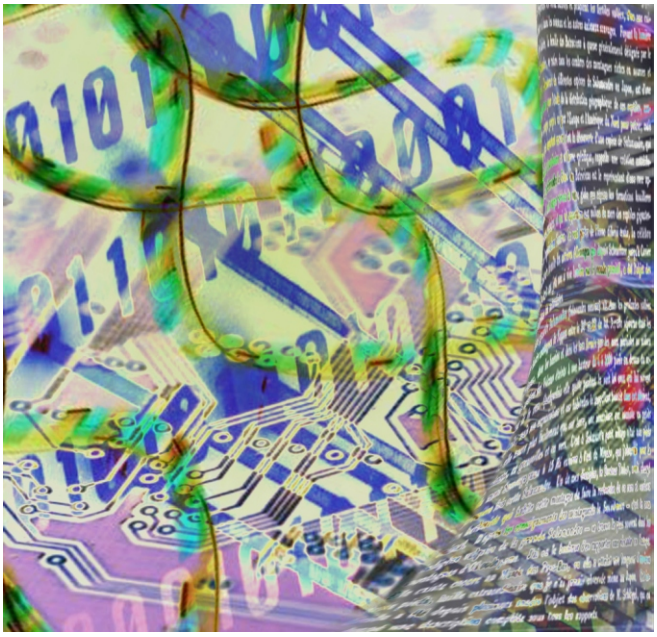
Comunicado Técnico 82

Agosto, 2007
Campinas, SP



ISSN 1677-8464

Utilizando o Rsync para atualizar os Bancos de Dados do Sting Millennium Suite

Fábio Danilo Vieira[1]
Edgard Henrique dos Santos[2]

O *rsync* é um utilitário capaz de transferir apenas as diferenças entre dois conjuntos de arquivos através de uma conexão de rede ou dentro de um mesmo computador. Ele analisa o interior desses arquivos e tenta transmitir apenas as diferenças entre versões. Uma das grandes vantagens do *rsync* em relação a outros utilitários de mesma função é que ele consegue preservar características dos arquivos originais, como permissões, proprietários e grupos, data e hora de modificação, além de arquivos de *links* e de dispositivos. O *rsync* pode ser instalado em diversos sistemas operacionais, sendo que no caso deste trabalho, foi utilizado em Linux e Irix.

O Núcleo de Bioinformática Estrutural - NBI, da Embrapa Informática Agropecuária tem como seu principal produto o Sting Millennium Suite - SMS. O SMS é acessível através da *web* e é usado para análise dos dados produzidos e/ou disponibilizados pelo NBI. O servidor principal, onde está instalado o SMS e sua base de dados de aproximadamente 100 GB, é uma Silicon Graphics Incorporated - SGI Origin com sistema operacional Irix, que fica na Embrapa Informática Agropecuária. No entanto, o SMS também está instalado em vários servidores espelhos pelo mundo, os quais se diferem em plataforma de *hardware* e *software*, ou seja, alguns deles possuem arquitetura com processadores MIPS - Microprocessor Whithout Interlocked Pipeline Stages (da SGI) e outros com arquitetura Intel, usada normalmente em microcomputadores pessoais. Além disso, uns possuem sistema operacional Irix e outros Linux, mostrando a boa portabilidade do SMS.

A base de dados do SMS é formada por diretórios e arquivos textos com estruturas padrões da comunidade de biologia molecular e genômica, e constantemente está crescendo em quantidade de arquivos e tamanho em *bytes*, pois toda semana são processadas e geradas novas informações, como contatos intramoleculares e intermoleculares, curvatura e hidrofobicidade. Essas informações são geradas a partir de de arquivos trazidos dos bancos de dados de domínio público, como o banco de dados de estrutura de proteínas PDB - Protein Data Bank (Berman et al., 2000), o banco de dados de estruturas secundárias derivadas por homologia HSSP - Homology-Derived Secundary Structure of Proteins (Sander & Schneider, 1991; Schneider et al.,1997) e o banco de dados de famílias e domínios de proteínas PROSITE (Hofmann et al., 1999).

[1] *Bacharel em Tecnologia de Processamento de Dados, Analista da Embrapa Informática Agropecuária, Caixa Postal 6041, Barão Geraldo - 13083-970 - Campinas, SP. (e-mail: fabiodv@cnptia.embrapa.br)*
[2] *Bacharel em Ciência da Computação, Analista da Embrapa Informática Agropecuária, Caixa Postal 6041, Barão Geraldo - 13083-970 - Campinas, SP. (e-mail: edgard@cnptia.embrapa.br)*

O objetivo deste trabalho é mostrar como o *rsync* foi utilizado para fazer a atualização das bases de dados dos espelhos do SMS a partir do servidor principal do NBI.

