Autenticação LDAP

Resumo

Este documento pretende ser um guia para a configuração de um servidor LDAP (principalmente um servidor OpenLDAP) para autenticação no FreeBSD. Isso é útil para situações em que muitos servidores precisam das mesmas contas de usuário, por exemplo, como substituto do NIS.

Índice

1. Prefácio

Este documento destina-se a fornecer ao leitor uma compreensão suficiente do LDAP para configurar um servidor LDAP. Este documento tentará fornecer uma explicação de net/nss_ldap e security/pam_ldap para uso com serviços de máquinas cliente para uso com o servidor LDAP.

Quando terminar, o leitor deve ser capaz de configurar e implantar um servidor FreeBSD que possa hospedar um diretório LDAP e configurar e implantar um servidor FreeBSD que possa autenticar em um diretório LDAP.

Este artigo não pretende ser um relato exaustivo da segurança, robustez ou considerações sobre práticas recomendadas para configurar o LDAP ou os outros serviços discutidos aqui. Embora o autor tenha o cuidado de fazer tudo corretamente, ele não aborda problemas de segurança além do escopo geral. Este artigo deve ser considerado para estabelecer as bases teóricas somente, e qualquer implementação real deve ser acompanhada por uma análise cuidadosa dos requisitos.

2. Configurando o LDAP

LDAP significa "Lightweight Directory Access Protocol" e é um subconjunto do X.500 Directory Access Protocol. Suas especificações mais recentes estão na RFC4510 e documentos amigáveis. Essencialmente, é um banco de dados que espera ser lido com mais frequência do que é escrito.

O servidor LDAP OpenLDAP será usado nos exemplos deste documento; embora os princípios aqui devam ser geralmente aplicáveis a muitos servidores diferentes, a maior parte da administração concreta é especificamente para OpenLDAP. Existem várias versões de servidor nos ports, por

exemplo net/openldap24-server. Os servidores clientes precisarão das bibliotecas net/openldap24-client correspondentes.

Existem (basicamente) duas áreas do serviço LDAP que precisam de configuração. A primeira é a configuração de um servidor para receber as conexões corretamente, e o segundo é adicionar entradas ao diretório do servidor para que as ferramentas do FreeBSD saibam como interagir com ele.

2.1. Configurando o servidor para conexões

Esta seção é específica do OpenLDAP. Se você estiver usando outro servidor, precisará consultar a documentação desse servidor.

2.1.1. Instalando o OpenLDAP

Primeiro, instale o OpenLDAP:

Exemplo 1. Instalando o OpenLDAP

```
# cd /usr/ports/net/openldap24-server
# make install clean
```

Isso instala os binários `slapd` e `slurpd`, juntamente com as bibliotecas OpenLDAP requeridas.

2.1.2. Configurando o OpenLDAP

Em seguida, devemos configurar o OpenLDAP.

Você desejará exigir criptografia em suas conexões com o servidor LDAP; caso contrário, as senhas de seus usuários serão transferidas em texto simples, o que é considerado inseguro. As ferramentas que usaremos suportam dois tipos muito semelhantes de criptografia, SSL e TLS.

TLS significa "Segurança da Camada de Transporte". Serviços que empregam TLS tendem a se conectar nas *mesmas* portas que os mesmos serviços sem TLS; assim, um servidor SMTP que suporte o TLS escutará as conexões na porta 25 e um servidor LDAP escutará no 389.

SSL significa "Secure Sockets Layer", e serviços que implementam SSL *não* escutam nas mesmas portas que seus equivalentes não-SSL. Assim, o SMTPS atende na porta 465 (não 25), HTTPS escuta na 443 e LDAPS na 636.

A razão pela qual o SSL usa uma porta diferente do TLS é porque uma conexão TLS começa como texto simples e alterna para o tráfego criptografado após a diretiva `STARTTLS`. As conexões SSL são criptografadas desde o início. Além disso, não há diferenças substanciais entre os dois.

Ajustaremos o OpenLDAP para usar o TLS, já que o SSL é considerado obsoleto.

