infra MAGAZINE

Edição 04 - Ano 01

DevMedia

Alta Disponibilidade e Recuperação de Desastres
A continuidade dos serviços de TI em casos de falhas

Autenticação centralizada com OpenLDAP
Controle de logins em uma única base de usuários

Windows 2008 Server Core
Uma nova opção de instalação do Windows Server

desvendando a criptografia

Usando o Forefront TMG
O Gateway de Internet segura da Microsoft

FreeBSD com proxy Squid
Controle, filtre e agilize o tráfego da rede

FÓRUM DEVMEDIA

O lugar perfeito para você ficar por dentro de tudo o que acontece nas tecnologias do mercado atual



No novo Fórum da DevMedia você vai encontrar canais específicos de Delphi, ASP.NET, Java, Banco de Dados e Engenharia de Software; além de ter contato com profissionais qualificados da área para troca de informações, sugestões e muito mais.

ACESSE AGORA
www.devmedia.com.br/forum

DevMedia

Sumário

Olá, eu sou o **DevMan**! Desta página em diante, eu estarei lhe ajudando a compreender com ainda mais facilidade o conteúdo desta edição. Será um prazer contar com sua companhia! Confira abaixo o que teremos nesta revista:



Infra

06 – Alta Disponibilidade e Recuperação de Desastres
A continuidade dos serviços de TI em casos de falhas
[*André Leon S. Gradvohl*]

Infra, Segurança

09 – Windows 2008 Server Core
Uma nova opção de instalação do Windows Server para administradores de rede
[*Edson Xavier*]

Segurança

18 – Desvendando a criptografia
Origem, evolução e características atuais dos métodos utilizados para transmissão de informações confidenciais
[*André Koide da Silva*]

Infra, Segurança

27 – Usando o Forefront Threat Management Gateway
O Gateway de Internet segura da Microsoft
[*Uilson Souza*]

Infra, Web, Segurança

33 – FreeBSD com proxy Squid
Controle, filtre e agilize o tráfego da rede com FreeBSD e Squid
[*Rogério Tomassoni de A. Junior*]

Segurança, Infra

42 – Autenticação centralizada com OpenLDAP
Controle de logins em uma única base de usuários
[*Flávio Alexandre dos Reis e Eduardo Pagani Julio*]

[Infra] Artigo que apresenta tecnologias relacionadas a questões de Infraestrutura.

[Segurança] Artigo dentro do contexto de Segurança: redes, servidores, sistemas operacionais, infraestrutura em geral.

[Web] Artigos sobre ou que envolvam tecnologias de infraestrutura para WEB.

infra MAGAZINE

Ano I • Edição 04 • 2012

Edição

Editor

Eduardo Spínola (*eduspinola@gmail.com*)

Produção

Jornalista Responsável Kaline Dolabella - JP24185

Atendimento ao leitor

A DevMedia possui uma Central de Atendimento on-line, onde você pode tirar suas dúvidas sobre serviços, enviar críticas e sugestões e falar com um de nossos atendentes. Através da nossa central também é possível alterar dados cadastrais, consultar o status de assinaturas e conferir a data de envio de suas revistas. Acesse **www.devmedia.com.br/central**, ou se preferir entre em contato conosco através do telefone 21 3382-5038.

Publicidade

publicidade@devmedia.com.br – 21 3382-5038

Anúncios – Anunciando nas publicações e nos sites do Grupo DevMedia, você divulga sua marca ou produto para mais de 100 mil desenvolvedores de todo o Brasil, em mais de 200 cidades. Solicite nossos Media Kits, com detalhes sobre preços e formatos de anúncios.

Fale com o Editor!

É muito importante para a equipe saber o que você está achando da revista: que tipo de artigo você gostaria de ler, que artigo você mais gostou e qual artigo você menos gostou. Fique a vontade para entrar em contato com os editores e dar a sua sugestão!

Se você estiver interessado em publicar um artigo na revista ou no site Easy Java Magazine, entre em contato com o editor, informando o título e mini-resumo do tema que você gostaria de publicar:

Eduardo Spínola - Editor da Revista
eduspinola@gmail.com

DevMedia

Editorial

Muito antes da existência do primeiro computador pessoas já se preocupavam em como guardar suas informações de forma que nem todos que tivessem acesso a elas conseguissem compreender o seu significado. Desde então, as técnicas para dificultar a leitura de informações amadureceram significativamente, assim como a importância de manter os dados criptografados, afinal, nenhuma empresa quer compartilhar seus dados com empresas concorrentes.

Assim, **Desvendando a criptografia** descreve os fundamentos da criptografia simétrica e assimétrica, os diferentes algoritmos conhecidos e os desafios computacionais relacionados ao processamento de suas operações matemáticas, bem como apresentar os certificados digitais e a infraestrutura de chave pública criada para o gerenciamento e a distribuição destes certificados.

Em **Alta Disponibilidade e Recuperação de Desastres** serão detalhados dois conceitos importantes para garantir a prestação contínua de serviços de TI: Alta Disponibilidade e Recuperação de Desastres.

Windows 2008 Server Core retrata como o modo Core, novo método de instalação do Windows Server 2008 e que possui funções limitadas, pode facilitar tarefas de administração e gerenciamento em ambientes que não necessitam de muitos recursos de servidor.

O artigo **Usando o Forefront Threat Management Gateway** apresenta o Forefront TMG, uma solução que aborda diversas questões de segurança e mantém a produtividade da corporação, sendo útil a profissionais de infraestrutura interessados em um produto de qualidade e com baixo custo.

Já **FreeBSD com proxy Squid** demonstra os procedimentos de instalação e configuração do Squid no sistema operacional FreeBSD. Após tais procedimentos, o Squid será responsável por controlar e gerenciar todos os acessos à internet, oferecendo inúmeras vantagens na administração de uma rede de computadores.

Por fim, **Autenticação centralizada com OpenLDAP** discorre sobre uma introdução ao uso do LDAP, apresentando sua origem, desenvolvimento e principais características do protocolo. Será abordado também como funciona a utilização dos serviços de diretórios como base central de autenticação a várias aplicações de rede.



Eduardo Oliveira Spínola
eduspinola@gmail.com
twitter.com/Java_Magazine

Dificuldades em encontrar conteúdo sobre .Net para iniciantes?



A DevMedia simplifica pra você!

easy .net magazine

Easy .Net Magazine:

A primeira revista digital feita especialmente para quem está dando os primeiros passos em .Net!

Você encontrará exemplos e modelos fáceis e práticos, noções introdutórias e conceitos básicos.

Conheça as vantagens:

- Vídeos e apresentações "embutidos" nas páginas;
- Ilustrações, diagramas e modelos apresentados em PowerPoint;
- Entrevistas em vídeo;
- Códigos-fonte que podem ser copiados com CTRL+C/CTRL+V.

A Easy.Net não é feita em PDF ou Flash!
Ela utiliza um aplicativo 100% online.



DevMedia group

Acesse agora a edição Nº 1 totalmente grátis e comprove

www.devmedia.com.br/easynet

Easy.Net Magazine, seu curso inicial de .Net!

Alta Disponibilidade e Recuperação de Desastres

André Leon S. Gradvohl

Nas duas últimas décadas, especialmente depois que a internet se tornou mais comercial, as informações têm aumentado em quantidade e em valor. Não bastasse isso, percebe-se que usuários e instituições têm investido grandes volumes de recursos financeiros para tornar essas informações acessíveis de forma rápida e dinâmica, isto é, disponíveis no momento em que se deseja e no dispositivo escolhido.

Ter acesso a essas informações no instante em que se deseja não é tão trivial quanto se imagina. Especialmente em instituições onde a informação é um ativo de grande valor, como as instituições financeiras e governamentais, por exemplo. Nessas instituições, perder informações pode trazer consequências incalculáveis, tanto do ponto de vista financeiro quanto político, ou seja, em termos de reputação.

Neste artigo serão detalhados dois conceitos importantes para garantir a prestação contínua de serviços de TI: Alta Disponibilidade (*High Availability* – HA) e Recuperação de Desastres (*Disaster Recovery* – DR). Define-se alta disponibilidade como os atributos de um sistema para prover serviços durante períodos definidos, em níveis aceitáveis ou acordados e mascarar interrupções não planejadas dos usuários finais. Na recuperação de desastres, por outro lado, o foco está em assegurar a capacidade de recuperação do sistema até mesmo na presença de falhas de múltiplos componentes. Portanto, são conceitos complementares.

Alta disponibilidade

De acordo com o *Service Availability Forum*, serviços ou aplicações de alta disponibilidade são aquelas que permanecem por 99,999% do tempo disponíveis (os cinco noves de disponibilidade). Considerando o tempo de um ano, com previsão de funcionamento de 7 dias por semana e 24 horas por dia, 99,999% significam que o sistema estaria desligado por 5 minutos e 35 segundos aproximadamente. É difícil atingir esse grau de alta disponibilidade, mas outros níveis – quatro ou três noves – também são aceitáveis. A **Tabela 1** resume esses níveis.

Para garantir a alta disponibilidade é necessário que o sistema não apresente um ponto único de falhas, isto é, devem existir software, dados e equipamentos redundantes o suficiente para permitir uma transição (*failover*) transparente de sistemas quando necessário. Entretanto, não é somente isso que basta. É preciso ter também ferramentas, procedimentos e habilidades para fazer a transição transparente. Aliás, em relação aos procedimentos, quanto mais padronizados, mais fácil é sua automatização.

Entre os requisitos para se implementar a alta disponibilidade estão a duplicação de centros de processamento de dados (CPD) em locais geograficamente distantes,



Nível de disponibilidade	Tempo "no ar"	Tempo máximo "fora do ar" por ano
Cinco noves	99,999%	5 minutos e 35 segundos
Quatro noves	99,99%	52 minutos e 33 segundos
Três noves	99,9%	8 horas e 46 minutos
Dois noves	99%	87 horas e 36 minutos
Um nove	90%	36 dias e 12 horas

Tabela 1. Os níveis de disponibilidade

espelhamento ou replicação dos dados entre os diferentes CPDs, sistemas de armazenamento seguro contra incidentes e altos níveis de segurança lógica e física de dados nos CPDs. Note que esses requisitos afetam outras características importantes dos sistemas que não devem ser negligenciadas, tais como a escalabilidade, flexibilidade, adaptabilidade e capacidade de recuperação.

Com as replicações e redundâncias exigidas pela alta disponibilidade, os sistemas se tornam menos escaláveis e menos flexíveis. Em outras palavras, qualquer demanda por aumento em sua capacidade de armazenamento ou de processamento, ou ainda qualquer mudança nos requisitos de software, implicará em investimentos dobrados ou triplicados em equipamentos ou sistemas. Portanto, é preciso cautela e um projeto muito bem concebido para evitar mudanças muito drásticas ou frequentes.

Recuperação de desastres

No conceito de recuperação de desastres, por sua vez, é preciso assegurar que a infraestrutura de suporte aos dados espalhada nos diversos sítios é consistente. Assim, em caso de falhas nos diversos componentes localizados em um sítio, um CPD em outro sítio poderá assumir as transações de forma transparente.

É preciso destacar que, diferente do conceito de alta disponibilidade, em uma situação de recuperação de desastres uma pequena perda de dados transacionais é aceitável. Dados transacionais são aqueles que descrevem uma transação (inserção, remoção, alteração) no banco de dados. Essa perda de dados transacionais é facilmente recuperável através dos *logs* gerados pelo sistema de bancos de dados. Em razão dessa característica, é possível considerar a replicação assíncrona de dados em ambientes adequados para a recuperação de desastres.

Replicação de dados em curta distância e em longa distância

Tanto no conceito de alta disponibilidade, quanto em recuperação de desastres, a replicação dos dados é fundamental para a continuidade dos serviços. Quanto menor a distância entre os sítios que mantêm os dados, maior é a velocidade de atualização dos dados em locais diferentes. Por outro lado, quanto mais próximos entre si, maior é a probabilidade de um evento indesejado atingir ambos os locais. Exemplos de eventos indesejados podem incluir desde a falta de energia prolongada (o temido apagão) até eventos da natureza como terremotos, enchentes e excesso de ventos.

Nesse texto, considera-se curta a distância entre 20 e 30 km. Acima disso, considera-se longa distância. A replicação em curta distância provê proteção suficiente para os requisitos de recuperação de desastres, é mais barata e mais simples de se implementar em comparação às soluções de longa distância.

Com a replicação síncrona, as atualizações feitas pela aplicação em um sistema de armazenamento em um sítio são – quase – simultaneamente feitas em outro sistema em local remoto. A resposta à atualização somente é fornecida pela aplicação quando os dados estão seguros em ambos os locais.

As replicações de dados em longas distâncias e assíncronas são mais indicadas para recuperação de desastres. A principal razão para isso é que se torna difícil e mais lento manter o sincronismo entre as transações quando os sítios estão distantes entre si. Ademais, conforme mencionado antes, em recuperação de desastres, uma pequena perda de dados transacionais é tolerável.

Topologia de CPD que combina Alta Disponibilidade e Recuperação de Desastres

Para combinar os conceitos de HD e DR, uma solução bastante comum para aplicações críticas é a topologia 3-sítios. Essa topologia permite algumas variações em função da independência entre os sítios. Considera-se um sítio independente dos demais quando existe uma independência de fornecimento de energia, sistemas de ar condicionado, sistemas de *backup*, conectividade das redes, segurança física e equipamentos, entre outros.

Uma das variações mais simples, descrita na **Figura 1**, mantém dois sítios (primário e secundário) em um único local físico, por exemplo, uma sala em um prédio; e um terceiro sítio em um local remoto. Os sítios primário e secundário compartilham a mesma infraestrutura física, mas possuem sistemas (hardware, redes, discos) redundantes. O sítio terciário, independente, permanece em um local remoto para prover recuperação de desastres, caso necessário.

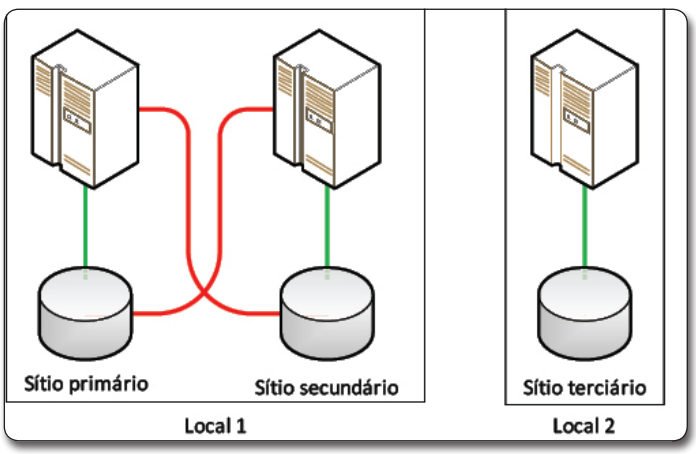


Figura 1. Variação 1 da Topologia 3-sítios

Uma segunda variação da topologia 3-sítios é ilustrada na **Figura 2**. Esta variação, baseada na primeira, considera também que os sítios primário e secundário estão no mesmo local, porém bem separados (cerca de 500 metros de distância entre eles). Além disso, contam com infraestruturas diferentes, por exemplo, em salas isoladas no mesmo prédio, com fornecimento de energia e conectividade independentes.

Na terceira variação da topologia, os três sítios são remotos e totalmente independentes uns dos outros, conforme mostrado na **Figura 3**. A alta disponibilidade é provida entre os sítios primário e secundário, enquanto o sítio terciário provê a recuperação de desastres. Nessa situação, a latência é um fator a ser considerado, pois a distância entre os sítios primário e secundário influencia o sincronismo das operações.

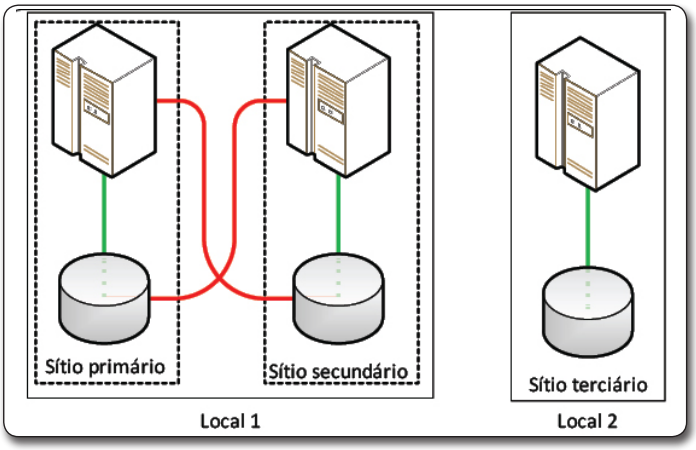


Figura 2. Variação 2 da Topologia 3-sítios

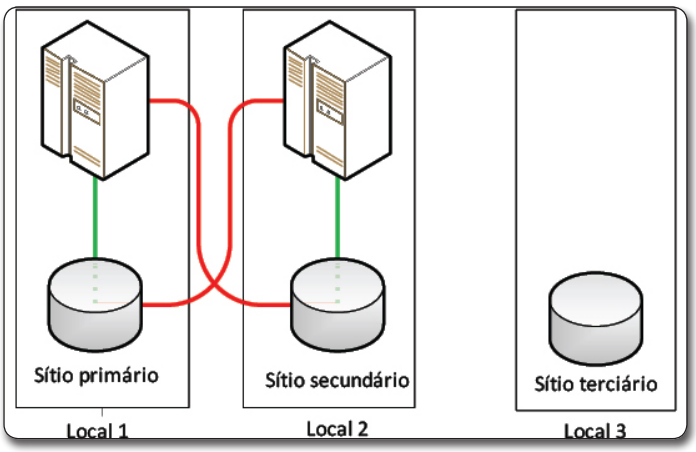


Figura 3. Variação 3 da Topologia 3-sítios

Existe ainda uma quarta variação, baseada na terceira, na qual o sítio terciário possui apenas o sistema de armazenamento de dados (*storage*), conforme descrito na **Figura 4**. Em caso de falha, o *storage* é acoplado ao sítio – primário ou secundário – que sofreu a pane. Essa quarta variação é mais barata, porém o tempo necessário para a recuperação é bem maior que as demais variações.



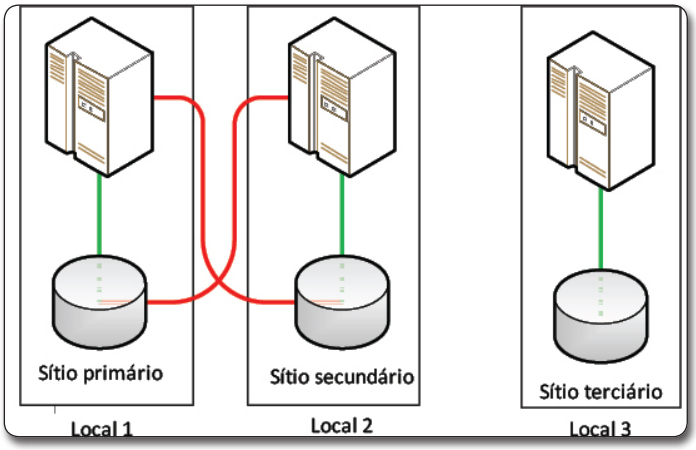


Figura 4. Variação 4 da Topologia 3-sítios

Conclusões

A continuidade dos serviços providos por uma instituição reflete diretamente na sua reputação e no seu valor de mercado. Por isso, é importante investir na disponibilidade dos servidores, bem como na rápida recuperação em caso de desastres. Entretanto, essas ações requerem planejamento. Afinal, nem todos os serviços são essenciais para a continuidade dos negócios. A redundância dos equipamentos e a topologia são os primeiros aspectos a serem considerados na implementação de alta disponibilidade e recuperação de desastres.



André Leon S. Gradvohl
gradvohl@ft.unicamp.br
É doutor em Telecomunicações, mestre e bacharel em Computação. Atua como professor e pesquisador da Faculdade de Tecnologia (FT) da Universidade de Estadual Campinas (Unicamp). Também é membro do Laboratório de Simulação e Computação de Alto Desempenho (LaSCADo) e do Grupo de Engenharia da Informação e Conhecimento (GEICon), ambos sediados na FT/Unicamp. É responsável pelo projeto "Metalinguagem para processamento de alto desempenho em arquiteturas híbridas" financiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP - 2011/00861-0).

Livro "Business Continuity and Disaster Recovery Planning for IT Professionals". Escrito por Susan Snedaker. Editora Syngress, 2007.

Livro "Crisis Management: Mastering the Skills to Prevent Disasters (Harvard Business Essentials)". Editora Harvard Business Press, 2004.

Normas ISO/IEC 27001 (Information Security Management System) e ISO/IEC 27002 (Information Technology - Security Techniques)
http://17799.standardsdirect.org

Windows 2008 Server Core

Uma nova opção de instalação do Windows Server para administradores de rede

Edson Xavier

Resumo DevMan

De que se trata o artigo:

Este artigo demonstra como o Windows 2008 Server Core pode facilitar o gerenciamento não só em escritórios remotos, que geralmente são as filiais de uma empresa, como em ambientes onde são demandados poucos recursos.

Em que situação o tema é útil:

Para profissionais que buscam segurança e estabilidade no seu ambiente de TI, reduzindo tarefas administrativas que antes demandavam muito tempo. Este artigo apresenta como o Server Core pode ser útil nesses casos.

Windows 2008 Server Core:

Este artigo aborda como o modo Core, novo método de instalação do Windows Server 2008 e que possui funções limitadas, pode facilitar tarefas de administração e gerenciamento em ambientes que não necessitam de muitos recursos de servidor, como ambientes que possuem servidores secundários de uma aplicação. São apresentados os principais recursos e funções que este modo de instalação permite e também os principais comandos a serem executados, pois não existe a opção de interface gráfica como no modo completo.

O Windows 2008 Server Core é um novo modo de instalação do Microsoft Windows 2008, o Sistema Operacional para servidores da Microsoft. Apesar de a instalação ser feita no modo gráfico, toda administração e gerenciamento do servidor é realizada através de linhas de comandos no console. A principal finalidade do Server Core é permitir a instalação do ambiente usando o mínimo de recursos, ou seja, somente o que é necessário para a utilização do sistema, diminuindo-se com isso, as tarefas administrativas dos gerentes de redes e servidores.

O objetivo deste artigo é apresentar como o Server Core facilita o gerenciamento e reduz tarefas que antes tomavam tempo dos administradores de rede como aplicação de patches de segurança, que no caso do Server Core diminuíram bastante; e configurações de Firewall, visto que no modo Core o Sistema Operacional é mais fechado e seguro. No entanto, não é aconselhável ter todo o parque de servidores com essa opção de instalação, pois muitos aplicativos utilizam da interface gráfica para funcionar. Por exemplo, o Windows SharePoint ou Windows Project Server.

Usando o Server Core como opção de instalação, a redução na manutenção desses servidores é considerável, pois apenas algumas funções estão disponíveis para serem utilizadas. São exemplos desses serviços o Active Directory Domain Services (AD DS), o Active Directory Lightweight Directory Services (AD LDS), o Dynamic Host Configuration Protocol (DHCP), o Domain Name System (DNS), o servidor de arquivos, o servidor de impressão, o Windows Media Services, o Hyper-V e o servidor Web. Esses serviços tornam esse Sistema Operacional ideal para escritórios remotos como, por exemplo, filiais que requerem poucos funcionários e não necessitam de muitos recursos. Na maioria das vezes, essas estruturas físicas podem ser reduzidas e não demandam muitos investimentos em hardware e gerenciamento de TI. Por isso, não se faz necessário profissionais com grande experiência em TI nessas estruturas visto que o Server Core permite que todo gerenciamento remoto seja feito através de consoles onde podem ser feitos ajustes e monitoramento de desempenho nos sistemas.

O Server Core

O Windows 2008 Server Core vem como um acréscimo muito atraente, diferente do modo completo de instalação, pois não contém a interface gráfica como padrão. As restrições em suas funcionalidades aumentam o nível de segurança do Sistema Operacional, pois reduz a possibilidade de ataques devido a suas funções reduzidas, principalmente na parte gráfica, onde pode conter diversas brechas de segurança. Não se trata de uma nova versão do Windows Server 2008. Por ser um modo de instalação com funções limitadas, fornece um servidor de baixa manutenção, que não exige muita intervenção técnica devido a apresentar poucas vulnerabilidades. O Windows 2008 Server Core encontra-se presente em todas as versões do Windows Server 2008: Standard, Enterprise e Datacenter.

Ao optar pela instalação do Windows Server 2008 no modo Core, conforme se observa na **Figura 1**, a maioria das empresas escolherão a versão Standard porque as principais diferenças encontradas nas funcionalidades das edições Enterprise e Data Center não estarão presentes no Server Core, pois se trata de um Sistema Operacional com recursos limitados.

Sendo assim, a edição mais utilizada no mercado é a Standard, que é ideal para ambientes simples que não necessitam de

muita expansão em hardware, suportando até 32 GB de memória e 4 processadores de 64 bits. A edição Enterprise permite que sejam utilizados até 8 processadores de 64 bits e 2 Terabytes de memória, bem como o recurso Clustering. Esse recurso é usado quando se tem necessidade de balancear a carga de trabalho entre dois ou mais servidores, ideal para ambientes críticos que necessitam de um sistema confiável. A edição Data Center, por sua vez, também é ideal para este tipo de ambiente, com a vantagem de trabalhar com virtualização em larga escala que traz como característica a escalabilidade, permitindo a expansão de hardware suportando até 64 processadores e 2 Terabytes de memória. A edição Data Center atende as necessidades de trabalho em larga escala como, banco de dados, e consolidação de servidores com o uso da virtualização.

