Supply Chain Research · Cornerstone Reference

Cloud vs On-Premise vs SaaS

Should I Run Supply Chain Software On-Premise, in the Cloud, or as SaaS?

The real question is not cloud or not-cloud. It is who controls upgrades, who owns security, where the data sits, and how the cost is shaped over time. Cloud-hosted and SaaS are not the same thing, and the difference decides most of what matters.

CONTENTS

01 The short answer
02 What do on-premise, cloud, and SaaS actually mean?
03 Is cloud-hosted the same as SaaS?
04 Who is responsible for security in each model?
05 What does each model do to cost and upgrades?
06 Which model fits which situation?
07 Frequently asked questions
08 Method, sources, and where to go deeper

01 The short answer

On-premise means you license the software and run it on infrastructure you control. Cloud-hosted means it runs on off-premise infrastructure, which may still be a dedicated, single-tenant instance you version and upgrade yourself. Multi-tenant software as a service means you subscribe to one shared, continuously updated service the vendor operates. The decision turns on four axes: who controls when upgrades happen, who is responsible for security, where the data resides, and how cost is shaped over time. The most common error is treating cloud and SaaS as one choice, when a single-tenant cloud deployment can carry the upgrade and version burden of on-premise software.

4	3	1
axes that actually decide the model: upgrades, security, data, cost	deployment models, not two: on-premise, hosted, multi-tenant SaaS	distinction most buyers miss: cloud-hosted is not the same as SaaS

Key takeaways

- **Cloud-hosted and SaaS are different decisions.** A single-tenant cloud instance keeps on-premise-style version control; multi-tenant SaaS does not.
- **SaaS removes upgrade projects and removes upgrade control.** You cannot defer or decline a versionless update, which is a benefit and a constraint.
- **Security is shared, and the split moves with the model.** The more the vendor operates, the more of the security burden shifts to them, but never all of it.
- **Data residency is not the same as data sovereignty.** Region of hosting does not by itself settle which laws can reach the data.
- **This is a hosting decision, not a cost or build decision.** Cost level belongs in a total-cost analysis; whether to build belongs elsewhere.

02 What do on-premise, cloud, and SaaS actually mean?

The clearest neutral vocabulary comes from the United States National Institute of Standards and Technology, whose cloud computing definition sets out three service models (infrastructure, platform, and software as a service) and four deployment models (private, community, public, and hybrid). Software as a service is the model in which the consumer uses the provider's applications running on cloud infrastructure and does not manage the servers, storage, or the release schedule. On-premise sits outside this framework: the software runs on infrastructure the organization owns or controls, and it manages everything from the hardware up.

Between those two sits a middle that causes most of the confusion. A single-tenant or private-cloud deployment places a dedicated instance on cloud infrastructure, but the customer still controls the version and schedules its own upgrades. It is hosted off-premise yet behaves much like on-premise software for upgrades and customization. Multi-tenant SaaS is the opposite: one shared instance serves many customers, the vendor updates it continuously, and no customer holds a private version. Naming which of the three a product is matters more than whether it is called cloud.

Figure 1. The three deployment models and the trade in each. The axis is control against burden: the more the vendor operates, the less you carry and decide.

The fourth deployment model that NIST names is the one most real supply chain estates end up in: hybrid, a composition of two or more distinct models bound together so that data and applications move between them. Few organizations run a single model cleanly. A planning suite may sit in multi-tenant SaaS while a warehouse system remains on-premise for latency or integration reasons, and a data platform bridges the two. Multi-cloud, running across more than one public cloud provider, is a related pattern driven by resilience or procurement policy rather than by the software itself. The useful consequence is that the deployment decision is rarely made once for the whole estate. It is made per system, against the four axes, and the estate is the sum of those choices.

03 Is cloud-hosted the same as SaaS?

No, and conflating them produces expensive surprises. Cloud-hosted describes where the software runs; software as a service describes how it is delivered, as a shared, vendor-managed, versionless subscription. A vendor can lift an on-premise application, place a dedicated single-tenant copy in a public cloud, and call it cloud, while the customer still owns the upgrade project, the version, and much of the operational burden. That is cloud-hosted but not SaaS.

The practical test is to ask whether other customers run the same instance you do, and who decides when it is upgraded. In true multi-tenant SaaS, you share the instance and the vendor upgrades everyone together on its schedule. In a single-tenant hosted deployment, your instance is yours, and its version can diverge from other customers and lag the vendor's latest release. Those are different commitments with different cost, customization, and upgrade consequences, and the word cloud does not distinguish them. Asking the tenancy question is the only reliable way to know which is on offer.

04 Who is responsible for security in each model?

Security is shared in every cloud model, and the division shifts with the service model rather than disappearing. The NIST cloud computing reference architecture states the point plainly: because of the service offerings a provider allows, there is a shift in the level of responsibility for some aspects of control, security, and configuration, and that responsibility is shared between provider and consumer. What changes across models is where the line falls, not whether a line exists.

In broad terms, the more of the stack the vendor operates, the more of the security burden moves to it. On-premise, the organization owns essentially all of it, from physical security to patching. In infrastructure or platform services the provider secures the underlying layers while the customer secures its configuration, access, and data. In multi-tenant SaaS the vendor operates and secures the application, while the customer remains responsible for its users, access controls, and data governance. A widely used industry framing distinguishes security of the cloud, which the provider owns, from security in the cloud, which the customer owns; that phrasing comes from a cloud provider rather than a neutral body, but it captures the division usefully.

The error to avoid is assuming SaaS outsources security entirely. It does not. The vendor takes the application and infrastructure layers, but access management, user provisioning, and the correct handling of your data remain yours, and most cloud security failures arise on the customer side of that line.

Data residency and data sovereignty are worth separating precisely, because they are frequently conflated in vendor assurances. Residency is where the data physically sits, which a provider can often guarantee by region. Sovereignty is which government's law can compel access to it, which region alone does not settle. Under the United States CLOUD Act, for example, a United States provider can face a lawful demand for data it controls regardless of where that data is stored, so hosting in a European region does not by itself place the data beyond United States legal reach. For a regulated buyer the implication is that the deployment question and the legal question must be answered together, and that a residency commitment in a contract is necessary but not sufficient for a sovereignty requirement.

05 What does each model do to cost and upgrades?

Cost shape differs more than cost level, and the shape is what the deployment choice determines. On-premise concentrates spend into upfront capital, licensing and hardware, followed by maintenance, with the organization carrying the infrastructure. Multi-tenant SaaS converts that into a recurring operating subscription with little upfront capital, usually with annual escalators. Single-tenant hosting sits between, combining subscription or hosting fees with more operational responsibility. This page addresses cost shape; the level and the full total cost of ownership belong in a dedicated analysis.

The upgrade consequence is the one buyers underweight. In multi-tenant SaaS, updates are continuous, mandatory, and versionless: upgrade projects largely disappear, but you cannot defer, decline, or fall behind a release, so a disruptive change arrives on the vendor's timetable. In on-premise and single-tenant deployments you control upgrade timing, which protects you from unwanted change but exposes you to version lock: falling so far behind that a future upgrade becomes a costly re-implementation. Neither is strictly better; they trade control for currency in opposite directions.

	On-premise	Single-tenant hosted	Multi-tenant SaaS
Upgrade control	You decide timing	You decide timing	Vendor decides; continuous
Cost shape	Capital plus maintenance	Hosting plus subscription	Operating subscription
Security burden	Almost entirely yours	Shared, more on you	Shared, more on vendor
Customization	Deepest	High	Constrained to configuration
Main exposure	You run everything	Version lock over time	No control over change

Table 1. The three models across the axes that decide the choice. Upgrade control is the row that most often surprises buyers, invisible until the first forced or deferred upgrade.

Exit is the cost shape buyers examine last and regret first. A multi-tenant SaaS deployment keeps the data in the vendor's shared environment, and retrieving it into a different system, or into a self-hosted deployment, can be a project in its own right, assuming the vendor offers an export path at all. On-premise and single-tenant deployments hold the data where the organization can reach it, which lowers exit cost at the price of higher operating burden. None of this argues against SaaS. It argues for settling export format, data-portability terms, and any assistance obligations in the contract before signing, because the leverage to negotiate them is at its highest before the deal closes and near zero afterward.

06 Which model fits which situation?

Profile and constraint decide the fit more than preference. Multi-tenant SaaS suits organizations that want the vendor to carry operations and upgrades, can work within configuration rather than deep customization, and accept continuous update. It is the market's default direction for good reasons, and the right answer for many operations, particularly where requirements are standard and internal infrastructure capability is limited.

On-premise and single-tenant hosting retain their place where specific constraints apply: strict data residency or sovereignty requirements that a shared service cannot meet, deep customization that a multi-tenant product cannot accommodate, integration with systems that must remain on-premise, or a regulatory

environment that requires control over upgrade timing and validation. The fair case for these models, which the momentum toward SaaS tends to drown out, is that control is itself a requirement in some industries, and a deployment that hands upgrade timing to a vendor is not acceptable where every change must be validated before it reaches production. The honest approach is to identify which of the four axes are genuine constraints for your operation, and let those decide, rather than adopting a model because it is the prevailing choice.

07 Frequently asked questions

What is the difference between cloud, hosted, and SaaS?

Cloud describes running on off-premise infrastructure. Hosted usually means a dedicated, single-tenant instance on that infrastructure, which you still version and upgrade. SaaS means a shared, vendor-managed, continuously updated service. A product can be cloud-hosted without being SaaS, and the difference drives upgrades, customization, and cost.

Can I still run supply chain software on-premise, and when does that make sense?

Yes. It makes sense where data residency or sovereignty rules require it, where deep customization is needed, where integration with on-premise systems is essential, or where regulation requires control over upgrade timing. It carries the full operational and security burden, so it is a deliberate choice rather than a default.

Does SaaS mean I lose control of when I upgrade?

In true multi-tenant SaaS, largely yes. Updates are continuous and versionless, so you cannot defer or decline them. That removes upgrade projects but means a disruptive change arrives on the vendor's schedule. If controlling upgrade timing is a hard requirement, single-tenant or on-premise deployment preserves it.

Who is responsible for security in SaaS?

It is shared. The vendor secures the application and the infrastructure it runs on; you remain responsible for user access, provisioning, and the governance of your own data. Assuming SaaS outsources security entirely is a common and costly error, since most cloud security failures occur on the customer side of that line.

Does hosting in my region guarantee compliance?

Not by itself. Data residency, where the data physically sits, is not the same as data sovereignty, which concerns whose laws can compel access to it. Hosting in a given region can be necessary for compliance without being sufficient, and the legal question should be checked separately from the technical one.

Can I move from SaaS back to on-premise later?

Rarely without difficulty. Multi-tenant SaaS keeps your data in the vendor's shared environment, and exporting it into a self-hosted deployment can be a significant project, if the vendor even offers a self-hosted option. Exit and data-portability terms are worth settling before signing, because the leverage to negotiate them disappears afterward.

What is a hybrid or multi-cloud deployment?

Hybrid means composing more than one deployment model, for example a SaaS planning tool alongside an on-premise warehouse system, with data moving between them. Multi-cloud means running across more than one public cloud provider, usually for resilience or procurement reasons. Most real supply chain estates are hybrid, because the right model differs by system, so the decision is made per application rather than once for everything.

How do I avoid lock-in when choosing SaaS?

Settle the exit terms before signing. Ask for the data export format, the portability obligations, and any assistance the vendor will provide on the way out, and get them into the contract. Multi-tenant SaaS concentrates switching cost because the data lives in the vendor's shared environment, so the protection is contractual, and it is far cheaper to secure before the deal closes than after.

08 Method, sources, and where to go deeper

Method

- Deployment and service-model definitions follow the United States National Institute of Standards and Technology, whose cloud computing definition and reference architecture are neutral, government-published standards rather than vendor material.
- The shared-responsibility framing is taken from the NIST reference architecture; the security-of-the-cloud versus security-in-the-cloud phrasing is identified in the text as vendor-originated and used only for its clarity.
- Supply Chain Research is independent and vendor-neutral. We accept no payment from the vendors or categories covered, and this page names no products.

Caveats

- Published figures on the cloud-versus-on-premise adoption split are unreliable and contradict each other. Different commercial publishers report materially different shares for comparable periods, so this page cites no single adoption figure and treats the direction of travel, not a precise number, as the only defensible claim.
- Cost is addressed here as shape rather than level. Any dollar comparison depends on the specific product, scale, and contract, and the widely quoted cost rules of thumb originate with interested parties. The total-cost question belongs in a dedicated analysis.
- Table 1 and Figure 1 are structural comparisons, not measured data. The security divisions describe common practice under each model rather than any single vendor's contract, which should be read directly.

Where to go deeper

Two SCR resources treat cost and integration dimensions this page only summarizes. The SCR editorial on total cost of ownership covers how to compare deployment models on cost level over time, including the escalators and staffing this page deliberately leaves out. The SCR editorial on vendor lock-in covers the exit and portability exposure that a multi-tenant deployment concentrates, which is the risk behind the question of moving back off SaaS. Readers scoping the wider stack should start with the SCR supply chain software category map.

Sources

[1] National Institute of Standards and Technology. The NIST definition of cloud computing, SP 800-145. Standards body; the neutral definition of the service and deployment models.

[2] National Institute of Standards and Technology. NIST cloud computing reference architecture, SP 500-292. Standards body; the neutral basis for the shared-responsibility model.

[3] Amazon Web Services. Shared responsibility model. Interested source: a cloud provider. Cited only for the security-of versus security-in framing.

[4] Association for Supply Chain Management. SCOR Digital Standard overview, used for process placement of technology enablement.

Supply Chain Research is an independent, vendor-neutral research platform for supply chain and technology leaders. We accept no payment from the vendors or categories covered. This page is analysis, not procurement or security advice, and its conclusions should be validated against your own circumstances before any decision.