Solution Overview

Scalr powers faster cloud adoption and application deployment at some of the world's largest enterprises. Companies such as Gannett, Expedia, NASA JPL and Centene achieve cost-effective, automated and standardized application deployments across multi-cloud environments with the Scalr Enterprise-Grade CMP.

Scalr utilizes a hierarchical, top-down approach to policy enforcement that empowers organizations to find the balance between the needs of finance, security, IT and development teams.

A multi-cloud infrastructure manifests in many different ways. In some enterprises, application teams adopt different clouds independently to fit their needs. In others, developers use one cloud for testing and a datacenter for production. The variations are endless. The point is – every enterprise does multi-cloud differently, and every enterprise has different constituencies with a wide variety of needs.

Scalr empowers enterprises to safely and rapidly deploy to the cloud by addressing the four major problem areas:

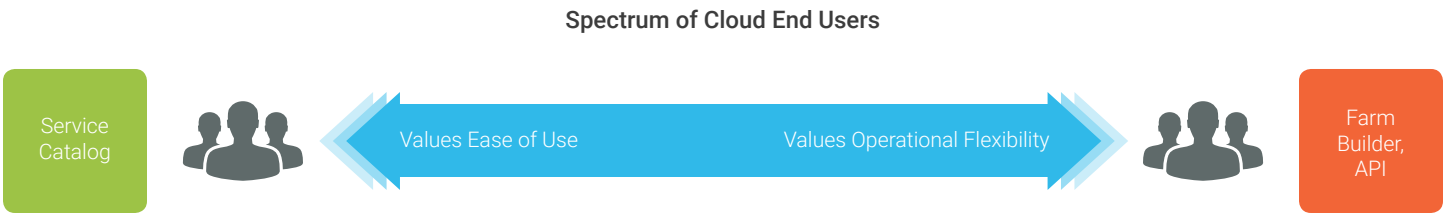
- **Cost Optimization & Visibility** – As Scalr is a part of the provisioning workflow, it's uniquely positioned to provide contextual financial information, and enforce sophisticated usage policies to reduce costs.
- **Governance, Security, and Compliance** – Scalr's Policy Engine creates re-usable guardrails around Access, Workload Placement, Application Lifecycle, Integrations, and Finance. End-users are exposed to relevant resources based on their identity and the cloud environments in which they operate.
- **Business Agility** – Scalr abstracts both provisioning and policy enforcement across multiple platforms. Scalr's abstractions make cloud skills more portable, helps reduce vendor lock-in, and gives app teams the flexibility to shift workloads between clouds. The Scalr Service Catalog fully enables self-service environments and reduces application delivery times.
- **Increased productivity** – Scalr's hierarchical model reduces operational overhead. Policies are set at a Business Unit/Department level are propagated to the relevant subset of applications and users. Scalr's policy inheritance model, paired with a single API that standardizes multiple clouds, allows DevOps, developers, and IT to move faster and focus on solution delivery. Users are matched with the provisioning methods that make them most productive, from one-click provisioning to complex application architectures.

SCALR

Types of enterprise users

Enterprise IT serves many types of users. Through our work with some of the world's largest enterprises, we found that most users fall in a spectrum between these two types of users:

- **Advanced users** who require more flexibility and operational freedom – *For example, DevOps engineers and Cloud Architects.*
- **Regular users** who require greater ease of use at Self-Service – *For example, QA Engineers or Sales Engineers.*



Scalr offers the only Cloud Management solution that serves all types of users. Users with varying needs and permissions can seamlessly operate in the same environments. Based on their identity and the environments they operate in, different users will only see the resources and provisioning options relevant to them.

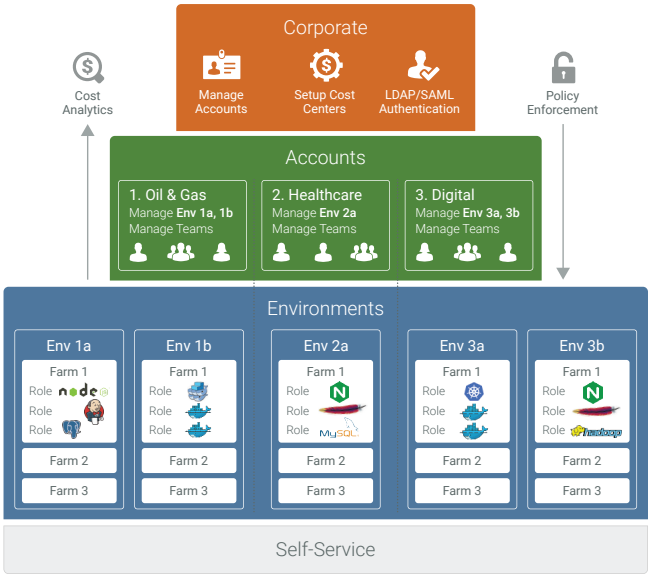
A DevOps engineer building a complex CI/CD pipeline, complete with multiple API calls to 3rd party tools and automated promotion of applications, will go through the Farm Builder or the Scalr API. This advanced user will have the operational freedom she needs to move fast and be more productive, while always remaining within the scope of policy. Policies can be set around all aspects of cloud usage, instance types, budgets, reclamation, and more.

