Acesso à rede mais seguro com 802.1X, Radius e LDAP

Batendo a porta (na cara dos estranhos)

O protocolo Radius é normalmente usado para autenticar usuários em provedores de acesso discado. Mas o Radius também é útil em ambientes corporativos: combinado com o protocolo 802.1X, ele força os usuários a se autenticar primeiro e, só depois, o switch abre uma porta para eles.

por Michael Schwartzkopff

www.sxc.hu

Ataques originados de dentro da própria rede são mais perigosos e difíceis de detectar do que os vindos de fora. Um invasor que possa "plugar" seu laptop em uma rede pelo lado de dentro ganha acesso instantâneo a uma porção de dados e serviços sem precisar de autenticação. Mesmo os dados "protegidos" por algum tipo de autenticação – por exemplo, quando o servidor de arquivos pede um "login" – estão em grande perigo, já que uma rede bastante ativa é uma mina de jóias preciosas – basta farejá-la com um *sniffer* para obter, em poucas horas, as senhas de quase todos os usuários. Além disso, é tarefa trivial fazer testes para verificar a presença de mecanismos de defesa (firewalls, IDSs) e mais fácil ainda "detonar" literalmente a rede, deixando-a fora de serviço.

Uma maneira de prevenir tais malandragens é implementar uma função de autenticação na camada 2 do modelo OSI (ou seja, no nível Ethernet) usando o protocolo 802.1X [1]. Um switch que reconheça esse protocolo e um servidor Freeradius é tudo de que você precisa para se divertir. Como qualquer mecanismo de autenticação na camada 2 opera na rede local – ou seja, precisa de conexão física – um intruso não conseguirá usufruir dos recursos disponíveis, mesmo que esteja conectado diretamente, sem autenticação.

O nome Radius significa *Remote Authentication Dial-in User Service Protocol* (Protocolo de Autenticação Remota para Usuários de Serviços Discados). Quando implementado em um servidor Linux, normalmente devolve ao cliente requerente um número IP e um *gateway* padrão para o usuário, mas o protocolo tem muito mais potencial que isso. Pode-se usar um servidor Radius para atribuir uma VLAN (rede local virtual) à porta do switch em que o usuário está conectado. Essa técnica evita a necessidade de uma infraestrutura de roteamento muito complexa, mas ainda assim restringe o tamanho do domínio de *broadcast*. Além disso, VLANs podem ser usadas para separar virtualmente os departamentos de uma grande empresa, melhorando em muito a segurança interna – uma vantagem adicional muito importante. Embora os usuários possam autenticar-se no sistema onde quer que estejam (do refeitório, por exemplo), eles conseguem ver *apenas* os recursos de seu próprio departamento, como se estivessem sentados em suas próprias mesas.

O protocolo 802.1x já é previsto para tratar da autenticação de usuários, desde que um programa externo ofereça na rede os serviços "AAA" (*Authentication, Authorization and Accounting* ou *Autenticação, Autorização e Contabilidade*). O servidor Freeradius se encarrega disso, acessando um catálogo ("diretório") OpenLDAP que guarda as informações sobre as contas. Com tal aparato, fica fácil gerenciar sem traumas um número impressionante de usuários. O sistema é compatível com clientes Windows e Unix/Linux indiscriminadamente e permite implementar redundâncias para garantir alta disponibilidade, usando proxies para os servidores Radius e replicação de catálogos para os bancos de dados LDAP.