Uma vez que o OpenLDAP esteja instalado via ports, os seguintes parâmetros de configuração em

/usr/local/etc/openldap/slapd.conf irão ativar o TLS:

```
security ssf=128

TLSCertificateFile /path/to/your/cert.crt
TLSCertificateKeyFile /path/to/your/cert.key
TLSCACertificateFile /path/to/your/cacert.crt
```

Aqui, `ssf=128` diz ao OpenLDAP para exigir criptografia de 128 bits para todas as conexões, tanto de pesquisa quanto de atualização. Esse parâmetro pode ser configurado com base nas necessidades de segurança do seu site, mas raramente é necessário enfraquecê-la, pois a maioria das bibliotecas de clientes LDAP oferece suporte à criptografia forte.

Os arquivos cert.crt, cert.key e cacert.crt são necessários para que os clientes autentiquem *você* como o servidor LDAP válido. Se você simplesmente quiser um servidor que seja executado, poderá criar um certificado auto-assinado com o OpenSSL:

Exemplo 2. Gerando uma chave RSA

```
% openssl genrsa -out cert.key 1024
Generating RSA private key, 1024 bit long modulus
.....................++++++
...++++++
e is 65537 (0x10001)

% openssl req -new -key cert.key -out cert.csr
```

Neste ponto, você deve ser solicitado para digitar alguns valores. Você pode inserir os valores que quiser; no entanto, é importante que o valor "Common Name" seja o nome de domínio totalmente qualificado do servidor OpenLDAP. No nosso caso, e os exemplos aqui, o servidor é *server.example.org*. Definir incorretamente esse valor fará com que os clientes falhem ao fazer conexões. Isso pode causar uma grande frustração, portanto, certifique-se de seguir atentamente estas etapas.

Por fim, a requisição de assinatura de certificado precisa ser assinada:

Exemplo 3. Auto-assinando o certificado

```
% openssl x509 -req -in cert.csr -days 365 -signkey cert.key -out cert.crt
Signature ok
subject=/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd
Getting Private key
```

Isso criará um certificado auto-assinado que pode ser usado para as diretivas em slapd.conf, onde cert.crt e cacert.crt são o mesmo arquivo. Se você for usar muitos servidores OpenLDAP (para

replicação via `slurpd`), você vai querer ver Certificados OpenSSL para o LDAP para gerar uma chave CA e usá-la para assinar certificados de servidor individuais.

Feito isso, coloque o seguinte em /etc/rc.conf:

```
slapd_enable="YES"
```

Em seguida, execute `/usr/local/etc/rc.d/slapd start`. Isso deve iniciar o OpenLDAP. Confirme que está escutando em 389 com

```
% sockstat -4 -p 389
ldap     slapd      3261  7  tcp4   *:389                 *:*
```

2.1.3. Configurando o Cliente

Instale o port net/openldap24-client para as bibliotecas do OpenLDAP. As máquinas cliente sempre terão bibliotecas OpenLDAP, já que é todo o suporte a security/pam_ldap e net/nss_ldap, pelo menos por enquanto.

O arquivo de configuração para as bibliotecas OpenLDAP é /usr/local/etc/openldap/ldap.conf. Edite este arquivo para conter os seguintes valores:

```
base dc=example,dc=org
uri ldap://server.example.org/
ssl start_tls
tls_cacert /path/to/your/cacert.crt
```

É importante que seus clientes tenham acesso ao cacert.crt, caso contrário, eles não poderão se conectar.

Existem dois arquivos chamados ldap.conf. O primeiro é este arquivo, que é para as bibliotecas OpenLDAP e define como falar com o servidor. O segundo é /usr/local/etc/ldap.conf e é para pam_ldap.

Neste ponto, você deve conseguir executar `ldapsearch -Z` na máquina cliente; `-Z` significa "usar o TLS". Se você encontrar um erro, então algo está configurado errado; muito provavelmente são seus certificados. Use os comandos `s_client` e `s_server` do openssl(1) para assegurar que você os tenha configurado e assinado corretamente.

2.2. Entradas no banco de dados

A autenticação em um diretório LDAP geralmente é realizada pela tentativa de vincular ao diretório como o usuário de conexão. Isso é feito estabelecendo um vinculo "simples" no diretório com o nome de usuário fornecido. Se houver uma entrada com o `uid` igual ao nome do usuário e o atributo `userPassword` da entrada corresponder à senha fornecida, o vinculo será bem-sucedido.