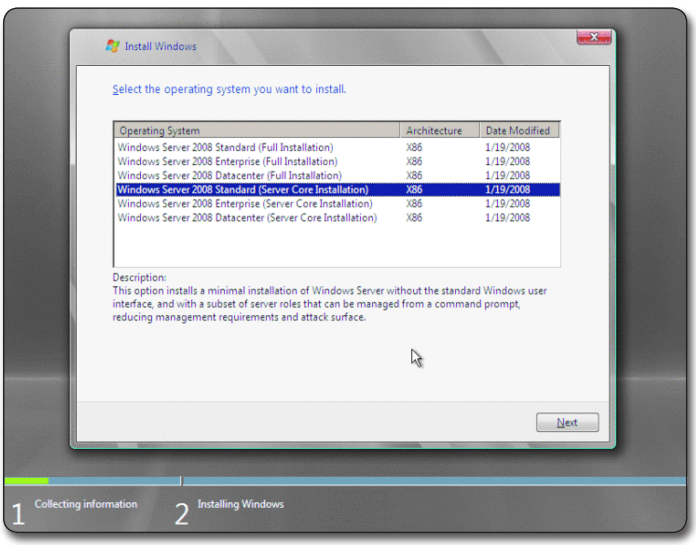


Figura 1. Modo de instalação Server Core

O Windows 2008 Server Core também pode ser utilizado em organizações onde o parque de servidores é muito grande e que exigem que certas tarefas sejam dedicadas, não sendo necessária a instalação completa que carrega diversos recursos que não serão totalmente utilizados como, por exemplo, a interface gráfica. Possui ainda como objetivo a redução do gerenciamento contínuo, atenuando o esforço administrativo empregado em tarefas operacionais, executadas diariamente. Dentre elas:

- Melhor gerenciamento do consumo de energia, pois utiliza poucos recursos de hardware;
- Administração remota:
  - Permite gerenciamento remoto aperfeiçoado através da área de trabalho remota;
  - Melhor gerenciamento remoto a partir da linha de comando e de scripts automatizados.
- Eficácia no gerenciamento automatizado e de linha de comando através do PowerShell versão 2.0.

Não há, no Server Core, o tradicional Windows Explorer, o navegador de Internet, tampouco suporte para códigos gerenciados.

O modo Server Core é considerado por alguns como a edição MS-DOS do Windows Server 2008. Esse modo torna necessário que os administradores de rede relembrem comandos que talvez não tenham sido usados em anos de utilização da interface gráfica do Windows.

O uso de comandos via modo texto possui a vantagem de que os mesmos comandos que são aplicados no modo Core possam ser utilizados no modo completo, com tela gráfica. Isso facilita a execução de tarefas administrativas, inclusive com a implementação de scripts que gerenciam o sistema de forma mais rápida e confiável.

Geralmente a instalação do Windows, mesmo no modo Server Core, é feita totalmente através da interface gráfica e os comandos são utilizados apenas após a instalação. Nesse ponto é feita toda a configuração inicial, tal como: definição do nome da máquina, endereço IP e nome de domínio, dentre outras. Em seguida todo o gerenciamento será feito via console, assunto a ser abordado posteriormente.

O Windows 2008 Server Core aceita somente a instalação limpa, isto é, iniciada a partir do zero. Portanto é impossível migrar ou atualizar um servidor no modo Core para o modo de instalação completo e vice versa.

Grande parte das funcionalidades do Sistema Operacional tradicional não é aplicada ao Server Core, eliminando todas as interfaces gráficas. O resultado final é um sistema mais robusto, que diminui a possibilidade de ataques à segurança e exige menos recursos de hardware, permitindo uma instalação em servidores pouco eficientes, que ainda contenham um hardware antigo. Esse fato é interessante quando acontece uma atualização de servidores no ambiente. Deste modo os servidores mais antigos podem ser usados com o artifício da instalação do Server Core.

Benefícios do Windows 2008 Server Core

Diversos benefícios como segurança, tarefas administrativas reduzidas e menos vulnerabilidades são oferecidas no modo Core, garantindo a estabilidade do ambiente de TI. A seguir são detalhados os benefícios:

- Redução de manutenção. Como são instalados apenas os itens necessários, menos atividades são exigidas. O principal gerenciamento será o da função ou recurso instalado, não sendo o caso da instalação completa;
- Diminuição de ataques. Como menos aplicativos são instalados, reduz-se também a possibilidade de ataques à segurança;
- Poucas portas de redes são abertas, mantendo o Sistema Operacional seguro e compacto;
- Menos itens a serem gerenciados, devido à implantação reduzida de aplicativos no servidor;
- O uso do disco é reduzido, utilizando apenas 3 GB para a instalação do Sistema Operacional e do restante das funções;
- Menos patches de atualização, pois é um sistema mais enxuto com poucas funções disponibilizadas, o que torna o Server Core mais seguro e menos vulnerável. Além disso, reduz a quantidade de paradas no ambiente para possíveis manutenções;

• Menor risco de bugs. Diminuir a quantidade de código pode ajudar a reduzir a quantidade de erros do sistema.

Com os benefícios apresentados anteriormente, a utilização do Windows 2008 Server Core em um ambiente de TI passa a ser considerado. Suas funções podem ser planejadas com facilidade, pois são limitadas. Pessoas envolvidas nesse planejamento são:
• Responsáveis pelo planejamento e análise de TI que estejam avaliando tecnicamente o produto;
• Planejadores e designers de TI empresarial para organizações;
• Responsáveis pela segurança de TI;
• Profissionais de TI que gerenciam determinadas funções como: Serviços do Active Directory, AD LDS, Servidor DHCP, Servidor DNS, Servidor de Impressão, entre outros.

Instalação

Os requisitos para instalação do Windows 2008 Server Core, citados na **Tabela 1**, são os mesmos para o modo de instalação completo. Porém no modo Core não existe a tela gráfica. Isso propicia ao Server Core operar com facilidade sobre os requisitos mínimos, sendo alocados menos recursos de hardware para o Sistema Operacional e consequentemente, obtendo um ganho de desempenho considerável.

Com um processador atual, como o Quad Core que possui quatro núcleos de processamento e é utilizado em grande parte dos novos servidores, com apenas 512MB de memória pode-se instalar um servidor controlador de domínio somente leitura (RODC) em uma máquina virtual. Isso requer pouquíssimos recursos de hardware e funciona bem nessas circunstâncias.

A opção de instalação no modo Core não inclui e nem modifica quaisquer definições de funcionalidades já existentes na instalação completa. No entanto, apenas algumas funções, como AD DS, AD LDS, DHCP, DNS, que serão mencionadas mais adiante, são suportadas e estão disponíveis nesse modo de instalação.

Gerenciamento

A administração remota de computadores e servidores são essenciais para manter o ambiente de TI eficiente e saudável. Neste contexto o gerenciamento do Server Core pode ser realizado de várias maneiras:

• Conectando-se remotamente através do RDC (*Remote Desktop Connection*), e então inserindo todos os comandos via console;
• Através das consoles MMC (*Microsoft Management Console*), trazendo à versão R2 do Server Core duas novidades a saber, o suporte ao .Net Framework 2.0 e a introdução do Power Shell, uma poderosa ferramenta de linha de comando;
• O *Sconfig.exe* mostrado na **Figura 2**, que consiste de uma interface na qual existem comandos preestabelecidos que facilitam a configuração do Sistema Operacional após instalação. O *Sconfig.exe* é um modo de configuração mais intuitivo que as linhas de comando tradicionais.

Por meio da área de trabalho remota, pode-se acessar a console e, a partir da versão R2, o *Sconfig.exe* nos permite efetuar diferentes configurações com poucos cliques. Com uma interface bem intuitiva, possibilita a interação àqueles administradores de redes que não possuem conhecimento com as linhas de comando.

Sendo assim, é possível, com alguns cliques:
• adicionar um servidor ao domínio;
• renomear o servidor;
• adicionar grupos locais de administração;
• configurar o gerenciamento remoto (via console MMC, Power Shell e Server Manager);
• configurar o modo de atualização (manual ou automático);
• configurar data e hora;
• especificar as configurações de rede;
• habilitar opção que permite configurar se a tela do *Sconfig.exe*, mostrada na **Figura 2**, será apresentada após o login;
• Failover Clustering Feature, função que permite configurar tolerância a falhas;
• Efetuar logoff de usuário;
• Reiniciar o servidor;
• Desligar o servidor.

O Server Core também possui suporte a RPC (Remote Procedure Call) e DCOM (Distributed Component Object Model), permitindo que o gerenciamento seja feito remotamente, a partir de um servidor ou de uma máquina cliente, através dos consoles de gerenciamento MMC. Basta então clicar com o botão direito na raiz do Server Manager e conectar-se a outro computador, conforme apresentado na **Figura 3**.

Componente	Requisito
Processador	Mínimo: 1.4 GHz (processador x86 64 bits) na versão R2. Observação: É necessário um processador Intel Itanium 2 para o Windows Server 2008 R2 for Itanium-Based Systems.
Memória Principal	Mínimo: 512 MB. Máxima: 8 GB (Foundation) ou 32 GB (Standard) ou 2 TB (Enterprise, Datacenter e Itanium-Based Systems).
Requisitos de Espaço em Disco	Mínimo: 32 GB. Observação: Computadores com mais de 16 GB de RAM precisam de mais espaço em disco para paginação, hibernação e armazenamento de arquivos.
Monitor	Super VGA (800x600) ou maior resolução.
Outros	Unidade de DVD, teclado e mouse, acesso à Internet para download de atualizações do Sistema Operacional.

Tabela 1. Requisitos de Instalação do Windows Server Core

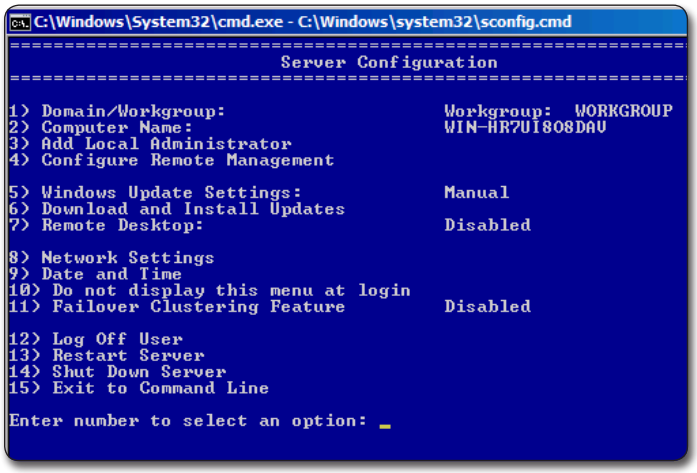


Figura 2. Sconfig.exe – Interface de linha comando pré-definida do Server Core

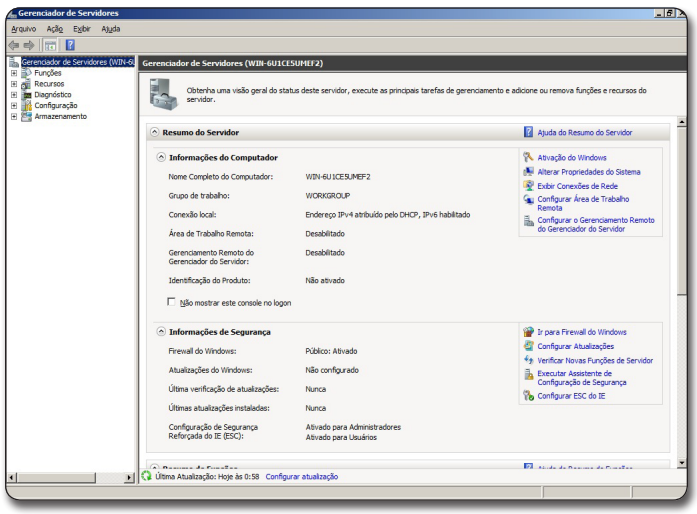


Figura 3. MMC (Microsoft Management Console) Interface de gerenciamento remoto do Server Core

A partir do Server Manager podem-se efetuar diversas tarefas administrativas, dentre as quais:

- visualizar o status de atualização do Windows;
- executar varreduras do analisador de práticas recomendadas em funções;
- verificar ou alterar o status do programa de aperfeiçoamento da experiência do usuário do Windows;
- gerenciar as configurações do firewall do Windows;
- gerenciar funções das páginas na web;
- verificar os parâmetros de configurações de segurança avançada do Internet Explorer;
- gerenciar funções a partir de home pages.

Além disso, há tarefas que não podem ser executadas no Server Core, através do Server Manager, por questões que envolvem a segurança do produto, tais como:

- adicionar ou remover funções de servidor;
- acrescentar ou retirar recursos de servidor;
- configurar as propriedades do sistema;
- verificar novas funções;
- alterar as configurações de atualizações automáticas do Windows;
- modificar as configurações de rede;
- alterar o nome do computador e a associação ao domínio;
- mudar os parâmetros de configurações de segurança avançada do Internet Explorer;
- executar o assistente de configuração de segurança, se o computador de origem for um servidor executando o Windows Server 2008 R2.

Existem diversas ferramentas de linhas de comando disponíveis na Internet que são suportadas pelo Server Core permitindo administrar o servidor remotamente, como as que vêm incluídas no pacote PSTools da Sysinternals, por exemplo. Dentre as diversas ferramentas de comandos, duas merecem destaque, a saber: o PsExec, que traz a linha de comando da máquina local e permite que comandos sejam executados como se realmente tivéssemos no servidor; e a PsList, que permite identificar quais processos estão sendo executados no servidor, além de finalizar os processos problemáticos.

O modo Core possui a capacidade de usar duas linguagens de script, VBScript e Windows PowerShell, bastante úteis no gerenciamento da instância do Server Core. Apesar do Server Core vir equipado com código necessário de tempo de execução VBScript, ou seja scripts que são executados em sequência diretamente da console do servidor, o mesmo não ocorria com o PowerShell. Contudo, a partir da versão R2 do sistema, já é possível executar os scripts via PowerShell direto da console.

O PowerShell 2.0 é a interface de linha de comando mais provável a ser utilizada, visto que já estamos na versão R2 do Server Core que suporta esta ferramenta. É possível, por exemplo, executar scripts em diversos computadores, principalmente permitir que vários profissionais de TI executem scripts em um único servidor, simultaneamente.

Dentre algumas melhorias do Windows 2008 Server Core R2 estão:

- aperfeiçoamentos em relação à versão 1.0;
- gerenciamento remoto aprimorado usando comunicação remota do PowerShell;
- segurança aprimorada para dados de gerenciamento, inclusive informações de estado e configuração, usando runspaces limitados;
- GUIs aperfeiçoadas para criar e depurar scripts do PowerShell, e exibir saída impressa na tela usando o Graphical PowerShell e o cmdlet Out-GridView;
- maior funcionalidade de script suportando a criação de scripts mais poderosos com menor esforço de desenvolvimento, devido à técnica de abreviamento de comandos e parâmetros;
- melhor portabilidade de scripts e cmdlets do PowerShell entre vários computadores.

Funções de Servidor

As funções também são conhecidas como papéis de servidor. Esses papéis definem qual função ou papel aquele servidor vai assumir no momento que definirmos qual será sua utilização.

Necessitando o mínimo de recursos possíveis, o Server Core apenas instalará um subconjunto dos arquivos binários necessários a determinadas funções específicas de servidor. Ou seja, o Server Core economiza o máximo de recurso de hardware e garante o máximo de desempenho da função instalada, alocando todo o recurso que ela precisa.

As principais funções suportadas pelo Server Core são:

- Active Directory Domain Services (AD DS);
- Active Directory Lightweight Directory Services (AD LDS);
- Server DHCP;
- Server DNS;
- File Server;
- Print Server;
- Hyper-V;
- Web Server (IIS).

É importante ressaltar que a função de Servidor Web (IIS) não suporta ASP.NET em instalações no modo Core. Devido ao fato de não suportar o código gerenciado, alguns recursos do IIS não estão disponíveis, como os listados a seguir:

- IIS-ASPNET;
- IIS-NetFxExtensibility;
- IIS-ManagementConsole;
- IIS-ManagementService;
- IIS-LegacySnapIn;
- IIS-FTPManagement;
- WAS-NetFxEnvironment;
- WAS-ConfigurationAPI.

Comparação das edições por opção de instalação do Server Core

De acordo com as funções disponibilizadas no modo Core, algumas delas são compatíveis apenas com algumas edições do Windows Server 2008 Server Core. Na **Tabela 2** podemos comparar todas as edições de acordo com cada função.

Recursos de servidor

Para os recursos opcionais a seguir, é exigido hardware apropriado, por que dependendo do recurso instalado algum outro equipamento ou até mesmo outro servidor pode ser usado:

- balanceamento de carga de rede;
- multipath I/O;
- criptografia de unidade de disco BitLocker.

Para os recursos a seguir não há pré-requisitos de hardware, pois como são recursos bem simples, somente um servidor será necessário para utilização:

- backup;
- protocolo SNMP;
- WINS;
- cliente Telnet;
- subsistema para aplicativos baseados no UNIX.

Core Configurator 2.0

Como informado anteriormente, o Server Core fora criado para ser totalmente administrado por linha de comando e remotamente através da console MMC do Windows. No entanto, o CodePlex, que mantém projetos de software Open Source, desenvolveu uma ferramenta para administrar o Server Core graficamente, facilitando assim à vida dos administradores de rede, mas fugindo parcialmente da realidade do Server Core.

Um ponto interessante é que por ser uma ferramenta Open Source, possibilita aos desenvolvedores aprimorá-la com o intuito de atender às suas necessidades, funcionando apenas no Windows Server 2008 R2 x86 64 bits.

O Core Configurator nada mais é do que uma ferramenta que possui uma interface gráfica como aquela mostrada na **Figura 4**, que permite que se façam todas as configurações no Server Core, evitando a utilização de comandos e da interface *Sconfig.exe*, citada anteriormente.

As principais funções do Core Configurator incluem:

- licenciamento de produtos;
- recursos de rede;
- ferramenta DCPromo;
- configurações iSCSI;
- funções de servidor e recursos;
- permissões de usuário e grupo;
- criação e exclusão de compartilhamentos;
- configurações dinâmicas de firewall;

Função de Servidor	Enterprise	Datacenter	Standard	Web	Itanium	Foundation
Serviços de Certificados do Active Directory	-	-	-	o	o	o
Serviços de Domínio Active Directory	-	-	-	o	o	o
Active Directory Lightweight Directory Services	-	-	-	o	o	o
Cache Hospedado do BranchCache	-	-	o	o	o	o
Servidor DHCP	-	-	-	o	o	o
Servidor DNS	-	-	-	-	o	o
Serviços de Arquivo	-	-	-	o	o	o
Hyper-V	-	-	-	o	o	o
Serviços de Mídia	-	-	-	-	o	o
Serviços de Impressão	-	-	-		o	o
Serviços Web	-	-	-	-	o	o

Tabela 2. Comparação das edições conforme funções instaladas

- configurações de vídeo e proteção de tela;
- adicionar e remover drivers;
- configurações de proxy;
- atualizações do Windows (incluindo WSUS);
- Multipath I/O;
- Hyper-V, incluindo miniaturas de máquinas;
- incluir servidor no domínio e renomeá-lo;
- adicionar e remover programas;
- serviços;
- WinRM;
- registro completo de todos os comandos executados.

Para instalar o Core Configurator, deve-se fazer o download da ferramenta no site do CodePlex e executar o arquivo *Start_CoreConfig.WSF*.

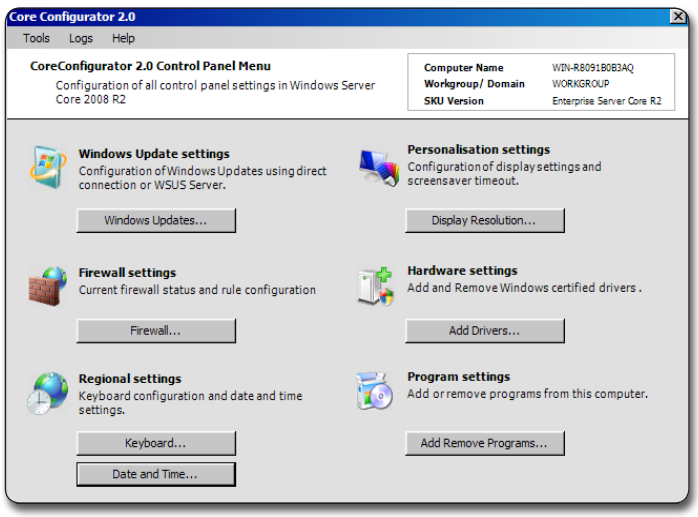


Figura 4. Core Configurator 2.0

Hyper-V

O Hyper-V é a principal ferramenta de virtualização da Microsoft. Pode-se instalá-la como uma função no modo Core e a partir daí trabalhar com máquinas virtuais. Atualmente o Hyper-V também funciona como um Sistema Operacional próprio, parecido com o Windows Server 2008 Server Core, contendo, portanto, o mesmo tipo de gerenciamento, por linha de comando ou através de consoles.

Configurações Iniciais

Nesta seção são apresentados os primeiros passos depois da instalação e os principais comandos para executar funções e recursos.

Como a interface do *Sconfig.exe* é bem intuitiva, vamos nos basear nos comandos do prompt do Windows, já que através dele é possível executar diversas tarefas administrativas, que auxiliam diariamente o administrador de redes.

Comandos e definições

A seguir veremos os principais comandos e definições para configurarmos o Server Core de forma adequada para utilização.

Definindo as configurações de IP

O primeiro passo após a instalação do Sistema Operacional de um servidor é definir as configurações de IP. A seguir, mostra-se como são feitas essas configurações:

Como visualizar as interfaces de Rede

Os servidores físicos geralmente possuem mais de uma interface de rede. Assim, é bom que se saiba quais interfaces estão disponíveis para configuração. Na **Figura 5**, o comando exibido mostra todas as interfaces de rede disponíveis no Sistema Operacional.

```
netsh interface ipv4 show interfaces
```



Figura 5. Visualizar as interfaces de redes disponíveis

Definindo o endereço IP para o Servidor

Muitos servidores utilizam aplicações que fazem algum tipo de acesso via endereço IP e, nesses casos, o endereço IP dos servidores não pode ser dinâmico. O ideal é que seja mantido sempre um endereço IP fixo nos servidores.

```
netsh interface ipv4 set address name="<ID>" source=static address=<IP Estático>
mask=<Máscara de Subrede> gateway=<IP do Gateway Padrão>
```

A **Figura 6** mostra como definir um endereço IP estático para o servidor. O ID, como mostrado entre aspas no comando anterior, é o número do Index, é definido no parâmetro **name**, e representa cada interface da placa de rede no Sistema. Também é mostrado o comando **ipconfig**, que exibe o endereço de IP do servidor.

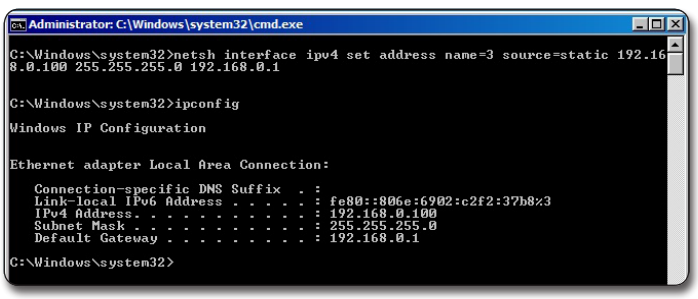


Figura 6. Definir um IP estático para o servidor

A **Figura 7** exibe o comando que faz com que o Sistema Operacional receba um endereço IP fornecido através do servidor DHCP. Também é mostrado o comando **ipconfig** com o parâmetro **/all**, que mostra as configurações de IP detalhadamente.

```
netsh interface ipv4 set address name="<ID>" source=dhcp
```

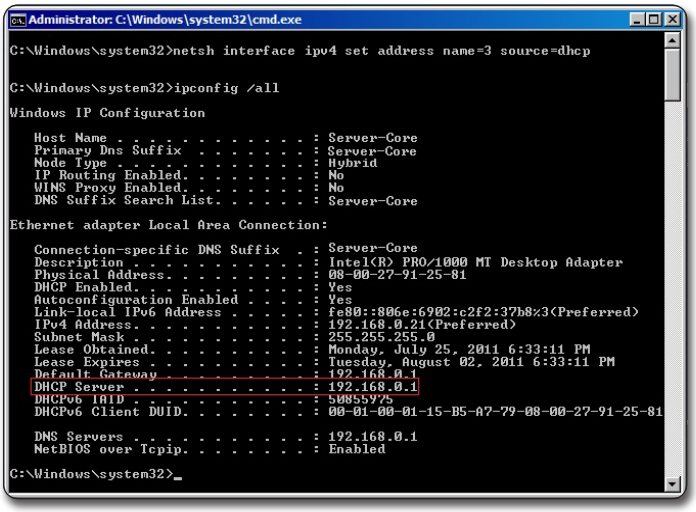


Figura 7. Define que o endereço IP é recebido de um servidor DHCP

O comando mostrado na **Figura 8**, por sua vez, configura o endereço IP de um servidor DNS em uma placa de rede citada no Index (ID). É possível configurar mais de um servidor DNS. Para isso, repete-se essa etapa para cada um que queira configurar, sempre alterando o Index= conforme o primeiro comando.

```
netsh interface ipv4 add dnsserver name="<ID>" address=<IP do Servidor DNS>index=1
```

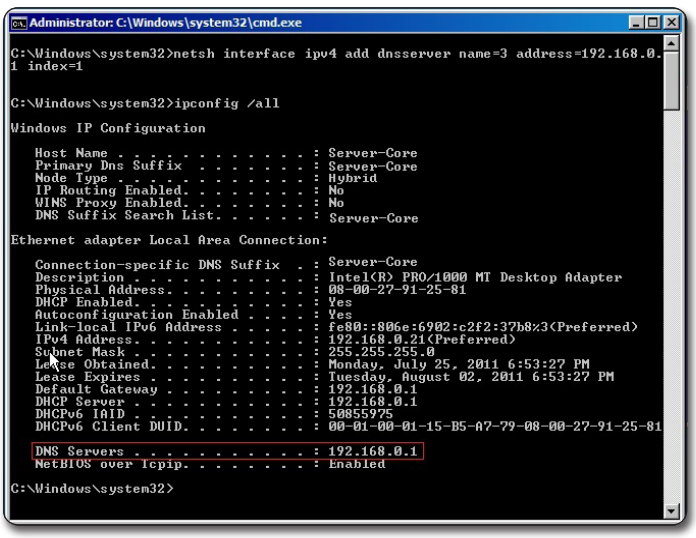


Figura 8. Define o endereço de IP do servidor DNS

Ingressar o Servidor em um domínio

Um dos passos mais importantes é inserir os servidores e computadores no domínio, pois facilita a administração e o gerenciamento da rede. Na **Figura 9** é demonstrado com inserir o servidor em um domínio.

