What is Automated Provisioning?

Step right up, Ladies and Gentlemen!

Are you getting the best bang for the buck out of your IT resources? Are there times when performance is poor and you know you have unused resources? Do you wish you could automatically redistribute those unused resources to the problem area? Do you want to do this by minimizing expensive manual efforts and reducing the possibility of human mistakes? Do you want to save time and money? Well, look no further, we have the cure for your problem(s): automated provisioning!

Ok, Ok—sales pitch set to OFF. Automated provisioning is definitely a step forward, especially for highly variable workloads and datacenters that are willing to shift resources from one application to another. This describes many networked configurations, including an increasing number of commercial apps. On the other hand, while IT is broadly on a journey toward the dynamic datacenter with automated provisioning one of its key tools, we haven’t arrived there quite yet.

Send in the Clones

Provisioning itself has been with us all along. The terminology started in the telco arena, where it means the act of acquiring a service, from the submission of a requirement (request) through the activation of service. In IT, it usually means the allocation and configuration of a computer system to provide a service; in practice, today, the focus is often on the installation and activation of appropriate software on an existing server, though the installation and configuration of servers, storage, networking gear, and other components is also included.

Armies of highly-trained, highly-paid IT specialists do provisioning every day—but manually. They usually walk to the device in question, equipped with the appropriate installation media, and tediously baby-sit the configuration and software deployment tasks required. Everyone knows that IT resources—both human

1. See, for example, U.S. Federal Standard 1037C. http://www.its.bldrdoc.gov/fs-1037/fs-1037c.htm
and nonhuman—are scarce and expensive. So IT departments would like to use these resources to their maximum potential. But doing so is hard because systems are usually a multi-vendor hodge-podge that often requires many different kinds of software and the skills of multiple sets of highly trained people.

Thus automated provisioning! The hard nut many vendors are trying to crack is the automation of installation and configuration procedures to better meet demands for simpler, more self-managed, and more dynamic computing environments. This automatic activation and setup process can be done either through scripting or by cloning (installing an exact copy of) a known working configuration.²

By allowing datacenters to quickly and efficiently allocate resources when they are needed, automated provisioning promises to improve resource utilization. It solves the dilemma of either initially overbuilding the IT infrastructure to cover all potential contingencies (costly), or under-specifying the infrastructure, potentially yielding poor performance (also costly). By automating what previously had been a manual process, it streamlines the system deployment (and redeployment) process, cutting deployment times from days to minutes, and also reduces labor costs. And by automating a complex set of notoriously error-prone manual tasks, it increases application availability. More efficient, more dynamic, and more reliable—what’s not to like?!

But wait, Dorothy, there are problems in the Land of Oz. True automated provisioning assumes a closed-looped system—one in which the entire process is untouched by human hands. That’s rather unlike the environment of today, and it will take a journey to get there. And unlike more mature closed-looped control systems in other areas (such as industrial process control), this one is still a youth, and has plenty of growing to do.

How Does it Work?

Today’s automated products certainly eliminate much of the grunt work of provisioning systems. They can automate the initial deployment of the operating system and applications. They can also perform automated upgrades and manage patches. They can configure the network and storage, too. Some of these products work in heterogeneous OS environments. Some even have an easy-to-use GUI browser-based interface that makes them easier to learn and use. When the IT infrastructure is large, the cost advantage of automating these tasks is obvious, and, as it grows, automation becomes essential.

Although there are considerable differences among the various products in the market today, they share some common concepts and components:

- Inventory, Configuration, and Network Topology Management
- Image and Deployment Script Management
- Change Management
- Automation Policies

Let’s consider each of these.

Inventory, Configuration, and Network Topology Management

The Inventory, Configuration, and Network Topology management functions keep track of where everything is located, whether it is in use, what is on it, and what it is currently doing. This database of information is the foundation required by essentially every automated provisioning product to automatically deploy and redeploy the IT resources.

At the physical level, many provisioning products can track what slot in what rack in what room in what building a server is physically located. At a logical level, many of the products have higher-level concepts such as “Server Inventory Pools.” Any particular server could be located in one of a

---

2. In practice, a little of each is required. Scripting the entire install process is not always perfectly do-able, and can be very labor-intensive. Copying a known working configuration can save time, but at least some information such as network address and host name must be configured for each and every system; cloning software often tightly integrates the ability to do some base-level customization.
number of different inventory pools, which might include production, spare, and maintenance pools. Other important information includes what software is installed and its version. Provisioning can also detect how a particular server is plugged into the network. Additional data include which storage resources—especially in Storage Area Networks—are being used.