A primeira coisa que temos que fazer é descobrir onde no diretório os nossos usuários irão estar.

A entrada de base para nosso banco de dados é `dc=example,dc=org`. O local padrão para usuários que a maioria dos clientes parece esperar é algo como `ou=people, base`, então é isso que será usado aqui. No entanto, tenha em mente que isso é configurável.

Assim, a entrada ldif para a unidade organizacional `people` será semelhante a:

```
dn: ou=people,dc=example,dc=org
objectClass: top
objectClass: organizationalUnit
ou: people
```

Todos os usuários serão criados como subentradas dessa unidade organizacional.

Alguma consideração pode ser dada à classe de objeto a que seus usuários pertencerão. A maioria das ferramentas, por padrão, usará `people`, o que é bom se você quiser simplesmente fornecer entradas para autenticar. No entanto, se você for armazenar informações do usuário no banco de dados LDAP, provavelmente usará `inetOrgPerson`, que possui muitos atributos úteis. Em ambos os casos, os esquemas relevantes precisam ser carregados em slapd.conf.

Para este exemplo, usaremos a classe de objeto `person`. Se você estiver usando `inetOrgPerson`, as etapas são basicamente idênticas, exceto que o atributo `sn` é necessário.

Para adicionar um usuário de teste chamado `tuser`, o ldif seria:

```
dn: uid=tuser,ou=people,dc=example,dc=org
objectClass: person
objectClass: posixAccount
objectClass: shadowAccount
objectClass: top
uidNumber: 10000
gidNumber: 10000
homeDirectory: /home/tuser
loginShell: /bin/csh
uid: tuser
cn: tuser
```

Eu inicio os UIDs dos meus usuários LDAP em 10000 para evitar colisões com contas do sistema; você pode configurar o número que desejar aqui, desde que seja menor que 65536.

Também precisamos de entradas de grupo. Eles são configuráveis como entradas do usuário, mas usaremos os padrões abaixo:

```
dn: ou=people,dc=example,dc=org
objectClass: top
objectClass: organizationalUnit
ou: people
```

```
dn: cn=tuser,ou=groups,dc=example,dc=org
objectClass: posixGroup
objectClass: top
gidNumber: 10000
cn: tuser
```

Para inseri-los em seu banco de dados, você pode usar `slapadd` ou `ldapadd` em um arquivo contendo essas entradas. Alternativamente, você pode usar o sysutils/ldapvi.

O utilitário `ldapsearch` na máquina cliente deve agora retornar essas entradas. Em caso afirmativo, o banco de dados está configurado corretamente para ser usado como um servidor de autenticação LDAP.

3. Configuração do Cliente

O cliente já deve ter bibliotecas do OpenLDAP do Configurando o Cliente, mas se você estiver instalando várias máquinas clientes, precisará instalar o net/openldap24-client em cada um deles.

O FreeBSD requer que dois ports sejam instalados para autenticação em um servidor LDAP, security/pam_ldap e net/nss_ldap.

3.1. Autenticação

O security/pam_ldap é configurado através do /usr/local/etc/ldap.conf.

Este é um *arquivo diferente* que o arquivo de configuração das funções da biblioteca OpenLDAP, /usr/local/etc/openldap/ldap.conf; no entanto, são necessárias muitas das mesmas opções; na verdade, é um superconjunto desse arquivo. Para o resto desta seção, referências a ldap.conf irão significar o arquivo /usr/local/etc/ldap.conf.

Assim, vamos querer copiar todos os nossos parâmetros de configuração originais do openldap/ldap.conf para o novo ldap.conf. Feito isso, queremos informar ao security/pam_ldap o que procurar no servidor de diretório.

Estamos identificando nossos usuários com o atributo `uid`. Para configurar isso (embora seja o padrão), defina a diretiva `pam_login_attribute` no ldap.conf:

Exemplo 4. Definindo `pam_login_attribute`

```
pam_login_attribute uid
```

Com esta definição, o security/pam_ldap pesquisará todo o diretório LDAP na `base` para o valor `uid=username`. Se encontrar uma e apenas uma entrada, ela tentará se vincular como aquele usuário com a senha que foi fornecida. Se vincular corretamente, então permitirá o acesso. Caso contrário,

falhará.

