D-SaaS

Deployable Software as a Service

Version: 1.1

Released: 14 December 2025

Executive Summary

Modern SaaS delivery models offer efficiency and scalability but often conflict with the requirements of compliance-driven environments. The core issue is the erosion of data sovereignty, defined as an organization's ability to retain direct technical ownership and enforcement authority over its data, infrastructure, and operational boundaries.

Vendor-hosted SaaS architectures centralize control within the provider's environment. This creates a structural misalignment with regulated environments that require in-boundary governance, auditability, and technically enforced policy controls.

This white paper introduces Deployable SaaS (D-SaaS), a software delivery architecture designed to preserve data sovereignty by deploying and operating SaaS applications entirely within customer-owned infrastructure. D-SaaS reframes SaaS not as a hosting model, but as a delivery and lifecycle model, enabling organizations to adopt modern service-based software without relinquishing control over identity, data, or enforcement boundaries.

Deployable SaaS is intended for organizations operating regulated private cloud environments where data custody, identity governance, and audit scope must remain under direct organizational control.

D-SaaS Certified

D-SaaS Compliant

Table of Contents

Background and Market Conditions

Over the past decade, regulated industries such as government, healthcare, finance, and critical infrastructure have aggressively adopted cloud technologies as a strategic and operational mandate. This adoption has been accompanied by a fundamental shift from software ownership and on-premises operation to vendor-hosted SaaS platforms.

Simultaneously, vendors have consolidated the SaaS, PaaS, and IaaS layers into monolithic, provider-controlled environments. While this streamlining improves efficiency and accelerates delivery, it centralizes control and externalizes core functions such as data storage, authentication, and telemetry outside the customer-controlled infrastructure boundary.

At the same time, regulatory requirements have intensified. Frameworks such as FedRAMP, HIPAA, RMF, and PCI-DSS require demonstrable boundary control, local auditability, and technical policy enforcement. These conditions are difficult or impossible to meet under traditional SaaS architectures.

While examples are drawn from common regulatory frameworks, Deployable SaaS (D-SaaS) is intentionally framework-agnostic and applicable wherever strict boundary control is required.

The market has created a paradox: cloud adoption is mandatory, yet compliance frameworks assume control models that modern SaaS architectures violate.

The Core Problem: Loss of Data Sovereignty

At the heart of this conflict is the loss of data sovereignty, defined here as an organization's ability to retain technical ownership and enforcement authority over its data, operations, and security boundaries. This sovereignty is fundamentally eroded by vendor-hosted SaaS delivery models.

Most Compliance frameworks implicitly assume:

- Organizations own and control the data they are responsible for.
- Data location and flow are known, constrained, and enforceable.
- Authentication, authorization, and audit occur within governed boundaries.
- Evidence and logs remain under customer custody.
- Trust is technically enforced—not inferred through contracts.

Vendor-hosted SaaS architectures violate these assumptions by design. Even when encryption, tenant isolation, or logical separation is employed, the underlying infrastructure, identity stack, logging systems, and runtime environment remain external to the customer's control.

As a result, audit scope expands beyond the organizational boundary, enforcement authority is diluted, and security posture becomes dependent on vendor implementation details that the customer cannot directly observe or control.

Resulting Impacts:

- Audit complexity and scope expansion
- Reliance on exception-based security
- Fragmented identity and authentication workflows
- Inconsistent MFA enforcement
- User frustration due to disjointed access models
- Unmanaged SaaS sprawl across departments

These are not procedural or operational failures. They are direct consequences of architectural misalignment between modern SaaS delivery models and environments that require strict boundary control.

Structural Limitations of Traditional SaaS

Traditional SaaS architectures are inherently centralized. Service providers retain operational control over:

- Application runtime and environment
- Platform services and tenancy models
- Infrastructure and networking layers
- Authentication and access control systems
- Monitoring and logging systems

This design places control outside the organization's governed boundary by default. Even offerings marketed as "private" or "single-tenant" SaaS frequently operate on provider-owned infrastructure, rely on externalized identity systems, and export logs and telemetry through vendor-controlled pipelines.

Configuration-level controls do not change this fundamental reality. When the runtime environment and control plane remain vendor-operated, the organization lacks the technical authority required to enforce its own boundaries.