Entendendo o Rsync

O *rsync* é um utilitário mais eficiente que seus concorrentes, *rdist*[3] e *rcp*[4], pelo fato de examinar o interior dos arquivos, individualmente, e procurar transmitir apenas as diferenças entre suas versões.

De acordo com Tridgell & Mackerras (2006), o *rsync* pode ser usado de oito maneiras diferentes:

- para copiar arquivos locais - é usada quando nem a fonte nem o caminho do destino contém um caracter dois pontos (:) como separador. (Exemplo: rsync -av /dir/origem /dir/destino);
- para copiar da máquina local a uma máquina remota usando um programa remoto de *shell* como o transporte (como o *ssh*[5] ou *rsh*[6]). É usada quando o caminho do destino contém apenas um caracter dois pontos (:) como separador. (Exemplo: rsync -avz -e ssh --delete /dir/origem maq_remota:/dir/destino);
- para copiar de uma máquina remota à máquina local usando um programa remoto de *shell*. Isto é invocado quando a fonte contém um caracter dois pontos (:) como separador. (Exemplo: rsync -avz -e ssh --delete maq_remota:/dir/origem /dir/destino);
- para copiar de um servidor remoto do rsync à máquina local. É usada quando o caminho da fonte contém dois caracteres dois pontos (::) como separador. (Exemplo: rsync maq_remota::nome_modulo/dir/origem /dir/destino);
- para copiar da máquina local a um servidor remoto do rsync. É usada quando o caminho do destino contém dois caracteres dois pontos (::) como separador. (Exemplo: rsync /dir/origem maq_remota::nome_modulo/dir/destino);
- para copiar de uma máquina remota, que tenha um servidor rsync, usando um protocolo remoto de *shell* como transporte. É usada quando o caminho da fonte contém dois caracteres dois pontos (::) como separador e a opção rsh=COMANDO. (Exemplo: rsync -av --rsh="ssh -l usuario_ssh" usuario_rsync@maq_remota::nome_modulo/dir/origem /dir/destino);
- para copiar da máquina local para uma máquina remota usando um protocolo remoto de *shell* como transporte, tendo um servidor rsync na máquina remota. É usada quando o caminho do destino contém dois caracteres dois pontos (::) como separador e a opção rsh=COMANDO. (Exemplo: rsync -av --rsh="ssh -l usuario_ssh" /dir/origem usuario_rsync@maq_remota::nome_modulo/dir/destino);
- para listar arquivos de uma máquina remota. Isto é feito da mesma maneira que transferências normais do rsync, exceto pelo fato que não se coloca um destino. (Exemplo: rsync -e ssh maq_remota:/dir).

É importante notar que em todos os casos (à exceção da listagem de arquivos), pelo menos um dos caminhos da fonte ou do destino deve ser local.

Como visto, o *rsync* pode ser utilizado de várias formas. Entretanto, a diferença básica entre elas é que umas utilizam um protocolo *shell* de conexão remota para transporte, outras utilizam o protocolo do próprio *rsync* (tendo um servidor *rsyncd* na origem ou no destino) e outras ambos os protocolos.

Usando o Rsync em modo Servidor

O *rsync* pode ser configurado como um processo servidor nas máquinas receptoras a partir de *inetd*. Esse servidor (na verdade apenas um modo diferente do *rsync* que deve ser instalado, tanto no mestre como nos clientes) é bem configurável, permitindo, por exemplo, que se restrinja acesso remoto a um determinado conjunto de diretórios e que o computador remoto prove sua identidade com uma senha (Nemeth et al., 2002).

Quando o *rsync* é utilizado em modo servidor, o comando *rsync* no computador que irá transferir arquivos exige um nome de módulo como o primeiro componente do destino, não um nome de caminho (Nemeth et al., 2002). Veja o exemplo a seguir:

```
% rsync -gopt /etc/arq.txt maq_destino::mod_teste
```

Neste exemplo, o comando *rsync* transfere o arquivo /etc/arq.txt para a máquina maq_destino no diretório especificado no módulo mod_teste. As opções -gopt preservam as permissões, posses e data/hora da modificação do arquivo. Os dois pontos duplos em maq_destino::nome_modulo fazem com que o *rsync* entre em contato com o servidor *rsync* na porta 873 de maq_destino.

O nome do módulo, assim como outras configurações para o servidor *rsync* em cada máquina cliente (ou seja, cada máquina que está recebendo arquivos da

[3] rdist - remote file distribution client program.
[4] rcp - remote file copy.
[5] ssh - secure shell.
[6] rsh - remote shell.

perspectiva de *rsync*, esses "clientes" são, na realidade, mais parecidos com servidores) seguem vários passos (Nemeth et al., 2002):

- adicione o número da porta de rsync a **/etc/service**.
- adicione o servidor (rsync --daemon) a **/etc/inetd.conf**.
- guarde autenticações de senhas em **/etc/rsyncd.secrets**.
- configure o servidor em **/etc/rsyncd.conf**.

