DebianZine
Contribua você também com esta idéia...
FanZine Eletrônico sobre o Debian
Debian Zine # 3 - Outubro/2005 – www.debianbrasil.org/zine

Nesta Edição:

Colaboraram nesta edição:

Denis Brandl (denisbr)
<denisbr@gmail.com>

Felipe Augusto van de Wiel (faw)
<felipe@cathedrallabs.org>

Fernando Ike (fike)
<fernando.ike@gmail.com>

Marco Carvalho (macs)
<marcoacarvalho@gmail.com>

Pablo Lorenzzoni
<spectra@debian.org>

Raphael Bittencourt S. Costa
<raphaelbscosta@yahoo.com.br>

Rodrigo Tadeu Claro (rlinux)
<rlinux@cipsga.org.br>

Quer publicar seu artigo na

Debian Zine?

Envie seu artigo em arquivo texto puro, sem formatação, para zine@debianbrasil.org

debian brasil
http://debianbrasil.org/

Subversion + Apache2/SSL no Debian 3.1

Recentemente resolvi aderir à turma do Subversion, depois da insistência de meu colega desenvolvedor Otávio Salvador. Estava já há algum tempo usando Arch (tla), no entanto tinha que permanecer usando CVS a maior parte do tempo (e Subversion outra parte), o que me impedia de aprender "de verdade" a usar o Arch e todas as suas funcionalidades.

Como usuário de CVS, suas limitações começaram a tornar-se insuportáveis para meu uso, e resolvi testar o Subversion e aproveitar o know-how de controle de versões que o CVS já havia me ensinado (considerei a proximidade de sintaxe do Subversion e do CVS).

Instalação do Subversion e de um repositório

Esta foi a parte mais fácil:

```
bash# aptitude install subversion subversion-tools
```

Ao contrário do pacote do CVS (que instala um diretório /var/cvs como repositório central), o pacote do Subversion não faz o mesmo. Nunca achei lógico que o CVS instalasse um repositório na instalação (uma vez que podemos ter quantos repositórios quisermos). Para instalar um repositório é simples. Eu utilizo um repositório central em /home/svn, no entanto o usuário pode substituí-lo pela localização que quiser.

```
bash# mkdir /home/svn
bash# svnadmin create /home/svn/repos
bash# ls /home/svn/repos
conf/  dav/  db/  format  hooks/  locks/  README.txt
```

Perceba que criei um repositório no diretório "repos" dentro de /home/svn. Mais sobre isso adiante, quando falarmos sobre a configuração do Apache2.

O Subversion não usa RCS como o CVS, mas usa um banco de dados como sistema de arquivos. Existem duas versões desse sistema de arquivos: o bdb e o fsfs. O bdb nada mais é do que um banco de dados Berkeley DB (com todas suas virtudes e problemas). O fsfs é um sistema de arquivos próprio. Versões mais antigas do Subversion só usam bdb (ou o tem como padrão). Na versão 1.2 o fsfs passou a ser o padrão. Para os objetivos desse artigo podemos ignorar as diferenças.

Para listar o conteúdo desse banco de dados (e não os arquivos que ele grava, como fizemos acima com ls), o usuário deve utilizar o Subversion:

```
bash# svn ls file:///home/svn/repos
bash#
```

Obviamente não temos nada no repositório ainda.

Vale a pena criar um grupo "subversion" para ser dono do repositório. Assim os usuários pertencentes ao grupo podem alterá-lo:

```
bash# addgroup subversion
Adding group `subversion' (1056)...
Concluído
```

```
bash# chgrp -R subversion /home/svn
bash# chmod -R 775 /home/svn
```

