Noções Tunning

O Que É Tuning?

No artigo de hoje vou comentar sobre o que é Tuning e como fazer Tuning em um Banco de Dados Oracle, passando alguns conceitos básicos e uma visão geral sobre o assunto. Tuning é um termo que desperta um interesse cada vez maior nos profissionais de TI, devido aos fatos que estão descritos abaixo:

- Aumento do legado de sistemas corporativos (ERPs, GEDs etc.) e sistemas web;

- Aumento da quantidade de usuários de BDs;

- **Aumento da quantidade de dados**.

Antes de começar, a 1ª coisa que o profissional de TI que trabalha com **Tuning** tem que aprender, é como escrever corretamente esta palavra. Muita gente escreve Tuning de forma errada, eu mesmo quando comecei a estudar o assunto, cometia este erro. A palavra **tuning** contém somente 2 Ns, portanto, escrevê-la com 3 Ns (Tunning) está errado! Tuning, na língua inglesa, é um substantivo derivado do verbo tune, e o erro na sua escrita, ocorre por causa de uma confusão do brasileiro ao aplicar o gerúndio (Present Participle) no verbo tune. Para entender melhor como fazemos essa confusão, sugiro a leitura do artigo Why do some words have double consonants while others have only one?.

Em TI, Tuning refere-se basicamente ao conceito de propor e aplicar mudanças visando **otimizar** o desempenho na recuperação ou atualização de dados. Em curtas palavras, **Tuning** (em TI) **é sinônimo de otimização**. Atualmente existem muitas técnicas e dicas de tuning que podem ser aplicadas para otimizar os sistemas corporativos, compreendo-os desde o nível do sistema operacional, até o nível do seu código-fonte. Para fazer um bom trabalho de Tuning, é necessário executar criteriosamente os seguintes processos:

1- Entender o problema;

2- Elaborar o diagnóstico;

3- Aplicar dicas e técnicas de otimização **que se relacionam ao diagnóstico elaborado**.

O objetivo principal do trabalho de tuning é **minimizar o tempo de resposta** e recuperação dos dados das aplicações. Em um Banco de Dados, os 3 tipos de atividades de tuning que podem ser realizadas, são:

1- Planejamento de performance:

Definição e configuração do ambiente em que o BD será instalado, considerando-se os seguintes itens: Hardware, Software, Sistema Operacional e Infraestrutura de rede.

2- Tuning de instância e BD:

Ajuste de parâmetros e configurações do BD (atividades que fazem parte do trabalho de um DBA).

3- SQL Tuning:
Otimização de instruções SQL.

Para desenvolver bem o 1º tipo de atividade, não há um treinamento específico, mas você verá alguns destes itens em um bom treinamento de Database Performance Tuning. Para se aprofundar é necessário estudar e pesquisar bastante sobre o assunto. Para adquirir os conhecimentos necessários p/ realizar o 2º tipo de atividade, recomendo o treinamento Database Performance Tuning. Para adquirir os conhecimentos do 3º tipo de atividade, recomendo o treinamento SQL Tuning em Bancos de Dados Oracle. Um ponto muito importante a ser ressaltado é que a maior parte dos problemas de performance estão em instruções SQL ruins (ver item Application na Imagem 1), portanto, dentro do 3º item.

