 안의 모든 Virtual System이 같은 VLAN 트렁크를 공유하는 구성입니다. High Availability에서는 관리자가 정한 우선순위와 상태에 따라 트래픽이 멤버로 향하고, VSLS에서는 VSLS 설정에 따라 부하가 멤버들 사이에 분산됩니다.



① 인터넷 ② 외부 이중화 스위치 ③ VSX Cluster ④ VSX Cluster Member 1 ⑤ VSX Cluster Member 2 ⑥ Bridge 모드 Virtual System들 ⑦ VS1 = Active ⑧ VS2 = Standby ⑨ VS3 = Backup ⑩ 내부 이중화 스위치 ⑪ VLAN 스위치 ⑫ 내부 네트워크

Virtual System을 Bridge 모드로 설정하기

여기서 가장 중요한 제약은, Bridge 모드는 Virtual System을 처음 만들 때 정해야 한다는 점입니다.

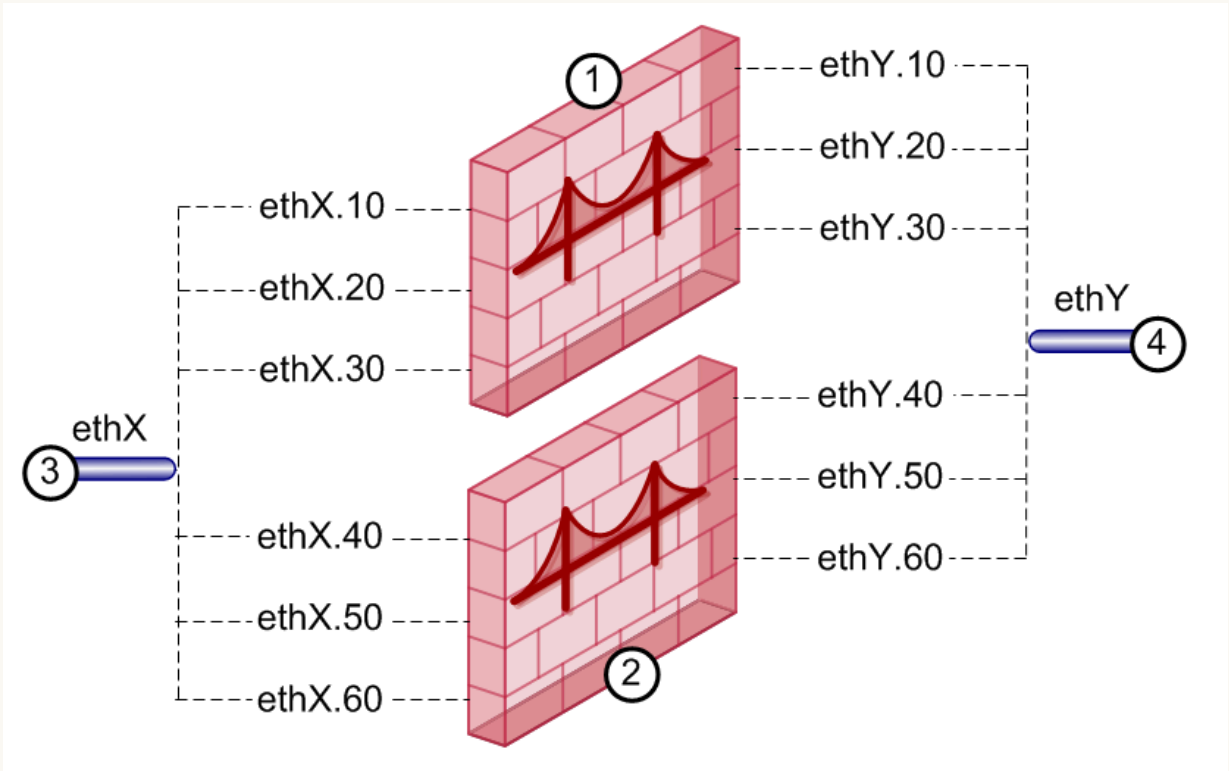
> *You cannot configure an existing non-Bridge Mode Virtual System to work in the Bridge Mode.*
>
> — AdminGuide, "Configuring Virtual Systems for the Active/Standby Bridge Mode" (p.275)

설정 자체는 단순합니다. 새 Virtual System 객체의 General Properties에서 **Bridge Mode**를 선택하고 다음으로 넘어가면 네트워크 설정 창이 열리는데, 여기서 외부·내부 인터페이스를 설정합니다. 필요하면 **Enable Layer 3 Bridge Interface Monitoring**을 켜는데, 이때 주는 IP는 고유해야 하고 보호 대상 네트워크와 같은 서브넷이어야 합니다. 그리고 마치면 됩니다. 참고로 R81부터는 `vsx_provisioning_tool` 로, R81.10부터는 SmartConsole로 **일반 Virtual System에도 브리지 인터페이스를 추가**할 수 있게 되었습니다.

Multi Bridge — 한 VS로 여러 VLAN 브리징

Multi Bridge는 하나의 Bridge 모드 Virtual System을 통해 여러 VLAN의 트래픽을 흐르게 하는 기능으로, VSX Gateway와 Active/Active·Active/Standby Bridge 모드 클러스터에서 지원됩니다. Bridge 모드 Virtual System에 물리·VLAN 인터페이스를 추가하다가 VLAN 인터페이스를 두 개보다 많이 추가하면 Multi Bridge가 자동으로 켜집니다. 브리징하려는 두 인터페이스에는 같은 VLAN 태그를 설정하면 됩니다.

여기에는 지켜야 할 규칙이 있습니다. 모든 인터페이스가 VLAN이어야 하고, 브리지는 두 VLAN 트렁크 사이에서만 만들 수 있으며, 하나의 Multi Bridge에 최대 64쌍의 VLAN 인터페이스를 넣을 수 있습니다. 그리고 그 두 VLAN 트렁크는 서로 짝으로만 함께 써야 하고, 다른 Bridge 모드 Virtual System의 다른 트렁크와 섞어 쓸 수 없습니다. 예를 들어 eth1.10, eth2.10, eth1.20, eth2.20을 정의하면 eth1과 eth2 트렁크는 묶인 것이라, eth1.30을 eth3.30과 브리징할 수는 없습니다. 참고로 Multi Bridge 인터페이스의 이름은 `brXXXXXX` (X는 숫자) 형태로 붙습니다.