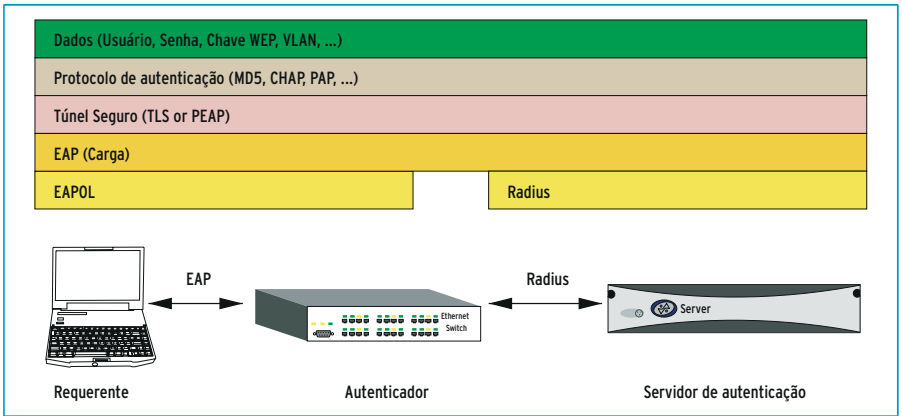

Figura 1: O protocolo 802.1X cobre múltiplas camadas. O EAP lida com a troca de dados entre o requerente e o autenticador; o Radius é responsável pela conexão entre o autenticador e o servidor de autenticação.

Com pouquíssimo esforço extra, a solução pode ser aplicada para garantir a segurança de redes sem fio (WLANs, ver quadro "o 802.1X e as redes sem fio"). Os administradores podem, ainda, se beneficiar das vastas opções de contabilidade disponíveis.

802.1X e EAP

O protocolo IEEE 802.1X oferece controle de acesso na camada 2 do modelo OSI (a camada MAC, normalmente ocupada pelo protocolo Ethernet). O IEEE 802.1X permite a autenticação de clientes enquanto estes estabelecem uma conexão com a rede. Isso garante que os indesejáveis sejam barrados antes mesmo de ganhar um endereço IP via DHCP (*Dynamic Host Configuration Protocol*). Entre outras coisas, o padrão especifica a forma como o protocolo de autenticação (EAP, *Extensible Authentication Protocol* ou Protocolo Extensível de Autenticação) é encapsulado no protocolo Ethernet, seja dentro do cabo ou da interface de rádio. É por isso que o 802.1X também é conhecido como EAPOL (*EAP over LAN*, EAP sobre LAN). Além do EAP, o 802.1X também trabalha com o protocolo PPP (*Point to Point Protocol*, ou Protocolo Ponto a Ponto). O PPP e o EAP são padrões da Internet (normatizados pelo IETF com documentos RFC), enquanto o 802.1X foi desenvolvido pelo IEEE (*Institute of Electrical and Electronics Engineers*).

O EAP é uma estrutura básica para vários métodos de autenticação, permitindo que sejam usadas mais informações do que a combinação usual de nome e senha. Ele usa um autenticador no padrão NAS – *Network Access Server* – para abrir um túnel até o servidor de autenticação através da rede. Isso permite que outros protocolos usem o túnel. O 802.1X define um grande número de termos para as entidades envolvidas no processo (ver **figura 1**):

- O cliente que requisita a autenticação é chamado de *Requerente* (*Supplicant*);
- O servidor que autentica o cliente é conhecido como *Servidor de Autenticação* (*Authentication Server*);
- O dispositivo intermediário entre essas duas entidades é o *Network Authentication Server* (NAS), também chamado de *Autenticador*.

Esse arranjo vai funcionar em qualquer rede na qual trafeguem pacotes Ethernet – sejam redes normais com cabos ou sem fio. O *white paper* dos Interopnet Labs dá uma bela introdução sobre o assunto [1].

Variações do EAP

O EAP define uma variedade grande de métodos de autenticação. O EAP/MD5 é o tradicional "digite seu nome e senha". O protocolo transfere um *hash* com o nome do usuário e uma semente arbitrária. O servidor usa uma senha em texto puro e a semente arbitrária para gerar seu próprio hash, que compara com o hash entrante. Esse método é simples de implementar, mas muito vulnerável a ataques de dicionário. Além disso, numa LAN sem fio, é importante criar chaves WEP usando EAP/MD5. Portanto, essa abordagem é apropriada apenas para pequenas redes comuns, com cabos de cobre.