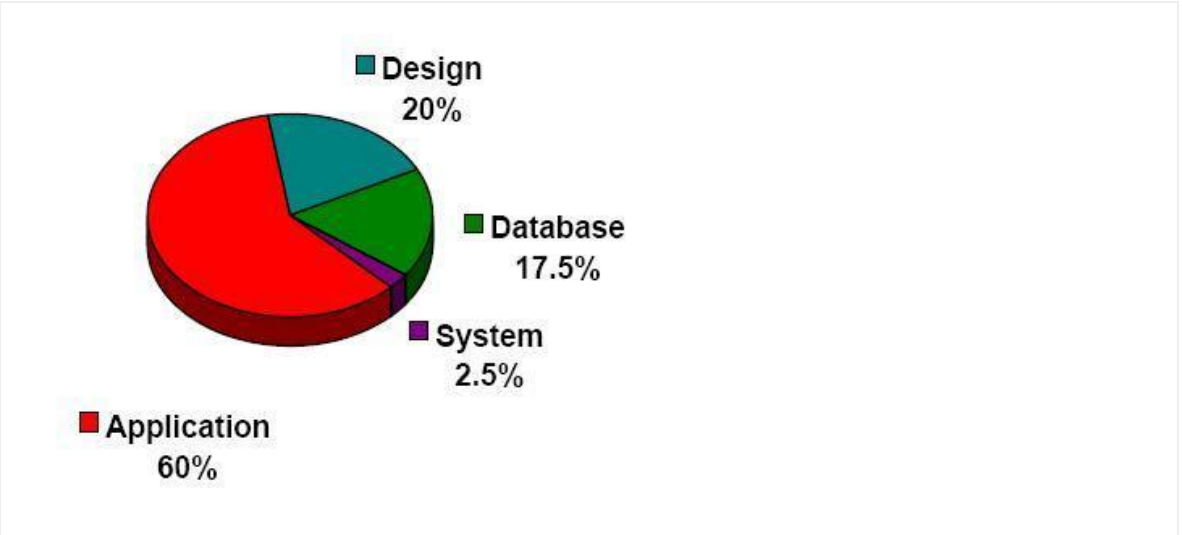

Imagem 1 - Performance Gains from Tuning
Fonte: Tuning When You Can't Touch The Code, Michael R. Ault

O ponto mais importante que eu costumo comentar para quem está começando nessa área, é que existem muitas dicas e técnicas de Tuning, e que não existe uma "receita de tuning" que podemos aplicar em qualquer BD e resolver os seus problemas de lentidão. Para fazer um bom trabalho de Tuning temos que saber usar as ferramentas disponíveis (Ex.: AWR, Statspack, SQL Traces, Visões de performance dinâmicas, Plano de execução etc.); elaborar um bom diagnóstico; e conhecer os recursos do BD, dicas e técnicas que podemos usar, de acordo com o problema encontrado e características da(s) aplicação(ões), como por exemplo:

- Quantidade de usuários concorrentes;

- Se ela faz muitas escritas ou somente leituras;

- Quantidade de dados que ela irá processar.

JBoss: Instalação, Arquitetura , Configuração, Tuning e Administração

Conheça a fundo o servidor JBoss do ponto de vista do administrador, vendo dicas de configuração e tuning, e aprenda o essencial para manter um ambiente de produção seguro, estável e perfomártico para aplicações Java EE.

Esse artigo faz parte da revista Java Magazine edição 46. Clique aqui para ler todos os artigos desta edição

JBoss de Ponta a Ponta

Instalação, arquitetura, configuração , tuning e administração

Conheça a fundo o servidor JBoss do ponto de vista do administrador, vendo dicas de configuração e tuning, e aprenda o essencial para manter um ambiente de produção seguro, estável e perfomártico para aplicações Java EE.

O JBoss Application Server (JBoss AS) é certamente o mais conhecido servidor de aplicações Java EE livre, competindo de igual para igual no mercado com produtos consagrados como a Websphere da IBM e o Weblogic da BEA. A liderança tecnológica do JBoss AS é confirmada pela participação ativa dos seus desenvolvedores nas definições da versão 5 da plataforma Java EE, em especial na especificação EJB 3. Com a compra do JBoss Group pela Red Hat e a parceria com a Exadel, a JBoss AS passou a ser a base de uma família de produtos que cobre todas as demandas de desenvolvimento e infra-estrutura de produção para aplicações Java EE. Isso sem abrir mão do modelo de negócios "Open Source Profissional" - isto é, os produtos com a marca JBoss continuam sendo fornecidos integralmente sob licenças livres, e a receita provêm integralmente da prestação de serviços sobre os produtos.

Todo o poder, flexibilidade e confiabilidade do JBoss AS vêm em um “pacote” muito simples de instalar. Rodar os primeiros servlets, EJBs e consumidores de mensagens JMS é igualmente simples. Mas, embora haja facilidade em se passar pelos passos iniciais com a JBoss AS, não podemos subestimar o esforço e conhecimentos necessários para se manter um ambiente de produção para aplicações Java EE.

Nesse contexto, iniciamos pela instalação do JBoss AS e apresentamos sua estrutura de diretórios, identificando as principais pastas de dados e de configuração do servidor de aplicações. Em seguida, são apresentadas dicas para a resolução de problemas na inicialização do JBoss AS, e como interagir com as aplicações de administração do servidor. Daí passamos para um exame detalhado da arquitetura interna do JBoss AS. Essa exploração fornece subsídios indispensáveis para conhecer algumas situações de configuração típicas em ambiente de produção, incluindo “fechar” o acesso administrativo (para obter maior segurança). Por fim, são apresentados os princípios básicos para o dimensionamento e monitoração do JBoss AS. Cada um destes tópicos é esmiuçado em uma seção própria.