Vale ressaltar que o usuário precisa ter as permissões necessárias para inserir o computador no domínio.

```
netdom join <Nome do computador> /Domain:<Domínio> /
UserD:<Login do Usuário> /passwordD:*
```

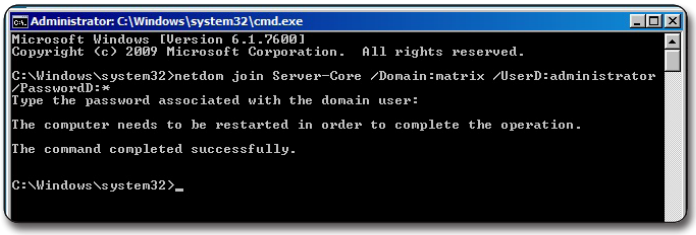


Figura 9. Inserir o servidor em um domínio

Inserir um usuário de domínio ao grupo local de administradores do Servidor

Para que se possa administrar o servidor recém-instalado o ideal é que o usuário faça parte do grupo local de administradores.

O comando seguinte define um usuário como membro do grupo local de administradores do servidor. Veja a **Figura 10**.

```
net localgroup administrators /add <Dominio>\<Login do Usuário>
```

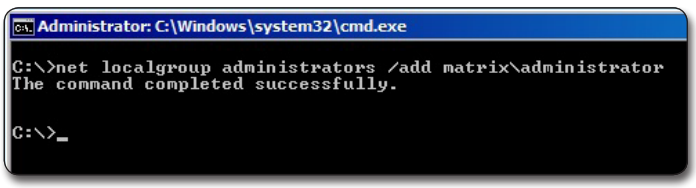


Figura 10. Adicionar usuário ao grupo de administradores locais do servidor

Reinicializar o servidor

Feitas as configurações de domínio, será imprescindível reinicializar o servidor. Para isso, usa-se o comando:

```
shutdown /r /t 10
```

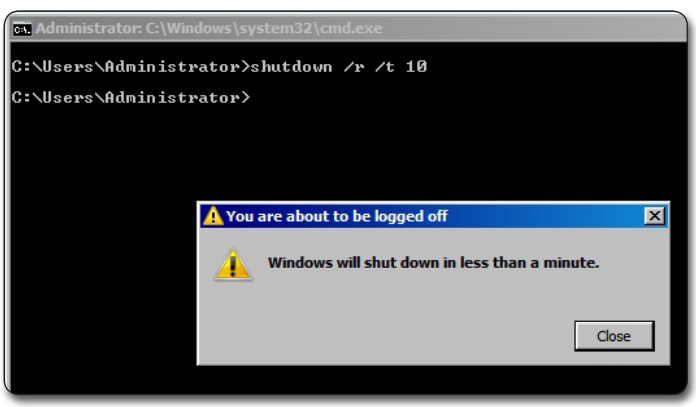


Figura 11. Reinicializar o servidor

Os parâmetros exibidos na **Figura 11** definem a reinicialização do sistema. A opção /r do comando shutdown informa que o servidor irá reiniciar e /t 10 informa que o tempo para que esta ação seja executada é de 10 segundos.

Renomear o Servidor

Como prática renomeamos os servidores de acordo com o padrão utilizado internamente pela empresa. O comando demonstrado na

Figura 12 mostra como se pode renomear o servidor. Após a execução do comando é exibida uma mensagem de confirmação para alteração do nome, pois alguns serviços podem estar atrelados ao nome do computador. Caso não se identifique algum problema, basta confirmar o comando e o nome é alterado.

```
netdom renamecomputer <Nome do Computador> /NewName:
<Novo Nome do Computador>
```

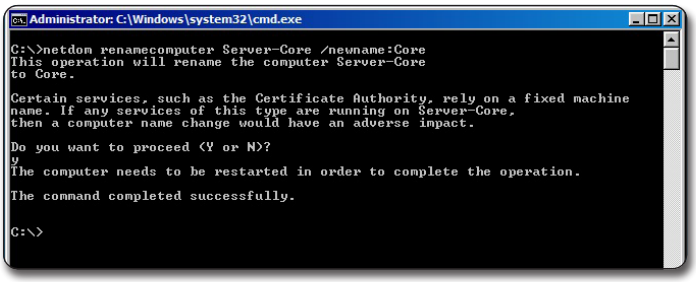


Figura 12. Alterar o nome do Servidor

Ativar uma chave de licença do Servidor

Como padrão em todo Sistema Operacional da Microsoft, é necessário ativá-lo com um número serial válido. Para isso se utiliza o comando:

```
slmgr.vbs –ato
```

Se a ativação for executada com sucesso, não é exibida nenhuma mensagem na tela.

Configuração de Firewall

A seguir, apresenta-se como ativar o gerenciamento remoto e habilitar o snap-in do Firewall no Windows 2008 Server Core:

• Ativar o gerenciamento remoto através de qualquer console MMC. Como padrão do Server Core, o gerenciamento remoto através da console vem bloqueado pelo firewall. Para efetuar o desbloqueio utiliza-se o comando:

```
netsh advfirewall firewall set rule group="Remote Administration" new enable=yes
```

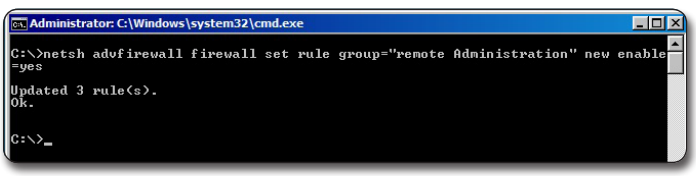


Figura 13. Habilitar gerenciamento remoto pela console

Na **Figura 13** demonstra-se como habilitar, via linha de comando, o gerenciamento remoto pela console. Este é um passo importante para que possamos ter acesso seguro remotamente.

• Habilitar o snap-in do firewall em um computador com Windows 7 ou Windows Server 2008. Para administrar o Server Core remotamente via console gráfica, basta executar o comando a seguir para habilitar o gerenciamento do firewall:

```
netsh advfirewall set currentprofile settings remotemanagement enable
```

Como demonstrado na **Figura 14**, é importante que este snap-in esteja liberado no firewall para se acessar remotamente de uma máquina com Windows 7 ou Windows Server 2008.

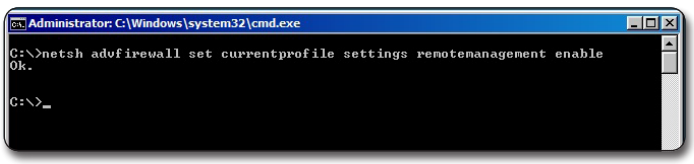


Figura 14. Habilitando snap-in do firewall no Windows 7 e Windows Server 2008

Administrando as funções do Server Core

O comando **oclist** lista as funções de servidor e os recursos opcionais disponíveis e instalados no momento. Geralmente ele é usado junto com o comando **ocsetup**, que instala as funções e os recursos de servidor. A **Figura 15** demonstra a execução de **oclist**.

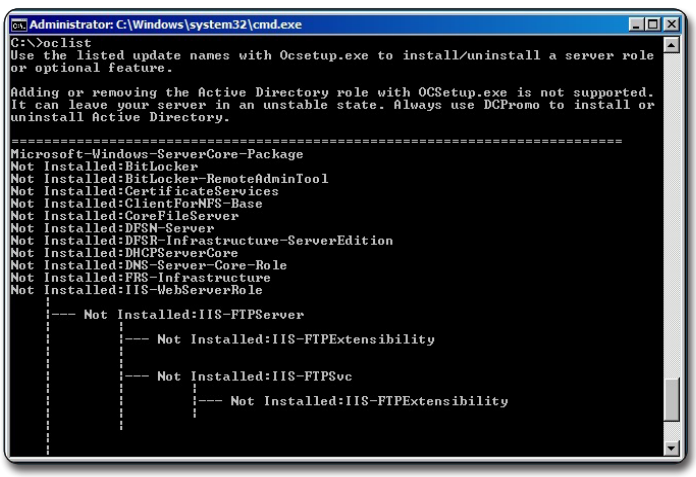


Figura 15. Listando as funções do servidor

A **Tabela 3** apresenta mais alguns comandos que podem ser utilizados para configurar o Windows 2008 Server Core.

Para finalizar, abaixo segue o comando padrão para instalação dos recursos, sendo necessário apenas inserir o nome do recurso desejado:

```
Dism /online /enable-feature /featurename:<Nome do Recurso>
```

Conclusão

O Windows Server 2008 Server Core é ideal para escritórios remotos devido ao alto nível de segurança e a gestão reduzida. Além disso, devido à instalação somente das funções necessárias, o gerenciamento do ambiente de TI proporciona um menor esforço administrativo. O Server Core funciona muito bem com hardware legado, pois como não possui interface gráfica, exige pouca memória e espaço em disco. Como é um Sistema Operacional com limitações em suas funcionalidades, utiliza somente as funções de Active Directory Domain Services (AD DS), Active Directory Lightweight Directory Services (AD LDS), Dynamic Host Configuration Protocol (DHCP), Domain Name

Recurso a ser instalado	Comando
Servidor DNS	Dism /online /enable-feature /featurename:DNS-Server-Core-Role
Servidor DHCP	Dism /online /enable-feature /featurename:DHCPServerCore
Servidor de Arquivo - Serviço de Replicação de Arquivos	Dism /online /enable-feature /featurename:FRS-Infrastructure
Servidor de Arquivo - Serviço de Sistema de Arquivos Distribuídos	Dism /online /enable-feature /featurename:DFSN-Server
Servidor de Arquivo - Serviço de Replicação DFS	Dism /online /enable-feature /featurename:DFSR-Infrastructure-ServerEdition
Servidor de Arquivo - Serviços de NFS	Dism /online /enable-feature /featurename:ServerForNFS-Base Dism /online /enable-feature /featurename:ClientForNFS-Base
Servidor de Impressão	Dism /online /enable-feature /featurename:Printing-ServerCore-Role
Servidor AD LDS	Dism /online /enable-feature /featurename:DirectoryServices-ADAM-ServerCore
Servidor de Domínio Active Directory	dcpromo /unattend:<Arquivo Autônomo>
Instalação padrão do IIS	Dism /online /enable-feature /feature-name:WebServerRole
Hyper-V	Dism /online /enable-feature /featurename:Microsoft-Hyper-V
BitLocker	Dism /online /enable-feature /featurename:BitLocker
Backup	Dism /online /enable-feature /featurename:WindowsServerbackup
Protocolo SNMP	Dism /online /enable-feature /featurename:SNMP-SC
Cliente Telnet	Dism /online /enable-feature /featurename:TelnetClient

Tabela 3. Configurando recursos do Windows 2008 Server Core

System (DNS), File Server, Print Server, Windows Media Services, Hyper-V, Web Server.

O Server Core consome pouco recurso de hardware e permite maior flexibilidade nas funções e recursos a serem instalados, de modo que pode vir funcionar como um servidor dedicado. Por padrão, nenhum recurso vem habilitado, tornando este, um dos focos principais do sistema.

O Server Core é totalmente gerenciável, via linha de comando, pois não existe interface gráfica. Isso ajuda bastante no desempenho e na segurança, pois sem a tela gráfica e com menos funções instaladas o sistema operacional torna-se mais seguro. Os principais ganhos com segurança estão relacionados à redução da possibilidade de ataques devido à necessidade da instalação de poucos recursos e a inexistência do ambiente gráfico, tornando-o um sistema mais robusto que a versão para instalação no modo completo. Deste modo, a tendência é que Sistemas Operacionais gerenciados via linha de comando e sem interface gráfica sejam adotados para uso como servidores dedicados devido ao alto nível de segurança e o baixo consumo de hardware, que aumenta o desempenho das tarefas dedicadas e mantém o Sistema Operacional muito estável.



Edson Xavier
edsonbxavier@hotmail.com
Trabalha a 4 anos na área de Tecnologia da Informação, atuando diariamente com administração de redes Microsoft, gerenciando ambientes virtualizados com Hyper-V. Atualmente participa de projetos de infraestrutura de redes. Graduado em Gestão de Redes de Computadores na Universidade Gama Filho, possui também certificações MCSA | MCDST | MCP.

CodePlex - Seção Core Configurator
http://coreconfig.codeplex.com/

Windows Server 2008
http://www.microsoft.com/windowsserver2008/pt/br/default.aspx

Passo a Passo de Instalação do Server Core
http://technet.microsoft.com/pt-br/library/cc753802(WS.10).aspx

Desvendando a criptografia

André Koide da Silva

Resumo DevMan

De que se trata o artigo:

Neste artigo será apresentado um breve histórico da criptografia, que é a arte (ou ciência) que permite tornar uma mensagem incompreensível para pessoas que não estejam autorizadas a recebê-la. Para tanto, podem ser utilizadas duas chaves distintas (criptografia assimétrica) ou uma única chave (criptografia simétrica). Em ambas as abordagens serão exemplificados os procedimentos empregados, bem como os principais algoritmos conhecidos. Também serão descritos os mecanismos que permitem a confidencialidade, a autenticidade, a integridade e o não-repúdio, e como estes são assegurados pelo uso dos sistemas criptográficos.

Em que situação o tema é útil:

Os pontos destacados neste artigo serão úteis para a compreensão dos conceitos básicos relativos aos sistemas criptográficos, certificados digitais e infraestrutura de chave pública, presentes em diversas aplicações Web, no correio eletrônico, no comércio eletrônico, na troca sigilosa de dados entre redes de diferentes corporações (redes virtuais privadas), entre outras.

Desvendando a criptografia:

Desde os tempos antigos, a comunicação sigilosa sempre foi uma necessidade e um fator estratégico. A troca de mensagens confidenciais durante as grandes guerras foi decisiva para os rumos tomados pela humanidade e também impulsionou a evolução dos sistemas criptográficos. O advento da Internet e sua posterior adoção comercial criaram novas demandas para a segurança da informação: confidencialidade, autenticidade, integridade e não-repúdio. Desta forma, este artigo tem por objetivo descrever os fundamentos da criptografia simétrica e assimétrica, os diferentes algoritmos conhecidos e os desafios computacionais relacionados ao processamento de suas operações matemáticas, bem como apresentar os certificados digitais e a infraestrutura de chave pública criada para o gerenciamento e a distribuição destes certificados.

A origem do termo criptografia está nas palavras gregas que significam escrita sigilosa. Muitos pesquisadores consideram a criptografia tão antiga quanto à própria escrita: os primeiros registros de sua utilização datam de 1.900 a.C., no antigo Egito. Durante muito tempo, o sucesso de sua utilização dependeu exclusivamente da habilidade de seus usuários em realizar os procedimentos que geravam a mensagem cifrada, pois a criptografia foi amplamente divulgada somente na década de 1970, com o surgimento dos computadores.

Neste artigo, serão descritos os sistemas criptográficos modernos que utilizam uma única chave, também denominados simétricos, as operações realizadas para criação do texto cifrado, os diferentes algoritmos empregados e as principais vantagens e desvantagens da utilização desta abordagem. Posteriormente, serão expostos os sistemas baseados em duas chaves, conhecidos como assimétricos, a maneira como asseguram a confidencialidade, a autenticidade, a integridade e o não-repúdio, e as características básicas dos algoritmos conhecidos. Por fim, serão apresentados os certificados digitais e a infraestrutura de chave pública criada para gerenciamento e distribuição destes certificados.

Introdução a criptografia

A criptografia é a arte (ou ciência) que permite tornar uma mensagem ou informação incompreensível para pessoas que não estejam autorizadas a recebê-la, ou seja, esta se caracteriza pela escrita em código ou cifras. A tentativa de recuperação do conteúdo criptografado, sem a autorização de seu originador, é chamada de criptoanálise, ou mais popularmente, de ataque.



Figura 1. Processos de cifragem e decifragem de uma mensagem

A origem do termo criptografia está nas palavras gregas *kryptós* (escondido ou oculto) e *graphos* (escrita). A cifragem da informação ocorre quando a mensagem original (também denominada texto claro) é disfarçada e oculta em códigos sem significado aparente, produzindo o texto criptografado (cifrado). Este pode ser transformado novamente em texto legível através do processo de decifragem (**Figura 1**).

Muitos pesquisadores consideram a criptografia tão antiga quanto à própria escrita. Segundo Jorge Loureiro Dias, em

seu artigo *Desenvolvimento Histórico da Criptografia*, os primeiros registros de sua utilização datam de 1.900 a.C., no antigo Egito. Nesta época, o faraó Khnumhotep II substituía trechos das escritas em argila que indicavam o caminho para seus tesouros armazenados nas pirâmides, possibilitando somente aos sacerdotes decifrá-las. Entre 600 e 500 a.C., os hebreus utilizaram um mecanismo simples, de substituição de uma letra por outra (conhecido como cifra de Atbash), para escrever o livro do Profeta Jeremias. Na Índia, o famoso livro erótico Kama-sutra, escrito pelo sábio hindu Vatsyayana, indicava as comunicações secretas, sejam escritas ou faladas, como uma das artes que as mulheres deveriam conhecer e praticar.

Na Roma Antiga, o imperador Júlio César utilizava um método para cifrar suas mensagens e enganar seus inimigos. O algoritmo simplesmente substituía as letras do alfabeto por outras localizadas três posições à frente. A **Figura 2** ilustra um exemplo de uma ordem cifrada segundo a técnica usada por Júlio César.

Este método funcionou durante algum tempo. Entretanto, sua lógica foi descoberta facilmente. A introdução do conceito de chave secreta permitiu melhorá-lo, pois, ao invés de realizar a permuta pela terceira letra subsequente, passou-se a realizar a substituição por outra, localizada N posições à frente. Desta forma, não bastava conhecer o algoritmo, mas a chave secreta N que foi utilizada para cifrar a mensagem (**Figura 3**).

O método criado pelo imperador é denominado cifra de substituição. Os algoritmos de criptografia modernos seguem esta mesma filosofia, ou seja, seus passos e procedimentos são conhecidos publicamente, sendo somente a chave compartilhada entre as entidades envolvidas na comunicação.

No século IX, o filósofo árabe Al-Kindi muito contribuiu para a criptoanálise, escrevendo sobre como utilizar a análise de frequência para decifrar mensagens. Segundo este, bastava avaliar as letras mais comuns e substitui-las por aquelas mais utilizadas habitualmente, como as vogais.

```
Alfabeto original:    A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Alfabeto Júlio César: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

     Texto claro:         VAMOS ATACAR HOJE
     Texto criptografado: YDPRV DWDFDU KRMH
```

Figura 2. Ordem cifrada segundo o algoritmo de Júlio César

```
Alfabeto original:    A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Alfabeto Júlio César: F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
Chave secreta (N):    5

Texto claro:         ATACAR
Texto criptografado: FYFHFW
```

Figura 3. Ordem cifrada segundo o algoritmo de Júlio César utilizando chave secreta

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
9	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
10	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
11	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
12	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
13	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
14	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
15	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
16	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
17	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
18	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
19	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
20	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
21	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
22	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
23	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
24	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
25	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figura 4. Distribuição de caracteres conforme as carreiras de Vigenère

A substituição polialfabética empregando palavras chaves, proposta no século XVI, foi marcada pelo trabalho de Blaise de Vigenère, e resistiu a todos os métodos de criptoanálise por mais de trezentos anos. A **Figura 4** ilustra um exemplo das carreiras de Vigenère, indicando como os caracteres eram distribuídos.

Segundo a técnica desenvolvida por Vigenère, um mesmo caractere pode assumir valores diferentes quando cifrado. No exemplo ilustrado na **Figura 5**, a mensagem "NEGOCIO OK" foi criptografada com a palavra chave "BUGRE". É importante notar que o texto produzido foi construído pela intersecção da coluna (texto claro) com a linha (caractere da chave utilizada), gerando desta forma as letras cifradas "F", "I" e "U", para a letra "O" da mensagem em texto claro.

A criptografia pré-computacional dependia exclusivamente da habilidade de seus usuários para realizar os procedimentos que geravam a mensagem cifrada. Em 1918, uma máquina conhecida como Enigma, patenteada pelo inventor alemão Arthur Scherbius, foi criada para promover a utilização da criptografia na comunicação entre comerciantes. Esta, embora não tenha obtido êxito comercial, foi adotada posteriormente pelas forças militares alemãs durante a Segunda Guerra Mundial, devido a sua

usabilidade e a suposta indecifrabilidade das mensagens cifradas.

A criptografia foi amplamente divulgada somente na década de 1970, com o surgimento dos computadores. Segundo Raul Fernando Weber, autor do artigo *Criptografia Contemporânea*, quando se usa sistemas computacionais para realizar a cifragem e decifragem, os processos de substituição e transposição podem ser empregados diversas vezes. Essa repetição de processos visa diminuir a frequência dos símbolos utilizados, aumentando a dificuldade dos criptoanalistas, pois a probabilidade de ocorrência destes tenderia a ser exatamente a mesma.

O advento da Internet e sua posterior adoção comercial criaram novas necessidades de segurança da informação, para assegurar:

• **Confidencialidade:** somente o destinatário da mensagem deve ter acesso ao seu conteúdo, ou seja, o acesso é restrito às entidades legítimas (aquelas autorizadas pelo proprietário da informação);

• **Integridade:** assegura que a mensagem mantenha suas características originais, afiançando que não foram criados, alterados ou removidos dados sem autorização, seja intencionalmente, indevidamente ou acidentalmente;

• **Autenticidade:** possibilita a confirmação da identidade dos envolvidos na comunicação;

• **Não-repúdio ou irretratabilidade:** garante que emissor ou receptor não possam negar a criação, a modificação ou a remoção das informações.

Sistemas de criptografia

Os sistemas criptográficos são agrupados em duas categorias, uma que emprega uma única chave e, a outra, que utiliza chaves distintas:

• **Criptografia simétrica:** igualmente conhecida como criptografia de chave secreta ou chave simples. Neste sistema, os envolvidos na comunicação (emissor e receptor) compartilham uma única chave, que é utilizada tanto para cifrar o texto claro quanto para decifrá-lo;

• **Criptografia assimétrica:** também denominada como criptografia de chave pública ou com duas chaves. Emprega duas chaves diferentes que possuem relação direta: a chave pública e a chave privada, sendo que esta última deve ser conhecida somente pela entidade que enviará/receberá as mensagens secretas.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
20	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
17	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D

Mensagem	N	E	G	O	C	I	O		O	K
Chave	B	U	G	R	E	B	U		G	R
Cifra	O	Y	M	F	G	J	I		U	B

Figura 5. Exemplo de aplicação da cifra de Vigenère (somente são apresentadas as cadeias relevantes para a chave "BUGRE").

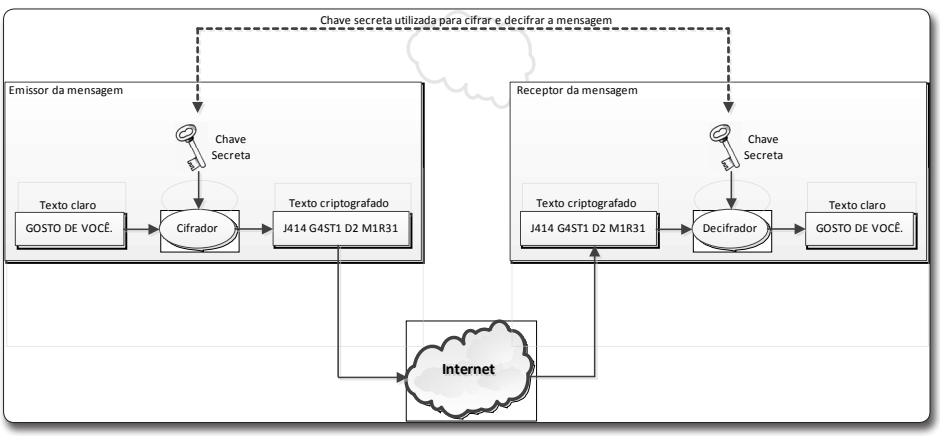


Figura 6. Criptografia simétrica: emprego da mesma chave para cifragem e decifragem da mensagem

Criptografia simétrica

Historicamente, a criptografia simétrica foi o primeiro método criado para cifragem de mensagens que necessitavam de sigilo (confidencialidade). Para tanto, é necessário que o emissor informe ao receptor do texto cifrado, qual chave secreta será utilizada no processo de cifragem, para que este possa decifrá-la (**Figura 6**).