Many of today’s products discover these resources at least somewhat automatically, though considerable manual operations are still required by many, especially to describe network topologies, appropriate asset allocations, and other data that have not been routinely recorded by earlier management tools and approaches. Auto-discovery functions are being improved over time.

**Image and Deployment Script Management**—There are two ways of deploying software on a server. One way is to use imaging technology to snapshot the entire contents of all software on a source server (a so-called “golden image”) so that it can be quickly deployed to a target server, which thereby becomes an exact clone of the source server. The other way to deploy software uses a script to automate the installation process. This is the automated equivalent of someone’s sitting at the server’s console and responding to the answers posed by the installation procedure.

A complete automated provisioning product needs to do both imaging and scripting, and it should be customizable to perform a combination of both.

That’s because there are pro’s and cons with both technologies; neither provides a complete solution. Server deployment using imaging technology is very fast. However, some applications cannot be imaged. Microsoft Exchange is an example; it cannot be imaged because it is so heavily tied into the Active Directory environment in which it is installed. Scripting, on the other hand, can handle a wider variety of hardware and software cases, but is far slower—making it less suitable for the largest server farms or most dynamic redeployments.

**Imaging and Scripting: Pros and Cons**

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Imaging</td>
<td></td>
</tr>
<tr>
<td>• Very fast deployment</td>
<td>• Cannot be used with all software</td>
</tr>
<tr>
<td>• Can easily capture subtle customizations to GUI, directory layout, user profiles, etc.</td>
<td>• Target and source server have to use identical hardware</td>
</tr>
<tr>
<td>• Can perform multicast clone operations (deployment of many servers simultaneously)</td>
<td>• Post-imaging operations such as computer name and IP address setup needed</td>
</tr>
<tr>
<td>Scripting</td>
<td></td>
</tr>
<tr>
<td>• Can install any software</td>
<td>• Slow deployment</td>
</tr>
<tr>
<td>• Less sensitive to hardware differences</td>
<td>• Customizations are not easily implemented</td>
</tr>
</tbody>
</table>

No matter which technique is used, an automated provisioning product has to be able to manage the scripts and the images it creates and deploys. An archive and an inventory of what images or scripts have been deployed and where are essential. It also needs to be able to handle bare-metal OS deployment, including the setup and initialization of any disk storage (for example, RAID sets) prior to the deployment.

**Change Management**—Software gathers no moss. It rolls endlessly, always changing. An automated provisioning product needs to keep track of new updates and versions, and to apply them automatically when necessary, convenient, and appropriate. For example, new viruses and worms are created every day and the provisioning product needs to be able to automatically update a server with security patches, based on established policies and rules. It must also have the capability to roll back a change if the change yields unexpected or unwanted results.

**Automation Policies**—Policies are at the heart of all this automation. Policies contain rules that trigger events, which automatically deploy or redeploy resources. An automated provisioning product has to determine which server to use and then deploy the appropriate software to this server. This auto-
information reduces human costs. All of this presupposes, of course, that a server is available to be deployed—most likely from a pool of spare servers, but possibly from a lower priority task—to solve the immediate performance problem. Automation may help server resources to be used more efficiently, but it certainly doesn’t eliminate the need for capacity planning.

As an example, a policy can be defined that monitors the CPU utilization of a particular server and then triggers an event if that utilization exceeds a predefined limit. The triggered event would then automatically deploy and configure an additional server to balance the workload. In order for the automated provisioning product to do this efficiently, it has to know:

• What is available (Inventory and Configuration)
• Where it should be placed in the network (Network Topology)
• Image or script source (Image and Deployment Script Management)
• Patches to be applied after the deployment (Change Management)

Policies thus depend on all of the fundamental components described earlier. In addition, although this example deals with server provisioning, the storage and network also have to be provisioned automatically to solve the overall problem.

The automated policies not only have to provision IT components, they also have to make better use of IT assets that aren’t currently deployed efficiently. Redeployment reclaims these underutilized IT resources and then reallocates them.

Problems in Oz

While it is a substantial step forward, automated provisioning is no panacea. Some of the problems include:

The Requirement of Scale- One needs to have a fair number of very similar (even essentially identical) systems to make automating their installation practical and useful, lest the initial setup cost be greater than the benefits of the automated solution. So while automation can be essential for service providers, enterprises, and large datacenters, it doesn’t always scale down well to small sites and small businesses.