Com a segunda variação, EAP/TLS, tanto o servidor como o cliente precisam de certificados X.509. Esse método é bem mais seguro, mas para que funcione é necessário que já exista uma infraestrutura de PKI (*Public Key Infrastructure* ou Infraestrutura de Chaves Públicas) na rede. O terceiro método mais comum é usando o PEAP (*Protected Extensible Authentication Protocol* ou Protocolo de Autenticação Protegido e Extensível). Com ele, apenas os servidores precisam do certificado; o protocolo usa o certificado para estabelecer uma conexão TLS e, por ela (que está criptografada), envia o nome e a senha do usuário. Internamente, usa-se o MSCHAPv2, *Microsoft Challenge Handshake Authentication Protocol*. Os administradores precisam, apenas, instalar o certificado dos servidores em cada cliente.

Quando o cliente sai do sistema ou encerra a conexão, o PEAP detecta a mudança e revoga a autorização. Isso deixa a conexão com um estado bem definido em ambos os lados: encerrado!

Dividindo a carga

Em redes exclusivamente de cobre, a combinação EAP/MD5 é quase sempre a melhor opção. É tudo de que se precisa para atribuir VLANs dinamicamente e – em contraste com o PEAP – é suportada por uma grande variedade de switches. Adicionalmente, o administrador tem que quebrar menos a cabeça para gerenciar o monstro, ao contrário dos complexíssimos PEAP e, em particular, EAP/TLS.

Um switch normalmente já possui funcionalidades de NAS, traduzindo o protocolo EAPOL (EAP over LAN) do requerente para o Radius, que é o que o servidor de autenticação espera. Muitos dispositivos oferecem essa opção quando configurados para 802.1X. Você precisará informar o endereço e a senha do servidor Radius. Em muitos casos, os administradores podem configurar múltiplos servidores para oferecer níveis bem altos de disponibilidade e backup.

O 802.1X e as redes sem fio.

O Freeradius, em combinação com o protocolo 802.1X, pode ser usado para implementar um esquema de autenticação em redes sem fio. Com esse arranjo, é possível trocar as fraquíssimas chaves WEP (*Wired Equivalent Privacy*) a cada meia hora para cada cliente. Mas isso requer que sejam instalados os protocolos PEAP (*Protected Extensible Authentication Protocol* ou Protocolo Extensível e Protegido de Autenticação) ou TLS, pois os hashes MD5 são incapazes de gerar chaves de criptografia. Embora não seja lá muito bem escrito, há um bom HOWTO sobre o assunto em [2].

Para poder usar o PEAP, o administrador precisa criar um certificado para o servidor Radius. Para clientes Windows®, o certificado precisa de um OID (*Object ID*) específico. Há um script chamado `CA.all` no subdiretório `scripts/` do diretório de códigos fonte do Freeradius que gera certificados de exemplo para servidores e clientes. Esses certificados são necessários *apenas* se os protocolos EAP/TLS forem usados.

Administração fácil com o TinyCA

Se você preferir uma ferramenta gráfica para gerenciar seus certificados, por que não testar o TinyCA [3], versão 0.6.4 ou posterior? Essa ferramenta permite que os administradores adicionem o OID apropriado (por exemplo 1.3.6.1.5.5.7.3.1 para o servidor) na caixa de diálogo avançada do certificado.

O próximo passo é copiar o novo certificado (o script cria um certificado chamado `srv-cert.pem`), bem como o certificado raiz do CA, `root.pem`, para o subdiretório `certs` dentro do diretório de configuração do servidor Radius. Para usar o TinyCA, é necessário também copiar o arquivo com a chave privada. Os usuários do Windows® precisam instalar o certificado `root.der`; basta clicar duas vezes sobre o arquivo. Isso também se aplica ao certificado do servidor Radius, `srv-cert.p12` (no formato PKCS#12).

Agora, ajuste as seções `tls` e `peap` da configuração do EAP no servidor Radius, conforme descrito em [2]. As linhas seguintes fazem parte da configuração padrão, mas estão desabilitadas. Se o Radius for executado em modo de depuração (`radiusd -X`), o serviço Radius mostrará na tela uma enxurrada de mensagens, o que pode ser de grande valia quando algo dá errado.

Listagem 1: Mapeamento de atributos

```
01 replyItem Tunnel-Type radiusTunnelType
02 replyItem Tunnel-Medium-Type radiusTunnelMediumType
03 replyItem Tunnel-Private-Group-Id radiusTunnelPrivateGroupId
```

Listagem 2: LDIF para a VLAN 2

```
01 dn:uid=vlan_02,ou=profiles,ou=radius,dc=domain,dc=br
02 uid: vlan_02
03 radiusTunnelMediumType: IEEE-802
04 radiusTunnelType: VLAN
05 radiusTunnelPrivateGroupId: 2
06 objectClass: radiusprofile
07 objectClass: top
```

Listagem 3: Referência à VLAN

```
01 dn:uid=testuser2,ou=users,ou=radius,dc=domain,dc=br
02 uid: testuser2
03 userPassword: password
04 objectClass: radiusprofile
05 objectClass: top
06 radiusProfileDn: uid=vlan_02,ou=profiles,ou=radius,dc=domain,dc=br
```

Freeradius

O Freeradius [4] é uma excelente escolha para servidor Radius. A versão 1.0 foi lançada recentemente e reconhece um grande número de EAPs, especialmente PEAP. Seu antecessor, o 0.93, não era lá muito compatível com quase nada, portanto foi um aprimoramento e tanto. Os desenvolvedores introduziram uma opção para autenticação contra um domínio Windows. E o Freeradius pode, ainda, obter as informações de autenticação de um grande número de fontes: o manjado `/etc/passwd` e mais LDAP, MySQL, PostgreSQL e Oracle.

A instalação é bastante simples e segue os três passos mágicos que todos conhecemos, basta digitar: `configure && make && make install` em um terminal. Se tudo correr bem durante a compilação, os arquivos de configuração estarão armazenados em `/usr/local/etc/raddb`. A primeira coisa de que o Freeradius precisa é a configuração para liberar – ou não – os clientes do Radius (em nosso caso, o NAS). O arquivo `clients.conf` cuida desse detalhe. A configuração para um switch com endereço IP estático `192.168.200.20` se parece com a mostrada a seguir: ➠

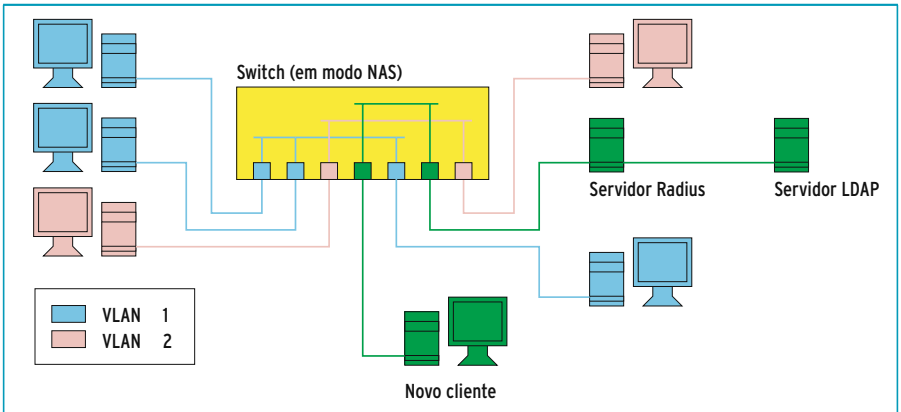

Figura 2: Novos clientes devem, primeiro, autenticar-se contra o NAS (ou seja, o switch), que age como um proxy para o sistema Radius. Este, por sua vez, consulta em um banco de dados LDAP se o usuário existe mesmo e quais recursos pode acessar.

```
client 192.168.200.20 {
   secret = testando123
   shortname = comutador03
}
```

O administrador precisa informar a senha do switch (*secret*, no exemplo anterior) na configuração do servidor Radius. O Freeradius aceita a notação CIDR para redes inteiras: 192.168.200.0/24.