No arquivo **services** basta adicionar a linha:

```
rsync                873/tcp
```

Já no arquivo **inetd.conf** se adiciona a seguinte linha:

```
rsync  stream  tcp  nowait  root      /usr/local/bin/rsync rsyncd --daemon
```

Em alguns sistemas operacionais Linux baseados em Red Hat, não existe o arquivo **inetd.conf**. Nesse caso, se configura o arquivo **/etc/xinetd.d/rsync** (Fig. 1).

```
service rsync
{
        disable = no
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/bin/rsync
        server_args     = --daemon
        log_on_failure  += USERID
}
```

Fig. 1. Arquivo rsync localizado no diretório /etc/xinetd.d de alguns Linux baseados em Red Hat.

Já o arquivo **rsyncd.secrets** deve conter uma única entrada (Nemeth et al., 2002):

```
root:senha
```

A senha utilizada em *rsync* deve ser diferente da senha real de *root*. O arquivo **rsyncd.secrets** deve ser legível apenas para *root*, isto porque a senha é armazenada em texto simples, sem criptografia alguma (Nemeth et al., 2002).

O último passo a ser feito é configurar um arquivo **/etc/rsyncd.conf** para informar ao servidor *rsync* (receptor dos arquivos) como se comportar. Uma configuração razoável é mostrada na Fig. 2 (Nemeth et al., 2002).

```
[mod_teste]
path = /etc
secrets file = /etc/rsyncd.secrets
read only = false
uid = root
gid = root
hosts allow = maq_envia1, maq_envia2
```

Fig. 2. Uma configuração básica do arquivo rsyncd.conf.

Outras opções podem ser configuradas, mas os padrões são suficientes. Essa configuração limitará as operações no diretório **/etc** e irá liberar acesso apenas para os hosts *maq_envia1* e *maq_envia2*.

Usando o Rsync com Protocolo Shell de Conexão Remota

Outra forma de utilização do *rsync* é através dos programas *rsh* ou *ssh*, os quais permitem uma maior independência da máquina transmissora em relação as máquinas receptoras, pois não é preciso se preocupar em configurar e executar o servidor *rsync* em cada uma das mesmas, como observado na explicação anterior.

No caso da base de dados do SMS do Laboratório de Bioinformática, a atualização dos diversos servidores espelhos instalados pelo mundo é feita através dessa forma de utilização do *rsync*, pois evita a necessidade de se ter conta de administrador nesses servidores, os quais possuem sistemas operacionais diferentes uns dos outros, evitando configurações extras, além de se utilizar transporte criptografado dos dados através da *internet*. Um exemplo de utilização do *rsync* com protocolo *shell* é dado a seguir:

```
% rsync -avze ssh /home/ maq_destino:/home
```

Esse comando irá sincronizar todo o conteúdo do diretório home do computador remoto maq_destino com o diretório home do computador local (onde foi executado o comando), usando como transporte o comando *ssh*, que é quem fará a validação de usuário e irá criptografar os dados transmitidos através da rede. A opção *-a* irá garantir que permissões e proprietários dos arquivos transferidos sejam preservados no computador destino. A opção *-v* fará com que a saída do comando *rsync* seja exibida na tela. A opção *-z* fará com que os dados sejam compactados antes de serem transferidos, sendo que no destino eles são automaticamente descompactados. Já a opção *-e* irá definir qual programa *shell* de conexão remota será usado, que nesse caso foi o *ssh*.

No caso da atualização da base de dados do SMS nos servidores espelhos onde está instalado, foi usado essa forma de *rsync*, pois, além de ser mais segura, evitou-se configurações adicionais nesses diversos servidores. Para que a atualização fosse automática, foram inseridas as seguintes linhas no *crontab* do servidor central, para os servidores espelhos de Nova Iorque, Madri e Fukuoka, respectivamente:

```
Rsync- avze ssh /db/ goran@trantor.bioc.columbia.edu:/razor/3/STING/diamond/data
rsync -avze ssh /db/ neshich@bossa nova.cnb.uam.es:SMS_database/data
rsync -avze ssh /db/ smsadmin@gibk26.bse.kyutech.ac.jp:SMS/SMS_backup/data
```