Os usuários cujo shell não está em /etc/shells não poderão efetuar login. Isto é particularmente importante quando o Bash é definido como o shell do usuário no servidor LDAP. O Bash não está incluído em uma instalação padrão do FreeBSD. Quando instalado a partir de um pacote ou port, ele está localizado em /usr/local/bin/bash. Verifique se o caminho para o shell no servidor está definido corretamente:

```
% getent passwd username
```

Existem duas opções quando a saída mostra `/bin/bash` na última coluna. A primeira é alterar a entrada do usuário no servidor LDAP para /usr/local/bin/bash. A segunda opção é criar um link simbólico no computador cliente LDAP para que o Bash seja encontrado no local correto:

```
# ln -s /usr/local/bin/bash /bin/bash
```

Certifique-se de que /etc/shells contenha entradas para ambos `/usr/local/bin/bash` e `/bin/bash`. O usuário poderá então efetuar login no sistema com Bash como seu shell.

3.1.1. PAM

PAM, que significa "Pluggable Authentication Modules", é o método pelo qual o FreeBSD autentica a maioria de suas sessões. Para dizer ao FreeBSD que desejamos usar um servidor LDAP, teremos que adicionar uma linha ao arquivo PAM apropriado.

Na maioria das vezes o arquivo PAM apropriado é /etc/pam.d/sshd, se você quiser usar SSH (lembre-se de definir as opções relevantes em /etc/ssh/sshd_config, caso contrário o SSH não usará o PAM).

Para usar o PAM para autenticação, adicione a linha

```
auth  sufficient  /usr/local/lib/pam_ldap.so  no_warn
```

Exatamente onde essa linha aparece no arquivo e quais opções aparecem na quarta coluna, determine o comportamento exato do mecanismo de autenticação; veja pam.d(5)

Com essa configuração, você deve conseguir autenticar um usuário em um diretório LDAP. O PAM executará uma ligação com suas credenciais e, se for bem-sucedido, informará ao SSH para permitir o acesso.

No entanto, não é uma boa idéia permitir que *todo* usuário no diretório dentro de *todo* computador cliente. Com a configuração atual, tudo o que um usuário precisa para efetuar login em uma máquina é uma entrada LDAP. Felizmente, existem algumas maneiras de restringir o acesso do usuário.

O ldap.conf suporta uma diretiva `pam_groupdn`; Cada conta que se conecta a essa máquina precisa ser membro do grupo especificado aqui. Por exemplo, se você tem

```
pam_groupdn cn=servername,ou=accessgroups,dc=example,dc=org
```

em ldap.conf, somente os membros desse grupo poderão efetuar login. Entretanto, há algumas coisas a serem lembradas.

Os membros desse grupo são especificados em um ou mais atributos `memberUid` e cada atributo deve ter o nome distinto completo do membro. Então `memberUid:someuser` não funcionará; deve ser:

```
memberUid: uid=someuser,ou=people,dc=example,dc=org
```

Além disso, essa diretiva não é verificada no PAM durante a autenticação, ela é verificada durante o gerenciamento de contas, portanto, você precisará de uma segunda linha em seus arquivos PAM sob `account`. Isso exigirá, por sua vez, que *todo* usuário seja listado no grupo, o que não é necessariamente o que queremos. Para evitar o bloqueio de usuários que não estão no LDAP, você deve ativar o atributo `ignore_unknown_user`. Finalmente, você deve definir a opção `ignore_authinfo_unavail` para que você não fique bloqueado em todos os computadores quando o servidor LDAP estiver indisponível.

Seu pam.d/sshd pode acabar ficando assim:

Exemplo 5. Exemplo pam.d/sshd

```
auth            required        pam_nologin.so          no_warn
auth            sufficient      pam_opie.so             no_warn no_fake_prompts
auth            requisite       pam_opieaccess.so       no_warn allow_local
auth            sufficient      /usr/local/lib/pam_ldap.so      no_warn
auth            required        pam_unix.so             no_warn try_first_pass

account         required        pam_login_access.so
account         required        /usr/local/lib/pam_ldap.so      no_warn
ignore_authinfo_unavail ignore_unknown_user
```

Como estamos adicionando essas linhas especificamente para pam.d/sshd, isso só terá um efeito nas sessões SSH. Os usuários LDAP não poderão efetuar login no console. Para mudar este comportamento, examine os outros arquivos em /etc/pam.d e modifique-os de acordo.