① VLAN 태그 10·20·30으로 브리지 3개를 가진 Bridge 모드 VS ② VLAN 태그 40·50·60으로 브리지 3개를 가진 Bridge 모드 VS ③·④ VLAN 트렁크(서로 짝으로만 사용)

새 Multi Bridge를 만들려면 해당 Virtual System을 관리하는 Security Management Server나 Target Domain Management Server에 SmartConsole로 접속해 새 Virtual System 객체를 만들고 **Bridge Mode**를 선택한 뒤, Interfaces 섹션에서 VLAN 인터페이스를 쌍으로 추가합니다. 이때 각 쌍의 두 인터페이스는 서로 다른 트렁크에서 온 같은 VLAN 태그여야 하고(예: `eth2.50/eth2.51`, `eth3.50/eth3.51`), 같은 두 VLAN 트렁크를 써야 합니다. 다 만들면 Access Control 정책을 설치합니다. 기존 브리지를 Multi Bridge로 바꿀 때는 객체를 열어 Bridge Configuration > Topology에서 물리 인터페이스가 있으면 제거하고 VLAN 인터페이스 쌍을 더 추가한 뒤 정책을 설치하면 됩니다.

VSX 클러스터를 Bridge 모드로 켜기

새 VSX 클러스터를 Active/Standby Bridge 모드로 만들려면, 각 멤버의 Gaia 최초 설정 마법사에서 **ClusterXL**을 선택합니다. 이후 VSLS에 필요한 Per Virtual System State 기능을 켰다면 Active/Standby Bridge 모드가 자동으로 켜지고, 켜지 않았다면 각 멤버의 CLI에서 `cpconfig`를 실행해 Active/Standby Bridge 모드 옵션을 활성화합니다. 그다음 SmartConsole에서 VSX 클러스터 객체를 열어 Other > VSX Bridge Configuration으로 가서 **Check Point ClusterXL**을 선택하면 ClusterXL의 루프 감지 알고리즘이 켜지고, 마지막으로 `<클러스터명>_VSX` 정책을 설치합니다.

기존 클러스터를 Active/Standby로 바꿀 때도 같은 화면에서 **Check Point ClusterXL**을 고르고 정책을 설치한 뒤, 각 멤버에서 `cpconfig`로 "Enable ClusterXL for Bridge Active/Standby"를 선택하고 재부팅합니다. 반대로 Active/Active로 바꾸려면 VSX Bridge Configuration에서 **Standard Layer 2 Loop Detection Protocols**를 고르고 정책을 설치한 뒤, 각 멤버에서 `cpconfig`로 "Disable ClusterXL for Bridge Active/Standby"를 선택하고 재부팅하면 됩니다.

```
cpconfig    # 멤버 CLI에서 Bridge Active/Standby 활성/비활성 선택 후 재부팅
```

13 VSX 진단과 트러블슈팅

VSX Diagnostics and Troubleshooting

VSX에서 문제가 생겼을 때 원인을 찾아가는 기본 진단 절차와, 자주 만나는 몇 가지 알려진 문제와 해결책을 다루는 챕터입니다. 대부분의 문제는 VSX Gateway·클러스터·가상 장치를 정의하는 과정의 설정 오류에서 비롯되고, 그다음으로 흔한 것이 네트워크 연결 문제입니다.

> *Most problems are caused by configuration errors occurring during the process of defining VSX Gateway, clusters and/or Virtual Devices.*
>
> — AdminGuide, "VSX Diagnostics and Troubleshooting" (p.283)

문제들은 보통 마주치게 되는 순서대로 정리되어 있습니다. 그래서 어떤 해결책을 시도하기 전에 그 앞 단계들이 모두 성공했는지 먼저 확인하는 것이 중요합니다. 초기의 한 문제가 뒤 단계의 여러 문제를 줄줄이 일으키는 경우가 많기 때문에, 무엇이 잘못됐는지 따질 때는 **근본 원인**을 찾는 데 집중해야 합니다.

> IMPORTANT
>
> SmartView Monitor에서 Firewall History와 System History 카운터는 Virtual System의 데이터를 보여 주지 않습니다.

기본 진단 순서

VSX 설정에 문제가 의심되면 다음 흐름으로 원인을 좁혀 갑니다. 여기서 쓰는 명령들은 명령어 레퍼런스 챕터에 자세히 나옵니다.

가장 먼저 각 VSX Gateway나 클러스터 멤버에서 `vsx stat -v`로 기본 설정을 점검합니다. 이 한 명령으로 모든 Virtual System이 빠짐없이 있는지, 모든 가상 장치가 Active인지, 각 VS에 올바른 보안 정책이 설치됐는지, 관리 서버와 SIC 신뢰가 맺어졌는지를 한 번에 확인할 수 있습니다. 이어서 각 VSX Gateway와 클러스터 멤버, 관리 서버에서 `cplic print`로 적절한 라이선스가 깔려 있는지 확인하고, 각 클러스터 멤버에서 `cphaprob stat`로 상태를 봅니다. 이때 멤버가 Active·Standby·Backup이 아닌 다른 상태로 나오면 R82 ClusterXL Administration Guide의 트러블슈팅 챕터를 참고합니다.

특정 Virtual System의 연결에 문제가 의심되면 반드시 `vsenv <VSID>`로 그 VS의 컨텍스트에 먼저 들어간 다음 진단해야 합니다. 컨텍스트를 맞춘 뒤 `fw getifs`로 그 VS의 인터페이스 목록을 보고, `ping`·`traceroute`·`tcpdump`·`ip route`·`ip link` 같은 표준 OS 도구로 연결 상태를 살핍니다. 이 도구들 중 일부는 **현재 컨텍스트(라우팅·출발지·목적지 IP)에 따라 다르게 동작**한다는 점을 기억해야 합니다.

인터페이스와 라우터가 모두 정상으로 보이는데도 문제가 남으면, 패킷이 시스템을 통과하는 과정을 직접 들여다봅니다. `fw monitor -v <VSID>`로 여러 지점에서 패킷을 캡처하면 같은 패킷이 여러 캡처 지점을 지나며 여러 번 보고되는데, 이 명령은 외부 Virtual Router로 향하는 패킷을 빼면 Virtual Router에 대해서는 보고하지 않습니다. 마지막으로 `tcpdump`로 Warp 인터페이스를 포함한 특정 인터페이스의 송수신 패킷을 보면 연결 문제의 단서를 자주 얻을 수 있습니다.

```
vsx stat -v                 # VS 목록·Active 여부·정책·SIC 한 번에 점검
cplic print                 # 라이선스 확인
cphaprob stat               # 클러스터 멤버 상태
vsenv <VSID>                # 진단 전 대상 VS 컨텍스트로 전환
fw getifs                   # 해당 VS의 인터페이스 목록
fw monitor -v <VSID>        # 여러 지점에서 패킷 캡처
tcpdump -i <interface>      # Warp 포함 특정 인터페이스 송수신 캡처
```

자주 만나는 문제들

VSX Gateway나 클러스터 멤버의 SIC 신뢰가 안 맺어질 때가 첫 번째입니다. VSX Gateway를 만들 때 `Certificate cannot be pushed. Connection error with wait agent.` 같은 오류가 나오는 경우인데, 먼저 VSX 쪽에서 관리 서버로 ping을 날려 연결을 확인하되 반드시 컨텍스트가 vrf 0인 상태에서 해야 합니다(반대로 관리 서버에서 VSX Gateway로 보내는 ping은 기본 정책 때문에 안 됩니다). 그래도 안 되면 케이블·경로·IP·중간 장비를 다시 점검하고, `cpwd_admin list`로 VSX Gateway의 모든 체크포인트 프로세스가 떠 있는지(PID가 0이 아닌지) 확인합니다. 재부팅 직후라면 프로세스가 아직 올라오는 중일 수 있습니다. 끝으로 `netstat -an | grep 18211`로 CPD 프로세스가 신뢰 설정 포트를 LISTEN하고 있는지 봅니다.

새 가상 장치의 SIC 신뢰 문제와 **VSX 생성 마법사의 정책 설치 오류**는 상당수가 관리 서버와 VSX Gateway의 시간·시간대 불일치가 원인입니다. SIC가 제대로 동작하려면 두 장비의 UTC/GMT 시각이 일치해야 합니다. 모든 장비에서 `/bin/date -u`로 UTC 시각을 확인하고, 시간대는 서로 달라도 되지만 UTC 기준이 맞도록 맞춰 줍니다. 정책 설치 오류의 경우 라이선스도 함께 의심해야 하는데, 관리 서버에서 `cplic check`로, VSX Gateway에서 `vsx stat`의 "Number of Virtual Systems allowed by license" 값이 0보다 큰지 확인합니다.

내부 호스트가 Virtual System을 ping하지 못할 때는 원인이 여럿입니다. 가장 흔한 것은 새로 만든 Virtual System에는 모든 트래픽을 막는 기본 정책(Default Policy)이 걸려 있다는 점입니다. 통신을 허용하는 정책을 설치하고 SmartConsole의 Logs & Events에서 VS가 트래픽을 허용하는지 확인해야 합니다. 그 밖에 스위치의 VLAN 설정이나 케이블 문제, 인접 라우터의 잘못된 라우팅, VS의 VLAN 인터페이스에 잘못 지정된 IP·넷마스크도 원인이 될 수 있습니다. 이때 `tcpdump`를 해당 VLAN 인터페이스에 걸어 트래픽이 VSX Gateway에 도착하고 나가는지 확인하면 도움이 됩니다.

가상 장치의 SIC 신뢰 다시 맺기

특정 가상 장치(Virtual System이나 Virtual Router)의 SIC 신뢰가 깨져 연결 문제가 생겼다면 수동으로 다시 맺을 수 있습니다(자세한 절차는 sk34098 참고). 큰 흐름은 게이트웨이 쪽에서 SIC를 리셋하고, 관리 서버 쪽에서 옛 인증서를 폐기한 뒤, SmartConsole에서 객체를 열고 OK를 눌러 새 인증서를 발급하는 것입니다.

먼저 VSX Gateway 또는 각 클러스터 멤버에 접속해 Expert 모드로 들어간 뒤 `vsx stat -v` 로 대상 가상 장치의 ID를 확인하고 `vsx sic reset <VSID>` 로 SIC를 리셋합니다. 그다음 관리 서버에서 Expert 모드로 들어가는데, MDS라면 `mdsenv <Domain Management Server>` 로 해당 가상 장치를 관리하는 Target Domain Management Server로 컨텍스트를 바꿉니다. 이어서 `cpca_client lscert` 로 가상 장치의 SIC 이름을 알아내고 `cpca_client revoke_cert` 로 인증서를 폐기합니다. 마지막으로 그 VSX 클러스터를 관리하는 Security Management Server나 Main Domain Management Server에 SmartConsole로 접속해 가상 장치 객체를 더블클릭하고 OK를 누르면, 새 SIC 인증서가 만들어져 게이트웨이에 저장됩니다.

```
# VSX Gateway / 클러스터 멤버에서
vsx stat -v                  # 대상 가상 장치 ID 확인
vsx sic reset <VSID>         # SIC 리셋

# 관리 서버에서 (MDS는 먼저 컨텍스트 전환)
mdsenv <Domain Management Server>
cpca_client lscert -stat valid -kind SIC | grep -i -A 2 <장치명>
cpca_client revoke_cert -n <CN=...,O=...>
# 이후 SmartConsole에서 객체 열고 OK → 새 인증서 발급
```

14 명령어 레퍼런스

Command Line Reference

VSX 운영에 쓰는 주요 CLI 명령을 정리한 챕터입니다. 전체 명령 구문은 R82 CLI Reference Guide에 있고, 여기서는 VSX에 핵심적인 것들만 용도별로 묶어 설명합니다. 모든 명령에서 한 가지를 늘 의식해야 하는데, 명령은 현재 들어가 있는 VS 컨텍스트 기준으로 동작한다는 점입니다. 그래서 거의 모든 작업은 "지금 어느 컨텍스트인가"를 확인하는 데서 시작합니다.

구문 표기에 대해 미리 알아두면, 중괄호 `{ }` 는 그중 하나를 골라 입력하는 항목(세로줄 `|` 로 구분), 꺾쇠 `< >` 는 사용자가 값을 채워야 하는 변수, 대괄호 `[ ]` 는 선택 항목을 뜻합니다.

컨텍스트를 바꾸는 vsenv

가장 자주 쓰는 명령입니다. `vsenv` 는 셸의 현재 컨텍스트를 지정한 가상 장치로 바꿉니다. 인자 없이 실행하면 기본 장치인 VS0으로 가고, ID나 이름을 주면 그 장치로 들어갑니다.

> *Changes the shell's current context to the specified Virtual Device.*
>
> — AdminGuide, "vsenv" (p.301)

어떤 장치가 있는지는 `vsx stat -v` 로 확인합니다.

```
vsenv                 # VS0(관리)로 전환
vsenv 2               # ID 2번 VS로 전환
vsenv <VS 이름>       # 이름으로 전환
```

전환하면 프롬프트의 끝(`:0` , `:2` 등)이 현재 컨텍스트를 보여 줘서, 이후 실행하는 `fw` · `ifconfig` · `route` 같은 명령이 그 VS의 관점으로 동작합니다. Scalable Platform에서는 해당 Security Group의 Expert 모드에서 실행해야 합니다.

VSX 상태와 설정을 다루는 vsx

`vsx` 는 VSX 설정을 보여 주고, 가져오고(fetch), 메모리 자원을 제어하는 명령으로 여러 하위 명령을 가집니다. 가장 많이 쓰는 것은 전체 상태와 VS 목록을 보여 주는 `vsx stat -v` 인데, 진단의 출발점으로 모든 VS가 있는지·Active인지·정책이 맞는지·SIC가 맺어졌는지를 한 번에 보여 줍니다. 현재 컨텍스트 정보는 `vsx get` 으로, 메모리 자원은 `vsx mstat` 로 봅니다.

