Dynamic Deployment Methodologies for Oracle RAC Databases

Dynamic provisioning allocates resources at the time they are needed. To improve provisioning efficiency, enterprise IT departments can organize resources into components and create an automated process for bare-metal deployments. This approach can prove especially effective for Oracle® Real Application Clusters (RAC) deployments because hardware resources can be dynamically allocated and reprovisioned based on various business priorities.

CREATING A DYNAMICALLY PROVISIONED CLUSTER INVOLVES BOOTING THE PRE-OS ENVIRONMENT, INSTALLING THE OS, INSTALLING THE NODE COMMUNICATION APPLICATION LAYER, AND FINALLY, PROVISIONING FOR UPDATES. Before building a dynamic provisioning infrastructure, however, enterprises should ascertain the need for dynamic deployments and examine their potential benefits.

Dynamic deployments can help reduce costs resulting from a large number of idle servers. This may occur in IT environments in which applications exhibit large spikes and variations in their loads. For instance, an application that allows students to check their grades might experience an extremely heavy load only at the end of a semester. Another example scenario is at the end of a business quarter, when an enterprise’s data center could see extremely high spikes in input and data manipulation. Such systems could be configured to allow reprovisioning to other applications during times of low usage.

Dynamic deployment can also be advisable when an enterprise needs a new cluster of servers and the administrator foresees that the enterprise will require additional servers with similar software configurations in the future. Designing systems that can be dynamically provisioned during the initial deployment may take additional time, but it facilitates adding or scaling out a cluster in the future.

In addition, an enterprise’s proactive approach to hardware management can justify dynamic provisioning. Hardware has a predictable life cycle. Administrators usually desire to minimize mean time between failure and consequent down time by periodically replacing older systems with new hardware.

This article explores steps to take when planning and implementing dynamic deployments. It also examines specific steps for dynamically deploying an Oracle Real Application Clusters (RAC) environment.

Planning component phases for deployment

Administrators should divide deployments into components: pre-OS installation, OS installation, and application-layer installation. Organizing dynamic deployment in component phases is essential. Although an imaging solution can be used to configure clusters once they have been deployed, this approach could lead to more work and less flexibility in the future than using a component-based deployment.
Dividing a dynamic deployment into components can provide several benefits. A component-based approach lets IT administrators mix and match software and OS components and specify which applications and services are deployed on each system. Conversely, a monolithic-image approach could leave an administrator with only one software stack and one hardware set on which to deploy it. For example, if an underlying hardware driver changes, an administrator must update the OS with the latest driver. If the administrator had imaged the systems monolithically, an entire new image snapshot of the system would need to be created, introducing additional risk to deployments. Conversely, creating an independent OS module means that only that particular module would need to be modified, leaving the post-OS, application installation, and patch-set modules intact.

Automating deployment with a component-based approach goes far beyond simply choosing installation packages. When designed correctly, a component-based methodology lets IT administrators use industry-standard components. This allows deployments to work properly on a myriad of hardware platforms—such as Linux* and Microsoft® Windows® operating systems—and permits flexibility in choosing application-layer components. The level of granularity varies based on the deployment. For example, with Linux and Windows, an administrator may choose an OS layer that deploys only the base OS installation. Creating another layer could permit inclusion of advanced components such as a professional or developer’s edition of the OS deployment. The best practices presented in this article address the rudimentary components of a dynamic deployment.

Defining the building blocks
A deployment that is broken down into several logical building blocks, as described in the following sections, allows administrators to modify those blocks without changing others.

Pre-OS environment
The pre-OS environment is the condition of the hard drive before any software resides on it—the bare-metal state. A pre-OS environment allows a system to boot up in the absence of an OS. This bootable image lets administrators install an OS or execute setup operations before the OS is deployed.

Various preinstallation operating systems are available, such as Windows Preinstallation Environment (WinPE) and Embedded Linux. The pre-OS environment is ideal for identifying the hardware configuration, choosing the right drivers to read and write to the system, and setting up the network so that each node can access a remote server.

Typically, the pre-OS environment requires the following technologies:

- **Wake-on-LAN:** If a BIOS is enabled for Wake-on-LAN and the system is turned off, it will turn itself on if a wake-up data packet is sent to the node. The packet must contain six bytes of a synchronization stream (FFh) followed by 16 copies of the node’s Ethernet address. This feature is particularly useful when no one is present at the data center and a need arises to reprovision or install an OS. Dell™ PowerEdge™ servers have built-in Wake-on-LAN functionality, which can easily be enabled to manage the servers remotely.

- **PXE:** Preboot Execution Environment (PXE) is used for dynamically booting or provisioning a server. PXE allows a system to boot over the network. Using PXE requires a BIOS and network interface card (NIC) that support the PXE framework. The network cards and BIOS in Dell PowerEdge servers have integrated PXE capabilities.

