

Horizon App Volume

1. 데이터센터 레이어

<p>The diagram illustrates the multi-layered architecture of Horizon App Volume. From top to bottom, the layers are: <ul style="list-style-type: none"> User Access: Client Access Devices, Self Service Portal, Location-Based Authorization. Application Resource Layer: Cloud-Based Application, Virtual Application, Enterprise Application, Locally Installed Application. Desktop Resource Layer: Departments, Contractors, External Users, DesktopTier 1, DesktopTier 2, DesktopTier X, Scale, Multi Site. Virtualization Layer: Hypervisor, Compute, Graphics, Network, Storage. Physical Layer: Server, Network, Storage. On the right side, vertical bars represent Mgt (Management) and Security components that span across the layers.</p>	<ul style="list-style-type: none"> • 데이터센터의 전반적인 아키텍처 레이어 <ul style="list-style-type: none"> ◦ Desktop Resource 레이어 (Horizon) <ul style="list-style-type: none"> ◦ 데스크톱 기반의 가상머신 제공 ◦ Application Resource 레이어 (App Volume & ThinApp) <ul style="list-style-type: none"> ◦ 어플리케이션 제공하는 서버 (Win) ◦ User Access 레이어 (Workspace ONE Access) <ul style="list-style-type: none"> ◦ 사용자 디바이스 및 소프트웨어 접근 제어
---	--

2. Omnissa 애플리케이션 가상화 방법

<p>The diagram shows the stack for application virtualization. At the bottom is the OS (Windows). Above it is the App Volumes Agent. The Apps layer includes icons for Chrome, Office, Teams, and Adobe Acrobat. The top layer is DEM (Desktop Environment Management), which includes Profile and Settings.</p>	<p>어플리케이션 가상화 장점</p> <ul style="list-style-type: none"> • 어플리케이션 중앙 관리로 인한 유지보수 간편 • 호환성 확장 <ul style="list-style-type: none"> ◦ OS ↔ 애플리케이션 간의 충돌 완화 • 배포 시간 단축 • 보안 강화 • 레거시 소프트웨어 지원
<p>레거시 방식</p> <ul style="list-style-type: none"> • 관리자가 하나하나 배포 • 골든 이미지 가상 머신 구축 > 여러 애플리케이션 설치 > VDI을 통해 VM 배포 <ul style="list-style-type: none"> ◦ 애플리케이션 패치 필요 시, 골든 이미지 수정하여 재배포 진행 	<p>현대화 방식</p> <p><App Volumes></p> <ul style="list-style-type: none"> • 어플리케이션 컨테이너 방식 사용 <ul style="list-style-type: none"> ◦ 어플리케이션 + 동작에 필요한 dll 파일을 vmdk (가상화 디스크) 형식으로 제공 ◦ 배포 시에 가상머신에 HDD 로 추가됨 <p><Thin App></p> <ul style="list-style-type: none"> • 어플리케이션 캡슐화 방식 사용 <ul style="list-style-type: none"> ◦ 어플리케이션 + 동작에 필요한 dll 파일, 레지스트리 값 등을 캡슐화 ◦ OS 와 Thin App 은 독립되어 있음

2.1. App Volumes 방식

<p>The diagram illustrates the App Volumes architecture. At the top, there are three 'App Container' boxes, each containing icons for different applications (e.g., Office, Java, Chrome). Below these is a 'User Settings' box. These components sit on top of the 'App Volumes' layer, which is connected to the 'OS' layer. The entire stack is supported by the 'Infrastructure' layer.</p>	<p>주요 특징</p> <ul style="list-style-type: none"> • 실시간 어플리케이션 전송 및 어플리케이션 수명 주기 관리 • 어플리케이션을 가상 디스크로 제공 (1:N 제공) <ul style="list-style-type: none"> ◦ 기존 이미지를 통해 어플리케이션 캡처 작업 필요 ◦ 어플리케이션이 실행되는 PC의 파일 시스템, 레지스트리 인식이 필요하여 OS 환경과 통합적으로 작동하는 방식 사용 • 사용자가 어플리케이션 요청 시, App Volumes 용 가상 디스크에 설치된 어플리케이션에 연결됨 <ul style="list-style-type: none"> ◦ 스토리지에 저장되므로 스토리지 성능에 영향을 받을 수 있음 <p>주요 사용 사례</p> <ul style="list-style-type: none"> • VDI 환경에서 여러 사용자에게 어플리케이션을 실시간으로 배포 • 대규모로 어플리케이션을 관리하는 환경
---	--

2.1. ThinApp 방식

<p>The diagram shows the ThinApp process flow in three steps: 'Prepare' (represented by a document icon with a checkmark), 'Create' (represented by an 'APP' icon with a lock), and 'Validate' (represented by a document icon with a checkmark and a plus sign). Arrows connect the steps in sequence.</p>	<p>주요 특징</p> <ul style="list-style-type: none"> • 개별 실행 파일 형태로 배포 <ul style="list-style-type: none"> ◦ 프로그램 실행에 필요한 전체 파일을 캡슐화 ◦ OS와 독립적으로 실행 가능 • 특정 사용자 또는 특정 환경에서 독립적으로 실행 • 개별 패키지 형태로 관리됨 • 휴대용 기기 또는 드라이브 공유 형식으로 사용 가능 <p>주요 사용 사례</p> <ul style="list-style-type: none"> • 어플리케이션을 OS와 완전히 분리하여 실행 필요 • 레거시 어플리케이션을 최신 OS에서 실행해야 하는 경우
---	---

🔄Revision #5

★Created 2025-09-30 23:25:19 KST by 박하림

✎Updated 2025-10-20 23:49:00 KST by 우승민