설정을 가져오는 계열로는 VSX Gateway 설정을 가져오는 `vsx fetch` , 특정 VS 설정을 가져오는 `vsx fetchvs` , 클러스터 피어로부터 모든 VS·VR의 정책을 가져오는 `vsx fetch_all_cluster_policies` 가 있습니다. 그 밖에 가상 장치의 네트워크 구성 스크립트를 보여 주는 `vsx showncs` , SIC를 리셋하는 `vsx sicreset` , 모든 정책을 내리는 `vsx unloadall` , 그리고 남은 설정 잔재를 정리하는 `vsx vspurge` 가 있습니다.

```
vsx stat -v           # 전체 상태·VS 목록 (진단 1순위)
vsx get               # 현재 컨텍스트 정보
vsx mstat             # 메모리 자원 (-vs <ID>, unit M/G, sort all 등)
vsx fetchvs <옵션>    # 특정 VS 설정 가져오기
vsx unloadall         # 모든 정책 내리기
```

`vsx mstat` 는 옵션이 풍부합니다. `-vs` 로 특정 VS만(예: `-vs 1` , `-vs 2 3` , 범위 `-vs 4-6`), `unit` 으로 단위(B/K/M/G, 기본 M), `sort all` 로 메모리 크기 순 정렬을 지정합니다. `enable` 은 메모리 자원 제어를 켜지만 재부팅이 필요하고, `disable` 은 즉시 적용됩니다.

시스템을 설정하는 cpconfig와 모니터링하는 cpview

`cpconfig`는 Check Point Configuration Tool을 띄우는 명령으로, 설치된 제품의 세부 설정을 메뉴로 다룹니다. VSX·클러스터에서 중요한 메뉴로는 SIC 설정, 클러스터 멤버십 비활성화, Per Virtual System State 활성화(VSLS에 필요), Bridge Active/Standby용 ClusterXL, CoreXL 등이 있습니다. 클러스터에서는 모든 멤버를 동일하게 설정해야 하고, Scalable Platform에서는 Security Group의 Gaia Portal에 접속해 설정합니다.

`cpview`는 CPU·메모리·디스크 같은 시스템 정보와 Software Blade 통계를 실시간으로 보여주는 텍스트 유틸리티입니다. 화면은 헤더·내비게이션·뷰 세 부분으로 나뉘고 화살표 키로 이동합니다. VSX에서는 VS0 컨텍스트에서 실행해 Advanced > VSX > VSs > Physical-Resources 로 들어가면 VS별 CPU·메모리 소비를 볼 수 있습니다. `R`로 갱신 주기를 바꾸고, `P`로 일시정지하며, `Q`로 종료합니다.

관리 서버에서 실행하는 vsx_util

`vsx_util`은 관리 서버에서 실행하는, VSX의 구조를 바꾸는 명령 모음입니다. 클러스터 멤버를 더하고 빼거나, 인터페이스·IP를 바꾸거나, 모드를 변환하거나, 업·다운그레이드하는 등 게이트웨이 자체로는 할 수 없는 작업을 합니다. 이런 작업은 관리 DB를 수정하므로, 다른 관리자가 접속해 DB를 잠그고 있으면 실패합니다. 그래서 보통 모든 SmartConsole을 닫고 진행합니다.

자주 쓰는 하위 명령을 보면, 클러스터 멤버를 더하는 `add_member`와 빼는 `remove_member`, 새 멤버를 관리 서버 기준으로 다시 구성하는 `reconfigure`(정책 push 실패 시 1순위 처방), 일반 클러스터를 VSX로 또는 VSLS↔HA를 변환하는 `convert_cluster`, VSLS에서 VS를 멤버에 재분배하는 `vsls`가 있습니다. 인터페이스를 본드 등으로 일괄 교체하는 `change_interfaces`, 클러스터 관리 IP·서브넷을 바꾸는 `change_mgmt_ip`·`change_mgmt_subnet`, 내부 통신 네트워크 IP를 바꾸는 `change_private_net`도 있습니다. 그 밖에 인터페이스를 보여 주는 `show_interfaces`, VS 설정을 보여 주는 `view_vs_conf`, 그리고 버전을 올리고 내리는 `upgrade`·`downgrade`가 있습니다.

```
vsx_util add_member          # 클러스터 멤버 추가
vsx_util remove_member       # 클러스터 멤버 제거 (먼저 라이선스 분리)
vsx_util reconfigure         # 게이트웨이 재구성 (정책 push 실패 시)
vsx_util convert_cluster     # 일반→VSX, 또는 VSLS⇔HA 변환
vsx_util vsls                # VSLS: VS를 멤버에 재분배
vsx_util change_interfaces   # 인터페이스 일괄 교체 (예: 본드)
vsx_util upgrade             # 버전 업그레이드
```

`downgrade`는 "변환 후 구성 변경이 없다" 같은 조건부라 의존하지 말고, 실무에서는 작업 전 Gaia Snapshot을 떠 두고 문제 시 revert하는 것이 안전합니다. MDS 환경에서는 관리자가 잠근 Domain Management Server를 건너뛰므로, 그런 경우 해당 Domain이 풀린 뒤 다시 실행해야 합니다.

대량 작업을 자동화하는 vsx_provisioning_tool

`vsx_provisioning_tool`은 VS 생성·인터페이스 배정 같은 작업을 스크립트나 CLI로 자동화하는 도구로, 가상 장치를 대량으로 만들고 바꿀 때 특히 유용합니다. SmartConsole에서 하나씩 클릭하는 대신 정의를 명령으로 적어 두면, 수십·수백 개의 VS를 일관되게 프로비저닝할 수 있습니다.

기본 형태는 관리 서버와 자격 증명을 준 뒤, 한 줄 명령은 `-o`로, 여러 명령을 모은 스크립트 파일은 `-f`로 실행하는 것입니다. `-o` 뒤에는 여러 작업을 쉼표로 이어 한 번에 넣을 수 있습니다.

```
vsx_provisioning_tool -s <관리서버> -u <사용자> -p <비밀번호> -o <작업>
vsx_provisioning_tool -s localhost -u admin -p mypw -f /var/log/vsx.txt   # 스
```

작동 단위는 트랜잭션입니다. 묶인 명령들이 하나의 트랜잭션으로 처리되어, 모두 성공하거나 모두 적용되지 않는 식으로 일관성을 지킵니다. 모든 명령은 `키 값` 쌍의 나열로 되어 있어 읽기 쉽습니다.