A criptografia simétrica é bastante eficiente e rápida, exigindo o uso menos intenso de processamento nos computadores. Basicamente, duas operações são utilizadas para cifrar uma informação:

• **Substituição:** os elementos do texto claro (bits, caracteres etc.) são transformados em outros. A cifra de César e as carreiras de Vigenère são exemplos de substituição;

• **Transposição:** nesta operação, o texto claro é rearranjado (embaralhado). Inicialmente, uma chave secreta deve ser escolhida para auxiliar as trocas de posições dos caracteres da mensagem. No exemplo da **Figura 7**, a chave "FUTEBOL" (sem aspas) é utilizada para identificar as colunas em ordem alfabética crescente. A seguir, a mensagem é escrita abaixo da chave, em linhas com sete caracteres (tamanho da chave) e o texto cifrado é lido na vertical, a partir da primeira coluna.

Segundo André L. B. Cavalcante, em seu artigo *Teoria dos Números e Criptografia*, a criptografia moderna continua baseada nas operações de substituição e transposição, mas busca algoritmos mais complexos, capazes de dificultar a criptoanálise. Alguns métodos empregam a álgebra linear, envolvendo propriedades e operações com matrizes, por exemplo, multiplicando a matriz mensagem por uma matriz chave secreta. Para construção destas, cada letra deve ser associada a um número, permitindo a criação da matriz mensagem (no exemplo, "ATAQUE CONFIRMADO") e da matriz chave secreta (suponha "ANTIARMAMENTISTA"), conforme ilustrado na **Figura 8**.

Mensagem: QUANDO SERÁ INICIADO O ATAQUE TERRESTRE?							
Chave	F	U	T	E	B	O	L
Ordenação crescente	3	7	6	2	1	5	4
Mensagem	Q	U	A	N	D	O	S
	E	R	A	I	N	I	C
	I	A	D	O	O	A	T
	A	Q	U	E	T	E	R
	R	E	S	T	R	E	?
Cifra: DNOTRNIOETQEIARSCTR?OIAEEAADUSURAQE							

Figura 7. Exemplo de aplicação da operação de transposição

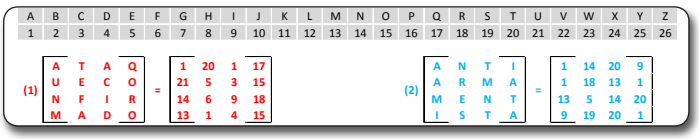


Figura 8. Matriz mensagem (1) e matriz chave secreta (2) construídas a partir da posição das letras no alfabeto, identificadas em ordem crescente

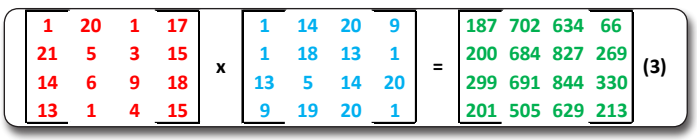


Figura 9. Matriz cifrada (3) resultante da multiplicação da matriz (1) pela (2)

A seguir, o texto criptografado é obtido a partir da multiplicação da matriz (1) pela (2). Este resultado, a matriz cifrada (3), é a mensagem que será transmitida ao receptor (**Figura 9**).

Para decifrar o texto, o receptor inicialmente deve encontrar a matriz inversa (4) da matriz chave secreta (2). Para obtê-la, este deve resolver a equação descrita na **Figura 10**.

```
[Matriz chave secreta] x [Matriz inversa] = [Matriz identidade]
```

Figura 10. Equação para cálculo da matriz inversa (4)

Lembrando que a matriz identidade (5) é uma matriz quadrada, onde os elementos que pertencem à diagonal principal são iguais a "1", e os demais, iguais à zero (**Figura 11**).



Figura 11. Matriz inversa (4), que será calculada, e matriz identidade (5)

A **Figura 12** ilustra a equação construída para o cálculo e o resultado obtido.

Por fim, o receptor utiliza a matriz inversa na equação ilustrada na **Figura 13** para recuperar a matriz mensagem (1), conforme apresentado na **Figura 14**.

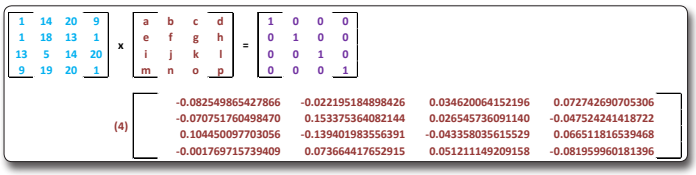


Figura 12. Equação construída para cálculo da matriz inversa (4)

```
[Matriz cifrada] x [Matriz inversa] = [Matriz mensagem]
```

Figura 13. Equação para decifrar a mensagem original

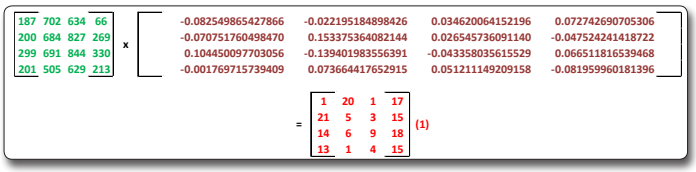


Figura 14. Matriz mensagem (1) resultante da multiplicação da matriz (3) pela (4)

Além da álgebra linear, a teoria dos números também é muito explorada na criptografia simétrica, especialmente em seus aspectos relacionados à fatoração de números extensos. Segundo Emílio Tissato Nakamura e Paulo Lício de Geus, autores do livro *Segurança de redes em ambientes cooperativos*, um problema matemático é considerado seguro quando a solução não é encontrada em até um milhão de anos, processado ininterruptamente por 50 mil computadores. Independentemente das propriedades empregadas, basicamente duas abordagens são suportadas para a cifragem e decifragem das mensagens:

- **Fluxo:** também conhecido como cifra em cadeia, opera a mensagem bit a bit ou byte a byte, de maneira incremental;
- **Bloco:** o texto claro é agrupado em blocos, com tamanho fixo, por exemplo, de 64 ou 128 bits, para aplicação das operações de substituição e transposição.

Atualmente, diversos algoritmos simétricos são utilizados. Entre os mais conhecidos, destacam-se:

- **DES (*Data Encryption Standard*):** proposto na década de 1970, a partir do algoritmo LUCIFER desenvolvido pela IBM. Padrão adotado pelos Estados Unidos em 1976 que cifrava em blocos com chave secreta de 56 bits. Atualmente, é considerado inseguro, pois teve sua primeira mensagem cifrada desvendada em 1997, em um concurso patrocinado pela empresa RSA Security, no qual criptoanalistas utilizaram a técnica de força bruta (ver **Nota DevMan 1**);
- **3DES (*Triple Data Encryption Standard*):** variação do algoritmo DES que opera em três estágios de cifragem em cascata. Com isto, foi possível empregar chaves com até 168 bits, tornando-se mais seguro que seu antecessor, porém mais lento;
- **IDEA (*International Data Encryption Algorithm*):** criado em 1991 e patenteado pela empresa suíça Ascom Systec, é estruturado conforme as mesmas linhas gerais do DES, porém, com chave de 128 bits;
- **AES (*Advanced Encryption Standard*):** sucessor do DES, este emprega o algoritmo Rijndael, escrito por Vincent Rijmen e Joan

Daemen, baseado em cifragem por blocos, com chaves de 128, 192 ou 256 bits. Em 2001, foi anunciado como novo padrão de criptografia utilizado pelos Estados Unidos, após cinco anos em processo de padronização;

•**RC6 (*Rivest Cipher* 6):** última versão de uma série de cifradores (RC2, RC3, RC4 e RC5) desenvolvidos pela RSA Security, destaca-se por sua simplicidade. Utiliza cifragem em blocos e caracteriza-se por sua flexibilidade e possibilidade de parametrização. A chave secreta também pode ter qualquer comprimento (por exemplo, 128, 192 ou 256 bits).



Nota do DevMan 1

Força Bruta: Força bruta é um algoritmo trivial que consiste em enumerar todos os possíveis candidatos de uma solução e verificar se cada um satisfaz o problema.

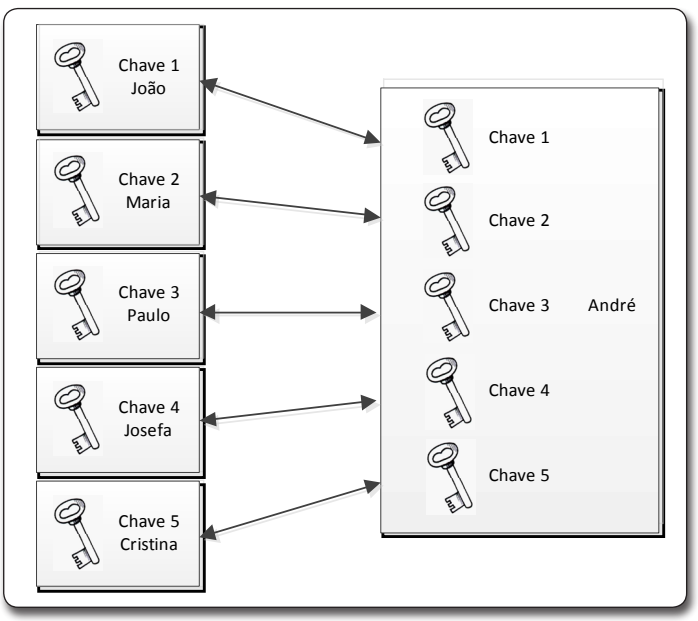


Figura 15. Chaves privadas utilizadas na criptografia simétrica

Conforme citado anteriormente, se por um lado, a principal característica da criptografia simétrica está relacionada ao seu baixo custo computacional, por outro, existem algumas restrições que dificultam sua adoção como único método para envio de mensagens sigilosas:

• As mensagens não podem ser assinadas, nem é possível o uso de certificados digitais (este tópico será exposto nas seções posteriores). Como assegurar sua autenticidade?

• Existem inúmeras dificuldades na distribuição da chave secreta entre o emissor e o receptor da mensagem. Como compartilhá-la sem depender de um terceiro envolvido?

• Cada comunicação requer a utilização de uma chave secreta diferente. Suponha um emissor que deseja enviar uma mensagem sigilosa para cinco destinatários diferentes (**Figura 15**). Como controlar e gerenciar o uso destas?

Assim, a criptografia assimétrica surge como alternativa para assegurar a confidencialidade das mensagens, além de possibilitar um mecanismo para distribuição de chaves, integridade, não-repúdio e autenticidade.

Criptografia assimétrica

Este método de criptografia é fundamentado em um par de chaves relacionadas matematicamente entre si: a chave pública – distribuída para qualquer usuário que deseja transmitir uma informação confidencial – e a chave privada ou secreta – mantida em segredo por seu proprietário. Sua principal característica baseia-se no fato de a mensagem que for cifrada com uma chave (pública ou privada) somente poderá ser decifrada com a outra (privada ou pública, respectivamente).

As chaves são criadas por meio de funções matemáticas de difícil inversão para impedir a descoberta da chave privada a partir da pública. Para tanto, emprega-se números inteiros aleatórios extremamente extensos, da ordem de centenas de dígitos decimais (centenas ou milhares de bits), impedindo que um sistema distribuído de computadores avançados possa testar todas as combinações possíveis em tempo hábil. Analogamente à criptografia simétrica, também são empregados problemas matemáticos de difícil solução para cifragem das mensagens, entre eles (segundo Raul Fernando Weber e Vinicius Gadis Ribeiro, em seu artigo *Problemas computacionais para esquemas de criptografia de chave pública*):

• **Fatoração de números inteiros:** dados dois números primos extensos, o cálculo de seu produto é uma operação simples. Entretanto, supondo que somente o resultado seja conhecido, a operação para determinar quais foram os números que o originaram (chamada fatoração) é complexa e onerosa para os computadores. Por exemplo, calcular o produto de 759.167 por 999.667 é uma operação simples que resulta em 758.914.197.389; contudo, a partir deste valor, não existe algoritmo eficiente para determinar quais foram os números primos que o geraram;

• **Mochila:** é um problema de otimização combinatória que consiste em preencher uma mochila com objetos de diferentes pesos e valores. O objetivo principal é carregar o maior valor possível, não ultrapassando o peso máximo estabelecido. Para tanto, pode-se empregar a enumeração exaustiva, determinando-se todas as soluções possíveis e posteriormente comparando-as para identificar a melhor. Porém, já se observou que mesmo para os problemas pequenos, com apenas 20 itens, precisam ser calculadas mais de um milhão de soluções, tornando a abordagem de enumeração exaustiva inviável;

• **Logaritmo discreto em campos finitos:** em termos simples, o logaritmo é o expoente que uma base deve ter para produzir certa potência. Por exemplo, o logaritmo de 81 na base 3 é 4, pois 4 é o expoente que a base 3 deve usar para resultar 81. O problema do logaritmo discreto consiste na resolução da equação ilustrada na **Figura 16**: dado um número primo extenso P e os números G e Y, determinar o valor do expoente A. Para este problema, não é conhecido algoritmo eficiente que possibilite identificar sua solução.

$$Y = G^A \bmod P$$

Figura 16. Equação do problema do logaritmo discreto

Para garantir a confidencialidade, o emissor deve utilizar a chave pública do receptor da mensagem para cifrar o texto claro que será transmitido. Desta forma, somente o destinatário será capaz de decifrá-la, por intermédio de sua chave privada, conforme ilustra a **Figura 17**.

Já para assegurar a autenticidade de uma mensagem, o emissor da informação deve utilizar sua chave privada para cifrá-la. Assim, qualquer pessoa que possua a chave pública deste remetente poderá decifrá-la. Observe que o sucesso na decodificação do texto criptografado indica que este foi gerado pela origem esperada, por meio da chave privada (**Figura 18**).

A obtenção de confidencialidade e autenticidade em uma mesma transmissão requer que o texto claro seja cifrado inicialmente com a chave privada do emissor e, a seguir, com a chave pública do receptor. O destinatário, ao receber a mensagem, deverá decifrá-la primeiramente com sua chave privada e, em seguida, com a chave pública do remetente (**Figura 19**).

Destarte, dependendo da maneira como a chave pública e a chave privada são utilizadas e por quem, diferentes objetivos são alcançados:

- **Envio sigiloso dos dados:** chave pública do destinatário;
- **Autenticidade de mensagem:** chave privada do remetente;
- **Decifragem do texto criptografado:** chave privada do destinatário;
- **Verificação da autenticidade da mensagem:** chave pública do remetente.

A criptografia assimétrica suporta a criação de um resumo criptográfico (*message digest*), também conhecido como função *hash*, que tem por objetivo assegurar a integridade de um documento digital, quando este for assinado digitalmente. Suas principais propriedades são:

- **Unidirecionalidade:** deve ser computacionalmente inviável realizar a operação inversa, ou seja, a partir do resumo, obter o texto claro;

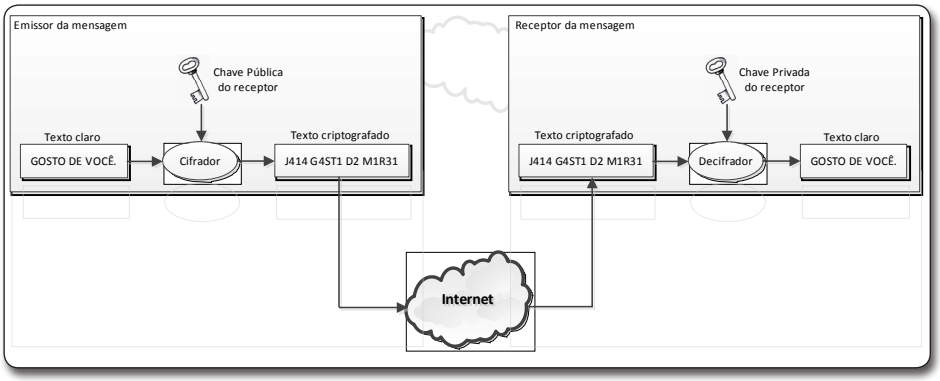


Figura 17. Aplicação da criptografia assimétrica para obtenção da confidencialidade

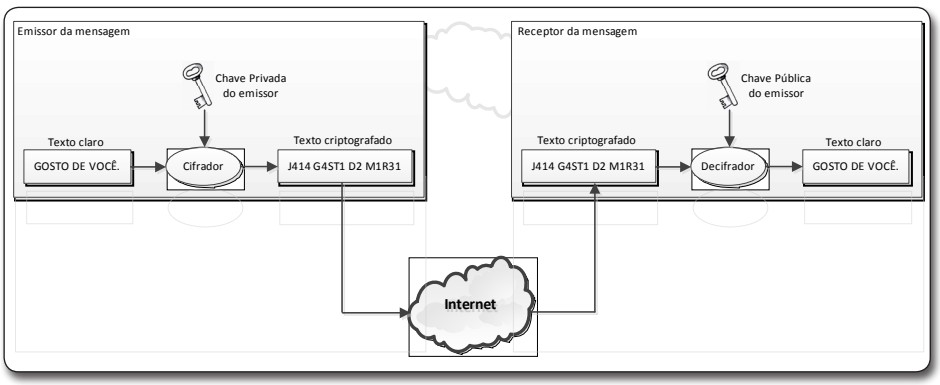


Figura 18. Implementação da criptografia assimétrica para obtenção da autenticidade

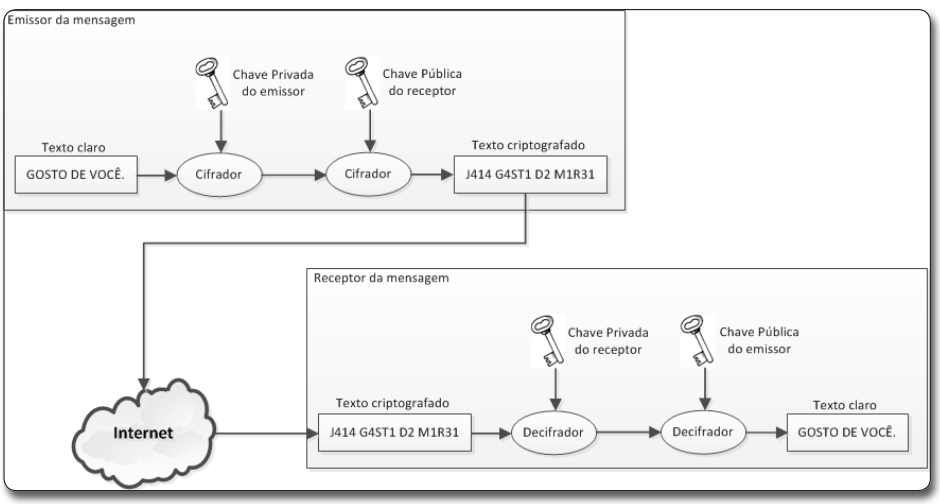


Figura 19. Uso da criptografia assimétrica para obtenção da confidencialidade e autenticidade

- **Difusão:** mensagens diferentes produzem resultados completamente distintos (a alteração de um único bit na informação original deve mudar aproximadamente metade dos bits do *hash*);
- **Facilidade de cálculo:** a obtenção do resumo criptográfico é simples e pouco onerosa em relação ao processamento computacional necessário para seu cálculo;
- **Compressão:** o resultado obtido possui tamanho fixo (por exemplo, 128 ou 256 bits), independentemente do tamanho do texto claro de entrada;
- **Colisão:** é computacionalmente impossível encontrar duas mensagens que gerem resumos criptográficos iguais.

Assim, ao invés de cifrar a mensagem completa com a chave privada do emissor

e, a seguir, com a chave pública do receptor, conforme ilustrado na **Figura 19**, para assegurar autenticidade e confidencialidade, pode-se gerar seu resumo (*hash*) e criptografar somente este último com a chave privada do emissor, obtendo-se a assinatura digital. Em seguida, esta deve ser adicionada ao texto claro, gerando a mensagem assinada, que será cifrada com a chave pública do receptor (**Figura 20**).

A recuperação da mensagem original e a verificação da assinatura digital estão descritas na **Figura 21**. O receptor decifra inicialmente o texto criptografado com sua chave privada, obtendo a mensagem assinada. Posteriormente, emprega a chave pública do remetente para recuperar o resumo criptográfico, que será comparado com o resumo calculado a partir do texto claro (que foi decifrado). Se forem iguais, a assinatura digital foi validada (garantindo a integridade, o não-repúdio e a autenticidade). Caso contrário, esta é invalidada.

Entre os algoritmos assimétricos mais conhecidos, destacam-se:

• **RSA (Ronald Rivest, Adi Shamir e Leonard Adleman fundadores da empresa RSA Security):** criado em 1977, foi o primeiro procedimento a suportar criptografia e assinatura digital. É um dos métodos de chave assimétrica mais utilizados atualmente. Seu funcionamento consiste no produto de dois números primos extensos, que para ser desvendado, depende da fatoração de números inteiros – um problema computacionalmente inviável. Em 1999, uma chave RSA de 512 bits foi revelada com o apoio de cientistas de diversos países. Para tanto, foram necessários sete meses de trabalho e processamento distribuído de aproximadamente 300 computadores;

• **ElGamal:** desenvolvido por Taher ElGamal em 1984 para ser utilizado no gerenciamento de chaves. Sua segurança advém do problema do logaritmo discreto, envolvendo a manipulação matemática de grandes quantidades numéricas;

• **Diffie-Hellman:** proposto por Whitfield Diffie e Martin Hellman em 1976, foi o primeiro algoritmo de chave assimétrica projetado para permitir a distribuição das chaves privadas utilizadas na criptografia simétrica. Também é baseado no problema do logaritmo discreto e, atualmente, permanece em uso;

• **Curvas Elípticas:** método apresentado em 1985 sugerindo a utilização de curvas elípticas para os sistemas criptográficos de chave assimétrica já existentes, como o ElGamal e o Diffie-Hellman. Estas curvas possuem o potencial de proverem sistemas mais seguros, com chaves de menor tamanho;

• **DSA (*Digital Signature Algorithm*):** destinado ao suporte das assinaturas digitais, foi descrito 1991 e patenteado pelo governo dos Estados Unidos. Permite a utilização de chaves com até 1024 bits e fundamenta-se no problema do logaritmo discreto.

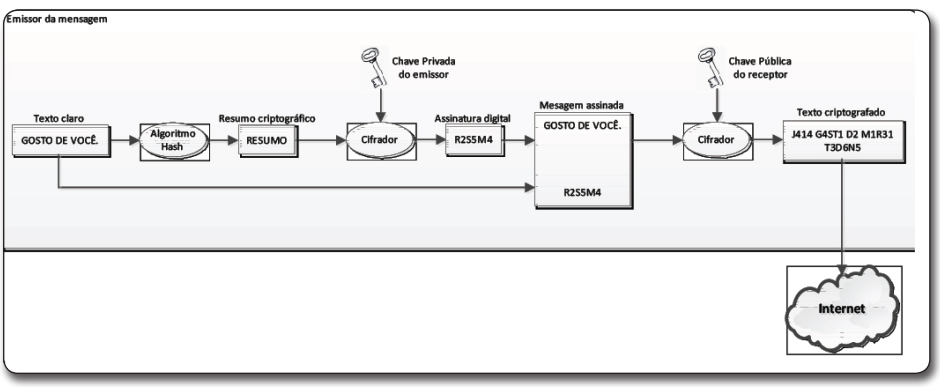


Figura 20. Criação da assinatura digital e da mensagem assinada para transmissão criptografada com a chave pública do receptor (confidencial)

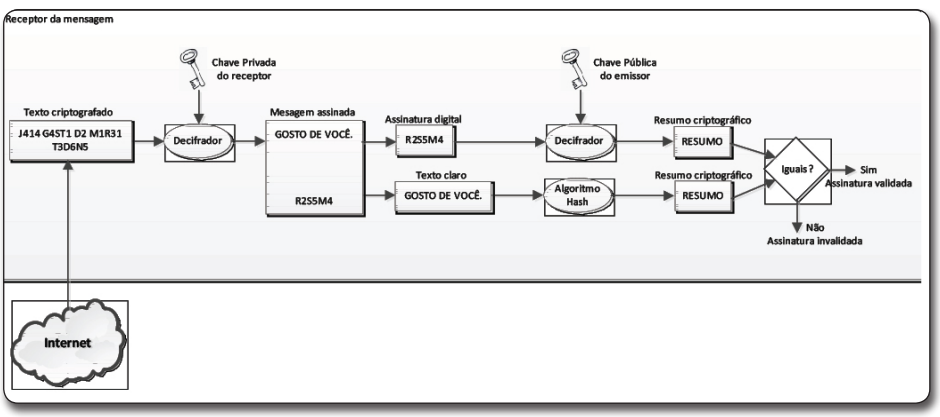


Figura 21. Recuperação e validação da assinatura digital

Para obtenção dos resumos criptográficos (funções *hash*), são comumente usados os seguintes algoritmos:

• **MD5 (*Message Digest Algorithm* 5):** função hash de 128 bits unidirecional desenvolvida pela RSA Security em 1991. Já existem vulnerabilidades conhecidas relacionadas às colisões dos resumos criptográficos. (Colisão é uma situação que ocorre quando duas mensagens distintas possuem o mesmo valor de *hash*.);

• **SHA (*Secure Hash Algorithm*):** a família SHA contempla as versões 0, 1 e 2. A função mais usada nesta família, a SHA-1 (160 bits), é suportada em diversas aplicações e protocolos de segurança, e foi considerada a sucessora do MD5, embora ambas tenham vulnerabilidades comprovadas. As variações SHA-224, SHA-256, SHA-384, e SHA-512 são conhecidas como SHA-2.