- **DHCP:** The task of a Dynamic Host Configuration Protocol (DHCP) server is to assign IP addresses to all nodes on a network so that each computer may communicate. Before a computer begins talking on the network, it must obtain an address that uniquely identifies it. It is analogous to a telephone, which cannot make or receive calls until it has received a number. Enabling the DHCP server for each node is the first step in dynamically adding a node to the infrastructure.

- **TFTP:** Trivial FTP (TFTP) is generally used for booting thin clients from a network host or transferring very small files. It uses User Datagram Protocol (UDP) on port 69 as its transport protocol and—unlike its more robust cousin, FTP—lacks features such as encryption, authentication, or the ability to list directory contents. However, like FTP it has the ability to read and write files from a remote server and can transfer both binary files and ASCII files. Each of the nodes uses TFTP to obtain a pre-OS environment as the first step toward deployment.

OS installation
Deploying an OS through scripted installation allows the installation to not be bound to a particular image, enabling flexibility for OS provisioning. For example, with a scripted OS installation, the same installation scripts can be used on multiple types of machines.

Two predominant methods are used to script an installation: Red Hat® kickstart installation using the ks.cfg file and Windows unattended installation using the unattend.txt file. Both allow scripted customization of an OS installation.
Figure 1. Typical Oracle RAC implementation with shared Oracle home

**Application-layer installation**

The application-layer phase establishes a means of connecting individual servers to a cluster. During OS installation, each server operates individually without knowledge of its peers. This phase involves reestablishing network connectivity and establishing shared storage among the server nodes. Many technologies exist that allow an enterprise’s cluster to share storage.

**Deploying Oracle RAC using the shared-home method and OCFS2**

After the OS is deployed, Oracle RAC can be deployed using the shared home method. This method installs all Oracle binaries in a common location and configures the RAC nodes to contain only node-specific information to supplement the common Oracle home. Each node can boot using the local disk that contains the OS, and at the same time, all Oracle binaries are located on a common storage area network (SAN) disk.

Oracle Cluster File System 2 (OCFS2) provides shared-home support for Oracle RAC environments. This file system enables Oracle RAC database files to be stored on shared storage accessible by all cluster nodes. Using OCFS2, all cluster nodes can access files used for database storage as well as the binary files for running the database. Consequently, application-layer installation is simplified because the Oracle database needs to be installed only once.

Because OCFS2 for a given node is specific to that RAC node, the OCFS2 parameter files must reside on the local disk. However, after OCFS2 is installed and the common attached storage disks are formatted for OCFS2, those partitions can be accessed by all nodes of the RAC cluster. Figure 1 shows a typical OCFS2-based shared-home implementation. The RAC nodes contain the OS and node-specific parameters such as host name, IP addresses, and so forth. They also contain the OCFS2 parameter files that are tied to the node. The common Oracle binaries are located in the common storage disk partition in the SAN.

When a need arises to expand RAC functionality, administrators simply deploy the OS on an additional node and set up the OCFS2 modules on that node. The node can then be added to the common storage subsystem where it can access the shared-home Oracle binaries.

The shared-home approach enables administrators to easily manage the Oracle RAC environment. Administrators can spend minimal time managing each RAC node individually, thereby enhancing administrator productivity. In addition, with storage backup functionality available on all back-end storage, an image of the shared home can be easily created, which can help provide effective disaster recovery.

**Oracle Universal Installer**

After the OS has been installed on systems that are part of the Oracle RAC environment, the Oracle binaries must be deployed to all the nodes of the cluster. This can be done using the Oracle Universal Installer (OUI). The OUI helps administrators install RAC and perform Oracle prerequisite checks. It also enables automated installation of Oracle RAC using response files.

**Provisioning servers dynamically**

The building blocks for dynamic provisioning include booting the pre-OS environment, installing the OS, and setting up node communication at the application layer. Dividing the deployment process into these components can give administrators flexibility in planning and managing an enterprise’s growing IT infrastructure.

Ujjwal Rajbhandari is a systems engineer in the Database and Application Engineering Department of the Dell Product Group. He is responsible for validating Oracle RAC solutions on Dell PowerEdge servers and Dell/EMC Fibre Channel storage. Ujjwal has a B.E. in Electrical Engineering from the Indian Institute of Technology, Roorkee, and an M.S. in Electrical Engineering from Texas A&M University.

David Mar is a senior software engineer in the Dell Database and Application Engineering Department of the Dell Product Group. His principal focus is developing deployment strategies for Oracle-based solutions on Dell PowerEdge servers, Dell PowerVault™ storage, and Dell/EMC storage. David has a B.S. in Computer Engineering and an M.S. in Computer Science from Texas A&M University.

FOR MORE INFORMATION

Oracle Real Application Clusters: