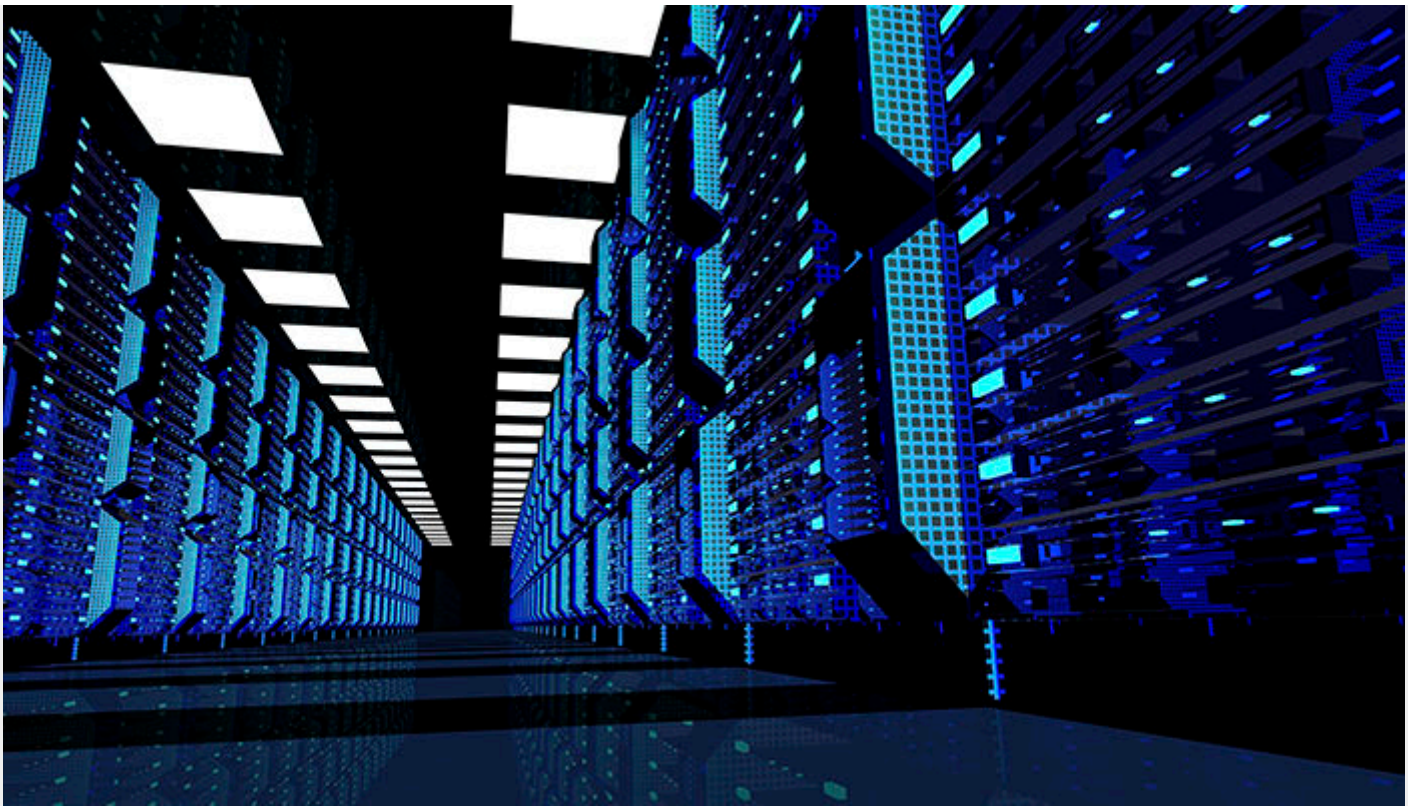
WHAT IS SERVER AUTOMATION? SERVER AUTOMATION EXPLAINED



Traditional server management can include hundreds of infrastructure elements managed by several systems and teams over various data centers. It's a complex network, often leading to delays and small errors in a time when the pressures on business technology are only growing. Businesses frequently demand a shift away from discrete devices in favor of delivering enterprise-wide services with greater flexibility, speed, and efficiency. But doing more with the same old tools just won't cut it. Server automation can reduce the complexity of manual server management while simultaneously increasing efficiency and effectiveness.