A criptografia assimétrica possui diversas vantagens, quando comparada à simétrica, entre elas: o fato da chave secreta não ser compartilhada, ou seja, esta é mantida em segredo por seu proprietário, que divulga apenas a chave pública (**Figura 22**); provê um mecanismo para distribuição (troca) das chaves privadas utilizadas na criptografia simétrica; e, possibilita autenticação, não-repúdio e integridade, através da assinatura digital.

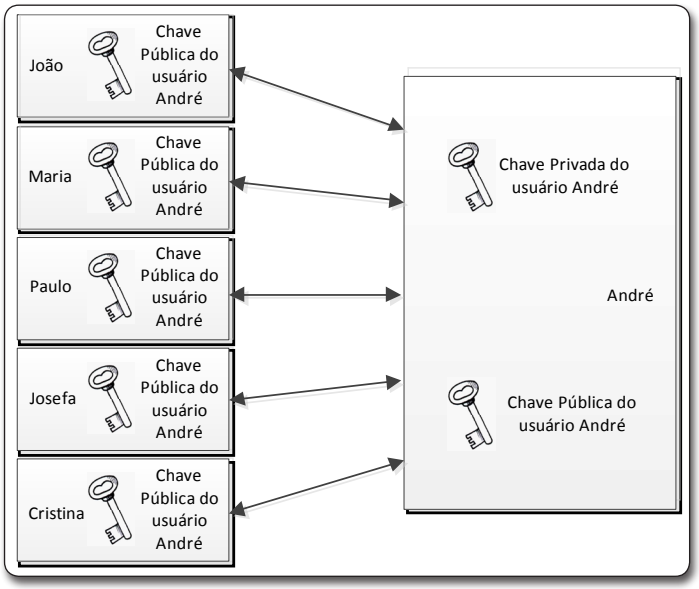


Figura 22. Chaves privadas e públicas necessárias na criptografia assimétrica

Por outro lado, a principal desvantagem da criptografia assimétrica está relacionada ao seu desempenho, pois os métodos são mais lentos e necessitam de maior capacidade de processamento dos sistemas onde são executados (pela sua natureza matemática, são computacionalmente intensivos, em média, cerca de 60 a 70 vezes mais lentos). Assim, os dois sistemas criptográficos são utilizados de maneira complementar: inicialmente, emprega-se um algoritmo assimétrico, para autenticação e estabelecimento de um canal seguro para o compartilhamento da chave privada, que será utilizada pela criptografia simétrica, que também assegura confidencialidade, mas com menor custo computacional.

O gerenciamento das chaves, em um sistema de criptografia assimétrica, deve considerar dois importantes aspectos: primeiro, a necessidade de localização da chave pública do destinatário da mensagem; segundo, a garantia de que a chave pública obtida seja realmente deste. Para resolver estas questões, foram criados os certificados digitais e a infraestrutura de chave pública (*Public Key Infrastructure* – PKI), que serão apresentados na próxima seção.

Certificados digitais e a infraestrutura de chave pública

Suponha que um usuário (Paulo) deseja enviar uma mensagem a outro (João). Para tanto, ele solicita a chave pública de João, a fim de cifrar a informação que será transmitida. Um atacante, notando este pedido, pode interceptar a resposta de João e falsificar a chave pública que será remetida a Paulo. Analogamente, este intruso pode fazer o mesmo com João, ou seja, expedir uma chave pública forjada. Deste modo, João e Paulo, ao invés de se comunicarem diretamente, na verdade estarão enviando o texto criptografado para um terceiro, que poderá decifrá-lo, observar seu conteúdo e cifrá-lo novamente. Ou ainda, alterar as informações antes de enviá-las ao destinatário legítimo.

Para evitar este tipo de ataque, foram criados os certificados digitais. Tais certificados consistem em chaves públicas assinadas por uma entidade de confiança (uma autoridade de certificação ou CA – *Certification Authority* – que atua como um cartório eletrônico), mitigando as tentativas de substituição de uma chave pública por outra. Além da chave pública, os certificados digitais contêm informações como: nome, endereço, validade do certificado, dados da autoridade certificadora, entre outras. Veja um exemplo na **Figura 23**.

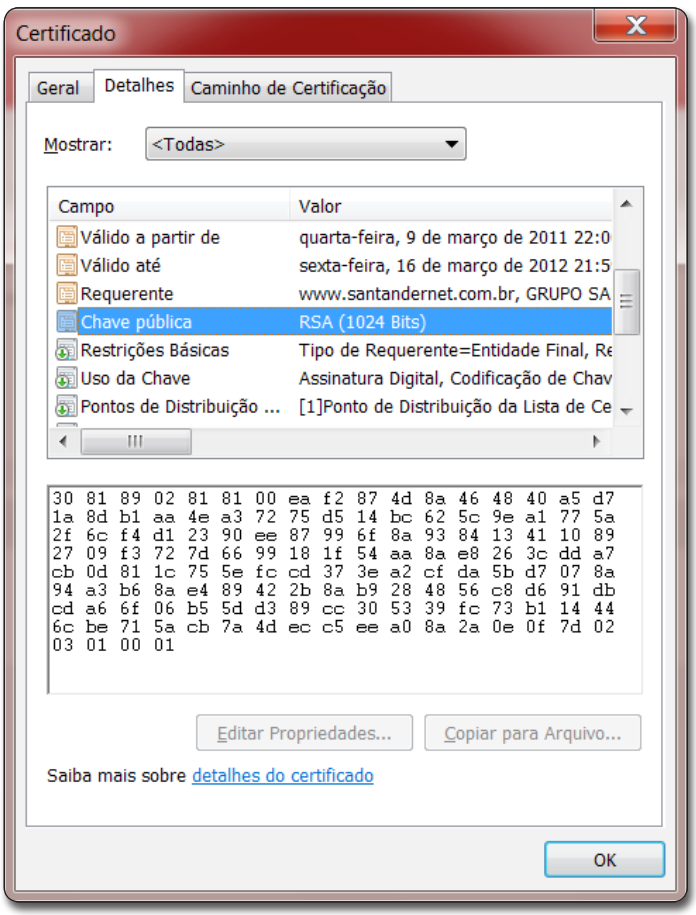


Figura 23. Exemplo de um certificado digital

Uma autoridade certificadora também tem a responsabilidade de manter e divulgar uma lista com os certificados revogados (*Certificate Revocation List* – CRL) – os quais podem ter sido roubados, perdidos, ou estarem sem utilidade. As CAs apresentam a possibilidade de estarem encadeadas em hierarquias de certificação, nas quais a CA de um nível inferior valida sua assinatura com uma CA de nível hierárquico superior.

Por fim, a complexidade relacionada às funções de gerenciamento dos certificados digitais resultou na criação de uma infraestrutura de chave pública, que é responsável pela distribuição, gerenciamento, expiração, nova emissão de certificados expirados, backup e revogação das chaves públicas e privadas de usuários, equipamentos ou serviços. Para tanto, a PKI define as técnicas, as práticas e os procedimentos a serem adotados pelas entidades a fim de estabelecer um sistema de certificação digital baseado em criptografia assimétrica.

Conclusões

A criptografia é a ciência que permite tornar uma informação incompreensível para pessoas que não estejam autorizadas a recebê-la. Embora tão antiga quanto à própria escrita, foi amplamente divulgada somente na década de 1970, com o surgimento dos computadores. Estes tornaram possível a repetição dos processos de substituição e transposição, para diminuir a frequência dos símbolos utilizados e aumentar a dificuldade dos criptoanalistas.

Os sistemas criptográficos são agrupados em duas grandes categorias: simétricos e assimétricos. Os algoritmos simétricos (DES, 3DES, IDEA, AES e RC6, entre outros) são eficientes e rápidos, exigindo o uso menos intenso de processamento nos computadores; já os métodos assimétricos (RSA, ElGamal, Diffie-Hellman, Curvas Elípticas, DAS, etc.), pela sua natureza matemática, em média, são de 60 a 70 vezes mais lentos.

A criptografia assimétrica possui diversas vantagens quando comparada à simétrica. Entre as mais relevantes: o fato da chave secreta não ser compartilhada; um mecanismo para distribuição das chaves privadas; e a assinatura digital. Por outro lado, sua principal desvantagem está relacionada ao desempenho. Assim, os dois sistemas criptográficos passaram a ser utilizados conjuntamente, aproveitando suas melhores características.

A evolução da criptografia foi motivada primariamente pela necessidade da transmissão de informações sigilosas, que subsidiavam a tomada de decisões estratégicas nas áreas militares e civis. O célere crescimento da Internet e sua utilização massiva também apresentaram novos desafios relacionados à segurança da informação. Neste contexto, os sistemas criptográficos são a melhor ferramenta para atender a estes requisitos atuais, presentes nas transações eletrônicas realizadas na rede mundial de computadores.



André Koide da Silva
andre@aksilva.com

Mestre em Engenharia pela Escola Politécnica da Universidade de São Paulo, especialista em Redes de Computadores pela Universidade Estadual de Campinas e bacharel em Sistemas de Informação pela Universidade Presbiteriana Mackenzie. Atualmente é especialista em redes de computadores pelo UOL Diveo, com ampla experiência em projetos e implementação de soluções Cisco Systems, Brocade Communications Systems e Juniper Networks. Atua também como professor na Faculdade Impacta Tecnologia. Trabalhou em empresas como Goldnet TI, EDS (Electronic Data Systems do Brasil) - atual HP Brasil (Hewlett-Packard), Alog Data Centers e Gennari & Peartree Projetos e Sistemas. Experiência de onze anos na área de redes de computadores e tecnologia da informação, além de cinco anos como docente do ensino superior nestas áreas. Certificado Cisco IP Communications Express Specialist, Cisco Certified Network Associate (CCNA), Extreme Network Associate (ENA), 3Com Certified Wireless Expert, Juniper Networks Certified Internet Specialist (JNCIS-M), Project Management Professional (PMP), Information Technology Infrastructure Library (ITILv3) e ISO/IEC 20000 - IT Service Management.

Artigo "Criptografia Contemporânea", escrito por Raul Fernando Weber
http://www.inf.ufsc.br/~mauro/curso/redes/cripto.doc

Artigo "Desenvolvimento Histórico da Criptografia", escrito por Jorge Loureiro Dias
http://www.uniplandf.edu.br/revista/vol3n3.pdf

Artigo "Problemas computacionais para esquemas de criptografia de chave pública", escrito por Raul Fernando Weber e Vinicius Gadis Ribeiro
http://dainf.ct.utfpr.edu.br/~maziero/static/ceseg/wseg04/2413.pdf

Artigo "Teoria dos Números e Criptografia", escrito por André L. B. Cavalcante
http://system7.upis.br/revistavirtual/Cavalcante_%20Teoria%20dos%20N%FAmeros%20e%20Criptografia_2005_UPIS.pdf

Livro "Segurança de redes em ambientes cooperativos", escrito por Emílio Tissato Nakamura e Paulo Lício de Geus - Editora Futura – 2° edição – 2003

Usando o Forefront Threat Management Gateway

Uilson Souza

Resumo DevMan

De que se trata o artigo:

O artigo apresenta o Forefront Threat Management Gateway, ou simplesmente TMG, uma solução que aborda diversas questões de segurança e mantém a produtividade da corporação.

Em que situação o tema é útil:

O TMG se faz útil para profissionais de infraestrutura ou segurança interessados em um produto do tipo Secure Web Gateway de qualidade e com baixo custo.

Usando o Forefront Threat Management Gateway:

Neste artigo debatemos sobre as funcionalidades e vantagens do Forefront Threat Management Gateway, além de conhecer boas práticas para configuração e sugestões de instalação.

Manter os negócios de uma corporação em harmonia com questões de segurança da informação representa hoje um desafio, tanto para quem provê TI, como para quem se preocupa em tempo integral com o negócio principal da empresa.

A questão da segurança hoje tem um peso considerável nas corporações, pois, o emprego desta acarreta em uma série de processos e procedimentos que necessitam de tempo, pessoal qualificado, comprometido e, acima de tudo, investimentos.

Entretanto, a falta de observância de aspectos de segurança pode acarretar em um prejuízo e em um custo muito maior para a corporação do que teria se fossem implementadas políticas e ferramentas adequadas.

Bem sabemos que uma corporação jamais para e seus colaboradores necessitam de informações rápidas, precisas e seguras.

Dessa forma, é necessária uma sintonia entre a área de TI e as demais áreas do negócio em várias vertentes, tais como:

1. A corporação precisa proteger dados importantes e seus pontos de distribuição. A área de TI precisa prover acesso seguro e fácil ao mais diversos tipos de dispositivos;
2. Para a continuidade dos negócios, a corporação precisa garantir acesso a seus colaboradores de qualquer lugar;
3. A corporação luta para diminuir custos com gerenciamento de recursos de segurança e a área de TI tenta diminuir a complexidade neste gerenciamento.

Para todos esses pontos, o Forefront Threat Management Gateway, ou simplesmente TMG, é a solução que aborda diversas questões de segurança e mantém a produtividade da corporação.

O que o Forefront TMG oferece

O produto pode funcionar como um proxy ou um firewall de borda, isto é, um firewall que protege a rede local da internet. O produto se comporta de forma efetiva na proteção da rede e dos dados.

O Forefront TMG atua como uma solução de Gateway Seguro para Web completa, provendo funcionalidades de Firewall, Web Cache, Proxy, Filtro de Conteúdo, Inspeção de redes, Inspeção de Malware, Detecção de Intrusão, além de prover segurança e proteção a seu Exchange Server.

Além das funcionalidades herdadas de seu antecessor (ISA Server 2006), o Forefront TMG traz consigo uma nova arquitetura que atua em três pilares distintos (veja a **Figura 1**):

1. Proteção de acessos de dentro para fora;
2. Proteção de acessos de fora para dentro;
3. Proteção da rede contra ameaças.

Proteção de acessos de dentro para fora

Neste tópico apresentamos a descrição das funcionalidades que ajudam a proteger os acessos internos da infraestrutura e suas definições:

- **Proteção de E-mails** – O Forefront TMG traz o serviço de proteção ao tráfego de e-mails contra todo o tipo de ameaças que hoje são enviadas em mensagens eletrônicas. E-mails com promoções, fotos de personalidades ou fatos do cotidiano podem conter ameaças que, ao penetrar no ambiente protegido, são capazes de um grande estrago;
- **Publicação Web Segura** – O Forefront TMG pode ser usado como um servidor de publicação para seus Servidores Web, protegendo as aplicações e evitando que um invasor chegue ao servidor de aplicativos web. O serviço de publicação Web segura usa certificado digital no servidor web e no servidor TMG para que todas as requisições sejam criptografadas. Neste caso a requisição para determinado site é encaminhada ao servidor TMG que redireciona para o servidor web através das regras de publicação;
- **Acesso VPN** – O acesso à rede virtual privada (do inglês, Virtual Private Network) é feito de forma rápida e segura, de modo que os colaboradores possam estar conectados ao ambiente de qualquer lugar para dar continuidade à determinada transação ou simplesmente para acesso a suas informações;
- **Integração NAP** – Quando o Forefront TMG é instalado, a funcionalidade do Network Policy Server também é instalada.

Dessa forma pode-se integrar à rede interna ou VPN ao NAP (Network Access Protection), fazendo com que o acesso só seja permitido caso a requisição passe por uma série de regras previamente definidas. Por exemplo, a conexão só será liberada se a estação possuir antivírus atualizado ou tenha um certificado digital emitido por uma autoridade certificadora.

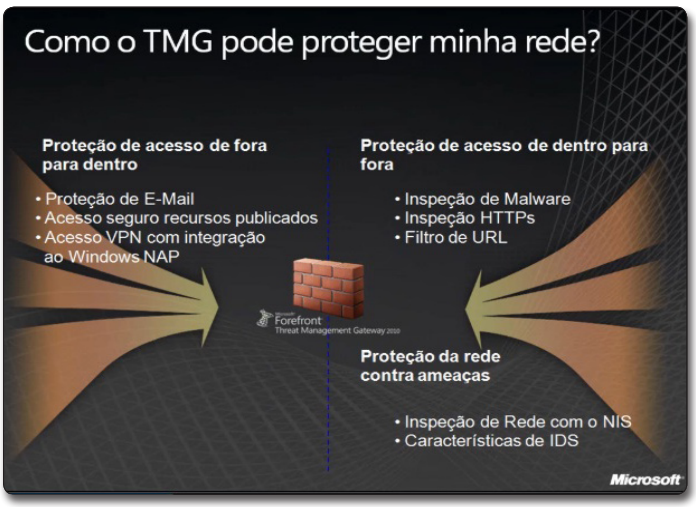


Figura 1. Visualização dos principais pilares do Forefront TMG

Proteção de acessos de fora para dentro

Por sua vez, neste tópico apresentamos a descrição das funcionalidades que protegem os acessos externos da infraestrutura e suas definições:

• **Inspeção Malware** – O serviço de inspeção malware protege seu ambiente das principais ameaças de vírus e worms. Quando uma página acessada contém vírus, o Forefront TMG tenta fazer uma limpeza do arquivo para que a página possa ser entregue ao usuário final. Quando isso não é possível, o download da página é bloqueado e o usuário recebe uma notificação;

• **Inspeção HTTPS** – O serviço de inspeção HTTPS verifica possíveis ameaças que podem vir encapsuladas no protocolo SSL (*Secure Socket Layer*). Com o serviço ativo, o Forefront TMG inspeciona o tráfego SSL, podendo bloqueá-lo caso encontre algo que possa ser prejudicial ao ambiente;

• **Filtro URL**– Este serviço analisa as categorias de cada website visitado, podendo bloquear ou não o seu acesso. Por exemplo, sites categorizados como pornografia, apologia a drogas ou redes sociais podem ser bloqueados e o usuário recebe uma notificação no browser.

Proteção da redes contra ameaças

Por fim, aqui, apresentamos a descrição das funcionalidades que protegem a rede contra as mais variadas ameaças:

• **Network Inspection System (NIS)** – Este serviço protege o ambiente contra vulnerabilidades Microsoft que possam haver nos servidores. Ele evita que um invasor explore uma vulnerabilidade Microsoft em seu servidor até que esta esteja corrigida através da aplicação de patches de segurança;

• **Detecção de Intrusões** – Quando um pacote ofensivo é detectado pelo Forefront TMG, o pacote é derrubado e um alerta é gerado.

Comparativo entre ISA Server e Forefront TMG

As vantagens da migração para o Forefront TMG vão além de suas funcionalidades. Itens como escalabilidade de memória e gerenciamento são motivos concretos para se pensar na migração (para as corporações que ainda usam o ISA Server 2006) ou a aquisição do produto.

A **Figura 2** apresenta um comparativo de todas as funcionalidades presentes do ISA Server 2006 e seu sucessor, o Forefront TMG.

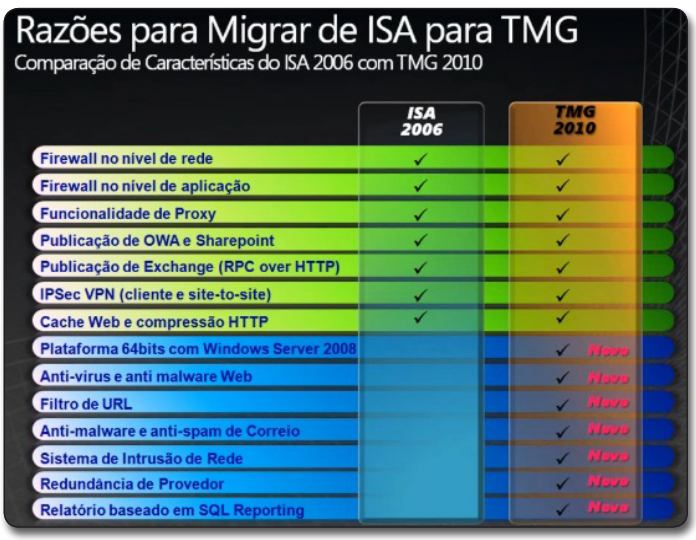


Figura 2. Comparativo entre ISA Server 2006 e o Forefront TMG 2010

Gerenciamento de redes no Forefront TMG

No gerenciamento de redes do TMG muitas funcionalidades permanecem iguais a seu antecessor, porém, o Forefront TMG traz uma característica muito interessante e que vem agregar valor a seu ambiente: o serviço de ISP Redundancy.

Esse serviço faz com que as requisições de acesso a sites na internet sejam direcionadas de um link para outro de acordo com especificações prévias.

O processo funciona para até dois provedores de serviço na internet e pode ser configurado no esquema de Failover (por exemplo, o link 2 é acionado em caso de queda ou demora na resposta do link 1) ou Balanceamento, isto é, à medida que o número de requisições no link 1 estiver alta, o TMG direciona o tráfego para o link 2.

Na prática, o Forefront TMG é configurado com uma placa de rede que irá monitorar as requisições internas e duas placas de rede que serão responsáveis pela navegação na internet em dois provedores distintos e gerenciados pelo ISP Redundancy.

Consoles de gerenciamento – EMS (Enterprise Management Server) e CSS (Configuration Storage Server)

Em uma infraestrutura de servidores Forefront TMG, é possível fazer a união entre os servidores através de arrays, ou seja, conectar

os servidores TMG a fim de que, tudo o que for publicado em um nó da rede seja automaticamente replicado em outro.

Além disso, é possível distribuir a carga entre esses nós de forma a manter a continuidade e desempenho da rede.

Agora, como gerenciar um agrupamento com vários servidores executando o Forefront TMG? Para quem tem esse tipo de cenário, o produto vem com ferramentas de gerenciamento que facilitam e centralizam a administração da infraestrutura de servidores TMG.

A seguir, uma breve descrição dessas ferramentas:

• **EMS -** O Enterprise Management Server é usado para gerenciar servidores Forefront TMG nas versões Enterprise ou Standalone. O EMS deverá ser instalado em um servidor membro de um domínio sem nenhum serviço do Forefront TMG. Não é preciso adquirir uma licença TMG para um servidor com o EMS instalado. Se o administrador está familiarizado com o ISA Server 2006 Enterprise, o Forefront TMG EMS é um pouco semelhante ao Configuration Storage Server (CSS) com algumas diferenças.

Com o Forefront TMG é possível conectar ou desconectar um TMG Server de um array. Desta forma a configuração fica mais flexível porque o administrador não precisará remover e reinstalar o Forefront TMG Server, caso queira conectá-lo em um array diferente do atual. Também é possível atualizar um Forefront TMG Standard para o Forefront TMG Enterprise sem a necessidade de reinstalação.

Lembrando que, instalações do EMS em um Windows Domain Controller não são suportadas pela Microsoft.

• **CSS -** O Configuration Storage Server é usado para instalações de TMG locais. Todo Forefront TMG Server possui um CSS local. Quando o administrador conecta o TMG Server a um TMG Array, ele usará o Enterprise Management Server (EMS). Quando o Enterprise CSS é aplicado, o CSS local (instância AD-LDS) é desativado.

Métodos de autenticação

O produto provê uma variedade de mecanismos de validação usados para pré-autenticar uma requisição. Os métodos de autenticação de um cliente no Forefront TMG são:

• No Authentication;
• Forms-Based Authentication;
• HTTP Authentication;
• Client Certificate Authentication.

O produto valida as credenciais do cliente – passadas em um dos formatos anteriormente citados – utilizando os seguintes provedores e protocolos, que devem ser empregados individualmente:

• **No Authentication** – permitindo que o servidor publicado internamente cuide da parte de autenticação;
• **Windows Active Directory** – fazendo a consulta na infraestrutura de servidores Active Directory;
• **RADIUS** e **SecureID** – Para autenticação externa.

Logging

A parte de *logging* é uma das mais importantes funções de um firewall para gerenciamento de rede e segurança. Além disso, é necessário que esta parte seja muito bem pensada e configurada.

Esse serviço oferece informações importantes acerca do que está acontecendo no firewall e ao seu redor, além de dar informações detalhadas sobre quem está acessando determinado recurso e possíveis ataques.

O *log* do *Forefront* TMG pode ser configurado de várias maneiras, que devem ser implementadas de forma a lhe proporcionar uma visão total do que acontece no ambiente.

Os *logs* podem ser gerados a partir de uma base em SQL Server Express (por padrão na instalação do produto), via arquivos texto ou ainda fazer com que o Forefront TMG direcione a geração de *logs* para um servidor SQL Server em sua rede.

Mas porque o serviço de *Logging* é importante? As empresas usam *logs* de todas as aplicações existentes para fins de auditoria e geralmente os armazenam por um prazo estabelecido em suas normas e processos. Este prazo varia de seis meses a um ano. Além disso, o serviço de *logging* permitirá que o administrador possa monitorar toda a atividade dos usuários, analisar as informações para fins de reparação de erros, verificar a saúde da rede, ou seja, como o tráfego está fluindo, e saber quais alterações foram efetuadas, para que, se necessário, possam ser revertidas.