A maiar parte dos conhecimentos apresentados neste artigo se aplicará igualmente as duas versões do servidor recomendadas hoje para produção, 4.0.x e 4.2.x, e também ao 5.0.x, que ainda está em beta.

Espera-se do leitor conhecimento básico para intermediário do Java EE, incluindo noções de JNDI, Servlets e EJBs. O conhecimento introdutório para desenvolvimento de aplicações com o JBoss[1] já foi apresentado antes na Java Magazine em vários artigos (veja o **quadro** “Para saber mais” no final), de modo que este artigo se foca nas tarefas do administrador de um ambiente de produção e no conhecimento aprofundado do funcionamento do JBoss. O administrador do JBoss AS precisa conhecer, além do básico do Java EE, o uso do JDK pela linha de comando, a edição de scripts nativos da sua plataforma (bat do Windows ou sh do Linux, Unix e Mac) e noções básicas de redes, por exemplo o uso e o significado de portas TCP.

Instalação, Início E Encerramento Do JBoss

Instalando O JBoss

Qualquer que seja o download escolhido, o resultado final da instalação é o mesmo código 100% Java, independente de plataforma. O instalador JAR pode até ser executado diretamente pela internet, utilizando Java Web Start, e fornece um assistente que ajuda na configuração de alguns perfis especializados, como rodar apenas o container web ou ativar o suporte a EJB

3[2]. A opção “advanced” do jem-installer permite ainda configurar parâmetros básicos de segurança para as ferramentas administrativas do JBoss, e fontes de dados para outros bancos da dados que não o HSQLDB (embutido no JBoss).

Apesar das facilidades fornecidas pelo jem-installer, a maior parte da configuração e tuning do servidor de aplicações ainda terá que ser feita depois da instalação, editando-se os vários arquivos de configuração. Além disso, administradores mais experientes irão apreciar a facilidade de se copiar uma instalação do JBoss de um computador para outro; portanto geralmente preferem baixar o ZIP. Os interessados em ver o jem-installer em ação podem consultar o quadro “O instalador gráfico do JBoss”.

Administradores mais experientes de sistemas Linux irão preferir instalar o JBoss AS utilizando os pacotes RPM fornecidos pelo projeto Jpackage (jpackage.org). Estes pacotes tem a vantagem de seguir as convenções do LSB (Linux Standards Base) para a distribuição de arquivos executáveis, de configuração, bibliotecas e logs, além de aderir a algumas práticas de seguranca essenciais da plataforma, como criar contas de usuários individuais para serviços de rede.

A Estrutura De Diretórios Do JBoss

No final das contas, instalar o JBoss consiste apenas em se descompactar o download em uma pasta qualquer. O layout de diretórios gerado será exatamente o mesmo, qualquer que seja o formato do instalador (veja o layout na **Figura 1**). Vamos agora descrever cada nível dessa estrutura de diretórios. Recomendamos que o leitor acompanhe esta parte verificando em sua instalação do JBoss os arquivos presentes em cada pasta, a medida que as pastas vão sendo apresentadas.

O primeiro nível de subdiretórios fornece arquivos utilizados pelo servidor, independentemente

da configuração ativa:

• bin - Scripts de inicialização e de término do JBoss, além de alguns utilitários de linha de comando para administração do servidor ou apoio ao desenvolvimento. Também são fornecidos scripts para inicialização do servidor como parte do boot de sistemas Unix.

• client - Pacotes JAR contendo APIs do Java EE e bibliotecas necessárias a um cliente remoto que se comunique com o JBoss utilizando serviços nativos do Java EE, por exemplo JAAS, JNDI, EJB ou JMS.

• docs - Modelos de arquivos de configuração do JBoss, como DataSources JDBC e provedores de persistência para o JMS, além de Schemas XML e DTDs, para os descritores de deployment do Java EE e para os arquivos de configuração do próprio JBoss. Apesar do nome, esta pasta não contém documentação de configuração e administração do servidor. Este material

pode ser baixado em formato PDF no site jboss.org/products/jbossas/docs.