O primeiro projeto

É prática comum, modernamente, utilizar subdiretórios para identificar porções de um projeto colaborativo. Comumente se utiliza os diretórios "branches", "tags" e "trunk" para identificar ramificações de um projeto, seus releases principais e seu repositório comum, respectivamente. Em "branches" guardamos as versões que se ramificaram do "trunk" original (não é coincidência que "branches" seja traduzido para "ramos" e "trunk" para "tronco"). Em "tags" armazenamos releases pontuais (por exemplo: RELEASE_1.0, COM_PATCH_DO_FULANO, etc). Então é uma boa ideia organizar nosso projeto dessa forma:

```
bash$ ls -R /tmp/meu_projeto
/tmp/meu_projeto:
branches/  tags/  trunk/

/tmp/meu_projeto/branches:

/tmp/meu_projeto/tags:

/tmp/meu_projeto/trunk:
Makefile  programa.c  programa.h

bash$ svn import /tmp/meu_projeto
file:///home/svn/repos/meu_projeto -m "inicio"
Adding         /tmp/meu_projeto/branches
Adding         /tmp/meu_projeto/tags
Adding         /tmp/meu_projeto/trunk
Adding         /tmp/meu_projeto/trunk/programa.h
Adding         /tmp/meu_projeto/trunk/programa.c
Adding         /tmp/meu_projeto/trunk/Makefile
-
Committed revision 1.
bash$
```

Podemos fazer o primeiro checkout do projeto para um diretório de trabalho de maneira muito simples:

```
bash$ mkdir work; cd work
bash$ svn checkout
file:///home/svn/repos/meu_projeto/trunk meu_projeto
A  meu_projeto/programa.c
A  meu_projeto/programa.h
A  meu_projeto/Makefile
-
Checked out revision 1.
bash$
```

Outros comandos similares ao CVS também funcionam: svn diff, svn commit, svn update.

Apache2, SSL, WebDAV e Subversion

O Subversion pode ser acessado através de um servidor próprio chamado svnserve. Pode fazer autenticação e criptografar a conexão com ssh, tal como o CVS. Se você criou um grupo "subversion" para ser "dono" do repositório, usuários com conta na sua máquina que pertençam a esse grupo podem acessar o repositório através de ssh. Não discutirei esses métodos de acesso, e uma leitura completa pode ser obtida do livro do Subversion [1].

Entretanto, a melhor diferença em relação ao CVS, talvez, seja a introdução do WebDAV para acesso ao repositório. Não é objetivo desse artigo discutir WebDAV. Para uma leitura introdutória, sugiro a Wikipedia [2].

Para utilizar o WebDAV temos de ativar o Apache2, o módulo SSL e configurar o local do repositório no apache. Primeiro, instale o Apache2 e o módulo para comunicação com o Subversion:

```
bash# aptitude install apache2 libapache2-svn
```

Crie os certificados e a configuração inicial para o seu site:

```
bash# apache2-ssl-certificate
bash# cd /etc/apache2/sites-available/
bash# cp default ssl
bash# a2ensite ssl
bash# echo "Listen 443" >> /etc/apache2/ports.conf
```

Certifique-se de que o módulo SSL esteja sendo chamado na inicialização do Apache2:

```
bash# a2enmod ssl
Module ssl installed; run /etc/init.d/apache2 force-
reload to enable.
bash#
```

Edite o arquivo de configuração /etc/apache2/sites-available/ssl para algo parecido com:

```
NameVirtualHost *:443
<VirtualHost *:443>
SSLEngine On
SSLCertificateFile /etc/apache2/ssl/apache.pem
...
<Location /svn>
   DAV svn
   SVNPath /home/svn/repos

   AuthType Basic
   AuthName "Meu Repositório Subversion"
   AuthUserFile /home/svn/dav_svn.passwd
   <LimitExcept GET PROPFIND OPTIONS REPORT>
       Require valid-user
   </LimitExcept>
</Location>
</VirtualHost>
```

Perceba, na tag "Location", que podemos utilizar o nome que quisermos. Eu utilizo /svn, mas poderia utilizar /subversion, /repositorio, /projeto, /svn/repos, etc. Esse é o sufixo do endereço pelo qual os usuários poderão acessar seu repositório. No exemplo:

```
https://meusite.homelinux.net/svn
```

Além disso, o arquivo dav_svn.passwd com os usuários ficará em /home/svn e não em /home/svn/repos. Esse arquivo fica escondido do apache, uma vez que SVNPath dá a localização do

repositório. Eis o motivo para termos criado o repositório em /home/svn/repos e não em /home/svn.