A QA Engineer who needs to repeatedly spin up the same test box will benefit more from an easy Self-Service workflow. With Scalr's Service Catalog, admins can set up blueprints for anything from a single server to a complete application stack, and deliver them to users as a one-click provisioning experience.

SCALR

The Scalr Hierarchical Model

Scalr is built to operate in a massive scale. This is made possible by the hierarchical policy inheritance model. When enforcing policies at a large scale and offering self-service cloud resources to hundreds or thousands of users, it doesn't make sense to tie policies to each individual application. When policies are defined at the application level, introducing changes becomes difficult, and the separation of responsibility becomes challenging.



Scalr uses a tiered model to map the company's organization structure. Like the federal government, Scalr employs a hierarchical model composed of three management scopes. At each level, the relevant admin can configure policies, catalog items, automation and more. The policies configured at a certain scope will be inherited by all lower scopes.

Scalr Enterprise Scopes

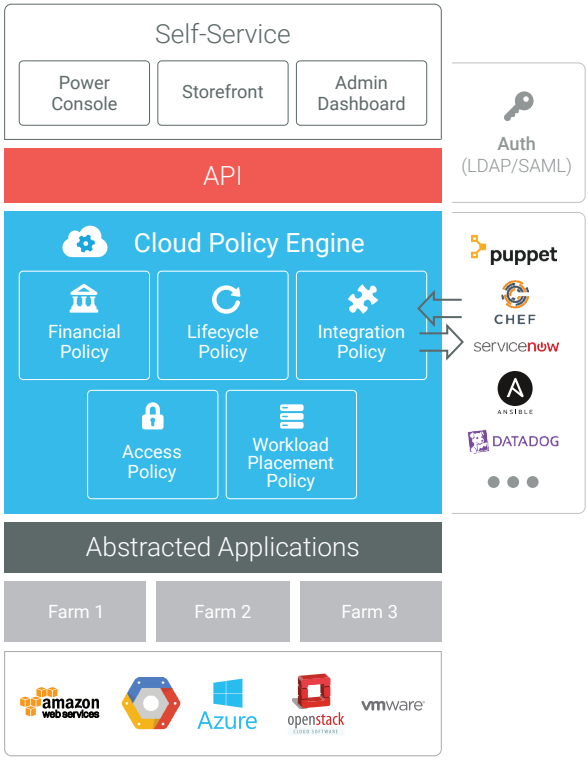
- **Corporate Scope** - The highest scope of management. This scope represents whoever owns cloud in the organization. From the Corporate scope admins can monitor cloud spend and set up budgets and reports, and manage policies that all affect all internal "cloud-customers" in the organization.
- **Account Scope** - A Scalr Account typically represents a Business Unit or Department in the enterprise. This is an compartmentalized cloud deployment that can contain multiple clouds, resources and policies, which are logically grouped as Environments. Any policy configured at the Account Scope can affect Environments within that Account.
- **Environment Scope** - A Scalr Environment is a logical grouping of Cloud resources and the policies that apply to them. Access to Environments can be governed by fine-grained RBAC policies that determine what users can do once they've logged in. End-users "live" in the Environment Scope, provisioning applications at self-service within company policy. Environments can be grouped by whatever makes sense for the organization. For example, Environments can be organized for Dev/Test, Staging and Production.

The Scalr Hierarchical model allows enterprises to safely use a multi-cloud infrastructure at scale and reduce operational overhead. Management can take a top-down approach and easily monitor usage, introduce new policies and allow users to focus on building great applications. This model also allows for greater flexibility when moving applications between Environments or changing application policy profiles.

SCALR

The Scalr Policy Engine

Scalr layers policies on cloud usage based on a user's identity and the environment she operates in. Policies adhere to Scalr's inheritance model, which means that a policy configured at a higher level will be propagated to all relevant environments. Once users with the proper permission log into these environments, RBAC policies can be applied based on their identity.