• lib - Bibliotecas Java utilizadas pelos scripts de inicialização do servidor ou pelos seus utilitários de linha de comando.

• server - Pacotes JAR e arquivos de configuração do servidor em si, além das aplicações e componentes hospedados pelo JBoss.

O administrador do servidor de aplicações passará a maior parte do seu tempo lidando com os subdiretórios da pasta server. Cada um desses subdiretórios é uma configuração completa do JBoss, incluindo todo o código Java e arquivos de configuração, além de dados destas configurações. Os downloads compactados pré-instalam três configurações-modelo, relacionadas a seguir:

• minimum - Fornece o estritamente mínimo para se iniciar o JBoss. Esta configuração não permite executar nenhuma aplicação, pois não traz serviços como container web ou pool de conexões JDBC. No entanto, esses recursos podem ser copiados de outras configurações e instalados aqui. Este diretório é o ponto de partida para se criar uma configuração especializada do JBoss (por exemplo, apenas o servidor JMS), ou para a construção de servidores de aplicações não-Java EE baseados no JBoss.

• default - Fornece todos os serviços previstos pela especificação Java EE, incluindo container web, container EJB, JTA, JMS, JavaMail, Web Services, Timers e monitoração de performance. Também traz alguns serviços extra-Java EE como Scheduler e AOP. Inclui ainda ferramentas administrativas com interface web.

• all- Inclui todos os serviços disponibilizados com o JBoss. Além dos componentes já relacionados na configuração default, merecem destaque o suporte a clustering (baseado no JGroups) e o agente SNMP.

O jem-installer fornece alguns modelos adicionais de configuração, mas apenas o modelo selecionado pelo usuário será descompactado para a pasta server (que ficará assim com um único subdiretório).

Para definir qual configuração será utilizada, deve-se passar a opção **-c** para o script de inicialização do JBoss, por exemplo **./run.sh -c minimal.** Os arquivos das demais configurações serão simplesmente ignorados. Mas, qualquer que seja a configuração escolhida, ela irá apresentar a estrutura de subdiretórios descrita a seguir (alguns destes diretórios só serão criados durante a primeira inicialização do JBoss):

• conf - Contém o arquivo de configuração do Microkernel JMX do JBoss (jboss-service.xml), arquivos de configuração para bibliotecas e APIs utilizadas pelo Microkernel, como Log4J, JNDI e JAAS, além de configurações padrão para os descritores de deployment proprietários do JBoss para EJBs. (Veja mais sobre o Microkernel na seção sobre a arquitetura do JBoss.)

• data - Arquivos de dados persistentes dos serviços, por exemplo bases de dados HSQLDB,

filas JMS e atributos de sessões HTTP.

• deploy - É nesta pasta que estão os serviços individuais do JBoss, além dos arquivos

de configuração desses serviços. Na instalação padrão, esta mesma pasta será usada pelo desenvolvedor para fazer o deployment dos seus aplicativos e componentes. Pacotes instalados aqui podem ser adicionados, atualizados e removidos sem necessidade de reiniciar o JBoss (recurso de hot deploy).

• lib - Contém bibliotecas e APIs utilizadas pelo próprio JBoss e por serviços individuais. Estas bibliotecas são visíveis também pelas aplicações e componentes do usuário, então aqui seria o lugar para colocar bibliotecas compartilhadas por várias aplicações, como as do Struts. Ao

contrário dos pacotes na pasta deploy, a atualização ou adição de pacotes na pasta lib exige o reinício do servidor.

• log - Destino padrão para os arquivos de log do servidor, que são gerados pelo Log4J.

• temp - Arquivos temporários diversos.

• work - Arquivos gerados internamente pelos serviços do JBoss, por exemplo páginas JSP compiladas pelo container web. Alguns destes arquivos são reaproveitados para acelerar o reinício, o que pode causar problemas após um término forçado do servidor. Neste caso, é seguro remover todo o conteúdo desta pasta.