Baseado em questões de desempenho é recomendado o uso do SQL Server Express (versão 2008 para o Forefront TMG 2010 e versão 2005 para o Forefront TMG MBE) ou o sistema de arquivos de texto.

Muitos problemas de desempenho entre servidores ISA Server 2006 e até mesmo Forefront TMG foram reportados em fóruns, blogs e listas de discussão por conta da opção por usar um SQL Server remoto.

Vários casos foram relatados por clientes e até mesmo situações expostas em blogs, onde um pequeno problema no servidor SQL Server causou problemas de lentidão de internet para uma grande quantidade de usuários.

Neste caso, a resolução do problema e sua análise em si são demoradas, pois, como o primeiro ponto de análise é sempre o servidor do Forefront TMG, as análises em cima do SQL Server remoto serão as últimas ações a serem tomadas.

Quando se transfere os *logs* do Forefront TMG para o SQL Server remoto normalmente não se usa um servidor dedicado. Os *logs* do Forefront TMG são armazenados em uma instância com outros bancos de dados que armazenam dados de outras aplicações. Dessa maneira a possibilidade de problemas de desempenho causados por esta forma de geração de *logs* é muito grande.

Além disso, não basta simplesmente direcionar o servidor Forefront TMG para o SQL Server remoto. É necessário executar alguns scripts no SQL Server remoto a fim de que a base seja criada corretamente. Este procedimento está disponível em *http://technet.microsoft.com/en-us/library/bb794867.aspx*.

Se, mesmo assim, o leitor optar pelo serviço de *logs* em SQL Server remoto, o suporte Premier Microsoft estará à disposição para ajudá-los!

Logging Queue

A funcionalidade de *logging queue* foi criada para evitar parada do Forefront TMG em caso de algum erro no serviço de *logs*.

O *Large Logging Queue* (*LLQ*) grava as entradas de log quando os mesmos não podem ser salvos em seu destino pré-definido (por default o SQL Server Express Edition). O *LLQ* roda em modo Kernel (parte do Sistema Operacional sem a intervenção do usuário) a partir do engine TMG (*FWENG.SYS*) e no Modo Usuário (Dispatcher). O processo no Modo Usuário somente lê os dados a partir do Disco Rígido, enquanto o modo do processo Fweng grava no HD.

Caso o acesso à área de *logs* (SQL Server Expression Edition ou outra base SQL) esteja indisponível, os arquivos começam a ser gravados em fila na pasta onde foi definida a gravação dos *logs* no Forefront TMG. Quando o acesso à área de *logs* em sua base de dados é restabelecido, os dados destes arquivos são gravados em sua base SQL Server e removidos do diretório.

Agora, o leitor pode questionar: se esta funcionalidade existe, então porque não é recomendado o uso de um SQL Server remoto, já que *log queueing* poderia funcionar de forma paliativa até que alguém resolva o problema do SQL usado como base para os logs.

Esclarece-se que, mesmo com possíveis problemas de desempenho ou algo qualquer no servidor remoto, o Forefront TMG ainda entenderá que o serviço está operacional, independente de qualquer coisa e desta forma continuará esperando respostas da base de dados alocada em outro servidor.

Outra questão é o momento em que o servidor remoto sai do ar e os arquivos LLQ começam a ser gerados. No momento em que a conexão for restabelecida, o Forefront TMG iniciará o processo de *commit* dos dados gerados nas filas para a base de dados. Dependendo da hora e da quantidade de dados a serem migrados, podem ocorrer problemas de desempenho no acesso à internet e até mesmo em outras aplicações, caso esteja usando uma instância compartilhada. Aí existe mais uma razão para evitar o uso de um SQL Server remoto para geração de *logs* em uma infraestrutura de servidores Forefront TMG.

Uma recomendação importante é o uso de um disco em um array exclusivo para a gravação de *logs*. De preferência, uma arquitetura RAID1 ou RAID10 para uma gravação com melhor desempenho.

A escolha entre as arquiteturas RAID1 e RAID10 ocorrerá de acordo com o tamanho da infraestrutura a ser montada.

Network Load Balance

Desde seu antecessor, o ISA Server 2006, o produto conta com o serviço de balanceamento de carga de rede que proporciona alta disponibilidade à infraestrutura de *proxies* em Forefront TMG.

Todas as recomendações, limites e limitações deste serviço são exclusivos do sistema operacional Windows, com algumas exceções.

Para a maioria das infraestruturas Windows Server com NLB, é possível adicionar até 32 servidores, mas, no caso do Forefront TMG, este número se limita a oito servidores. Por esta razão, é necessário um bom entendimento e planejar antes de implementar.

Apenas para relembrar alguns pontos: cada nó participante de um cluster NLB tem um endereço IP único, conhecido como DIP (*Dedicated IP Address*). Todos os nós em um array NLB compartilham um conjunto de endereços IP comuns, conhecido como VIP (*Virtual IP Addresses*).

A **Figura 3** descreve a arquitetura NLB.

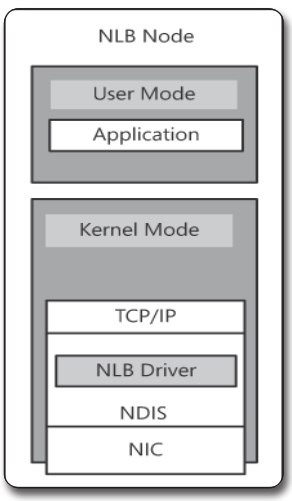


Figura 3. Arquitetura NLB

O serviço de NLB tem três modos de operação, os quais são usados para determinar como será a comunicação com os clientes e entre os nós no array. Estes modos são denominados **Unicast**, **Multicast** e **Multicast com IGMP**. Independente do modo a ser usado, o endereço MAC (endereço físico dos NIC´s) utilizado no tráfego é configurado para ser o mesmo para todos os nós.

O serviço de NLB dentro do Forefront TMG pode ser usado nos modos **Integrated** e **Non-Integrated**. O modo Integrated é aquele configurado através da console do produto. O modo Non-Integrated é configurado através do Sistema Operacional.

Quando o modo Integrated é habilitado, a comunicação ocorre em modo Unicast. Desta forma, os pacotes de requisição são entregues paralelamente a todos os nós e o driver NLB filtra os pacotes que não deverão ser processados por um dos nós em particular. No modo Multicast o acesso ocorre via endereço MAC para os NIC´s de todos os nós participantes do cluster NLB.

É importante lembrar que o NLB Windows, tanto o integrado ao TMG quanto o não integrado, suporta tráfego de até 500 Mbps. Acima disso é recomendado o uso de dispositivos que gerenciem o balanceamento de carga entre os nós.

Caso a corporação opte por usar um balanceador externo, é importante que o produto trabalhe com o conceito de **stickiness**,

que mantém uma determinada sessão sempre indo e voltando pelo mesmo nó. Dessa forma o usuário não sofrerá com queda de conexão em um acesso a sites de banco ou algum outro acesso em que tenha formulários para preencher.

Uma dica para quem vai implementar NLB é sobre sua pilha de switches. É necessário que, de modo a garantir a convergência entre os nós, o switch seja de layer 2. Caso a pilha seja layer 3, é necessário que uma das portas simule o layer 2. Se não for usar NLB na infraestrutura de servidores Forefront TMG, poderá usar qualquer um dos dois.

Para informações mais detalhadas sobre o funcionamento do NLB, consulte a documentação em *http://technet.microsoft.com/en-us/library/cc756878.aspx*. E para informações sobre as definições de switches de layer 2 e 3, consulte a documentação em *http://www.malima.com.br/article_read.asp?id=30*.

Conceitos de Caching

Uma das principais razões para implementação de um *Proxy* é a necessidade da redução de consumo de largura de banda da rede. O Forefront TMG oferece um controle granular de grande qualidade sobre o gerenciamento de arquivos e conteúdos de cache.

O uso do cache é um dos fatores a se considerar acerca de desempenho nos acessos e downloads dos conteúdos das páginas web. O cache no Forefront TMG faz com que conteúdos frequentemente acessados estejam previamente disponíveis no próprio servidor e não use sua banda de rede toda vez que forem solicitados.

Entre os benefícios do cache, se pode dizer que o acesso à internet fica mais rápido, reduz o tráfego e proporcionará menor consumo da banda de rede.

Além disso, o administrador pode definir quais sites poderão ter seu conteúdo armazenado em cache e periodicamente verificar se determinada página teve seu conteúdo atualizado.

Para infraestruturas do Forefront TMG interligadas em arrays ou NLB, é possível usar o serviço do CARP (*Cache Array Routing Protocol*). Assim, se a requisição entrar pelo nó 1 e o serviço do CARP estiver habilitado, o Forefront TMG verificará a existência da página a ser acessada no mesmo nó. Caso não encontre a referência à página, o Forefront TMG então faz a leitura no cache do nó 2. Caso o serviço de cache do nó 2 esteja fora e a requisição entrar por este nó, o Forefront TMG lerá os dados armazenados no cache do nó 1.

Esta funcionalidade foi herdada do ISA Server 2006, continua usual e muito útil também no Forefront TMG.

Versões do produto

O produto é comercializado nas versões Standard e Enterprise. No entanto, somente é possível de ser instalado em um servidor com Windows Server 2008 – 64 bits.

Atualmente o produto está no Service Pack 2, e mesmo com notícias da Empresa Gartner acerca da descontinuidade do mesmo, este ainda possui pelo menos mais 10 anos de suporte e atualizações.

Para ambientes menores, existe também o Forefront TMG Medium Business Edition (MBE), ideal para médias e pequenas empresas. Porém, é um produto que possui algumas limitações.

Esta versão somente pode ser instalada em um servidor membro de domínio. Além disso, não dá suporte para array e o CSS usado só será aquele instalado no mesmo servidor do MBE.

A instalação completa do MBE é composta pelo próprio produto em si, o Active Directory Lightweight Directory Services (LDS) e o SQL Express 2005 (usado para a parte de relatórios).

Alguns destes componentes poderão ser removidos, mas a Microsoft recomenda que todos eles sejam mantidos para um bom funcionamento do MBE. É possível também verificar se as instalações dos componentes estão pré-definidas a serem feitas no drive C. Se for preciso, pode-se mudar a localização das pastas, porém, o SQL 2005 Express e o LDS permanecerão instalados na pasta padrão *%ProgramFiles%*.

Apesar da existência desta versão, é recomendado que seja usada a versão *full* do produto. Mesmo em pequenas e médias empresas.

Planejamento

A instalação do Forefront TMG deve ser cuidadosamente planejada para que se evitem problemas de desempenho.

A Microsoft oferece gratuitamente uma planilha que ajuda na escolha do melhor equipamento e formas de instalação, e que pode ser baixada no endereço *http://www.microsoft.com/download/en/details.aspx?id=15196*.

O Forefront TMG 2010 Capacity Planning Tool permite o cálculo da banda WAN e do número de usuários que um determinado hardware vai suportar, além de recomendar o hardware ideal para sua instalação.

Os requisitos de hardware para servidores com Forefront TMG variam e são dependentes de alguns fatores:

- As funcionalidades que o usuário habilita;
- O número de conexões concorrentes;
- Link de rede disponível.

Para detalhes adicionais, a planilha traz todas as instruções de preenchimento na aba "Instructions".

Licenciamento

O valor da licença é estabelecido por processador. Os serviços de Web Filtering, Network Inspection e Malware Inspection são licenças cobradas à parte, pois utilizam o serviço do MRS (*Microsoft Reputation Services*) para sua atualização. Para mais detalhes sobre a aquisição do produto, contate o representante Microsoft.

Conclusão

Este artigo buscou apresentar as funcionalidades e vantagens que o Forefront Threat Management Gateway pode trazer às corporações, além das melhores práticas para configuração e

sugestões de instalação. O Forefront TMG pode ser usado como firewall de borda ou simplesmente ser o Proxy da rede. Também é possível publicar aplicações web com o objetivo de aumentar o nível de segurança no acesso.

Os serviços de Inspeção SSL, Malware e Redes vêm como diferencial do produto para tornar o tráfego interno e externo mais seguro e confiável. Por sua vez, o serviço de filtro de conteúdo garante que sites mal intencionados ou que não sejam interessantes aos objetivos da corporação não sejam acessados.



Uilson Souza
souzajr.nc@uol.com.br
http://uilson76.wordpress.com
Atua no ramo de tecnologia e infra estrutura há mais de 16 anos, mantenedor de blog técnico voltado a implementação e suporte de Forefront TMG, ISA Server e demais ferramentas Microsoft, atua como Analista de Projetos. É Tecnologo em Processamento de Dados pela UNIBAN, Microsoft Certified: MCTS ISA Server 2006 Configuring. Colunista dos portais TI Especialistas e GrupoIT4All, atua também na Comunidade Técnica Microsoft-SP e nos fóruns TechNet Brasil. Também colabora com artigos técnicos para o portal Microsoft TechNet Wiki. Desde Outubro/2011 é membro do programa MTAC (Microsoft Technical Audience Contributor).

ISATools.Org: Site do Engenheiro Jim Harisson com diversos scripts e ferramentas que irão auxiliar no dia a dia do administrador ISA Server ou Forefront TMG.
www.isatools.org

Artigo "Deployment do Forefront TMG"
www.microsoft.com/forefront/en/us/deployment.aspx

Blog "Equipe ISA Server/TMG"
http://blogs.technet.com/b/isablog

Site "TechNet Virtual Lab:Forefront Threat Management Gateway (TMG)
http://cmg.vlabcenter.com/default.aspx?moduleid=3f9b93f1-f9d2-47bc-b2e2-e64cf45bf19e

Site "Estudos de caso Forefront TMG"
www.microsoft.com/forefront/threat-management-gateway/en/us/case-studies.aspx

Blog "Yuri Diógenes – Senior Technical Writer – Microsoft Texas"
http://blogs.technet.com/b/yuridiogenes

Blog "Uilson Souza – Blog MCTS"
http://uilson76.wordpress.com

Livro "Microsoft Forefront Threat Management Gateway (TMG) – Administrator´s Companion"
http://www.microsoft.com/learning/en/us/Book.aspx?ID=13148

Conhecimento faz diferença!



Faça já sua assinatura digital ! | www.devmedia.com.br/es

Faça um *upgrade* em sua carreira

Em um mercado cada vez mais focado em qualidade, ter conhecimentos aprofundados sobre requisitos, metodologia, análises, testes, entre ouutros, pode ser a diferença entre conquistar ou não uma boa posição profissional. Sabendo disso a DevMedia lançou uma publicação totalmente especializada em Engenharia de Software. Todos os meses você pode encontrar artigos sobre Metodologias Ágeis; Metodologias tradicionais (document driven); ALM (application lifecycle); SOA (aplicações orientadas a serviçoes); Análise de sistemas; Modelagem; Métricas; Orientação à Objetos; UML; testes e muito mais. **Assine Já!**

DevMedia

FreeBSD com proxy Squid

Controle, filtre e agilize o tráfego da rede com FreeBSD e Squid

Rogério Tomassoni de A. Junior

Foi publicado na primeira edição da Infra Magazine um artigo sobre como instalar o sistema operacional FreeBSD para atuar como um gateway. O gateway é o host que atua como um concentrador de tráfego, situado na fronteira entre a rede interna e a externa, sendo por meio dele que as máquinas da rede interna acessam a internet.

Para complementar o artigo da primeira edição, será mostrado como adicionar um servidor proxy ao gateway com o intuito de controlar o acesso e manter a correta utilização da internet, em função do uso a que esta se destina.

Outra utilidade propiciada por seu uso é o controle de acesso, que se baseia em diferentes métodos de autenticação, como: fornecimento de usuário e senha, pelo domínio da rede, pela estação de trabalho, em função do horário do acesso, por agrupamento de regras, entre outros.

Além dos benefícios citados, o Squid aumenta a economia da largura de banda, o que melhora a velocidade de entrega do conteúdo solicitado pelo cliente, permite a criação de regras de controle de acesso à internet e autenticação do usuário, e armazena o conteúdo solicitado para que futuras solicitações não necessitem buscar as mesmas informações na internet, fazendo a leitura diretamente de seu cache. Neste caso, quando é requisitado um conteúdo que sofreu atualização após seu armazenamento em cache, o Squid atualiza tais dados e entrega-os ao cliente, garantindo economia de banda.

Todas as funcionalidades do Squid geram logs que possibilitam ao proxy a disponibilização de relatórios completos, tornando mais fácil a identificação de possíveis problemas, bem como o acompanhamento do desempenho da rede.

Resumo DevMan

De que se trata o artigo:

Este artigo visa demonstrar os procedimentos de instalação e configuração do proxy Squid no sistema operacional FreeBSD. Após tais procedimentos, o Squid será responsável por controlar e gerenciar todos os acessos à internet, oferecendo inúmeras vantagens na administração de uma rede de computadores.

Em que situação o tema é útil:

O Squid é muito útil em ambientes onde existe a necessidade de controle sobre o acesso à internet, sobretudo de seu conteúdo, proporcionando economia de banda, uma vez que possui um cache dos conteúdos acessados. Dentre seus benefícios, destaca-se a melhoria na segurança da rede, em função do controle de acesso e uma sensível diminuição do esforço administrativo.

FreeBSD com proxy Squid:

Este artigo abordará de forma simples e prática como instalar e configurar o proxy Squid, para que atue como cache dos conteúdos acessados na internet. Sendo assim, será possível entender os arquivos de configuração do Squid e a forma como este serviço se comportará na infraestrutura de rede, o que possibilitará ao leitor fazer customizações e adequar esta leitura às necessidades de seu ambiente.

O proxy Squid

O Squid é um servidor proxy, open source (código aberto), empregado em atividades relacionadas ao tráfego de pacotes na rede. Dentre as funcionalidades disponíveis, destacam-se: o filtro de conteúdo, que registra em arquivo todo o conteúdo acessado pelas estações da rede; o bloqueador de conteúdo, cuja funcionalidade é restringir o acesso a informações consideradas indesejadas pelo administrador; o autenticador de acesso, que concede acesso à internet somente a usuários conhecidos; e, uma de suas principais características, o cache de conteúdo, que armazena conteúdos recentemente visitados, proporcionando menor tempo de resposta às solicitações do usuário e diminuindo o tráfego entre a rede local e a internet. Cada uma das *features* destacadas pode ser utilizada individualmente ou em conjunto. Dessa forma, é mantido o controle e registro das atividades dos usuários que fazem uso da internet, permitindo investigações futuras e o próprio monitoramento de acesso em tempo real.

Tantos serviços oferecidos e a disponibilidade de suas versões para os sistemas operacionais das famílias Unix e Windows, além do suporte aos navegadores mais utilizados no mundo, como: Firefox, Internet Explorer e Google Chrome, e a facilidade que o usuário tem em configurar seu browser para o uso deste proxy, fazem do Squid uma excelente escolha para gerenciar a internet como recurso de rede.

O cenário

Como ferramenta para a elaboração deste artigo, foi utilizada uma rede de microcomputadores contendo um host com a funcionalidade de gateway e os demais hosts para atuarem como clientes deste gateway.

O servidor proxy foi instalado no gateway com o objetivo de disponibilizar o acesso à internet mantendo certo nível de controle quanto aos conteúdos acessados.

As solicitações de acesso à internet feitas pelos clientes acontecem de forma controlada, e possíveis tentativas de acesso indevido aos recursos da rede são negadas e registradas em arquivos de log, objetivando a identificação do usuário.

Neste contexto, cabe aos administradores a definição das regras que serão aplicadas ao Squid, sejam eles profissionais ou educacionais, a fim de restringir determinados acessos, a exemplo de conteúdo pornográfico.

Para tanto, será utilizado um microcomputador com o sistema operacional FreeBSD, conforme a **Figura 1**, compilado com parâmetros do kernel, para atuar como gateway (na primeira edição da Infra Magazine foi mostrado como realizar esta compilação). Após este procedimento, o proxy Squid será instalado e configurado com algumas de suas funcionalidades básicas.

Para atuar como gateway e possivelmente como um proxy, um microcomputador bem modesto atenderá aos requisitos de hardware. Porém, é indicado o uso de um que o disco rígido esteja em bom estado, já que ele será altamente exigido pelo cache do proxy.

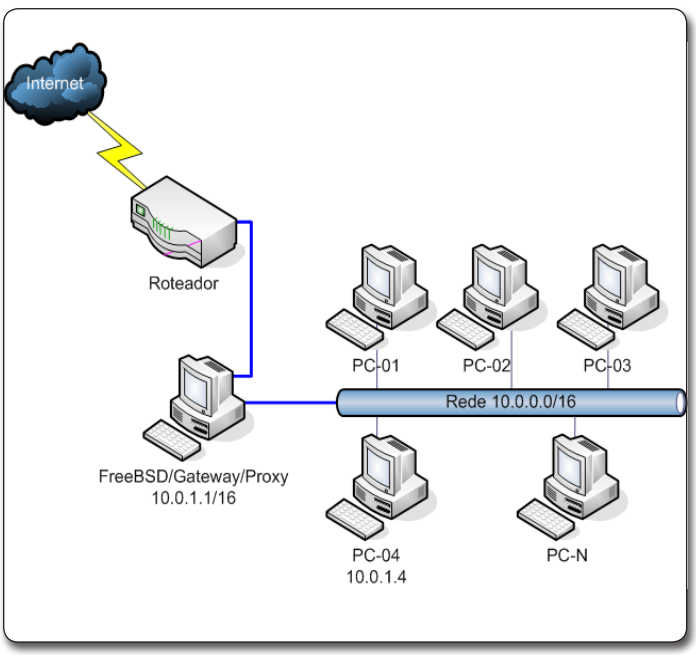


Figura 1. O cenário da rede

O microcomputador usado no cenário foi um Intel(R) Atom(TM) CPU 330 @ 1.60GHz, 1GB de RAM, HD Maxtor 80GB, com duas placas de rede, sendo uma onboard e outra offboard: sis 900 e RT8139(A/B/C/810x/813x/C+) Fast Ethernet Adapter. A conexão com a internet é via *cable modem*, onde o provedor fornece o endereço IP automaticamente via DHCP (*Dynamic Host Configuration Protocol*).

Vale ressaltar que o tráfego das estações será direcionado ao gateway que, por sua vez, redirecionará, por meio do firewall IPFW (Firewall padrão do FreeBSD), todo o tráfego destinado à web pela porta padrão do Squid (3128), que aplicará suas regras – chamadas de ACL (Lista de Controle de Acesso).

A instalação

A versão do sistema operacional utilizada foi a FreeBSD 7.2-RELEASE i386 (veja a seção **Links**), e a versão do Squid foi a 2.6.STABLE23 (veja seção **Links**).

Neste exemplo, a instalação do proxy será a partir do código fonte do Squid, para que seja possível realizar os passos aqui descritos também em outras distribuições Linux, tais como Red Hat, Slackware, dentre outras, já que o processo de instalação e configuração é idêntico. Há administradores que optam por realizar a instalação através dos ports (*/usr/ports*), no entanto, ambas as formas levam a resultados idênticos.

Instalando o Squid

Após fazer o download do Squid em *www.squid-cache.org/Versions/v2/2.6/*, deve-se acessar o diretório onde foi salvo o arquivo referente a seu código fonte, por exemplo */home/dmedia/squid-2-.6.STABLE23.tar.gz* e descompactá-lo com o utilitário tar (*tar -jxvf squid-2.6.STABLE23.tar.gz*). Feito isso, será criado um diretório chamado *squid-2.6.STABLE23*.

Deve-se logar com o super usuário do sistema (*su -*) e, em seguida, executa-se o script de configuração do Squid. Em função da grande quantidade de parâmetros e funcionalidades, este artigo mostrará os mais utilizados para um servidor proxy simples, porém funcional.

O script de pré-configuração do Squid chama-se *configure*. Através dele serão realizadas as configurações básicas de pré-instalação, como a localização do diretório raiz para instalação, tipo de sistema de arquivos que será reconhecido pelo Squid, dentre outras. Porém, algumas funções não estão disponíveis através do arquivo de configuração (*squid.conf*), a exemplo dos tipos de sistema de arquivos reconhecidos, que deverá ser informado neste ponto, através do script de configuração, da seguinte maneira:

```
./configure --prefix=/usr --exec-prefix=/usr --sbindir=/usr/sbin --sysconfdir=/etc
--mandir=/usr/share/man --bindir=/usr/sbin --libexecdir=/usr/lib/squid --enable-snmp
–enable-storeio=aufs,ufs,null --enable-auth=basic --enable-basic-auth-helpers=NCSA
--enable-ident-lookups --with-large-files --enable-arp-acl
```

Quando executado, uma longa lista de arquivos e procedimentos de pré-compilação será exibida, estando inclusos nela a verificação de todos os requisitos necessários para o Squid funcionar com os parâmetros indicados. Então, para que sejam evitados problemas com dependência de arquivos, a instalação do FreeBSD foi feita no modo standard.

Conforme mostrado no comando "./configure...", foi passada como parâmetro uma lista de informações de configuração, cuja sintaxe a ser seguida é a seguinte: *variável=parâmetro*, onde variável é uma palavra reservada do script de configuração do Squid, e parâmetro pode ser um caminho completo para um diretório ou o nome de um protocolo, assim como mostrado na **Tabela 1**, onde se tem uma breve explicação sobre cada parâmetro utilizado.