Scalr policies generally fall into five categories:

- **Access Policies** – Permissions assigned to teams. Once users have logged into their provisioning portal, what can they see? What actions can they perform? Can they build their own applications or just use the service catalog? Can they modify security groups or SSH keys? Can they see workloads provisioned by other teams? These types of questions are answered by Access Policies.
- **Workload Placement Policies** – Usage and provisioning restrictions in Cloud environments. Which server sizes are available for Windows machines? Which networks can be used? Must storage be encrypted? Which locations are available? These types of questions are answered by workload placement policies. Workload Placement Policies include both platform specific objects like Azure Resource Groups and VMware Datastores, and cloud agnostic objects like Chef servers.
- **Integration Policies** – How other tools interact with the provisioning workflow. Examples for Integration Policies include logging actions to CMDBs, leveraging configuration management tools such as Chef, Puppet and Ansible, and more.
- **Application Lifecycle Policies** – Automation that governs application lifecycle from provisioning to termination. Application lifecycle policies cover all aspects of app automation from bootstrapping servers with scripts, ongoing maintenance, auto-scaling and scheduled app termination.
- **Financial Policies** – Cost reduction and cost metering policies. Financial policies include budgeting tools, notifications around budget consumption, showback/chargeback and financial reports. Cost reduction policies tie in to other Scalr policies such as reclamation of unused resources, application lifetime, ensuring usage of appropriate server sizes and more.

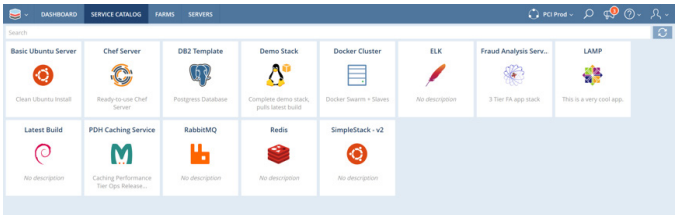
SCALR

Self-Service Powers Faster Application Deployment

When we push a button on a vending machine and get our soda, we often don't think of everything that goes into that process behind scenes. Similarly, it's easy to forget that Cloud Self-Service involves multiple enterprise groups.

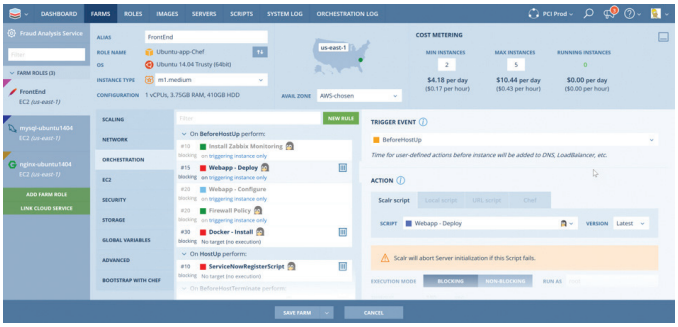
To achieve fast, easy and cost-effective self-service, Scalr matches users with the provisioning method that makes them most productive based on identity and operating environment.

Service Catalog – The Scalr Service Catalog is a simple provisioning method that makes it easy for any user to log into an environment and provision an application in a few clicks. Administrators can grant users access to multiple catalogs. For example: Jane the QA engineer has access to both the "TEST" environment where she can provision an app stack for testing with the latest build, and the "SANDBOX" environment where she has more operational freedom to try out new technologies.



Catalog items can represent a single server, or an application stack complete with cloud services, automation rules and auto-scaling. The complexities of the stack are abstracted away from the end-user.

Farm Builder – The Scalr Farm Builder is a more advanced way for developers, DevOps engineers, cloud architects and administrators to describe the desired state of their applications. With the Scalr Farm Builder, users have more operational flexibility to customize every aspect of their stack, while still operating within company budget and policy.



The Farm Builder uses a declarative style to create the application's desired state. The desired state can be captured in a YAML file and includes all aspects of app deployment including platforms, images, security, networking, storage, automation rules and auto-scaling.

Once an application is fully designed, it can be offered up as a Service Catalog item.

About Scalr

The Scalr Enterprise-Grade Cloud Management Platform enables today's enterprises to achieve cost-effective, automated and standardized application deployments across multi-cloud environments. Scalr utilizes a hierarchical, top-down approach to policy enforcement that empowers administrators to find the balance between the needs of finance, security, IT and development teams. Founded in 2007, leading global organizations have selected the Scalr platform, including Samsung, Expedia, NASA JPL, Gannett and Food & Drug Administration.