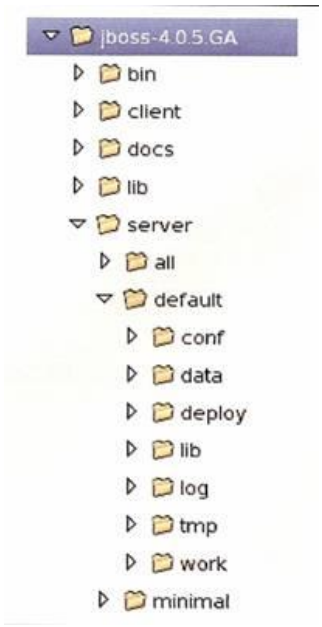

Figura 1. Pastas criadas pela instalação / descompactação do JBoss AS

É importante se familiarizar com a estrutura de diretórios do JBoss, para poder localizar rapidamente os arquivos a serem modificados, inclusos ou removidos de modo a satisfazer cada necessidade de configuração. No final deste artigo serão apresentados alguns casos mais comuns. Para detalhes adicionais, o administrador deve consultar a extensa documentação do JBoss, para conhecer arquivos e parâmetros de configuração mais especializados.

Iniciando o JBoss

Para iniciar o JBoss, basta entrar na pasta bin criada pela descompactação ou instalação (por exemplo, jboss-4.0.5GA/bin) e então executar o script **run** (run.sh ou run.bat). Para encerrar o servidor, e seguro usar Ctrl+C ou então rodar o script **shutdown** (shutdown,sh ou shutdown.bat). (Falaremos mais sobre o encerramento do JBoss adiante).

Não sendo especificada uma configuração, o JBoss tentará usar a pasta default. A **Figura 2** ilustra a inicialização do servidor em ambiente Linux. O processo será igual no Windows – exceto pela aparência da janela de prompt de comandos, claro.

A mensagem inicial, advertindo sobre a falta do pacote tools.jar no classpath, e sugerindo a definição da variável de ambiente JAVA_HOME, poderá ser ignorada na maioria dos casos, especialmente em sistemas Unix. No Windows, a instalação do JDK sempre instala também um JRE para uso pelo Internet Explorer, e pode acontecer de o JBoss detectar este JRE em vez do JDK. Para evitar este problema, configure a variável JAVA_HOME, apontando para a sua instalação do JDK, utilizando as propriedades do Meu Computador, na aba Avançado, ícone Variáveis de Ambiente. Defina então JAVA-HOME como uma variável de sistema (não de usuário).

Todas as mensagens exibidas no prompt de comandos são também replicadas no arquivo log/server.log, dentro da configuração ativa, e detalham cada etapa da inicialização do servidor. No final, a seguinte mensagem indica que o servidor está pronto para uso:

23:38:36,264 INFO [Server]JBoss (MX MicroKernel)

[4.0.5.GA (build:CVTag=Branch_4_0 date=200610162339)]

Started in 59s:727ms

É importante observar cuidadosamente as mensagens exibidas durante a inicialização do servidor. Se alguma coisa impedir a inicialização completa e correta de algum dos serviços, serão mostradas mensagens de erro na forma dos conhecidos stack traces de exceções Java, por exemplo[3]:

23:47:13,547 ERROR [Http11BaseProtocol] Error starting endpoint

java.net.BindException: Address already in use:8080

at org.apache.tomcat.util.net.PoolTcpEndopoint.initEndopoint(

PoolTcpEndopoint.java:297)

Havendo mensagens de erro relativas a portas TCP, finalize o servidor com Ctrl+C, ou fechando a janela do prompt de comandos, e tente determinar qual outro programa está utilizando a mesma porta que o JBoss. Usuários de Linux, Mac OS X e Unix poderão usar o comando **netstat-anp** para identificar o programa que está ocupando a porta. Usuários Windows terão que buscar ajuda em utilitários de terceiros, como a popular suíte SysInternals (microsoft.com/technet/sysinternals), recentemente adquirida pela Microsoft.

Embora seja possível configurar o JBoss para usar portas de rede alternativas, em um primeiro momento sera mais fácil apenas desativar o programa causador do conflito. Além disso, espera-se que um servidor de produção seja dedicado ao JBoss, e portanto que ele não terá instalados outros programas que possam usar as mesmas portas TCP.

É importante destacar que o JBoss não é um simples servidor, dedicado a fornecer um único serviço de rede, tal como um servidor web ou servidor de e-mail. Ele fornece um conjunto amplo de serviços de rede mais ou menos independentes, cada qual com seu próprio protocolo de aplicação, porta TCP e parâmetros de configuração e tuning. Pense no JBoss como vários servidores embutidos em um só produto. Então um conflito de portas ou outro problema de configuração pode passar despercebido até que seja necessário o serviço específico que foi afetado.

Figura 2. Inicialização do servidor JBoss