3.2. Switch de serviço de nome

NSS é o serviço que mapeia atributos para nomes. Assim, por exemplo, se um arquivo é de propriedade do usuário `1001`, um aplicativo consultará o NSS para o nome de `1001`, e ele pode obter `bob` ou `ted` ou qualquer que seja o nome do usuário.

Agora que nossas informações sobre o usuário são mantidas no LDAP, precisamos dizer ao NSS para

procurar lá quando perguntado.

O port net/nss_ldap faz isso. Ele usa o mesmo arquivo de configuração como security/pam_ldap e não deve precisar de nenhum parâmetro extra depois de instalado. Em vez disso, o que resta é simplesmente editar é /etc/nsswitch.conf para aproveitar o diretório. Simplesmente substitua as seguintes linhas:

```
group: compat
passwd: compat
```

com

```
group: files ldap
passwd: files ldap
```

Isso permitirá que você mapeie nomes de usuários para UIDs e UIDs para nomes de usuários.

Parabéns! Agora você deve ter autenticação LDAP em funcionamento.

3.3. Ressalvas

Infelizmente, a partir do momento em que isso foi escrito, o FreeBSD não suportava a mudança de senhas de usuário com passwd(1). Por causa disso, a maioria dos administradores estão deixando para implementar uma solução por conta própria. Eu forneço alguns exemplos aqui. Observe que, se você escrever seu próprio script de alteração de senha, há alguns problemas de segurança dos quais você deve estar ciente; veja Armazenamento de Senha

Exemplo 6. Script de shell para alteração de senhas

```
#!/bin/sh

stty -echo
read -p "Old Password: " oldp; echo
read -p "New Password: " np1; echo
read -p "Retype New Password: " np2; echo
stty echo

if [ "$np1" != "$np2" ]; then
  echo "Passwords do not match."
  exit 1
fi

ldappasswd -D uid="$USER",ou=people,dc=example,dc=org \
  -w "$oldp" \
  -a "$oldp" \
  -s "$np1"
```

Esse script dificilmente faz qualquer verificação de erros, mas, o mais importante, é muito indiferente sobre como ele armazena suas senhas. Se você fizer algo assim, ajuste pelo menos o valor de sysctl `security.bsd.see_other_uids`:

```
# sysctl security.bsd.see_other_uids=0
```

Uma abordagem mais flexível (e provavelmente mais segura) pode ser usada escrevendo um programa personalizado, ou até mesmo uma interface web. A seguir, parte de uma biblioteca Ruby que pode alterar senhas LDAP. Ele vê o uso na linha de comando e na web.

Exemplo 7. Script Ruby para Alterar Senhas

```
require 'ldap'
require 'base64'
require 'digest'
require 'password' # ruby-password

ldap_server = "ldap.example.org"
luser = "uid=#{ENV['USER']},ou=people,dc=example,dc=org"

# get the new password, check it, and create a salted hash from it
def get_password
  pwd1 = Password.get("New Password: ")
  pwd2 = Password.get("Retype New Password: ")

  raise if pwd1 != pwd2
  pwd1.check # check password strength

  salt = rand.to_s.gsub(/0\./, '')
  pass = pwd1.to_s
  hash =
"{SSHA}"+Base64.encode64(Digest::SHA1.digest("#{pass}#{salt}")+salt).chomp!
  return hash
end

oldp = Password.get("Old Password: ")
newp = get_password

# We'll just replace it.  That we can bind proves that we either know
# the old password or are an admin.

replace = LDAP::Mod.new(LDAP::LDAP_MOD_REPLACE | LDAP::LDAP_MOD_BVALUES,
                        "userPassword",
                        [newp])

conn = LDAP::SSLConn.new(ldap_server, 389, true)
conn.set_option(LDAP::LDAP_OPT_PROTOCOL_VERSION, 3)
conn.bind(luser, oldp)
```

```
conn.modify(luser, [replace])
```

Apesar de não ter a garantia de estar livre de falhas de segurança (a senha é mantida na memória, por exemplo), isso é mais limpo e mais flexível do que um simples script `sh`.