Para que cada um desses *rsync* pudessem ser colocados no *crontab* evitando que o *ssh* pedisse uma senha toda vez que se tivesse que executar cada um deles, foram criadas chaves públicas para cada um dos usuários de cada servidor espelho, utilizando um comando específico para isso, que é o *ssh-keygen*. O comando *ssh-keygen* faz parte da instalação do conjunto de ferramentas que vem junto com o programa *ssh*. A sintaxe usada para criação das chaves e a saída (em destaque) foram as seguintes:

```
% ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):<teclar Enter>
Enter same passphrase again: <teclar Enter>
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx root@gaviao.cbi.cnptia.embrapa.br
```

Esse comando foi executado pelo usuário *root*, e através dele foi gerada uma chave pública do tipo *RSA* (um algoritmo de encriptação de dados, que deve o seu nome a três professores do Instituto MIT, Ron Rivest, Adi Shamir e Len Adleman, que inventaram este algoritmo (Wikimedia Foundation, 2006) versão 2, que foi armazenada no arquivo *id_rsa.pub* no diretório *.ssh* do usuário *root* (como mostra a saída do comando acima). O conteúdo desse arquivo foi copiado para os servidores espelhos e armazenado no arquivo *authorized_keys* do diretório *.ssh* de cada usuário de acesso.

Conclusões

A atualização da base de dados do SMS nos servidores espelhos ficou muito mais eficiente e confiável depois da utilização do *rsync*, isto porque, além de transferir somente os arquivos novos e os que foram modificados na origem, ele o faz de forma rápida e se utilizando de um protocolo de criptografia muito seguro, no caso o *ssh*.

Antes da utilização do *rsync*, foram feitas outras tentativas de transferências dos dados, novos e modificados, para os espelhos. Entre estas tentativas estavam programas que usavam os protocolos ftp - File Transfer Protocol e http - HyperText Transfer Protocol, que apesar de mostrarem rapidez no transporte, não ofereciam nenhuma ferramenta adicional que pudesse fazer uma comparação entre os dados da origem e do destino, o que acabou causando significativas diferenças entre os arquivos que existiam no servidor central e nos espelhos.

Foi realizado um teste para se comparar a eficiência de cada protocolo. Neste teste se transferiu para o servidor de Nova Iorque um diretório da base de dados do SMS, com cerca de 40 mil arquivos textos, totalizando 800 MB de tamanho. Numa primeira transferência, o protocolo *http* foi um pouco mais rápido que o *ftp* e o *ssh* usado pelo *rsync*.

Apesar da maior rapidez dos protocolos *http* e *ftp* em relação ao *ssh* utilizado pelo *rsync*, em transferências posteriores, para atualização da base de dados, o *rsync* precisou de apenas poucos segundos para atualizar a base, pois só transferiu os arquivos novos e modificados, não necessitando de nenhuma ferramenta ou *script* extra para fazer a comparação dos arquivos na origem e no destino, como já dito anteriormente.

Referências Bibliográficas

BERMAN, H. M.; WESTBROOK, J.; FENG, Z.; GILLILAND, G.; BHAT, T. N.; WEISSIG, H.; SHINDYALOV, I. N.; BOURNE, P. E. The Protein Data Bank. *Nucleic Acids Research*, v. 28, n. 1, p. 235-242, 2000.

HOFFMANN, K.; BUCHER, P.; FALQUET, L.; BAIROCH, A. The PROSITE database, its status in 1999. *Nucleic Acids Research*, v. 27, n. 1, p. 215-219, 1999.

NEMETH, E.; SNYDER, G.; SEEBASS, S.; HEIN, R. T. *Manual do administrador do sistema Unix*. Porto Alegre: Bookman, 2002. p. 560-562.

SANDER, C.; SCHNEIDER, R. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins*: Structure, Function, and Genetics, v. 9, n. 1, p. 56-68, 1991.

SCHNEIDER, R.; DE DARUVAR, A.; SANDER, C. The HSSP database of protein structure-sequence alignments. *Nucleic Acids Research*, v. 25, n. 1, p. 226-230, 1997.

TRIDGELL, A.; MACKERRAS, P. *Manage of RSYNC*. Disponível em: <http://optics.ph.unimelb.edu.au/help/rsync/rsync.html>. Acesso em: 28 set. 2006

WIKIMEDIA FOUNDATION. *RSA Wikipedia*. Disponível em: <http://pt.wikipedia.org/wiki/RSA>. Acesso em: 28 set. 2006.

Comunicado Técnico, 82

Ministério da Agricultura, Pecuária e Abastecimento

GOVERNO FEDERAL

Embrapa Informática Agropecuária
Área de Comunicação e Negócios (ACN)
Endereço: Caixa Postal 6041 - Barão Geraldo
13083-970 - Campinas, SP
Fone: (19) 3789-5743
Fax: (19) 3289-9594
e-mail: sac@cnptia.embrapa.com.br

1ª edição on-line - 2007