Versões mais antigas do Freeradius pediam a definição de um dicionário, `dictionary.tunnel`, para os valores de retorno da VLAN. Na versão 1.0 isso não é mais necessário: a configuração do EAP não usa o arquivo central de configuração, `radiusd.conf`; em vez disso há um `eap.conf` que usa MD5 por padrão – mas que pode ser modificado.

Autenticando Usuários

O arquivo `users` especifica o tipo de autenticação a ser usado para controlar o acesso pelos usuários. A linha a seguir deve servir para o teste inicial:

```
testuser Auth-Type := Local, ➥
User-Password == "senha"
  Reply-Message = "Bem-vindo, %u"
```

O servidor Radius deve ser colocado em modo de depuração. O comando `radiusd -X` ordena ao servidor que mostre, na tela, toda e qualquer mensagem de erro ou aviso. O `radtest` é um cliente de testes bastante útil para pôr à prova suas configurações iniciais.

Com os ajustes mencionados até agora, emita o seguinte comando:

```
radtest testuser senha localhost ➥
0 testando123.
```

Os últimos três parâmetros referenciam o servidor Radius – para que isso funcione, é preciso adicionar `localhost` ao arquivo `clients.conf` para que seja reconhecido como um cliente válido do Radius. O Freeradius deve responder com a mensagem *Access-Accept*.

Colocando uma VLAN na resposta do Radius

É claro que o protocolo pode dar a você mais do que uma simples mensagem de resposta. Mais que isso, é possível fazer com que o programa arbitre o número de uma VLAN, por exemplo, permitindo que o switch interprete esse número e aceite (ou rejeite) o cliente nessa VLAN. Para que isso aconteça, é necessário modificar a resposta dada pelo Radius para o usuário de teste. Observe que quaisquer valores de retorno têm que ser separados por vírgulas e indentados com espaços:

Listagem 4: Configuração do Radius

```
01 modules {
02   ldap {
03     server = "ldap.domain.br"
04     identity = "cn=freeradius,ou=admins,ou=radius,dc=domain,dc=br"
05     password = senhasenha
06     basedn = "ou=users,ou=radius,dc=domain,dc=br"
07     filter = "(&(uid=%{Stripped-User-Name:-%{User-Name}})(objectclass=radiusprofile))"
08     start_tls = no
09     dictionary_mapping = ${raddbdir}/ldap.attrmap
10     password_attribute = userPassword
11   }
12 }
13 authorize {
14   preprocess
15   ldap
16   eap
17   suffix
18   files
19 }
20 authenticate {
21   eap
}
```

```
testuser Auth-Type := Local, ➥
User-Password == "senha"
  Reply-Message = "Bem-vindo, %u",
  Tunnel-Medium-Type = IEEE-802,
  Tunnel-Private-Group-Id = 1,
  Tunnel-Type = VLAN
```

Com isso, dizemos ao switch que ele precisa associar a VLAN 1 ao usuário `testuser` depois de autenticado. Isso é praticamente tudo. É óbvio que isso funciona bem com um número reduzido de usuários: o administrador pode simplesmente adicionar configurações manualmente para os usuários restantes. Em redes maiores, entretanto, a coisa fica impraticável.

OpenLDAP

O OpenLDAP [5] é uma boa escolha para o banco de dados do Freeradius. A arquitetura para essa implementação é mostrada na **figura 2**. Dusty Doris escreveu um excelente tutorial sobre como combinar o Freeradius e o OpenLDAP, que pode ser lido em [6]. A documentação do Freeradius também possui um exemplo disso, mais precisamente em `doc/rlm_ldap`.