주요 작업을 보면, 게이트웨이나 클러스터를 만드는 `add vsx`(`type gateway` 또는 `type cluster`), 가상 장치를 만드는 `add vd`(`type vsw` / `vr` /생략 시 VS), 지우는 `remove vd`, 설정을 바꾸는 `set vd`(예: CoreXL 인스턴스 수, main IP)가 있습니다. 인터페이스 쪽은 `add interface`(물리·VLAN·VPN 터널), 두 인터페이스를 브리지로 묶는 `attach bridge`, 떼는 `remove interface`, 바꾸는 `set interface`, 물리 인터페이스를 VLAN 트렁크로 지정하는 `set physical_interface`가 있고, 라우트는 `add route`·`remove route`로 다룹니다.

```
# VSX 게이트웨이 생성
... -o add vsx name VSX_GW1 type gateway main_ip 192.168.20.1 version R82 sic

# VSLS 클러스터 생성
... -o add vsx name VSX1 type cluster cluster_type vsls main_ip 192.168.1.1 v

# Virtual Switch 생성 / VS에 인터페이스·라우트 추가
... -o add vd name VirtSwitch1 vsx VSX_GW1 type vsw
... -o add interface vd VirtSystem1 name eth4.100 ip 1.1.1.1/24
... -o attach bridge vd VS1 ifs1 eth2 ifs2 eth3
... -o add route vd VS1 destination default leads_to VR3
```

이렇게 객체 생성부터 인터페이스·브리지·라우트 배정까지를 한 스크립트에 담아 두면, 동일한 구성을 반복하거나 대규모로 배포할 때 큰 도움이 됩니다. 구체적인 활용 예는 구성 예제 챕터에서 볼 수 있습니다.

15 구성 예제

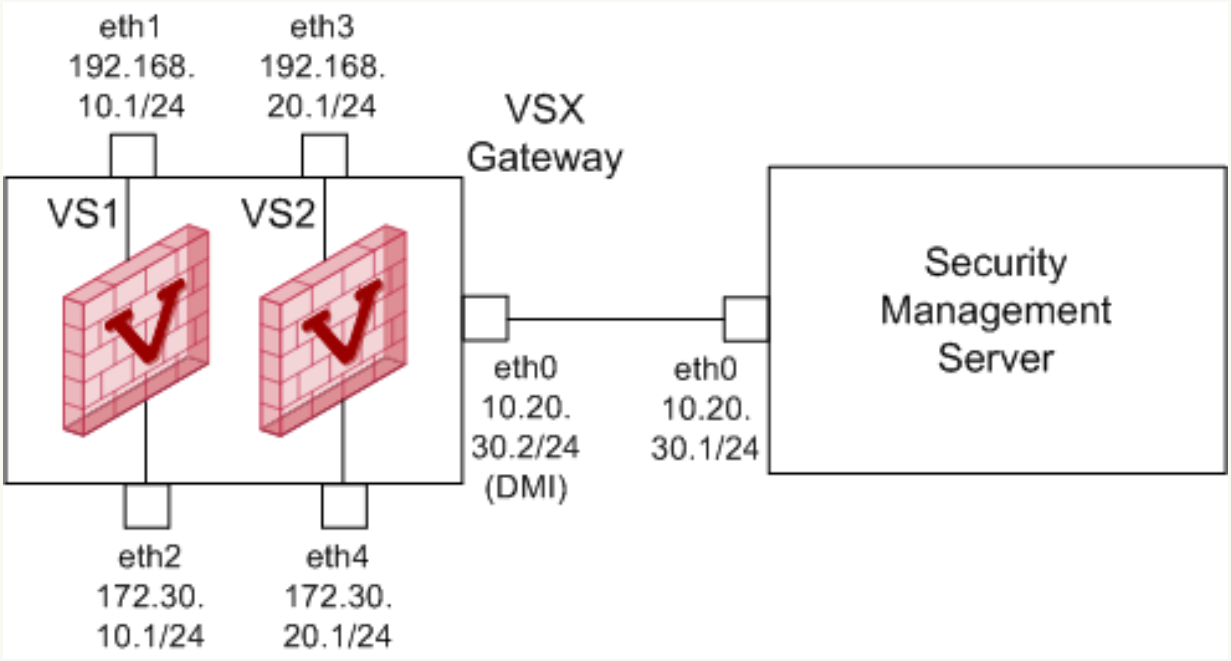
Configuration Examples

지금까지 본 개념과 절차가 실제로 어떻게 맞물리는지, 두 가지 완성된 시나리오로 따라가 보는 챕터입니다. 첫 번째는 Security Management Server로 관리하는 단일 VSX Gateway이고, 두 번째는 Multi-Domain Server로 관리하는 VSX Cluster 입니다. 두 예제 모두 "관리 서버 설치 → VSX 게이트웨이 설치 → SmartConsole에서 객체 생성 → 가상 장치 생성 → 정책 설치"라는 같은 큰 흐름을 따르되, 규모와 관리 모델이 다릅니다.

예제 1 — SMS로 관리하는 VSX Gateway

이 예제는 단순하면서도 핵심을 보여 줍니다. DMI 관리 연결을 쓰는 VSX Gateway 한 대에 Virtual System 두 개를 올리는데, 각 VS는 게이트웨이의 물리 인터페이스에 직접 연결됩니다. 그리고 한 VS에는 IPsec VPN Blade를, 다른 VS에는 Mobile Access Blade를 켜서 서로 다른 보안 기능을 가진 가상 방화벽을 한 장비 안에 둡니다. 이 모두를 하나의 Security Management Server가 관리합니다.

> *One Security Management Server manages both the VSX Gateway and the two Virtual Systems.*
>
> — AdminGuide, "Example 1: VSX Gateway managed by Security Management Server" (p.402)



예제 1 토폴로지

작업은 관리 서버부터 세우는 것으로 시작합니다. 어플라이언스나 Open Server에 Gaia OS를 설치하고 최초 설정 마법사에서 관리 인터페이스(예: eth0, 10.20.30.1/24)를 잡은 뒤 Installation Type에서 Security Management를 고릅니다. 라이선스를 설치하고 SmartConsole로 접속해 관리 Blade를 설정하면 관리 서버가 준비됩니다.