Compliance cannot be restored through policy overlays, contractual assurances, or compensating controls. As long as the runtime and control-plane are vendor-owned, boundary enforcement remains indirect and incomplete.

Deployable SaaS (D-SaaS): Architectural Definition

Deployable SaaS (D-SaaS) is a software delivery architecture in which SaaS applications are provided as self-contained artifacts that are deployed and operated entirely within the customer's infrastructure.

D-SaaS departs from traditional SaaS by explicitly decoupling:

- Hosting from service delivery
- Operation from ownership
- Capability from vendor control

Rather than operating the application on behalf of the customer, the provider delivers the software as a deployable unit while retaining responsibility for lifecycle activities such as updates, vulnerability remediation, and product support.

The customer operates the service fully within its governed environment, preserving data sovereignty, boundary control, and compliance enforcement without sacrificing modern service-based software delivery.

Architectural Invariants

Deployable SaaS is defined by a strict set of architectural invariants. These are hard requirements that form the foundation of the D-SaaS compliance-aligned delivery model. The invariants are non-negotiable. If any are violated, the product does not qualify as Deployable SaaS.

Customer-Owned IaaS Tenancy

The application must be deployed entirely within infrastructure owned and controlled by the customer. This includes compute, storage, and network boundaries.

No Vendor-Operated Runtime or Tenant

The vendor must not host, operate, or share the runtime environment. All execution, orchestration, and service operation occur within the customer's control plane.

Data Remains In-Boundary

All data, including configuration data, operational data, metadata, and telemetry, must always remain within the customer's defined boundary of authority.

Identity Integrates with Customer IdP

Authentication must be integrated directly with the customer's identity provider. This enables enforcement of local identity policies such as role-based access control and multi-factor authentication.

Logs and Audit Artifacts Remain Local

All operational logs, telemetry, and security-relevant evidence must be generated, retained, and accessible within the customer environment. Logs must be forensically complete and suitable for audit and review.

No Required External Control Plane

The system must operate fully without reliance on a vendor-operated control plane or external orchestration service. External connectivity must not be required for normal operation.

Artifact-Based Updates Under Customer Control

All updates, patches, and configuration changes must be delivered as cryptographically signed artifacts. The customer retains full control over update timing, approval, and deployment.

Data and Log Preservation During Updates

All critical operational data and security-relevant logs must be preserved across upgrades, patches, and redeployments. A zero-regression posture is required to maintain forensic continuity and evidentiary integrity.

Air-Gap Compatible Deployment

The system must support air-gapped and disconnected environments. Vendors must assume that external repositories, update endpoints, and validation services may be unreachable and must provide offline-compatible installation, update, and maintenance workflows.

Offline-Verifiable, Time-Bound Licensing

Licensing must be time-bound and enforceable without requiring external validation or license server communication. Offline-verifiable cryptographic mechanisms, such as signed license tokens using public and private key pairs, must be supported to enable licensed functionality in fully disconnected environments.

Note: Any product that fails to meet all of the above invariants cannot be considered a Deployable SaaS solution.

D-SaaS Deployment Models

Deployable SaaS supports multiple deployment approaches to align with customer infrastructure preferences and operational constraints.

Container-Based Deployments

Delivered as Docker or OCI-compliant containers and deployed using Kubernetes or functionally equivalent orchestration platforms. The customer retains full control over scheduling, scaling, networking, and runtime configuration.

Virtual Machine Appliance Deployments

Provided as virtual machine images compatible with common enterprise and cloud-native hypervisors, including VMware, KVM, and equivalent platforms. These deployments support environments where container orchestration is unavailable or prohibited and accommodate software designs that cannot rely on container-based execution.

Supported Private Cloud Environments

Designed for deployment into customer-owned private cloud infrastructure, including environments built on AWS, Azure, OpenStack-based platforms, or equivalent private cloud implementations.

Connected and Disconnected Operation

Supports both connected and fully disconnected environments. Updates, patches, and configuration changes are delivered as signed artifacts and applied under customer control without requiring continuous network connectivity.

This deployment flexibility enables D-SaaS solutions to operate consistently across diverse regulatory, compliance, and operational contexts while preserving strict boundary control.