Com estas configurações, será possível que o Squid faça o cache, filtre os pacotes por endereço MAC, reconheça arquivos grandes, permita autenticação básica com nome de usuário e senha,

Parâmetro	Descrição
--prefix=/usr	Indica o prefixo, isto é, o diretório raiz para instalação dos componentes do Squid.
--exec-prefix=/usr	Indica o prefixo, isto é, o diretório raiz para instalação dos componentes executáveis do Squid.
--sbindir=/usr/sbin	Indica onde devem ser colocados os componentes binários, para manipular o Squid.
--sysconfdir=/etc	Indica a localização dos principais arquivos de configuração do Squid (cachemgr.conf, squid.conf e outros – cuja explicação será apresentada mais adiante).
--libexecdir=/usr/lib	Indica em que local manter as bibliotecas usadas pelo Squid.
--bindir=/usr/sbin	Indica onde estará o principal binário do Squid.
--enable-snmp	Habilita o protocolo SNMP (Simple Network Management Protocol), interessante para manter monitoramento.
--enable-storeio=aufs,ufs,null	Indica quais são os tipos de sistemas de arquivos que o Squid usará na manipulação do cache em disco (existem outras opções como: coss e diskd).
--enable-auth=basic	Habilita o modo de autenticação básico.
--enable-basic-auth-helpers=NCSA:	Indica ao Squid qual protocolo será utilizado para autenticação, no caso NCSA (NCSA-style é um arquivo cujo formato é um nome de usuário e sua senha, sendo um por linha).
--enable-ident-lookups	Habilita o Squid a utilizar o protocolo ident (RFC 1413). Este protocolo identifica o nome de usuário do sistema operacional de modo transparente, ou seja, não é preciso que sejam feitas novas solicitações.
--with-large-files	Habilita suporte a arquivos grandes de cache, maiores que 2GB.
--enable-arp-acl	Habilita suporte a filtragem por MAC (Media Access Control).

Tabela 1. Parâmetros de configuração do Squid

realize a identificação pelo nome do usuário logado no sistema operacional, habilite o uso do protocolo de gerenciamento SNMP, e habilite o reconhecimento dos sistemas de arquivos mais comuns e utilizados.

Após executar o script *configure*, será necessário compilar o Squid com o comando *make* e posteriormente fazer sua instalação com o comando *make install*.

Para agilizar este processo, os comandos serão utilizados de forma sequencial, *make && make install*. O símbolo *&&* diz que só se deve executar o comando mais a direita se o da esquerda for bem sucedido, ou seja, se seu retorno for 0. Se tudo ocorrer bem com o comando acima, haverá um retorno em tela semelhante à **Figura 2**.

```
Making install in ntlm_auth
Making install in negotiate_auth
Making install in digest_auth
Making install in external_acl
Making install in tools
test -z "/usr/sbin" || ../cfgaux/install-sh -c -d "/usr/sbin"
  /usr/bin/install -c 'squidclient' '/usr/sbin/squidclient'
test -z "/usr/lib/squid" || ../cfgaux/install-sh -c -d "/usr/lib/squid"
  /usr/bin/install -c 'cachemgr.cgi' '/usr/lib/squid/cachemgr.cgi'
/usr/bin/install -c -m 644 ./cachemgr.conf /etc/squid/cachemgr.conf
freeBSD#
```

Figura 2. Compilando e instalando o Squid

A configuração do Squid

Toda configuração do Squid é feita através de arquivos em formato texto, e o principal arquivo é o *squid.conf* . Nesta instalação, ele deve estar situado no local informado como parâmetro na variável *sysconfdir*, logo, estará em */etc/squid/squid.conf*.

Configurando o Squid

Como boa prática, deve-se fazer uma cópia do arquivo original antes de iniciar as modificações, pois caso aconteça qualquer tipo de problema com o arquivo manipulado o arquivo original continua à disposição para restauração. Para proceder com as configurações, é imprescindível criar um usuário e grupo para que o processo do Squid o utilize, pois não é seguro deixá-lo com usuários *root* ou *nobody*. Usuários com privilégios além do necessário podem ser explorados em falhas/bugs por pessoas maliciosas para causar danos ao sistema.

Ao especificar um usuário, deve-se delimitar seu acesso somente aos recursos necessários para o funcionamento do sistema. Pensando nisso, será criado um grupo com o comando *pw groupadd squid* e, em seguida, um usuário, já o inserindo no grupo e com o *shell* (programa que interpreta comandos e os repassa ao kernel do sistema operacional) inválido (Squid não utiliza o shell), através do comando *pw adduser squid -g squid -s /dev/null*. Feito isso, deve-se modificar o dono dos arquivos de configuração com o comando *chown*, a fim de que o usuário *squid* possa acessá-los.

Conforme observado na **Figura 3**, o usuário não necessita de um shell válido (*-s /dev/null*), o que melhora a segurança, uma vez que foi restringido que este não terá como executar qualquer comando referente ao sistema operacional, caso ele venha a ser utilizado para fins maliciosos.

```
freeBSD# pw groupadd squid
freeBSD# pw adduser squid -g squid -s /dev/null
freeBSD# chown squid /etc/squid/squid.conf
freeBSD# chown -R :squid /etc/squid/squid.conf
freeBSD# ls -la /etc/squid/squid.conf
-rw-r--r--  1 squid  squid  155769 Jul 28 20:01 /etc/squid/squid.conf
```

Figura 3. Criando o grupo, usuário e editando as permissões

Criando a estrutura de diretórios para o cache

É possível criar os diretórios necessários ao funcionamento do Squid em discos separados, o que aumenta a performance em função de os discos trabalharem em paralelo, e a segurança.

Contudo, para este artigo, o disco será apenas particionado, técnica conhecida como *slice*, que divide um disco físico logicamente fazendo com que o sistema operacional trabalhe como se houvesse tantos discos quanto as partições que foram criadas.

O Squid faz uso do disco proporcionalmente à quantidade de usuários e acessos. Logo, quanto maior o número de usuários e seus acessos, maior será o número de registros escritos em disco, o que demanda especial atenção à área em disco destinada ao cache.

O Squid necessita que o administrador crie uma estrutura de diretórios para que ele possa utilizar durante suas operações. Esta estrutura servirá para armazenamento de dados de cache, registros de log, entre outros, conforme mostrado abaixo:

• Para o cache deve haver o diretório */var/spool*. Para tanto os comandos: *cd /var/spool* e, em seguida, *mkdir squid* devem ser utilizados;

• Para arquivos de registros/log, onde serão mantidos os históricos de acesso, deve-se criar o diretório */var/log/squid*, através dos comandos *cd /var/log* e *mkdir squid*. Após sua criação, deve-se alterar o dono dos diretórios recém-criados para que o Squid possa escrever nos mesmos, conforme os comandos mostrados na **Figura 4**.

```
freeBSD# cd /var/spool/
freeBSD# mkdir squid
freeBSD# cd /var/log
freeBSD# mkdir squid
freeBSD# chown squid /var/spool/squid/
freeBSD# chown squid /var/log/squid/
```

Figura 4. Criando diretórios

Editando o squid.conf

Em princípio, será mostrada a configuração básica do Squid, o que permitirá o uso de seu cache, bem como o acesso total pelos usuários, ou seja, sem qualquer restrição, mantendo todo registro de quem, onde e quando foi realizado cada acesso, porém forçando a configuração do proxy no navegador do usuário.

Para tal configuração é necessário editar o arquivo *squid.conf*, inserindo as linhas descritas na **Listagem 1**. Nos próximos tópicos serão atribuídas mais configurações ao Squid.

Para melhor entendimento, segue uma breve descrição sobre os parâmetros do arquivo *Squid.conf*:

• **http_port 3128:** Define a porta que será utilizada pelo Squid, sendo 3128 a porta padrão. Pode-se mudá-la desde que as portas sejam acima de 1024 e, no máximo, 65535, pois abaixo dessa faixa é reservada para outros serviços. Para verificar portas e serviços, consulte o arquivo */etc/services*. Nele existe o número, protocolo e descrição de cada serviço;

• **cache_mem 256 MB:** É a quantidade de memória utilizada para manter os objetos mais recentes, chamados de objetos quentes, e objetos em transição (por exemplo, dados de páginas web). Desta forma o Squid consulta a memória ao invés do disco, o que melhora seu desempenho, já que o acesso a memória é muito mais rápido que o acesso em disco.

Listagem 1. Squid.conf básico.

```
http_port 3128
cache_mem 256 MB
hierarchy_stoplist cgi-bin ?
maximum_object_size_in_memory 8 KB
minimum_object_size 0 KB
maximum_object_size 250 MB
dns_nameservers endereço1 endereço2
error_directory /usr/share/errors/Portuguese
visible_hostname filtro.de.conteudo.FreeBSD.DevMedia
shutdown_lifetime 1 second

cache_dir aufs /var/spool/squid 5000 32 512
coredump_dir /var/spool/squid

cache_effective_user squid
cache_effective_group squid

cache_access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log
pid_filename /var/run/squid.pid

acl all src 0.0.0.0/0.0.0.0
acl redeInterna src 10.0.0.0/255.255.0.0
acl localhost src 127.0.0.1/255.255.255.255

acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY

acl manager proto cache_object
acl SSL_ports port 443 563          # https,snews
acl Safe_ports port 80              # http
acl Safe_ports port 21              # ftp
acl Safe_ports port 443 563         # https
ts port 70                          # gopher
acl Safe_ports port 210             # wais
acl Safe_ports port 1025-65535      # unregistered ports
acl Safe_ports port 280             # http-mgmt
acl Safe_ports port 488             # gss-http
acl Safe_ports port 591             # filemaker
acl Safe_ports port 777             # multiling http
acl purge method PURGE
acl CONNECT method CONNECT
no_cache deny SSL_ports

http_access allow manager localhost
http_access deny manager all
http_access allow purge localhost
http_access deny purge
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports

http_access allow localhost
http_access allow redeInterna
http_access deny all
```

• Para especificar o tamanho ideal, deve-se considerar que para sistemas de 32 bits o Squid usa aproximadamente 10MB de memória por GB no total dos diretórios de cache físico *cache_dir*. Logo, recomenda-se ter o dobro de memória RAM física com relação à especificada, e como os dados são armazenados em blocos de 4 KB, deve-se sempre especificar o tamanho como múltiplo de 4.

• O **cache_mem** nunca deve ser maior que o tamanho físico do *cache_dir*. Para um **cache_mem** de 128MB, tem-se um *cache_dir* de 512MB, dada a fórmula 128 x 4 = 512.

• Caso o cálculo seja feito em função do tamanho destinado ao *cache_dir*, por exemplo, 5GB, tem-se acréscimo à fórmula de 10MB por GB, ou seja, 50MB, e o cálculo é dado pelos seguintes valores: ((128 MB + 50 MB) * 4 KB = 712 MB), logo o valor ideal, neste caso, será de 712MB.

• Como a memória deve ser utilizada também pelo sistema operacional, tem-se como boa prática configurar o dobro. Portanto, este servidor deve ter 1,5GB de memória para não gerar transtornos ou lentidão. Nota-se que ter um cache em disco grande com pouca memória não é vantajoso, uma vez que o Squid fará muito swap comprometendo o desempenho;

• **hierarchy_stoplist cgi-bin ?:** Define quais palavras que, quando encontradas em uma URL, não devem ser armazenadas em cache, pois possuem poucas chances de serem novamente requisitadas, o que fará com que esta configuração poupe o cache, aumentando o desempenho do Squid;

• **maximum_object_size_in_memory 8 KB:** Tamanho máximo de um objeto em memória. Objetos maiores não ficarão no cache de memória. Logo, o parâmetro deve ter um valor razoável e múltiplo de quatro, conforme mostrado anteriormente. O valor de 8 KB é o padrão, sendo ideal para servidores com bastantes requisições e com a quantidade de memória especificada acima;

• **minimum_object_size 0 KB:** Define que os objetos menores que o tamanho configurado não serão salvos em disco. O valor 0 (zero) indica que não há limite. Neste caso todos os objetos serão salvos;

• **maximum_object_size 250 MB:** Define que os objetos maiores que o tamanho determinado não serão salvos em disco. Manter esse valor alto exigirá muito da largura de banda. Portanto, deve-se ajustar de acordo com o ambiente, sendo interessante que seja um valor médio referente à largura de banda;

• **dns_nameservers endereco1 endereco2:** Determina os servidores DNS utilizados pelo Squid. Os endereços devem ser separados por espaço. O DNS configurado no sistema operacional (*/etc/resolv.conf*) será ignorado se essa opção for encontrada no Squid. Logo, deve-se inserir tanto o DNS configurado no *resolv.conf* quanto os adicionais no *Squid.conf* ou remover essa linha para que seja usado apenas o DNS configurado no *resolv.conf*;

• **error_directory /usr/share/errors/Portuguese:** Decide onde o Squid deve procurar as páginas de erro a serem exibidas quando uma regra for aplicada. Existem páginas próprias para cada tipo de erro que foram traduzidas em várias línguas, sendo possível customizá-las ou desenvolver novas páginas que devem ser visualizadas quando regras específicas forem aplicadas, o que melhora o entendimento do usuário sobre o correto uso da rede;

• **visible_hostname filtro.de.conteudo.FreeBSD.DevMedia:** Define o nome do host/servidor que o Squid exibirá, quando solicitado;

• **shutdown_lifetime 1:** Define quanto tempo o Squid deve aguardar até que as requisições sejam finalizadas para encerrar seu próprio processo. Deve-se definir um valor baixo para que a ação seja rápida e não cause transtornos por demora na restauração;

• **cache_dir ufs /var/spool/squid 5000 32 512:** Um dos parâmetros fundamentais ao funcionamento do Squid (o parâmetro de instalação *--enable-storeio* afeta diretamente esta opção). É ele quem define como será o cache, e sua sintaxe é explicada na **Tabela 2**.

Sendo assim, foi definido que o tipo de armazenamento é o **aufs** (*Assíncrono Unix File System*), a localização do diretório raiz do cache será */var/spool/squid*, e o tamanho máximo do disco, que pode ser usado para os diretórios */cache*, será de 5GB.

O parâmetro seguinte se refere ao número de subdiretórios a partir do diretório raiz. Por último, o número de diretórios de nível 2, criados a partir dos subdiretórios de nível 1, ou seja, definiu-se 32 diretórios, sendo que cada um conterá 512 subdiretórios. Tal diretiva pode ser definida múltiplas vezes, para múltiplos caches.

• **coredump_dir /var/spool/squid:** Quando o Squid é iniciado são criados alguns arquivos chamados de *core-dump*, que em princípio são armazenados no próprio diretório do Squid. Com este parâmetro é possível alterar a localização padrão dos arquivos *core-dump*, indicando apenas o novo caminho;

• **cache_effective_user squid:** Conta de usuário utilizado pelo processo do Squid;

• **cache_effective_group squid:** Identifica o grupo de usuários que será utilizado pelo processo do Squid;

• **cache_access_log /var/log/squid/access.log:** Define onde será armazenado o arquivo que mantém todos os registros de acesso. É uma boa prática manter este arquivo em uma partição diferente da que está o sistema operacional e o Squid para que se possa melhorar o desempenho de acesso ao disco.

• Há a possibilidade de desativar os registros usando a diretiva *cache_access_log none*. O Squid também disponibiliza um comando específico para rotacionar o arquivo de log, e este pode ser usado juntamente com o agendador de tarefas, como o *cron* por exemplo, para rotacionar os arquivos a cada período de tempo, através do comando *squid -k rotate*;

• **cache_log /var/log/squid/cache.log:** Determina em que local será armazenado o arquivo que mantém todos os registros referentes ao cache e informações como número de objetos carregados, expirados, requisições inválidas, dentre outros. Também pode ser desabilitado;

• **cache_store_log /var/log/squid/store.log:** Decide onde será armazenado o arquivo que mantém todos os registros

cache_dir	Diretiva de configuração para o diretório de cache.
aufs	Tipo de sistema de arquivo que o Squid utilizará.
/var/spool/squid	Caminho do diretório raiz do cache.
5000	Tamanho máximo de disco que o Squid usará para cache.
32	Número de diretórios de primeiro nível
512	Número de diretórios de segundo nível, a partir do diretório de primeiro nível.

Tabela 2. Parâmetros cache_dir

referentes ao processo de manipulação de disco e que contém informações completas referentes a quais objetos foram salvos, por quanto tempo e quais foram expirados. Aqui podem ser encontradas as informações em caso de falha de leitura e escrita no disco. Esse registro também pode ser desativado (*cache_store_log none*);
• **pid_filename:** Define onde será criado o arquivo que contém o PID do Squid, que será configurado mais adiante no script de manipulação do serviço do Squid.

Após a configuração do cache deve-se ajustar as ACLs (*Access Control List*), que são as regras utilizadas para liberar ou bloquear os acessos.

As ACLs permitem manipular os pacotes de dados, isto é, os acessos, por meio de endereços de origem e/ou destino, usuários, domínios, portas e expressões regulares. Ainda, admite a definição dos horários de acesso e métodos de autenticação.

O Squid é bastante flexível quanto às suas configurações de controle de acesso. Por isso, vale uma consulta ao material oficial (veja a seção **Links**). A ACL, quando definida, é utilizada pela diretiva **http_access**, permitindo ou negando o tráfego de pacotes entre origem e destino caso coincidam com alguma de suas regras.

O formato da ACL é: **acl <nome_acl> <tipo_acl> <"argumento ou arquivo">**, e o formato da diretiva é: **http_access <nome_da_acl_a_que_se_aplica> <permissão ou negação>**.

Embora existam vários tipos de ACL, a forma de uso é a mesma para todos. O próprio arquivo *squid.conf* contém exemplos e explicações para cada tipo. Vejamos alguns deles:
• **acl all src 0.0.0.0/0.0.0.0:** Define a ACL de nome **all**, do tipo **src** (IP de clientes/origem), cujo IP e faixa abrangem todas as faixas de rede A, B, C, D e E. Esta regra será utilizada com a diretiva **http_access** para negar todo o acesso e bloquear toda a rede;
• **acl redeInterna src 10.0.0.0/255.255.0.0:** Determina uma ACL de nome **redeInterna**, do tipo **src**, pertencente à faixa de IPs utilizada na rede interna. Esta regra com a diretiva **http_access** servirá para bloquear todo o tráfego da rede, com exceção dos IPs da rede definida aqui;
• **acl localhost src 127.0.0.1/255.255.255.255:** Define uma ACL de nome **localhost**, do tipo **src**, cuja faixa de IP/rede é a interface loopback, servindo para permitir os pacotes gerados pelo próprio servidor;
• **acl QUERY urlpath_regex cgi-bin \?:** Determina uma ACL de nome **QUERY**, do tipo **urlpath_regex**, que será utilizada juntamente com o recurso de expressão regular para procurar ocorrências das palavras **cgi-bin** ou **?** em uma URL;
• **no_cache deny QUERY:** Define uma regra de negação (deny), do tipo **no_cache**, para a ACL de nome **QUERY**. Ela determina que não deve ser feito cache para conteúdos cgi, pois como normalmente são conteúdos dinâmicos, é importante evitar que informações inválidas ou antigas sejam mantidas e possivelmente acessadas no futuro;
• **acl manager proto cache_object:** Cria uma ACL de nome **manager**, do tipo **proto**, que define uma regra para o gerenciador de objetos em cache, como por exemplo o **squidclient**, usando um protocolo especial (*cache_object*);
• **acl SSL_ports port 443 563# https,snews:** Define uma ACL de nome **SSL_ports**, do tipo **port**, e seus elementos confiáveis que são as portas seguras;
• **acl Safe_ports port 80#http:** Define ACLs chamadas **Safe_ports**, do tipo **port**, que são as portas permitidas pelo proxy Squid. Por exemplo: **acl Safe_ports port 21# ftp**;
• **acl purge method PURGE:** Define uma ACL de nome **purge**, do tipo **method**. **purge** é um método específico do Squid que permite que o administrador force a remoção dos objetos do cache. Esta regra será utilizada para permitir apenas solicitações de origem do próprio servidor;
• **acl CONNECT method CONNECT:** Determina uma ACL de nome **CONNECT**, do tipo **method**, utilizada para permitir conexões diretas através do proxy;
• **no_cache deny SSL_ports:** Define uma ACL de negação, do tipo **no_cache**, às portas seguras, ou seja, não permite que seja feito cache dos objetos, evitando que informações inválidas ou antigas sejam consultadas no futuro, como por exemplo, dados dinâmicos em portas https;
• **http_access allow manager localhost:** Cria uma nova diretiva (**http_access**) que permite (**allow**) à ACL chamada **manager** aceitar todas as solicitações referentes ao gerenciamento dos objetos em cache, desde que elas sejam originadas da rede configurada através da ACL **localhost** (127.0.0.1/255.255.255.255, o próprio servidor);
• **http_access deny manager all:** Cria uma diretiva que nega (**deny**) o conteúdo da ACL **manager**, ou seja, proíbe ao restante da rede o gerenciamento dos objetos em cache configurados na ACL **all**;
• **http_access allow purge localhost:** Cria uma diretiva que aceita a ACL **purge**, ou seja, permite que os IPs configurados na ACL **localhost** possam gerenciar os objetos em cache do Squid através do método **purge**;
• **http_access deny purge all:** Cria uma diretiva que nega a ACL **purge** para o restante da rede através de seus próprios hosts;
• **http_access deny !Safe_ports:** Cria uma diretiva que nega as portas que não estão definidas na ACL **Safe_ports**. O símbolo de exclamação ("!") inverte a lógica, ou seja, tudo que for diferente do definido na ACL **Safe_ports**, como por exemplo, negando as portas de número 81 e 82. Logo, é preciso observar se existem aplicativos na rede que utilizem portas especificas, sendo necessário liberá-las para que eles funcionem corretamente;
• **http_access deny CONNECT !SSL_ports:** Cria uma diretiva negando conexões diretas às portas que não estão definidas na ACL **SSL_ports**;
• **http_access allow localhost:** Cria uma diretiva que permite a ACL **localhost**, ou seja, libera todo o tráfego ou requisição cuja origem seja o próprio host, uma vez que a ACL **localhost** aponta para o IP de loopback;
• **http_access allow redeInterna:** Cria uma diretiva que permite o que foi definido na ACL **redeInterna**, ou seja, libera todo o tráfego

ou requisição proveniente da rede definida na mesma ACL (desde que atenda às regras definidas anteriormente);

• **http_access deny all:** Cria uma diretiva que nega todo o tráfego de qualquer origem para qualquer destino, com exceção da rede definida na ACL **redeInterna**, pois, como o Squid lê as regras de forma sequencial e a ACL **redeInterna** está posicionada antes desta, logo, já foi liberada. Então, lê-se a regra assim: a rede interna foi liberada e houve bloqueio de todas as outras faixas de endereços possíveis.

Após a realização das configurações deve-se executar o binário do Squid para que este leia as configurações e então possa construir o cache/swap e, por fim, inicializar o Squid.

Para construir ou reconstruir o cache é necessário parar o Squid e utilizar o comando */usr/sbin/squid –z*. Sua inicialização é feita através do comando */usr/sbin/squid*, conforme a **Figura 5**.

Para quaisquer alterações no arquivo de configuração (*squid.conf*) é preciso fazer com que o Squid releia o referido arquivo e aplique as novas alterações. Para realizar essa tarefa, deve-se utilizar o comando */usr/sbin/squid -k reconfigure*. Assim as conexões ativas não serão interrompidas.

Neste ponto o Squid deve estar funcionando, como indicado na **Figura 5**. Caso contrário, deve-se verificar os caminhos e as permissões dos diretórios: tanto os configurados no *squid.conf*, quanto os próprios diretórios e arquivos de configuração, incluindo as contas e os grupos de usuários.

Contudo, apesar do Squid estar configurado e funcionando, não implica dizer que o tráfego esteja sendo filtrado, pois é necessário que todas as requisições que passem pelas portas HTTP (80), HTTPS (443) e 8080 (as mais utilizadas) sejam redirecionadas para a porta em que o Squid foi configurado, o que é possível através do parâmetro http_port (3128) existente no mesmo arquivo de configuração. O redirecionamento será feito pelo IPFW (Firewall padrão do FreeBSD).

Para fazer o monitoramento dos logs do Squid (vide arquivo */var/log/squid/access.log*) será utilizado o comando *tail –f /var/log/squid/access.log*. Este comando monitora o arquivo passado como parâmetro e, a partir deste momento, quaisquer alterações sofridas pelo arquivo serão exibidas em tempo real.

```
freeBSD# /usr/sbin/squid -z
2011/07/31 02:31:13| Creating Swap Directories
freeBSD# /usr/sbin/squid
freeBSD# ps -aux | grep -v grep | grep squid
root      1617  0.0  0.4  7088  3532  ??  Ss    2:31AM   0:00.00 /usr/sbin/squid
squid     1619  0.0  0.7 13232  7152  ??  S     2:31AM   0:00.06 (squid) (squid)
squid     1620  0.0  0.1  3216   912  ??  Ss    2:31AM   0:00.00 (unlinkd) (unlinkd)
```

Figura 5. Criando cache e inicializando o Squid

```
#!/bin/sh

ipfw -f flush
ipfw add 500 deny ip from any to any 3128 via rl0
ipfw add 501 fwd 127.0.0.1,3128 tcp from not me to any 80,443,8080
ipfw add 502 divert natd via rl0
```

Figura 6. Script com regras para o IPFW

Para finalizar o processo de monitoramento do arquivo */var/log/squid/access.log* deve-se pressionar *ctrl + c*. É importante lembrar que o arquivo de log do Squid permanecerá vazio até que o redirecionamento do tráfego das portas citadas seja feito.

Redirecionando o tráfego

Para redirecionar o tráfego serão utilizadas regras de firewall do IPFW, o firewall padrão do FreeBSD.