Crie os usuários que terão permissão para mexer no repositório de fora (perceba que a opção -c só é necessária para a criação do arquivo):

```
bash# htpasswd2 -c -m /home/svn/dav_svn.passwd
spectra
bash# htpasswd2 -m /home/svn/dav_svn.passwd
pablo
bash# htpasswd2 -m /home/svn/dav_svn.passwd
otavio
```

Não esqueça de fazer o dono do Apache2 o dono do seu repositório:

```
bash# chown -R www-data /home/svn
```

Finalmente, reinicie o apache:

```
bash# /etc/init.d/apache2 restart
```

Seu repositório deverá estar acessível em:

```
https://localhost/svn
```

Você verá algo como:

```
Revision 1: /

    * meu_projeto/

Powered by Subversion version 1.1.4 (r13838).
```

Agora, quaisquer usuários podem acessar o repositório com um simples:

```
bash$ svn checkout
https://meusite.homelinux.net/svn/meu_projeto/t
runk
```

Usuários criados com htpasswd2 terão permissão de escrita no repositório, e poderão ajudar no projeto diretamente.

Revisões por repositório e futuros artigos

As revisões do Subversion são orientadas ao repositório. Portanto, se importarmos outro projeto dentro do mesmo repositório a revisão do repositório sobe para 2, mesmo sem mexermos em meu_projeto. Isso pode parecer meio estranho no início, mas é realmente uma questão de costume. Se preferir pode criar um repositório para cada projeto, mas isso já está fora do escopo desse artigo.

Ao mudar de Arch para Subversion, obtive a grande vantagem da sintaxe muito parecida com a do CVS (ao mesmo tempo que evitava a sintaxe alienígena do Arch). No entanto perdi a maior vantagem do Arch: a descentralização. No próximo número do DebianZine vou falar sobre uma ferramenta que supre essa necessidade: svk [3]. Até lá.

Notas e Referências

```
[1] http://svnbook.red-bean.com/ [2]
http://en.wikipedia.org/wiki/WebDAV [3]
http://svk.elixus.org/
```


Autenticacao Linux Workstation no Active Directory

Anteriormente, este artigo era para descrever sobre implantação de um Servidor LDAP (OpenLDAP) com estação autenticando nele, baseado apenas em software livre, mas infelizmente o mundo corporativo não é tão homogêneo como gostaria e a necessidade foi configurar um Servidor LDAP (Active Directory) proprietário (argh!). Não tendo muita escolha, este artigo tenta ajudar os profissionais que passam pela mesma situação.

Neste artigo estamos usando o Debian 3.1 (Sarge).

Pacotes necessários

```
# aptitude install ssh libdb4.2 libsasl2 nscd
# aptitude install ldap-utils libldap2 libkrb53
krb5-user krb5-config
```

Perguntas feitas pelo debconf para o autenticação Kerberos

1 - REALM do Active Directory
`DOMINIO.COM.BR`

2 - Servidor DNS da rede

```
servidor_responsavel_AD
```

3 - Servidor Master do Active Directory
```
servidor_responsavel_AD
```

```
#aptitude install winbind smbfs smbclient
```

Perguntas feitas pelo debconf para autenticação samba/winbind:

1 - Domínio Microsoft Windows:
```
dominio
```

2 - apontar o WINS pelo DHCP?
```
não
```

3 - Excutar como serviço (daemon)>
```
sim
```

Arquivo de configuração do smb.conf

```
[global]
workgroup = dominio
server string = %h server (SAMBA/WINBIND %v)
wins server = ip_do_servidor_wins
dns proxy = no
log file = /var/log/SAMBA/WINBIND/log.%m
max log size = 1000
syslog = 0
display charset = UTF8
panic action = /usr/share/SAMBA/WINBIND/panic-
action %d
security = ads
password server = servidor_responsavel_AD
realm = DOMINIO.COM.BR
encrypt passwords = yes
passdb backend = tdbsam guest
obey pam restrictions = yes
invalid users = root
passwd program = /usr/bin/passwd %u
passwd chat = *Enter\snew\sUNIX\spassword:* %n\n
*Retype\snew\sUNIX\spassword:* %n\n .
load printers = yes
printing = cups
printcap name = cups
socket options = TCP_NODELAY SO_SNDBUF=8192
SO_RCVBUF=8192
winbind uid = 10000-20000
winbind gid = 10000-20000
winbind use default domain = yes
template homedir = /home/%U
template shell = /bin/bash
restrict anonymous = no
guest ok = yes
domain master = no
preferred master = no
max protocol = NT
ldap ssl = No
server signing = Auto
```

Reinicie o samba e o winbind
```
# net ads join -U root
```

Para testar se o samba/winbind estão pegando os usuários do AD
```
#wbinfo -u (lista os usuários)
#wbinfo -g (lista os grupos)
```

Com a comunicação entre o samba/winbind e o Active Directory funcionando, vamos editar o /etc/nsswich.conf para a PAM autenticar via samba/winbind

De:
```
passwd: compat
shadow: compat
group:  compat
```

Para:
```
passwd: files winbind
shadow: files
group:  files winbind
```

Agora, editamos o /etc/pam.d/common-auth para ficar parecido com as duas linhas abaixo:

```
auth    sufficient    pam_winbind.so
auth    required     pam_unix.so     nullok_secure
   use_first_pass
```

O arquivo /etc/pam.d/common-account também:

```
account sufficient    pam_winbind.so
account required      pam_unix.so
```

E por fim o /etc/pam.d/common-session também:

```
session required      pam_mkhomedir.so
skel=/etc/skel umask=0027
session required      pam_unix.so
```

Para testar autenticação PAM+SAMBA/WINBIND+AD execute:

```
#getent passwd
#getent group
```

O resultado deve exibir os usuários e grupos do Active Directory. Parabéns, sua máquina está aparecendo na rede Microsoft Windows.

Referências
```
http://www.solis.coop.br/modules/ldap/files/files/sa
mbaldap.pdf
http://www.solis.coop.br/modules/ldap/files/files/pa
m_ldap.txt
```

Criptografia com chaves GNUPG

Introdução

Este artigo pretende alertar os utilizadores de que o email não é um canal seguro de comunicação, no entanto, pode sê-lo usando o GnuPG. Como a maioria dos utilizadores já devem estar cientes, os pacotes com mensagens de correio eletrônico, entre outros, viajam livremente pela Internet, uma rede reconhecidamente insegura, até atingirem seus respectivos destinos. Neste percurso, um intruso pode facilmente interceptá-los e ler, ou até alterar seu conteúdo. Basta pensarmos um pouco numa troca de emails nunca podemos ter a certeza de que a mensagem enviada não foi interceptada e lida por terceiros. Estes terceiros podem ser, os nossos bem amados fornecedores de internet (Internet Service Providers), lamers bisbilhoteiros, invasores, ou mesmo alguém utilizando o seu PC quando está ausente. Assim, as nossas mensagens podem ser lidas em vários locais.

No nosso próprio PC, na caixa postal do ISP (as mensagens que recebemos e enviamos ficam armazenadas num diretório do servidor), no PC do destinatário, além de, neste longo caminho as mensagens ficarem ainda armazenadas nas máquinas pelas quais vão passando até chegarem ao seu destino. Desta forma podemos concluir que a rede não é um canal seguro para trocas de mensagens confidenciais. Graças à criptografia, a rede pode ser utilizada para troca de mensagens confidencias.

Porque é que eu devo criptografar os meus documentos?