4. Considerações de segurança

Agora que suas máquinas (e possivelmente outros serviços) estão autenticando em seu servidor LDAP, este servidor precisa ser protegido pelo menos tão bem quanto /etc/master.passwd seria em um servidor regular, e possivelmente mais ainda, uma vez que um servidor LDAP corrompido quebraria todos os serviços do cliente.

Lembre-se, esta seção não é exaustiva. Você deve revisar continuamente sua configuração e procedimentos para melhorias.

4.1. Definindo atributos somente leitura

Vários atributos no LDAP devem ser somente leitura. Se deixado gravável pelo usuário, por exemplo, um usuário poderia alterar seu atributo `uidNumber` para `0` e obter acesso ao `root`!

Para começar, o atributo `userPassword` não deve ser legível por todos. Por padrão, qualquer pessoa que possa se conectar ao servidor LDAP pode ler esse atributo. Para desabilitar isso, coloque o seguinte em slapd.conf:

Exemplo 8. Ocultar senhas

```
access to dn.subtree="ou=people,dc=example,dc=org"
  attrs=userPassword
  by self write
  by anonymous auth
  by * none

access to *
  by self write
  by * read
```

Isso não permitirá a leitura do atributo `userPassword`, enquanto ainda permite que os usuários alterem suas próprias senhas.

Além disso, você desejará impedir que os usuários alterem alguns de seus próprios atributos. Por padrão, os usuários podem alterar qualquer atributo (exceto aqueles que os próprios esquemas LDAP negam alterações), como `uidNumber`. Para fechar este buraco, modifique o acima para

Exemplo 9. Atributos somente leitura

```
access to dn.subtree="ou=people,dc=example,dc=org"
  attrs=userPassword
  by self write
  by anonymous auth
  by * none

access to attrs=homeDirectory,uidNumber,gidNumber
  by * read

access to *
  by self write
  by * read
```

Isso impedirá que os usuários se disfarçam como outros usuários.

4.2. Definição da conta `root`

Geralmente, a conta `root` ou a conta de administrador para o serviço LDAP será definida no arquivo de configuração. O OpenLDAP suporta isso, por exemplo, e funciona, mas pode causar problemas se o slapd.conf estiver comprometido. Pode ser melhor usar isto apenas para se autoinicializar no LDAP, e então definir uma conta `root`.

Melhor ainda é definir contas com permissões limitadas e omitir totalmente uma conta `root`. Por exemplo, os usuários que podem adicionar ou remover contas de usuário são adicionados a um grupo, mas não podem alterar a participação desse grupo. Essa política de segurança ajudaria a mitigar os efeitos de uma senha perdida.

4.2.1. Criando um grupo de gerenciamento

Digamos que você queira que seu departamento de TI possa alterar os diretórios pessoais dos usuários, mas não deseja que todos eles possam adicionar ou remover usuários. A maneira de fazer isso é adicionar um grupo para esses administradores:

Exemplo 10. Criando um grupo de gerenciamento

```
dn: cn=homemanagement,dc=example,dc=org
objectClass: top
objectClass: posixGroup
cn: homemanagement
gidNumber: 121 # required for posixGroup
memberUid: uid=tuser,ou=people,dc=example,dc=org
memberUid: uid=user2,ou=people,dc=example,dc=org
```

E então mude os atributos de permissões em slapd.conf:

Exemplo 11. ACLs para um grupo de gerenciamento de diretório inicial

```
access to dn.subtree="ou=people,dc=example,dc=org"
  attr=homeDirectory
  by dn="cn=homemanagement,dc=example,dc=org"
  dnattr=memberUid write
```

Agora `tuser` e `user2` podem alterar os diretórios home de outros usuários.

Neste exemplo, demos um subconjunto de poder administrativo a certos usuários sem dar a eles poder em outros domínios. A idéia é que em breve nenhuma conta de usuário tenha o poder de uma conta `root`, mas todo poder que root tem seja tido por pelo menos um usuário. A conta `root` torna-se desnecessária e pode ser removida.