Server automation consolidates task and process automation into a single solution for more efficient application deployment as well as ongoing management across virtual and physical systems. It enables end-to-end management through the creation of workflows that automatically coordinate tasks and processes, leading to a reduction in human-caused errors. It is particularly useful for small developer teams who need to constantly keep production rolling to enable a continuous delivery process.

Below you'll find additional details on what server automation is, the benefits it can provide, and how to get started.

Features and Benefits of Server Automation

Clean Architecture

The primary server-agent architecture of an automated server consists of a core and an agent. The core is the physical server component which can run on one or multiple servers and saves information about server configurations. Distributing the core components across multiple servers scales the functionality while increasing performance.

Meanwhile, the agent side of the architecture is a software component that runs on the server's operating system. Its primary purpose is to communicate with the core, allowing the core to manage the server's life cycle.

Compare this to non-automated situations where a data center of thousands of servers requires manual management through a long list of different tools. Installing a server automation core and deploying server automation agents into an environment reduces these requirements and streamlines management.

Secure and Reliable

With the simple automated architecture of server automation comes increased reliability. Humans are naturally prone to mistakes and slip-ups for any number of reasons, but a machine performs to perfection given proper programming. And this perfect performance frees up a good deal of time and effort that the developer team would have previously devoted to checking for errors.

Because the team can worry less about double-checking, fewer users will need access to the server. The automated server is essentially the only one that needs access, so it can eliminate risky extended user access. Reduced access permissions and effective error checking lead to a secure working environment for the developer team.

Easier and More Reliable Environment Setup

Speaking of user error, manual installation of servers can lead to unique setups and behaviors. While uniqueness is beneficial in many contexts, it's better to have homogenous servers that avoid the emergence of distinct and unknown bugs.

Server automation allows such homogeneity, leading to more consistent performance. If there are any bugs, they will be the same throughout all servers, thus being easily remedied. This results in a consistent disaster recovery process when it comes to hardware problems.

Instant Feedback

As the developers work alongside server automation, they can take advantage of its all-seeing eye to generate immediate notifications and reports based on every task the server completes. The options for notification warnings are flexible in that they can be directed to the commit author individually or the whole team.

For example, a developer committing code to the central repository for other team members to see can receive warning notifications if the automated server determines that the commit causes problems for the build. This addresses the problem at the source rather than allowing it to cause

further issues down the road, requiring the interference of other team members.

Use of Software Policies

Server automation allows for the modeling of software through the use of policies. Various specifications can be included within a software policy, including which configurations will apply to managed servers and which packages and patches will be installed, as well as users, groups, files, and scripts.

There is also a framework within the server automation, ensuring compliance with defined policies. The framework helps identify and fix specific instances where the server does not comply with the software policies.

Range of Interfaces and Tools

You have a few options when it comes to making use of server automation. The most widely-used tool by most server administrators is the SA Client. It is a Windows desktop application that uses java for provisioning, policies, and software management. Similarly, there is the SAS Web Client that provides a web-based interface where users can perform tasks such as server management, track configuration changes, and manage user permissions.

Additionally, an SA Command Line Interface (OCLI) provides an efficient means of file management within the software repository and a Data Center Markup Language (DCML). Exchange Tool is great for transferring information from one core to another. For creating and uploading ISMs, the ISM Development Kit contains various useful command-line libraries and tools.

Testing and Rollback of Deployments

One of the biggest issues many companies have is successfully packaging and delivering their software to the customer. Larger companies usually turn to automation, but smaller teams might see it as too great an expense or unnecessary for their smaller workload. That's not the best perspective to take, though, as many developers focus so much of their time on enhancing an app's features, they often neglect to make improvements in delivery. Server automation can help to facilitate and speed up the deployment process in various ways, starting with the creation of test environments.

Test environments that use the same processes as production enable test deployments where developers can see new features perform before sending them to the final customer. This process optimally reveals any bugs or malfunctions, thus enhancing the software quality of the final product. Server automation also allows the implementation of a rollback process to avoid errors after the test environment is replicated into production.