Pelas razões acima referidas, suponha que alguém tem acesso ao seu PC, estará apto para roubar as senhas, os seus números de cartões de crédito, bisbilhotar os seus documentos pessoais, e etc. Se esta informação estiver criptografada não significará nada para o invasor e pode ter a certeza que este terá um trabalho bastante árduo para furar o mecanismo de segurança.

Quem utiliza o GnuPG?

Todas as pessoas que dão valor à privacidade, políticos, jornalistas, empresas, homens de negócios, etc. O PGP é ainda utilizado em transações financeiras como as que são feitas através da Internet. "Não tenho nada a esconder, por que preciso de privacidade"? Não acredito, apresente-me alguém que não tenha absolutamente nada a esconder da sua família, dos seus vizinhos ou dos seus colegas. Apresente-me alguma empresa que não tenha segredos a esconder dos seus concorrentes. Suponha que não é o único a utilizar o seu computador. Vai deixar os seus documentos pessoais abertos a pessoas estranhas? Bem, me parecia que não. Introdução à criptografia, de um modo geral, é uma ciência ou a arte de cifrar e decifrar informações, mantendo-as em segredo e garantindo que somente pessoas autorizadas tenham acesso a elas. Com a criptografia, podemos também criar mecanismos de autenticação com assinaturas

digitais e métodos para verificação da integridade dos dados. Tão antiga como a escrita, a criptografia tem evoluído desde a invenção do computador, transformando-se numa ferramenta imprescindível nestes tempos de Internet e correio eletrônico. Criptografia (do Grego Kryptos) é a arte de escrever secretamente, ou seja, por meio de cifras e sinais convencionais.

A criptografia existe desde muito tempo atrás. Quando Júlio César enviava mensagens aos seus generais fazia-o de forma criptografada, ou seja, substituía o A pelo D, o B pelo E e assim sucessivamente até ao fim do alfabeto. Desta forma, só quem conhecia o código de descriptografia poderia ler as mensagens. Assim, nasceu a criptografia. A informação que pode ser lida sem o uso de qualquer mecanismo extra tem o nome de texto plano ou texto limpo. O método de codificar essa informação chama-se criptografia. Ao codificar texto simples obtém-se algo indecifrável. Ao processo de descodificação, ou seja, tornar o texto original novamente legível, dá-se o nome de descriptografia. A tecnologia da criptografia não mudou muito até a Segunda Guerra Mundial. Com a invenção do computador, a área cresceu rapidamente. Aliás, muitos afirmam que o computador moderno é uma criação da criptografia, pois algumas das

primeiras máquinas foram construídas pelos aliados para quebrar mensagens militares que tinham sido codificadas pelos alemães, durante a Guerra. A ciência de quebrar códigos e decifrar a informação sem conhecer a chave utilizada é conhecida como criptoanálise. A criptologia é a união da criptografia com a criptoanálise.

Assim sendo, nos dias atuais, nada mais importante do que mantermos nossos contatos, rede de amizades e principalmente negócios sob a vigilância da segurança da informação e, para conseguirmos um patamar aceitável, dispomos desta excelente ferramenta chamada GnuPG no Debian.

Como dito, esta ferramenta gera automaticamente um par de chaves (uma pública e outra privada) para que possamos criptografar nossas mensagens, arquivos e até diretórios inteiros. Mas vamos ao que interessa, a prática:

1) Primeiro você vai precisar do pacote ´gnupg´, então instale-o com o comando:

```
#aptitude install gnupg
```

2) Depois você precisa gerar um par de chaves, uma pública e outra privada.