Entretanto, há uma "pegadinha" no esquema do LDAP, `Radius-LDAPv3.schema` (também presente em `doc`). Esse esquema não é estrutural, ou seja, só funciona em conjunto com outro esquema, como por exemplo o `inetorgperson.schema`. Já o esquema alternativo disponível em [6] foi criado para ser usado especialmente com o Radius, mas não compreende os tipos de extensões que se usa no dia-a-dia.

Para que o Freeradius entenda as respostas para as VLANs devolvidas pelo LDAP, precisamos fazer mapeamentos (`ldap.attrmap`) na configuração do Radius (ver **listagem 1**).

A figura 3 mostra a estrutura do catálogo no LDAP. Os perfis (`ou=profiles`) contêm as configurações das VLANs; a **listagem 2** mostra o arquivo LDIF para a `uid=vlan_02` (veja a primeira linha). Para a configuração dos usuários (`ou=users`), precisamos simplesmente referenciar o perfil do usuário com o perfil de VLAN ao qual ele pertence (**listagem 3**, linha 6). Agora, basta informar ao servidor Radius que ele pode acessar os serviços do LDAP para gerenciamento de usuários. O arquivo de configuração `radiusd.conf` dá conta disso.

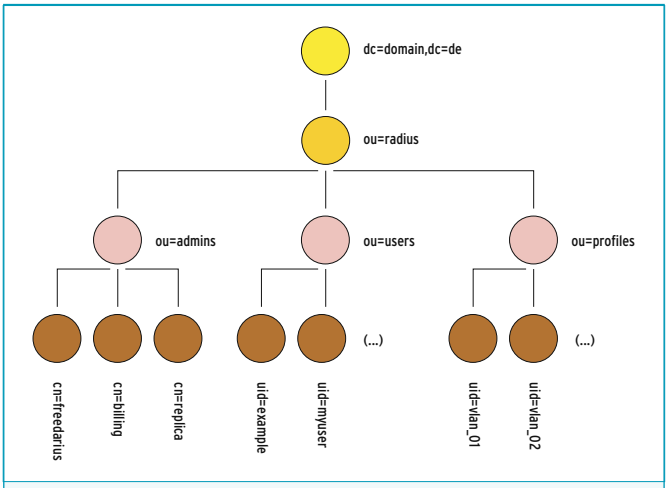

Figura 3: Essa estrutura de LDAP contém os perfis do usuário e configurações para as VLANs. Em cada um dos perfis há uma chave `radiusProfileDn` que aponta para a configuração correta de VLAN.

O exemplo na **listagem 4** ordena ao Radius que se inscreva no servidor LDAP (`ldap.domain.br`, linha 3) com a identidade e a senha configuradas (`identity` e `password`, respectivamente, linhas 4 e 5) e então autentique o usuário `filter` com a senha `password_attribute` (linhas 7 e 10). Se isso funcionar, o servidor LDAP devolverá os parâmetros do perfil adequado ao usuário (mapeado como `radiusProfileDn`).

Um novo item `DEFAULT` no gerenciamento de usuários do Radius (arquivo `users`) garante que as tentativas de autenticação usando EAP terão sucesso:

```
DEFAULT Auth-Type == EAP
   Fall-Through = yes
```

Ainda precisamos preparar os clientes para a autenticação via protocolo 802.1X. O projeto Open 1X [7] desenvolveu um software para Linux que trata exatamente disso. O Windows® 2000 SP4 e o Windows® XP SP1 têm, ambos, funções nativas de autenticação para esse propósito. O Windows® 2000 precisa, entretanto, que o serviço `Wireless Configuration` esteja ativo.