Trust Boundaries and Control Planes

Deployable SaaS enforces a strict separation of responsibility between the Customer Plane and the Provider Plane. This separation is fundamental to preserving data sovereignty, auditability, and enforcement authority.

Customer Plane Responsibilities:

The customer retains full control and responsibility for all operational aspects of the deployed service, including:

- Executing and operating the application runtime
- Managing identity, authentication, and access control
- Maintaining network segmentation and boundary enforcement
- Collecting, storing, and reviewing logs, evidence, and telemetry
- Enforcing compliance, security policy, and operational governance within the environment

All operational data and enforcement mechanisms remain within the customer's boundary of authority.

Provider Plane Responsibilities:

The provider retains responsibility for product integrity and lifecycle management, including:

- Developing, building, and cryptographically signing release artifacts
- Supplying vulnerability remediation, patches, and functional updates
- Providing documentation, operational guidance, and product support
- Maintaining source code, secure build pipelines, and product lifecycle processes

The provider does not operate or access the customer's runtime environment.

Boundary Enforcement

No operational data, telemetry, identity information, or audit artifacts cross from the Customer Plane into the Provider Plane during normal operation. Interaction between planes is limited to artifact delivery and support workflows explicitly initiated and controlled by the customer.

This separation ensures that trust is enforced through architecture rather than assumed through contractual agreement, reinforcing data sovereignty, auditability, and accountability.

Compliance Alignment Through Architecture

Deployable SaaS aligns with compliance requirements through architectural design rather than procedural overlays or compensating controls.

Key Compliance Enablers:

Boundary Control

All sensitive operations, including data processing, identity enforcement, and audit generation, occur entirely within the customer's governed environment.

Audit Scope Reduction

Because the application runtime, data, and logs remain within customer-owned infrastructure, audit scope is limited to systems under direct organizational control.

Policy Enforcement

Identity, access control, logging, and retention policies are enforced internally using the customer's existing governance and security frameworks.

Elimination of Exception-Based Security

Architectural alignment removes the need for security exceptions, waivers, or compensating controls commonly required by vendor-hosted SaaS models.

By preserving ownership of infrastructure, identity, and operational boundaries, D-SaaS makes compliance an intrinsic outcome of the architecture rather than a condition imposed after deployment.

Secondary Effects and Operational Benefits

While the primary drivers behind Deployable SaaS are data sovereignty, boundary control, and compliance alignment, the architecture also produces meaningful secondary operational benefits.

Reduced SaaS Tenant Sprawl

Service deployment within customer-owned infrastructure consolidates SaaS capability under organizational governance and eliminates uncontrolled external tenants.

Consistent Authentication and Multi-Factor Enforcement

All access is mediated through existing identity systems, enabling uniform authentication, authorization, and multi-factor authentication policies across services.

Improved User Experience

Users interact with services through a consistent access model, avoiding cross-platform identity friction and fragmented authentication workflows.

Simplified Auditing and Evidence Collection

All logs, telemetry, and audit artifacts remain locally accessible within the governed environment, reducing audit preparation effort and complexity.

EASIER INTEGRATION WITH EXISTING OPERATIONS

Deployments align naturally with established IT, security, and DevSecOps workflows without requiring architectural exceptions or specialized handling.

These benefits are not additional features or optimizations. They are direct consequences of architectural alignment and strict boundary control inherent to the D-SaaS model.

Comparison Matrix

The following matrix contrasts Deployable SaaS with common software delivery models to highlight differences in ownership, control, and compliance alignment.

Capability	**Traditional SaaS**	**Single-Tenant Hosted SaaS**	**On-Prem Software**	**Deployable SaaS (D-SaaS)**
Data Ownership	Vendor-controlled	Shared	Customer-owned	Customer-owned
Boundary Control	External	Partial	Internal	Internal
Identity & MFA	External provider	Mixed	Customer-managed	Customer-managed
Logging & Audit	External or limited	Partial	Local	Local
Deployment Flexibility	Vendor-defined	Limited	High	High
Delivery Format	Web application	Hosted service	Workstation/ server installable	Container or VM appliance
Lifecycle Model	Continuous delivery (remote)	Managed by provider	Manual upgrades	Artifact-based updates under customer control

Table 1: Software Delivery Model Comparison

Is D-SaaS Just Like Legacy Software?