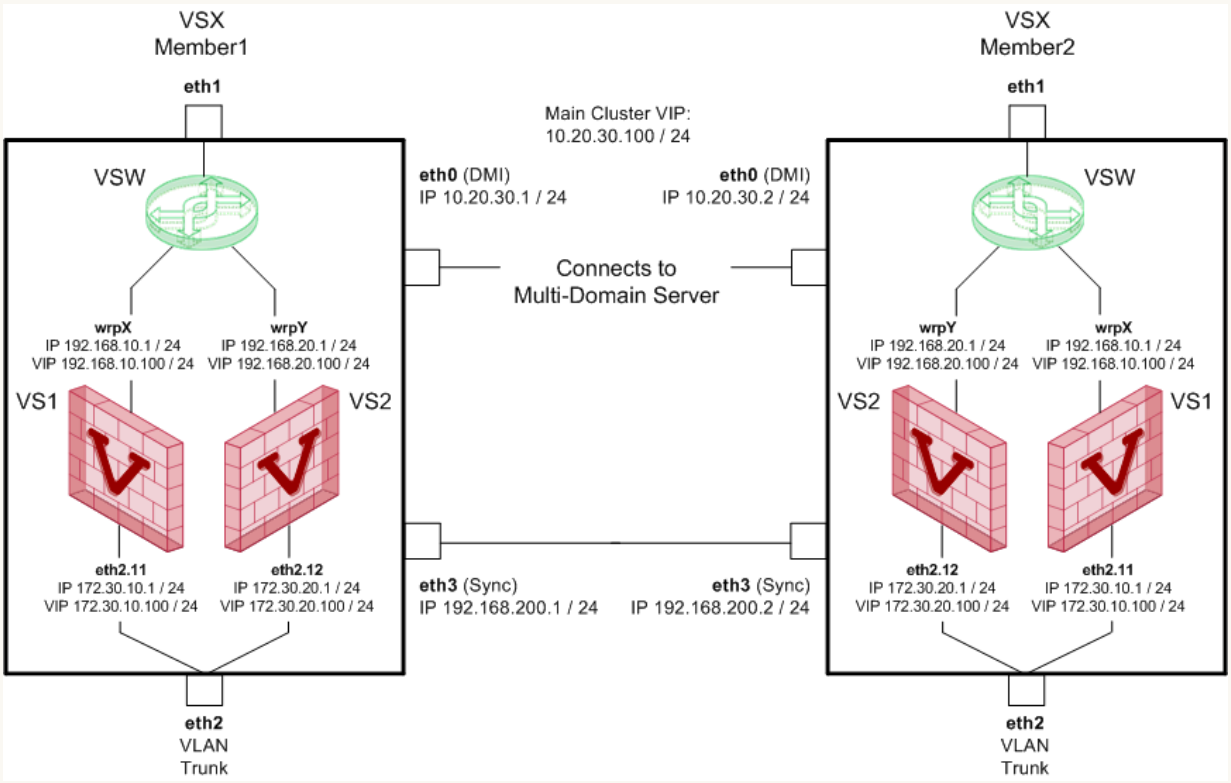
다음은 VSX Gateway 설치입니다. 마찬가지로 Gaia를 설치하되, 최초 설정 마법사에서 DMI 관리 연결용 인터페이스(예: eth0, 10.20.30.2/24)를 잡고 VS가 직접 연결될 물리 인터페이스에는 IP를 주지 않습니다. Products에서 Security Gateway를 고른 뒤, Gaia Portal이나 Clish(`set interface <이름> state on`)에서 쓸 물리 인터페이스를 활성화하고 라이선스를 설치합니다.

그다음 SmartConsole에서 VSX Gateway 객체를 만듭니다. New > VSX > Gateway로 마법사를 시작해 General Properties에서 이름(예: MyVsxGw)과 IPv4(최초 설정 때와 같은 10.20.30.2), 버전(R82)을 입력하고, SIC 단계에서 최초 설정 때와 같은 활성화 키 를 넣어 Initialize합니다. 신뢰가 안 맺어지면 게이트웨이 CLI에서 `cpconfig` 의 SIC 메뉴로 키를 다시 설정하고 마법사에서 Reset 후 재시도합니다. 이어서 물리 인터페이스를 확인하고 마법사를 마칩니다.

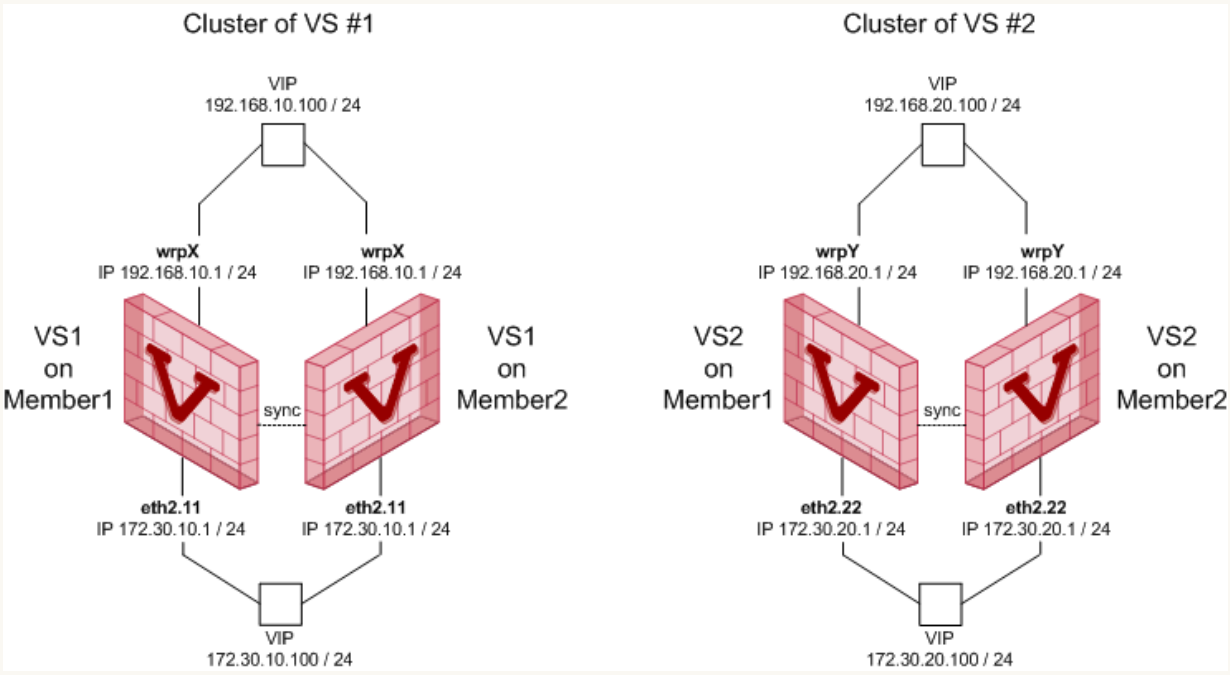
마지막으로 두 Virtual System을 만듭니다. Virtual System Wizard로 각 VS를 생성하면서 외부·내부 인터페이스를 직접 연결된 물리 인터페이스에 배정하고, 한 VS에는 IPsec VPN을, 다른 VS에는 Mobile Access Blade를 켭니다. 각 VS에 Access Control 정책을 설치하면 두 개의 독립된 가상 방화벽이 한 장비 위에서 서로 다른 역할로 동작하게 됩니다.