4.3. Armazenamento de Senha

Por padrão, OpenLDAP armazenará o valor do atributo `userPassword` conforme ele armazena quaisquer outros dados: puro texto. Na maioria das vezes, ele é codificado na base 64, o que fornece proteção suficiente para impedir que um administrador honesto conheça sua senha, mas pouco ainda.

É uma boa idéia, então, armazenar senhas em um formato mais seguro, como o SSHA (salted SHA). Isso é feito por qualquer programa que você use para alterar as senhas dos usuários.

Apêndice A: Ajudas Úteis

Existem alguns outros programas que podem ser úteis, especialmente se você tiver muitos usuários e não quiser configurar tudo manualmente.

O security/pam_mkhomedir é um módulo PAM que sempre é bem-sucedido; Sua finalidade é criar diretórios pessoais para usuários que não os possuem. Se você tiver dezenas de servidores clientes e centenas de usuários, é muito mais fácil usar isso e configurar diretórios esqueletos do que preparar cada diretório inicial.

O sysutils/cpu é um utilitário do tipo pw(8) que pode ser usado para gerenciar usuários no diretório LDAP. Você pode chamá-lo diretamente ou encapsular os scripts em torno dele. Ele pode manipular tanto o TLS (com o sinalizador `-x`) quanto o SSL (diretamente).

O sysutils/ldapvi é um ótimo utilitário para editar valores LDAP em uma sintaxe semelhante a LDIF. O diretório (ou subseção do diretório) é apresentado no editor escolhido pela variável de ambiente `EDITOR`. Isso facilita a ativação de alterações em grande escala no diretório sem a necessidade de escrever uma ferramenta personalizada.

O security/openssh-portable tem a capacidade de contatar um servidor LDAP para verificar as chaves SSH. Isso é extremamente bom se você tiver muitos servidores e não quiser copiar suas

chaves públicas em todos eles.

Apêndice B: Certificados OpenSSL para o LDAP

Se você estiver hospedando dois ou mais servidores LDAP, provavelmente não desejará usar certificados auto-assinados, já que cada cliente precisará ser configurado para trabalhar com cada certificado. Embora isso seja possível, não é tão simples quanto criar sua própria autoridade de certificação e assinar os certificados de seus servidores com isso.

Os passos aqui são apresentados como eles são, com muito pouca tentativa de explicar o que está acontecendo - mais explicações podem ser encontradas em openssl(1) e aplicações iguais.

Para criar uma autoridade de certificação, simplesmente precisamos de um certificado e chave auto-assinados. As etapas para isso novamente são

Exemplo 12. Criando um Certificado

```
% openssl genrsa -out root.key 1024
% openssl req -new -key root.key -out root.csr
% openssl x509 -req -days 1024 -in root.csr -signkey root.key -out root.crt
```

Estas serão sua chave e certificado de CA raiz. Você provavelmente desejará criptografar a chave e armazená-la em um local seguro; qualquer pessoa com acesso a ele pode se passar por um dos seus servidores LDAP.

Em seguida, usando as duas primeiras etapas acima, crie uma chave ldap-server-one.key e a solicitação de assinatura de certificado ldap-server-one.csr. Depois de assinar o pedido de assinatura com root.key, você poderá usar o ldap-server-one.* nos servidores LDAP.

Não se esqueça de usar o nome de domínio totalmente qualificado para o atributo "common name" ao gerar a solicitação de assinatura de certificado; caso contrário, os clientes rejeitarão uma conexão com você e poderá ser muito complicado diagnosticar.

Para assinar a chave, use `-CA` e `-CAkey` em vez de `-signkey`:

Exemplo 13. Assinando como uma autoridade de certificação

```
% openssl x509 -req -days 1024 \
-in ldap-server-one.csr -CA root.crt -CAkey root.key \
-out ldap-server-one.crt
```

O arquivo resultante será o certificado que você pode usar em seus servidores LDAP.

Finalmente, para os clientes confiarem em todos os seus servidores, distribua root.crt (o *certificado*, não a chave!) para cada cliente, e especifique-o na directiva `TLSCACertificateFile` no ldap.conf.