While Deployable SaaS shares the concept of customer-managed deployment with traditional boxed software, the two models differ fundamentally in architecture, delivery format, and operational expectations.

Aspect	**Legacy Software (Box Model)**	**Deployable SaaS (D-SaaS)**
Target Environment	Workstations, servers	Private/hybrid cloud, orchestrated IaaS
Deployment Format	Installers (EXE, MSI, ISO)	Containers, VM appliances

Aspect	**Legacy Software (Box Model)**	**Deployable SaaS (D-SaaS)**
Update Model	Manual patches or media	Signed artifacts, controlled rollout
Compliance Alignment	Optional or indirect	Architectural by design
Connectivity Assumptions	Standalone or networked	Airgap-compatible
Licensing	Product keys, dongles, or activations	Cryptographic, offline validation

Table 2: Legacy Software vs Deployable SaaS

Deployable SaaS is purpose-built for modern private cloud environments. It supports air-gapped operation, identity federation, audit-grade logging, and continuous vendor lifecycle responsibility while preserving customer ownership of infrastructure and enforcement boundaries. It is not a rebranded legacy software model. It is a modern service-architecture designed for sovereignty and compliance.

Relationship with Related Terms

Deployable SaaS (D-SaaS) exists within a broader ecosystem of terms used to describe alternative software delivery and deployment models. Some of these terms are used informally or interchangeably in industry discourse. This section clarifies how D-SaaS relates to, and differs from, commonly referenced adjacent concepts.

Self-Hosted SaaS

“Self-hosted SaaS” is an imprecise term generally used to describe software that is deployed within customer infrastructure rather than operated entirely by the vendor. In practice, self-hosted SaaS implementations often retain one or more vendor-controlled elements, such as external control planes, licensing servers, telemetry pipelines, identity dependencies, or remote management capabilities.

D-SaaS is not equivalent to self-hosted SaaS. While both may be deployed within customer infrastructure, D-SaaS imposes strict architectural invariants that prohibit vendor-operated runtimes, externalized control planes, or out-of-boundary enforcement mechanisms. Deployment location alone is insufficient to meet the D-SaaS standard.

SaaS Anywhere

“SaaS Anywhere” is commonly used as a marketing or product positioning term to indicate that a SaaS offering can be deployed across multiple environments, including public cloud, private cloud, or customer-managed infrastructure. The term emphasizes portability or deployment flexibility but does not define ownership, enforcement authority, or audit responsibility.

D-SaaS differs fundamentally in intent and scope. It is not a statement about where software can be deployed, but about who retains control over execution, identity, data, logging, and lifecycle management. A SaaS Anywhere solution may still rely on vendor-operated identity services, update pipelines, or telemetry endpoints. Such dependencies are explicitly disallowed under the D-SaaS architectural invariants.

On-Prem Software

Traditional on-premises software is typically delivered as installable binaries or packages intended for workstation or server deployment. While this model places software inside the customer boundary, it often lacks modern service architecture characteristics such as API-first design, artifact-based lifecycle management, cloud-native deployment patterns, and compliance-aligned observability.

D-SaaS is not a return to legacy on-prem software. It is a modern service-oriented delivery model designed specifically for private and hybrid cloud environments, supporting containerized and virtualized deployment, identity federation, audit-grade logging, and air-gapped operation while preserving customer ownership and enforcement authority.

Terms such as self-hosted SaaS and SaaS Anywhere describe deployment flexibility or hosting arrangements. Deployable SaaS defines an architectural and governance model.

D-SaaS establishes enforceable constraints on runtime ownership, control planes, identity integration, data locality, auditability, and lifecycle management. These constraints are essential for regulated and boundary-controlled environments and are not guaranteed by adjacent or informal delivery models.

For this reason, D-SaaS should be understood as a distinct, compliance-aligned architectural standard rather than a synonym for existing deployment terminology.