The Requirement of Regular Resources- While it’s theoretically possible to do automated provisioning on a hodgepodge of existing gear, unless there are clear patterns or similarities among the gear, the IT department is going to have to invest considerable time to inventory and systematically describe the equipment available to be provisioned. Moreover, software cloning usually requires that the systems being cloned be identical hardware, or at least be very, very similar. Differences as minor as those needing slightly different BIOS or driver versions may require a whole, separate software image. But datacenters filled with identical hardware are rare. IT managers typically have to deal with rooms of dissimilar systems, which can make image management quite the challenge. So automated provisioning suggests either significant new “Greenfield” equipment purchases, or waiting for a natural equipment upgrade cycle to start a deployment.³

³ This is one reason HP’s Utility Data Center, for example, is most often sold with neat, regular, pre-configured racks of new servers. While UDC can work with existing gear in theory, it’s much easier to set up with a clean slate environment. While competitors such as IBM and Sun have criticized UDC on this front, we believe they are themselves now discovering the same practical tradeoff.
**Image Skew**—Provisioning by cloning works well. At the moment the image process completes, the target and source servers are almost exactly alike. However, random “out-of-band” changes—little updates here, configuration changes there—often soon take place. While useful or necessary individually, the group of initially identical systems can quickly diverge to be just “sort of identical”—which is to say, not identical at all. That increases the management complexity. Frequent upgrades and patches make tracking, recording, and executing server images a full-time job. Images do not remain static for long; every new mandatory patch for a virus or a bug requires that existing images be modified or new images be created. Although images make server deployment fast, they carry a high ongoing maintenance cost.

**Early Days and Limited Scope**—This technology is also a toddler yet, with many current toolsets limited in scope. They often handle initial server and software provisioning (including the setup of operating systems, middleware, and such) very well. But that’s only part of the job; it’s server provisioning, rather than the service provisioning that more fully matches the telco history of the term, as well as customer desires. Provisioning mechanisms tend to focus on initial installations, and don’t necessarily tie systems into all of the monitoring and management systems one might want. This is just one of the issues that relates to the relative immaturity of automation process in IT.

**Seeking Equilibrium and Global Optima**—Few systems or techniques exist for understanding the complexities and interactions between applications and other software components. As with any closed-loop control system, it’s important to guard against local optimizations that have untoward macro effects, such as oscillating patterns of making and then unmaking changes based on short-lived perturbations in IT system usage. In geek-speak, they have to build in hysteresis and work toward equilibrium.

**A Required Cultural Shift**—Most IT managers are not at the point where they feel comfortable automating many of their critical processes and removing human interaction. This is understandable, but nevertheless retards deployments. Trust will have to come gradually as this solution matures and is deployed and tested in real-life situations—and broadly prove that it works as advertised, even in extreme and “corner” cases.

**Integration Akimbo**—Today, there are many separate tools to detect a problem, deploy or script a server, keep track of inventory, etc. But mechanisms for integrating one toolset with another—especially across vendor and product line boundaries—are immature and often non-existent. These tools were developed separately. With Web and Grid services, there is hope that the situation will improve over time, but efforts such as DCML and OGSA notwithstanding, many vendors are not interested in working closely with their perceived competitors, so that improvement is coming slowly at best.

**Conclusions**

*Automated provisioning* can help sophisticated IT departments respond more efficiently and rapidly to problems arising from unpredictable and fluctuating computer resource usage. It can save money by eliminating human interaction, slashing provisioning time, and simultaneously decreasing the potential errors inherent in manual processes. IT resources get allocated efficiently, automatically, and correctly by established policies. The IT department establishes these policies in a move to proactive, not reactive, management. Save money and time. Respond more quickly. Reduce errors. Automated provisioning sure is great stuff, right?

It is—when implemented properly, with appropriate support and a long view. Enterprises have
begun using a variety of automated provisioning products and techniques today as a better alternative to past manual procedures, and they will use much more of it in the coming years. But, as with all automation, this solution is only as good as the policies and procedures created. There is a significant up-front investment required, and there is a learning curve. In addition to the cost of the tools and establishing the right policies and procedures, many shops will have to overcome ingrained behaviors that have historically favored manual process and \textit{ad hoc} tweaking.

Sophisticated policy management, global optimization, integration with complementary management tools, and cross-product interoperability increasingly have become the next challenge and competitive battleground, rather than base-level server provisioning. Automated provisioning is still very much a developing technology. Nonetheless, it is a powerful approach—one that will increasing define how servers and applications are deployed.