예제 2 — MDS로 관리하는 VSX Cluster

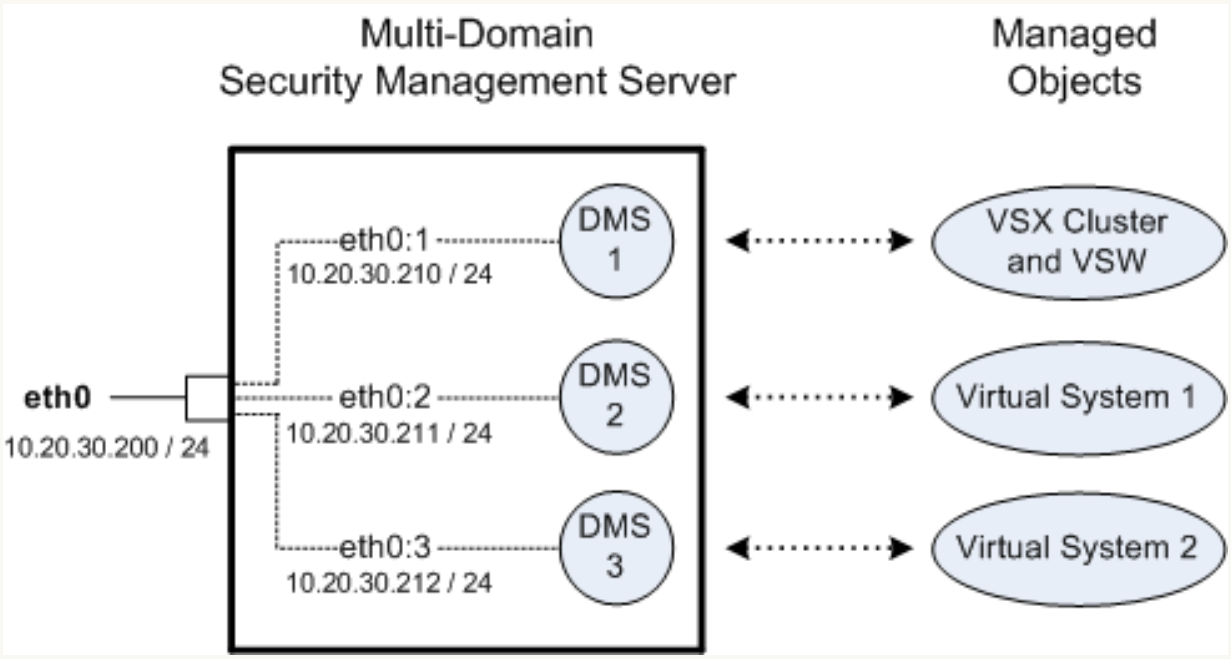
두 번째 예제는 한 단계 더 나아갑니다. Multi-Domain Server가 관리하는 VSX Cluster 구성으로, 이중화(클러스터)와 다중 도메인 관리(MDS)를 함께 보여 줍니다. 서비스 제공업체나 대기업처럼 여러 도메인의 가상 방화벽을 중앙에서 관리하면서 무중단을 보장 해야 하는 환경에 해당합니다.



예제 2 토폴로지



예제 2 토폴로지 (이어서)



예제 2 구성 상세

큰 흐름은 예제 1과 같지만, MDS 특유의 단계가 앞에 더 붙습니다. 먼저 Multi-Domain Server를 설치·설정하고, VSX Cluster를 관리할 Domain과 그 Main Domain Management Server를 만든 뒤 거기에 SmartConsole로 접속해 VSX Cluster 객체를

생성합니다. 클러스터이므로 VSX 클러스터 구성에서 본 대로 두 멤버를 설치하고 동기화 네트워크와 Cluster IP, 내부 통신 네트워크를 설정합니다.

가상 장치는 각 도메인을 관리하는 Target Domain Management Server에 접속해 만듭니다. 즉 VSX Cluster 본체는 Main Domain Management Server가, 그 위의 개별 Virtual System들은 각자의 Target Domain Management Server가 관리하는 식으로 관리 책임이 도메인별로 나뉩니다. 각 VS를 만들고 정책을 설치하면, 여러 도메인의 가상 방화벽이 하나의 이중화된 VSX Cluster 위에서 각자 독립적으로, 그리고 무중단으로 동작하게 됩니다.

두 예제가 보여 주는 것

두 예제를 나란히 놓으면 VSX 구축의 공통 골격이 또렷이 드러납니다. 항상 바깥 틀(관리 서버 → 게이트웨이/클러스터 본체)을 먼저 세우고, SIC로 신뢰를 맺고, 그 안에 가상 장치를 만든 뒤, VS마다 정책을 설치 하는 순서입니다. 차이는 규모에 있을 뿐입니다. 단일 도메인이면 SMS로 충분하고, 여러 법인·도메인을 다루거나 무중단이 필요하면 MDS와 클러스터로 확장합니다. 실제 현장에서는 이 두 예제의 요소들을 배포 전략 챕터에서 본 조직 유형에 맞게 조합해 쓰게 됩니다.

16 커널 파라미터 다루기

Working with Kernel Parameters

이 챕터는 VSX 가이드 안에서는 짧게 안내만 합니다. **커널 파라미터를 다루는 방법은 VSX 전용이 아니라 일반 Security Gateway와 동일**하기 때문입니다. 따라서 자세한 내용은 별도 문서로 안내됩니다.

> *See the R82 Quantum Security Gateway Guide > Chapter "Working with Kernel Parameters".*
>
> — AdminGuide, "Working with Kernel Parameters" (p.435)

정리하면 커널 파라미터 관련 작업이 필요할 때는 **R82 Quantum Security Gateway Guide의 "Working with Kernel Parameters" 챕터**를 참고하면 됩니다. VSX 환경이라고 해서 다르게 다룰 필요는 없되, 명령을 실행할 때 어느 VS 컨텍스트인지만 늘 의식하면 됩니다.

17 Security Gateway 커널 디버그

Kernel Debug on Security Gateway

이 챕터 역시 VSX 가이드에서는 안내만 하고 자세한 절차는 별도 문서로 넘깁니다. **커널 디버그 방법도 VSX 전용이 아니라 일반 Security Gateway와 같기** 때문입니다.

> *See the R82 Quantum Security Gateway Guide > Chapter "Kernel Debug on Security Gateway".*
>
> — AdminGuide, "Kernel Debug on Security Gateway" (p.436)

따라서 커널 디버그가 필요하면 **R82 Quantum Security Gateway Guide의 "Kernel Debug on Security Gateway" 챕터**를 참고하면 됩니다. 다만 VSX에서는 디버그를 시작하기 전에 `vsenv <VSID>` 로 대상 VS의 컨텍스트에 먼저 들어간 뒤 진행해야 한다는 점만 기억하면 됩니다.