D-SaaS Compliance and Certification Model

Deployable SaaS establishes two distinct and complementary designations to indicate conformance with the D-SaaS architectural standard: D-SaaS Compliant and D-SaaS Certified. These designations are intentionally differentiated to reflect different levels of assurance while preserving the clarity and integrity of the standard.

D-SaaS Compliant

indicates that a software product conforms to the Deployable SaaS architectural standard based on a self-assessment against the published D-SaaS invariants.

This designation reflects architectural alignment and intent, not independent validation.

Key Characteristics

- Self-attested by the product owner or developer
- Demonstrates architectural alignment with the D-SaaS standard
- Deployable into customer-owned infrastructure, including private cloud IaaS
- Data, identity, logging, and operational control boundaries remain under customer control
- No vendor-operated runtime, shared tenancy, or external control plane is required

D-SaaS Compliant is an architectural declaration, not a certification. It does not imply independent verification, formal assurance, or endorsement by Lab 0x2A beyond conformance with the published standard definition.

Permitted Usage

The D-SaaS Compliant mark may be used in:

- Product documentation
- Architecture and system design diagrams
- White papers and technical briefs
- Marketing or informational materials that accurately describe architectural alignment

D-SaaS Certified

indicates that a software product has undergone independent verification by Lab 0x2A and has been validated against the D-SaaS standard and its defined acceptance criteria.

This designation represents a higher level of assurance than self-attestation.

Key Characteristics

- Verified against formal D-SaaS acceptance and validation criteria
- Includes review of deployment architecture, control boundaries, and runtime behavior
- Confirms adherence to D-SaaS architectural invariants in practice
- Certification may be version-specific, environment-specific, and time-bounded

- Certification may be suspended or revoked if the product no longer conforms

D-SaaS Certified signals that the product has been evaluated beyond self-assessment and meets the requirements of the D-SaaS standard as verified by the standard's steward.

Permitted Usage

The D-SaaS Certified mark may be used only:

- After successful completion of the Lab 0x2A verification process
- In formal documentation, proposals, and compliance-relevant materials
- In contexts where architectural assurance or independent validation is required

Reference Implementation and Stewardship

The Deployable SaaS architecture was originally conceived under Hodge IP & Holdings Co. LLC and implemented through Lab 0x2A LLC, which together continue to develop secure software products that conform to the D-SaaS architectural model.

Deployable SaaS is not a proprietary framework. It is an open architectural standard intended to be adopted, implemented, and evaluated by any vendor or organization that requires compliance-grade data sovereignty and strict boundary control.

Lab 0x2A serves as the reference implementer and steward of the D-SaaS standard. Its role is to maintain the architectural definition, publish conformance criteria, and provide optional verification for organizations that seek independent validation of D-SaaS compliance.

The objective of D-SaaS is to establish a repeatable and defensible software delivery pattern for regulated and boundary-controlled environments, enabling modern service-based capabilities without relinquishing ownership of infrastructure, identity, or enforcement authority, while reducing operational overhead and increasing customer adoption and acceptance.

Conclusion

Deployable SaaS was created in response to the structural limitations of traditional SaaS delivery models in regulated and compliance-intensive environments.

By decoupling SaaS capability from vendor-hosted infrastructure, D-SaaS restores data sovereignty, preserves auditability, and aligns software delivery with the architectural assumptions embedded in modern compliance frameworks.

D-SaaS is not a workaround or an incremental adaptation of existing models. It is a structural correction to a systemic problem and a viable path forward for delivering secure, compliant, and modern service-based software without relinquishing control of infrastructure, identity, or enforcement authority.

Glossary

Data Sovereignty

The principle that an organization retains technical ownership and enforcement authority over its data, including how and where it is processed, stored, and audited.

Boundary Control

The ability to define, enforce, and govern technical perimeters for data, identity, access, and operational authority within an environment.

Tenancy

The logical and infrastructural separation between different customers, environments, or operational domains within a shared or multi-tenant platform.

Control Plane

The system layer is responsible for managing configuration, identity integration, policy enforcement, updates, and orchestration of services.

Deployable SaaS (D-SaaS)

A software delivery architecture in which SaaS applications are deployed and operated entirely within customer-owned infrastructure, with no vendor-operated runtime, shared tenancy, or external control-plane dependency.