Para gerar suas chaves, na sua área (ou diretório pessoal) digite:

```
$ gpg --gen-key
```

Ele vai criar umas coisas e pedir para digitar novamente, digite:

```
$ gpg --gen-key
```

Ele vai te fazer algumas perguntas, sobre tipo de encriptação de quantos bits e dados seus, vou por as que eu escolhi aqui, mas você pode colocar as opções de acordo com as que desejar.

```
Please select what kind of key you want: (1) DSA
and ElGamal (default) (2) DSA (sign only) (4)
ElGamal (sign and encrypt) (5) RSA (sign only) Your
selection?
```

Escolha 1

```
DSA keypair will have 1024 bits.  About to generate
a new ELG-E keypair.  minimum keysize is 768 bits
default keysize is 1024 bits highest suggested
keysize is 2048 bits What keysize do you want?
```

Escolha 1024

```
Please specify how long the key should be valid.  0
= key does not expire <n> = key expires in n days
<n>w = key expires in n weeks <n>m = key expires in
n months <n>y = key expires in n years Key is valid
for?
```

Escolha 0

Agora ele pergunta se as configurações que colocou estão corretas, diga que sim, então ele pedirá seus dados.

```
Real Name: Email address: Comment: (Seu nick por
exemplo)
```

No meu caso deixei: Rodrigo Tadeu Claro / rlinux@cipsga.org.br / rlinux

Agora ele pergunta se está tudo ok, diga ***OK***

```
Enter passphrase:
```

Nesta ETAPA você terá que digitar sua "frase-secreta" que você vai usar para assinar e ou criptografar as mensagens, isso é feito para que, se alguém conseguir sua chave privada, não a use para enviar e-mails assinados por você sem saber essa frase-secreta.

Agora, espere um pouco até ele gerar suas chaves.

Pronto, as chaves estão geradas e dentro do dir ~/.gnupg.

ATENÇÂO: faça um backup dos arquivos "pubring.gpg" e "secring.gpg". (entretanto, lembre-se que estes arquivos são suas chaves pública e privada respectivamente. Nunca divulgue sua chave privada em servidores de chaves, apenas a sua chave pública para que seus amigos/clientes possam baixá-las e utilizá-las para criptografar mensagens/arquivos destinados apenas à você. Lembre-se ainda que estes arquivos estarão em um diretório oculto no seu sistema (o .gnupg) e para poder visualizá-lo digite no seu diretório):

```
$ls -la | grep .gnupg
```

DICA: Copie este diretório oculto para um disquete, pois nele estará seu par de chaves que são identificados como secring.gpg e pubring.gpg:

1) Monte o disquete e depois rode o comando:

```
$cp /home/.gnupg /dev/fd0
```

Para poder utilizar essas chaves para enviar mensagens criptografadas e/ou assinadas, você precisa de um leitor de e-mails que trabalhe com GnuPG, o Sylpheed-claws, Kmail e Evolution são de excelente qualidade e dão suporte a chaves gnupg.

A minha chave pública pode ser encontrada em http://pgp.mit.edu (um servidor de chaves público mantido pelo MIT). Para encontrá-la, basta digitar no campo search "meu nome" :)

Para listar as chaves que você possui em seu chaveiro basta digitar o seguinte comando no prompt do seu terminal:

```
$gpg --list-keys
```

E deverá aparecer algo como abaixo:

```
pub 1024D/D33084F2 2002-11-05 Rodrigo Tadeu Claro
(rlinux) rlinux@cipsga.org.br Key fingerprint =
61C1 EE0F AC5A 5711 F44E ABFE 4B61 24E1 D330 84F2
sub 1024g/D7A56C5A 2002-11-05
```

NOTA: se você possuir mais chaves no seu chaveiro pessoal, então aparecerão todas. Atenção: depois de baixar minha chave pública do servidor de chaves do MIT, você poderá conferir no seu leitor de email's se o conteúdo das mensagens enviadas por mim são realmente de minha autoria pois, aparecerá a palavra "assinatura válida". Contudo, antes de mais nada você deverá baixar/importar minha chave pública para seu chaveiro pessoal, fazendo assim:

```
$gpg --keyserver=pgp.mit.edu --recv-keys D33084F2
```

Qualquer dúvida, sinta-se à vontade para entrar em contato.

Mais informações poderão ser encontradas em minha página [1].

```
[1]
http://www.rlinux.com.br/modules.php?name=Sections&o
p=viewarticle&artid=19
```