Clientes Windows

Há um campo chamado `Authentication` nas propriedades de conexão de rede. Nele, escreva `EAP` com o tipo apropriado (`MD5`, `PEAP` or `TLS`). Os usuários podem especificar como o sistema deve responder se as credenciais dos usuários não estiverem disponíveis – por exemplo, quando ainda não houver sido dado o login. Ao selecionar essa opção, o computador tentará se registrar na rede usando o nome da máquina.

Em contraste com a opção EAP/MD5, o PEAP e o TLS oferecem opções adicionais. Os usuários precisam especificar o certificado CA a ser usado. O Windows® também pode usar o nome e a senha locais (ou seja, do próprio Windows) para entrar no sistema com autenticação PEAP – assim, não é preciso "se logar" duas vezes. Essa facilidade está localizada nas opções avançadas de autenticação.

Entretanto, nesse caso o sistema usa uma combinação `Domínio/Usuário`. Se não estiver certo disso, a variante exata será registrada nos logs do servidor Freeradius depois da primeira tentativa de login. Para entender esse formato, o Radius precisa de algumas dicas. Se você tem interesse em aprender mais sobre o assunto, experimente o livro sobre Radius em [8].

Atrasadinhos...

Os clientes 802.1X da Microsoft têm todos um problema crônico. Eles primeiro se inscrevem ("logam-se") em seu próprio domínio e só depois se autenticam na rede. Mas, como a rede não está disponível no momento que ele se "loga" no domínio, a tentativa de login falha e causa um erro. O servidor de domínio tem que estar, então, numa VLAN aberta ao público – em outras palavras, um atentado aos pudores básicos de segurança.

Há algumas ferramentas de terceiros que tentam resolver o problema, colocando na rede requerentes 802.1X adicionais. Esses clientes tipicamente permitem uma configuração bem mais granular – permitindo que os clientes Windows® sejam integrados ao processo de login. Este começa autenticando os usuários contra a rede baseada no protocolo 802.1X e, só depois disso, permite a autenticação normal no domínio Windows®. Entretanto, alguns programas ainda não estão totalmente lixados e polidos, portanto teste bastante antes de comprar.

Clientes e Servidores para Linux

Para Linux há uma implementação do protocolo 802.1X que podemos considerar como "bem madura", o *open1x* [7]. O projeto, que conta com o apoio comercial de empresas como a 3Com e a HP, atingiu a versão 1.0 há pouco tempo e possui uma vasta documentação. Com ele, deve ser bem fácil integrar seus computadores com Linux à rede.

O servidor Freeradius possui uma gama enorme de opções para autenticação e controle de acesso baseado nos requisitos técnicos de cada rede. O exemplo que demonstramos neste artigo é apenas uma das muitas abordagens possíveis. Pelo fato de usar um serviço de catálogo LDAP para gerenciamento de usuários, o projeto de segurança também é apropriado para redes bastante grandes em que o risco de ataques originados de dentro é assustadoramente alto. ■

Informações

[1] Artigo sobre o protocolo 802.1X – Interopnet Labs, "What is 802.1X?": http://www.ilabs.interop.net/WLANSec/What_is_8021x-lv03.pdf

[2] Freeradius e o Windows XP: http://text.broadbandreports.com/forum/remark,9286052~mode=flat

[3] TinyCA: http://tinyca.sm-zone.net/

[4] Freeradius, servidor Radius Open Source: http://www.freeradius.org/

[5] OpenLDAP: http://www.openldap.org/

[6] Freeradius e o OpenLDAP: http://doris.cc/radius/

[7] Implementação livre e aberta do 802.1X: http://www.open1x.org/

[8] *Radius – Securing Public Access to Private Resources* Jonathan Hassell; O'Reilly, 2002. http://www.oreilly.com/catalog/radius/index.html

Sobre o autor

Michael Schwartzkopff trabalha para a Multinet Services GmbH, na Alemanha, como consultor de segurança e redes. Sua especialidade é o protocolo SNMP. Ele pegou a "gripe do Linux" lá atrás, em 1994, depois de trabalhar com a Yggdrasil, uma das primeiras distribuições Linux.