18 용어 정리

Glossary

VSX를 읽다 보면 비슷비슷한 용어가 자주 나와 헷갈리기 쉽습니다. 여기서는 VSX에서 특히 중요한 용어를 주제별로 묶어 풀어 설명합니다. 일반적인 체크포인트 용어(각 Software Blade 등)는 마지막에 짧게 모았고, 전체 표준 용어 정의는 PDF 원문의 Glossary를 참고하면 됩니다.

VSX의 기본 객체

VSX(Virtual System eXtension) 는 물리 장비 한 대(또는 클러스터) 위에 Security Gateway와 네트워크 장치들을 가상으로 올려 두는 체크포인트의 가상 네트워킹 솔루션입니다. 이 가상 장치들은 물리 장비와 똑같은 기능을 제공합니다.

VSX Gateway 는 그 VSX 가상 네트워크를 호스팅하는 물리 서버입니다. 모든 가상 장치를 담으며, 적어도 하나의 Virtual System을 갖는데 그것이 바로 VS0 입니다.

Virtual Device(가상 장치) 는 물리 네트워크 장치의 기능을 흉내 내는 논리 객체를 통칭하는 말로, **Virtual System·Virtual Router·Virtual Switch** 셋 중 하나입니다.

Virtual System(VS) 은 Security Gateway의 기능을 구현한 가상 장치, 곧 가상 방화벽입니다. **Virtual Router(VR)** 는 물리 라우터처럼 동작하는 가상 장치이고, **Virtual Switch(VSW)** 는 물리 스위치처럼 동작하는 가상 장치입니다.

Warp Link(WRP) 는 Virtual System과 Virtual Switch/Router 사이에 자동으로 생기는 논리 인터페이스입니다. 한편 **Warp Jump**는 같은 Virtual Switch나 Router에 연결된 두 Virtual System 사이에서, 한 VS 뒤 네트워크의 트래픽이 다른 VS 뒤 네트워크로 갈 때 Switch/Router를 거치지 않고 VS에서 VS로 곧장 "점프" 하는 동작을 가리킵니다.

클러스터와 이중화

Cluster(클러스터) 는 High Availability나 Load Sharing으로 이중화 구성된 둘 이상의 Security Gateway이고, 그중 하나의 게이트웨이를 **Cluster Member(클러스터 멤버)** 라 합니다.

VSLS(Virtual System Load Sharing) 는 Virtual System들의 트래픽을 서로 다른 Active 클러스터 멤버에 나눠 배정하는 VSX 클러스터 기술입니다.

관리 — 누가 무엇을 관리하는가

Security Management Server(SMS) 는 단일 관리 도메인 안에서 객체와 정책을 관리하는 전용 서버이고, **Multi-Domain Server(MDS)** 는 Domain Management Server라 불리는 가상 관리 서버들을 호스팅하는 전용 서버입니다. **Management Server** 는 이 둘을 통칭하는 말입니다.

MDS 환경에서는 역할에 따라 이름이 갈립니다. **Main Domain Management Server** 는 VSX Gateway나 클러스터 객체를 설정한 Domain Management Server 이고, 이때 그 안의 Virtual System들은 다른 Domain Management Server에 정의됩니다. 반대로 **Target Domain Management Server** 는 Virtual System 객체를 설정한 Domain Management Server 이며, 이 경우 VSX Gateway·클러스터 객체는 다른(Main) Domain Management Server에 있습니다. **MDLS(Multi-Domain Log Server)** 는 MDS 환경에서 로그를 저장·처리하는 전용 서버입니다.

DMI(Dedicated Management Interface) 는 관리 트래픽 전용의 별도 물리 인터페이스로, 관리 서버가 VSX Gateway에 직접 연결되는 통로입니다. 반대로 **Non-DMI(Non-Dedicated Management Interface)** 는 운영 트래픽과 공유하는 인터페이스인데, R80.40 이하에서만 지원되며 Virtual Router나 Switch가 필요했습니다(현재는 폐기).

SIC(Secure Internal Communication) 는 체크포인트 컴포넌트끼리 SSL로 서로를 인증하는 보안 통신 메커니즘 으로, 관리 서버의 ICA(Internal Certificate Authority)가 발급한 인증서에 기반합니다.

네트워크와 성능

CoreXL 은 멀티코어 환경에서 여러 방화벽 인스턴스를 병렬로 돌려 성능을 높이는 기술입니다. 복제된 각 방화벽 커널을 **CoreXL Firewall Instance** 라 하며, 각 인스턴스가 하나의 CPU 코어에서 독립적으로 트래픽을 검사합니다. **CoreXL SND(Secure Network Distributor)** 는 네트워크 인터페이스에서 들어온 트래픽을 받아 방화벽 인스턴스들에 분배하는 부분으로, 출발지·목적지 IP와 프로토콜을 기준으로 연결을 정적으로 배정합니다. **SecureXL** 은 게이트웨이를 통과하는 트래픽을 가속하는 기술입니다.

Bridge Mode 는 기존 토폴로지에 손쉽게 끼워 넣기 위해 L2 브리지처럼 동작하는 Security Gateway나 Virtual System입니다.

운영 환경

Gaia 는 체크포인트의 보안 운영체제이고, **Gaia Clish** 는 그 기본 명령줄 셸(역할 기반으로 제한됨), **Expert Mode** 는 전체 root 권한을 주는 상위 셸, **Gaia Portal** 은 웹 인터페이스입니다.

그 밖의 일반 용어

다음은 VSX 전용은 아니지만 자주 등장하는 체크포인트 공통 용어입니다. **Software Blade** 는 특정 보안 기능 모듈을 가리키며, 게이트웨이에서는 트래픽의 특정 측면을 검사하고 관리 서버에서는 관리 기능을 제공합니다. 대표적인 게이트웨이 Blade로는 침입 방지의 **IPS**, 애플리케이션 제어의 **Application Control(APPI)**, 웹사이트 접근 제어의 **URL Filtering(URLF)**, 백신의 **Anti-Virus(AV)**, 봇 차단의 **Anti-Bot(AB)**, 정보 유출 방지의 **DLP**, 샌드박스 분석의 **Threat Emulation(TE)** 과 악성 콘텐츠 제거의 **Threat Extraction(TEX)**, 신원 기반 제어의 **Identity Awareness(IDA)**, SSL 검사의 **HTTPS Inspection** 등이 있습니다. 관리 도구로는 정책·장치·이벤트를 다루는 GUI인 **SmartConsole**, 정책의 규칙 묶음인 **Rule Base**, 그리고 Gaia OS 업데이트 엔진인 **CPUSE** 가 자주 쓰입니다.