É importante ressaltar que a política adotada pelo firewall é a totalmente aberta, onde todo o tráfego é permitido através do gateway em qualquer porta e em qualquer direção (entrada e saída).

O próximo passo consiste na criação de um script de firewall cujo objetivo é ajustar as regras para que todo o tráfego das portas 80, 443 e 8080 sejam redirecionados à porta 3128 do Squid. Este script deve estar localizado em */usr/local/etc/rc.d/rc.firewall*, conforme mostrado na **Figura 6**.

Como podemos observar, as regras são simples e servem para redirecionar o tráfego. As regras são: de número 500, que nega requisições de qualquer origem para qualquer destino que forem realizadas diretamente na porta 3128, através da interface de rede rl0; a regra de número 501, que faz com que todo o tráfego destinado às portas 80, 443 e 8080, desde que a origem não seja o próprio servidor (not me), sejam redirecionadas à porta 3128 do próprio servidor – 3128 é a porta onde o Squid está "escutando/aguardando" o tráfego; e a última regra, de número 502, que realiza o NAT (Network Address Translation) através da interface rl0 – o NAT irá mascarar os endereços dos microcomputadores da rede local de maneira que seja possível a comunicação entre a rede interna LAN (Local Area Network) com a rede externa WAN (Wide Area Network, ou rede).

É importante enfatizar a regra de número 501, pois ela é responsável por forçar o uso da configuração do proxy no navegador. Se o proxy não for informado, configurado, não será permitido a navegação, uma vez que requisições diretamente nas portas especificadas (80, 443 e 8081) não serão aceitas.

Esta configuração de firewall aumenta a segurança, já que se algum browser/navegador for desconfigurado pelo usuário, este não conseguirá navegar. Em contrapartida, quando o parque de máquinas é amplo ou existe um fluxo muito grande de dispositivos móveis e não há mecanismos como servidor DHCP (Dynamic Host Configuration Protocol) configurados para atribuir o proxy de modo automático para os dispositivos, poderá exigir maior esforço por parte dos administradores e aos usuários leigos.

Adicionadas as regras de firewall, será necessário ativá-las com o script ou reiniciando o micro. Após o servidor ter sido configurado, será necessária a configuração

do proxy nos navegadores dos clientes. A partir deste momento pode-se verificar novamente o arquivo de log do Squid com o comando *tail -f /var/log/squid/access.log* (veja **Figura 7**), pois os logs serão registrados durante os acessos.

É importante mostrar aos usuários leigos que tenham necessidade de conectar seus dispositivos em outras redes como remover as configurações de proxy.

Como complemento, vale a pena estudar sobre proxy automático, através dos scripts PAC (*Proxy Auto-Configuration*) e/ou WPAD (*Windows Proxy Auto Discovery*) juntamente com o DHCP.

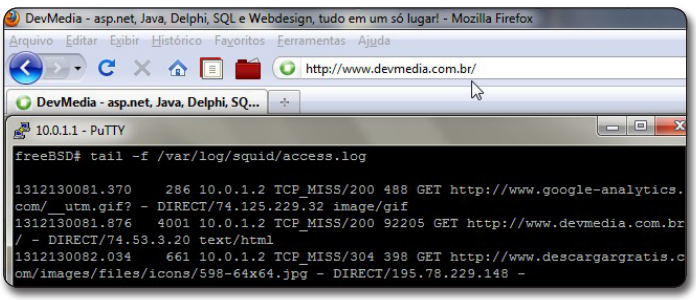


Figura 7. Realizando um acesso e verificando o arquivo de log

Com o proxy já configurado no navegador, conforme se observa na **Figura 7**, foi possível fazer a simulação do retorno do browser informando ao usuário que o proxy não está configurado, o que pode ser visto na **Figura 8**.

É possível perceber que a página de erro exibida (**Figura 8**) está no idioma português, pois o caminho para as páginas foi configurado através do arquivo *squid.conf*, por meio do parâmetro *error_directory*, e o nome do host configurado através do parâmetro *visible_hostname*, o que demonstra facilidade em personalizar as páginas de erros. Para tanto, é necessário apenas um modesto conhecimento de HTML (*HyperText Markup Language*) para que se possa editar as páginas existentes ou mesmo criar novas páginas de erro.

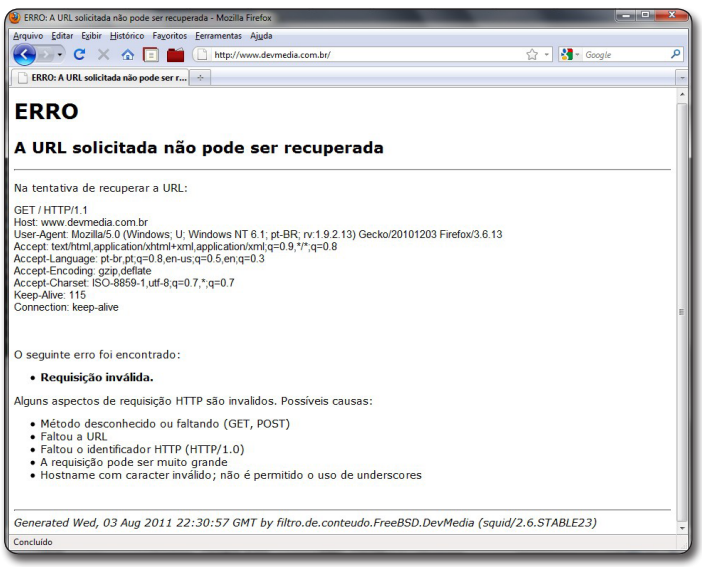


Figura 8. Tentativa de acesso sem proxy configurado

Neste ponto tem-se então um gateway com proxy funcional, surgindo agora a necessidade de se adicionar novas regras a fim de controlar o acesso por endereço IP, por endereço MAC, bloquear palavras contidas em URL e domínios. Mas antes é preciso fazer com que o Squid seja inicializado junto ao sistema operacional.

Iniciar o Squid automaticamente junto ao FreeBSD

Embora o Squid tenha sido configurado corretamente, é necessário fazer com que ele fique disponível automaticamente a cada nova inicialização do sistema operacional. Reiniciar o sistema operacional não é uma atividade rotineira, mas causa transtornos à empresa, por isso, quanto mais rápido e automatizado este processo acontecer, menor será seu impacto.

Sendo assim, será utilizado o sistema de inicialização *rc.d* do FreeBSD para fazer do Squid um serviço que possa ser iniciado automaticamente através de script e do arquivo *rc.conf*, assim como feito com outros serviços, a exemplo do firewall. Para tal, será criado um arquivo dentro do diretório */usr/etc/rc.d* com o nome *squid*. Seu conteúdo pode ser visto na **Listagem 2**.

Listagem 2. Script para inicialização do Squid no boot.

```
#!/bin/sh
#
# PROVIDE: squid
# REQUIRE: ipfw
# KEYWORD: shutdown

. /etc/rc.subr
name="squid"
rcvar=`set_rcvar`
command="/usr/sbin/squid"
load_rc_config $name
squid_enable=${squid_enable-"NO"}
squid_pidfile=${squid_pidfile-"/var/run/squid.pid"}
pidfile="${squid_pidfile}"
run_rc_command "$1"
```

Após a criação do arquivo *squid*, deve-se ajustar suas permissões de modo a torná-lo executável, alterar seu dono para o usuário *root* e seu grupo para *Wheel* através dos seguintes comandos:

```
# chown -R root:wheel /etc/rc.d/squid
# chmod 755 /etc/rc.d/.squid.
```

Feito este procedimento, deve-se adicionar a variável *squid_enable="YES"* no arquivo *rc.conf* localizado em */etc/rc.conf*.

Logo após, deve-se executar o comando */etc/rc.d/squid rcvar* com o intuito de verificar se a configuração está correta, pois a partir deste ponto pode-se manipular o Squid com os comandos */etc/rc.d/squid start|stop|status*. Como o foco deste artigo não é o sistema de inicialização do FreeBSD, mais detalhes sobre este assunto podem ser obtidos através da seção **Links**.

Vale ressaltar que a variável *squid_pidfile* possui o seguinte conteúdo: *"/var/run/squid.pid"*, e que este deve ser o mesmo caminho e arquivo de PID que é ajustado no arquivo de configuração do

Squid, através da diretiva *pid_filename*. É através dele que será possível finalizar o processo do Squid com a opção *stop*.

Outra diretiva que sofre influência direta da opção *stop* é a diretiva *shutdown_lifetime*, que configura o tempo de finalização do processo. Quanto maior o valor para *shutdown_lifetime*, maior será o tempo para finalização do Squid.

Para um teste efetivo, deve-se reiniciar o FreeBSD e verificar se o Squid foi iniciado automaticamente.

Este teste pode ser feito realizando-se um simples acesso a algum site pelo navegador, através do comando *ps aux | grep squid* no console do servidor ou verificando o arquivo de log do Squid */var/log/messages*.

O arquivo *messages* contém as mensagens referentes a todo o processo de inicialização. Logo, é mais indicado verificá-lo, pois se houver qualquer tipo de erro, é através dele que será possível identificar o maior número de detalhes para se atuar na correção.

Caso tudo esteja correto, o arquivo *messages* deverá conter o registro da inicialização do Squid, semelhante à **Figura 9**.

Neste ponto o Squid está instalado e configurado de modo a registrar todos os acessos da rede, porém ele está aberto, o que quer dizer que todo e qualquer conteúdo poderá ser acessado sem qualquer restrição. Por isso, em edições posteriores da Infra Magazine serão publicados novos artigos que detalharão como podem ser adicionados mais controles e regras no Squid.

```
freeBSD# cat /var/log/messages | grep squid
Aug  7 15:17:15 freeBSD squid[853]: Squid Parent: child process 856 started
Aug  7 15:17:15 freeBSD squid[853]: Squid Parent: child process 856 exited with status 1
Aug  7 15:17:18 freeBSD squid[853]: Squid Parent: child process 896 started
freeBSD# ps aux | grep squid
root    853  0.0  0.8 11184  7688  ??  Is    3:17PM   0:00.00 /usr/sbin/squid -D
squid   896  0.0  1.2 17328 11764  ??  S     3:17PM   0:00.25 (squid) -D (squid)
squid   897  0.0  0.1  3216   912  ??  Is    3:17PM   0:00.00 (unlinkd) (unlinkd)
root    936  0.0  0.0   388   268  p0  R+    3:29PM   0:00.00 grep squid
```

Figura 9. Registro de inicialização correta do Squid no /var/lo/messages e a verificação do processo

Conclusão

Redes empresariais, escolares e de acesso público, requerem o controle do conteúdo acessado através da rede de dados (Internet).

Neste contexto a utilização de um servidor proxy é conveniente e vantajosa quando é preciso manter os registros de identificação do usuário, datas e horários em que foram realizados os acessos, além do registro do que foi acessado ou da tentativa de acesso.

Tais informações possibilitam ao administrador a identificação do perfil de seus usuários, e a restrição do acesso ao tipo de uso a que a rede se destina. Além disso, auxilia no dimensionamento da rede, quanto à largura de banda, uma vez que se pode mensurar o número de acessos, e ajuda a restringir o tempo de conexão dos usuários, quando apropriado.

Sendo assim, o Squid se mostra eficiente na realização das tarefas cotidianas, bastante aderente ao sistema operacional, com baixo custo de implantação, além de ser confiável e robusto, como o FreeBSD.



Rogério Tomassoni de A. Junior
extomass@gmail.com
Bacharel em análise de sistemas, formado pela FAI - Faculdade de Administração e Informática, certificado LPIC2 000160469, certificado ITIL Foundation, atualmente exerço função de administrador de redes, membro ativo do site viva o Linux.

Infra Magazine 01 – Gateway com FreeBSD
www.devmedia.com.br/websys.4/webreader.asp?cat=62&revista=inframagazine_1#a-3509

Endereço para download do ISO FreeBSD
ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/ISO-IMAGES/7.2/7.2-RELEASE-i386-disc1.iso

Endereço para download do Squid
http://www.squid-cache.org/Versions/v2/2.6/squid-2.6.STABLE23.tar.gz

Como configurar a Squid ACL
http://www.squid-cache.org/Doc/config/acl/

Autenticação centralizada com OpenLDAP

Flávio Alexandre dos Reis e Eduardo Pagani Julio

Resumo DevMan

De que se trata o artigo:

Este artigo tem como objetivo demonstrar uma introdução ao uso do LDAP, apresentando sua origem, desenvolvimento e principais características do protocolo. Será abordado como funciona a utilização dos serviços de diretórios como base central de autenticação a várias aplicações de rede, tais como servidor de e-mail, FTP, Proxy, SAMBA, dentre outras. Nos exemplos será utilizado o projeto OpenLDAP para demonstrar alguns exemplos de uso do protocolo.

Em que situação o tema é útil:

Este serviço é usado para armazenar todos os dados de usuários em uma rede, como senhas, IDs, nomes, endereços, dentre outros. Dessa forma, as pesquisas e consultas serão centralizadas. Essa centralização é a chave para abrir um caminho que leva à praticidade na administração de uma rede, independente de seu tamanho e complexidade.

Autenticação centralizada com OpenLDAP:

A utilização dos serviços de diretórios é necessária quando se deseja centralizar a autenticação de serviços utilizados em uma rede de computadores, evitando assim a necessidade de várias bases de usuários. O LDAP oferece a integração com protocolos de comunicação como o IPv4 e IPv6, transferência de arquivos como o FTP, além da integração com banco de dados, chaves criptográficas, dentre outras ferramentas que fazem com que o LDAP possa ser implementado de forma segura e funcional, possuindo a capacidade de armazenamento de dados de usuários.

Quando é abordado um serviço de diretórios, a primeira ideia que surge é uma base de dados em que podem ser armazenadas informações de usuários. De certa forma, isso é verdade. Em um servidor de diretórios, podem ser armazenados dados de usuários que poderão ser recuperados a qualquer momento, assim como em um SGBD (Sistema de Gerenciamento de Banco de Dados) convencional, como Oracle e PostgreSQL.

O protocolo LDAP (*Lightweight Directory Access Protocol*) ou Protocolo Leve de Acesso a Diretório trata-se, como o nome já diz, de um protocolo que rege a forma de acesso a serviços de diretórios e seus respectivos clientes. Em outras palavras, fornece a comunicação entre usuários e serviços de diretórios, centralizando deste modo toda uma base de dados de autenticação.

Dentre as funcionalidades do LDAP, o que pode ser considerado como um destaque principal é a capacidade de oferecer a autenticação de usuários usando sua base de dados. Com ela, podem-se acessar as referências de todas as informações dos usuários da rede em um único lugar, permitindo também que todos os protocolos e serviços de diretórios vinculados a ele possam utilizar seus dados para a autenticação de seus clientes. Isso gera o que se denomina de centralização, pois a autenticação de todos os serviços de rede será concentrada em uma única árvore de informações e, como consequência, facilita o trabalho do gerente de redes.

Para esclarecer melhor o que é um diretório, pode-se comparar com uma lista telefônica, onde se procura, por exemplo, o telefone de uma pizzaria. Nesse caso, através do telefone você irá conseguir o caminho até o destino da pizzaria. Portanto, pode-se definir um diretório como um serviço de armazenamento hierárquico de informações com o objetivo principal de facilitar a pesquisa e a recuperação dessas informações.

De forma semelhante, um diretório pode ser uma lista de informações sobre objetos organizados ou catalogados em uma ordem, e que fornece o acesso aos dados dos objetos. O diretório permite que usuários ou aplicações possam encontrar recursos no ambiente com características necessárias para um tipo de tarefa particular.

Há algumas comparações equivocadas sobre o serviço de diretórios, sendo descritas algumas delas abaixo:

- **Banco de dados:** um banco de dados é criado para otimizar a leitura e escrita de dados com o mesmo grau de eficiência. Um diretório, ao contrário, é otimizado apenas para leitura, podendo ocasionalmente inserir novos dados. Seu sistema de transações é muito simples quando comparado a um sistema de banco de dados;
- **Sistema de arquivos:** O sistema de arquivos possui otimizações para manipulação de arquivos. Um arquivo grande não precisa ser completamente carregado na memória, sendo possível apontar para apenas uma região naquele instante. O diretório não possui esse tipo de otimização;
- **FTP e servidores *web*:** seguem a filosofia de ler muito e gravar pouco. Mas ao ser analisada a questão do tamanho do arquivo, observa-se o mesmo problema do item anterior, o tamanho do arquivo a ser carregado. Servidores *web* são extensíveis, podem ser usados como base de desenvolvimento de aplicações mais complexas. Um diretório, por sua vez, não possui essa natureza de extensão.

Nos itens seguintes serão abordados alguns conceitos importantes para um bom conhecimento dos serviços de diretórios.

O Serviço de Diretórios

Entre um serviço de diretórios e um SGBD existem diferenças importantes que devem ser observadas:

• **Otimização de leitura:** um gerenciador de base de dados deve levar em consideração uma série de operações de escrita e travamento de registros, ao passo que os diretórios devem disponibilizar prontamente os dados solicitados;

• **Hierarquia:** uma base de dados possui tabelas com campos e registros, ao passo que no diretório cria-se uma estrutura em formato de árvore, de modo que cada registro será armazenado dentro de um ramo específico da árvore;

• **Modelo distribuído para armazenamento de informações:** com base na raiz da árvore, podem-se criar estruturas independentes para cada unidade (ramo);

• **Forte padronização:** os dados armazenados em um diretório seguem uma padronização, normalmente estabelecida em uma RFC;

• **Capacidade avançada de pesquisa:** efetuar pesquisas pelo seu conteúdo de forma exata ou aproximada, levando em consideração até mesmo variações fonéticas.

Em alguns casos, uma análise deverá ser feita antes de escolher entre o uso de um Diretório ou de um SGBD. Assim, pode-se saber quem terá melhor desempenho, afinal o SGBD tem estrutura muito similar a servidores de Diretórios, compartilhando muitas características como pesquisas rápidas e base de fácil personalização.

Neste contexto, os seguintes aspectos devem ser observados:

• **Volume de dados:** Servidores de diretórios normalmente não têm suporte a métodos avançados para controle de transações ou esquemas de *rollback*, sendo a baixa performance em escrita seu ponto fraco;

• **Hierarquia:** A estrutura em forma de árvore permite mobilidade aos dados que podem ser agrupados por departamentos ou mesmo por unidades geograficamente diferentes;

• **Administração descentralizada:** Possibilita delegar a administração de parte do Diretório a outros usuários (ramos, unidades), podendo assim estar localizados em outros servidores.

X.500

O X.500 é um padrão de protocolos de serviços de diretórios utilizado em redes de computadores, tendo sido elaborado para trabalhar em cima do modelo OSI (veja a **Nota DevMan 1**) e incorporado ao pacote de protocolos do mesmo. Designado para dar suporte ao padrão X.400, que define a troca de mensagens eletrônicas entre os usuários da rede local, sua função é prover serviços de diretórios para a rede centralizando a base de dados dos usuários em um servidor.

O protocolo de acesso a diretórios – DAP (*Directory Access Protocol*) faz parte das especificações do padrão X.500, tendo sido desenvolvido para trabalhar junto a todas as camadas do modelo OSI, e tinha por objetivo definir o acesso de usuários aos serviços de diretórios que seu padrão provia.

Assim como o OSI e o X.500, o DAP foi desenvolvido antes do advento da Internet e originalmente não foi preparado para trabalhar com o TCP/IP, visto que a aplicação do mesmo, além de ser de difícil implementação, gerava aplicações complexas e lentas. Além do mais, o estilo da organização da árvore de diretórios do X.500 não foi preparado para a utilização de diretórios distribuídos.

Quando se trata de redes de computadores, tudo gira em torno dos protocolos, a Internet, e-mail, intranets, transferência de arquivos, acessos a diretórios e outras aplicações providas pela pilha TCP/IP. A grande vantagem da utilização de protocolos se resume na compatibilidade que eles oferecem, tanto entre hardware quanto entre software de diversos fabricantes, o que os torna compatíveis. Um exemplo prático disso é a possibilidade de se estabelecer a comunicação entre Sistemas Operacionais diferentes, ou partindo para o nível de hardware, que é, por exemplo, fazer com que placas de rede de fabricantes distintos possam se comunicar.

Já é aceito que a implementação do Modelo OSI utiliza uma gama maior de recursos de rede, onde se trafega uma grande quantidade de dados desnecessários, e a pilha TCP/IP, por sua vez, trabalha de forma mais leve, exigindo menos dos recursos da rede, tanto que, com a disseminação da Internet, o TCP/IP passou a ser usado como um padrão internacional. O sucesso foi tanto que os protocolos do padrão X.500 foram adaptados para que as redes TCP/IP pudessem trabalhar com os servidores X.500. No entanto, posteriormente percebeu-se a necessidade da criação de protocolos que se encaixassem melhor em suas características.

O LDAP foi criado como uma alternativa ao DAP, para prover acesso aos serviços de diretórios do X.500 pelos protocolos da pilha TCP/IP. O LDAP é mais fácil de ser implementado do que o DAP, além de exigir menos dos recursos da rede e memória. Ele foi desenvolvido, e não adaptado (como o DAP), para aplicações TCP/IP. Deste modo, o LDAP obteve um maior desempenho. Por esses motivos, recebeu o nome *Lightweight Directory Access Protocol* (protocolo leve de acesso a diretórios), que é o nome de seu antecessor acrescentado de *Lightweight* (leve).

Posteriormente foram criados servidores de diretórios voltados para o TCP/IP e o LDAP. O servidor *Slapd* (*Stand-alone LDAP daemon*) foi escolhido como a melhor opção. Com sua utilização, passa a se colocar em prática um software provedor de serviços de diretórios específicos para o TCP/IP e o LDAP, deixando de lado o X.500, que é uma mera adaptação de um padrão desenvolvido para o modelo OSI. Com isso, há um ganho em desempenho, funcionalidades e melhor integração com o LDAP.

Deste modo, o LDAP passou a ser a melhor forma de se obter o acesso a serviços de diretórios, e foi padronizado em julho de 1993 na RFC 1487.

LDAP

O serviço de Diretórios é executado em um servidor especializado, onde todas as informações contidas são consultadas

obedecendo a uma série de convenções definidas inicialmente pelo protocolo X.500. Este protocolo determinou uma série de informações e controles devido ao seu desenvolvimento ter sido baseado no modelo OSI, porém não serão abordados neste artigo.



Nota do DevMan 1

Modelo OSI: Foi uma das primeiras organizações a definir formalmente uma forma comum de conectar computadores. Sua arquitetura é chamada OSI (Open Systems Interconnection), Camadas OSI ou Interconexão de Sistemas Abertos.

O protocolo LDAP está em sua versão 3 (LDAPv3), publicada no final da década de 1990, e acrescenta melhorias em relação a sua versão anterior, algumas delas listadas a seguir:

- Autenticação e serviços de segurança de dados via SASL (veja a **Nota DevMan 2**);
- Autenticação via certificados digitais e serviço de segurança de dados via TLS (SSL) – veja a **Nota DevMan 2**;
- Internacionalização usando a tabela UNICODE;
- Resolução de *schemas*;
- Extensibilidade;
- Dentre outras melhorias.



Nota do DevMan 2

SASL e TSL: Simple Authentication and Security Layer (SASL) é um quadro para a prestação de serviços de segurança e autenticação de dados em protocolos orientados a conexão.

Transport Layer Security ou TLS (Segurança da Camada de Transporte) é um protocolo criptográfico que confere segurança de comunicação na Internet para serviços como e-mail (SMTP), navegação por páginas (HTTP) e outros tipos de transferência de dados.

Imagine o cenário de uma empresa com 100 funcionários, onde toda base de e-mail (contatos do *outlook*) é compartilhada com todos eles. Assim, sempre que um novo contato é adicionado, o administrador deverá informar a todos os usuários, repassando a informação de alteração via e-mail. Esse processo pode ser centralizado em um servidor de Diretórios. Dessa forma todos os usuários serão atualizados ao mesmo tempo.

Há algumas características a serem consideradas para adoção dos serviços de diretórios. Dentre elas, pode-se citar:

- Base de autenticação para serviços de rede;
  - Login de rede (autenticação em um servidor de Domínio);
  - E-mail (pop, smtp, imap);
  - Proxy (autenticação para uso da Internet);
  - Intranet (uso de aplicações e compartilhamentos internos);
- Permissão para administrar parte do Diretório (direitos administrativos centralizados; dessa forma as permissões de acesso são referenciadas no LDAP).

Estrutura do servidor de Diretórios

No LDAP, as entradas do Diretório são organizadas em uma estrutura semelhante a uma árvore, lembrando a estrutura de um DNS ou do sistema de diretórios do GNU/Linux. Tal estrutura foi desenvolvida para representar limites geográficos ou organizacionais.

Quando se pensa em uma estrutura física, como exemplo, entradas representando países, esses devem obrigatoriamente aparecer no topo da estrutura, conforme apresentado na **Figura 1**.

Abaixo dessas estruturas deverão existir entradas representando os Estados, como pode ser observado na **Figura 2**.

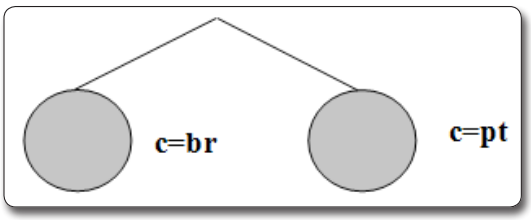


Figura 1. Exemplo de estrutura de Diretórios, dois níveis representando países

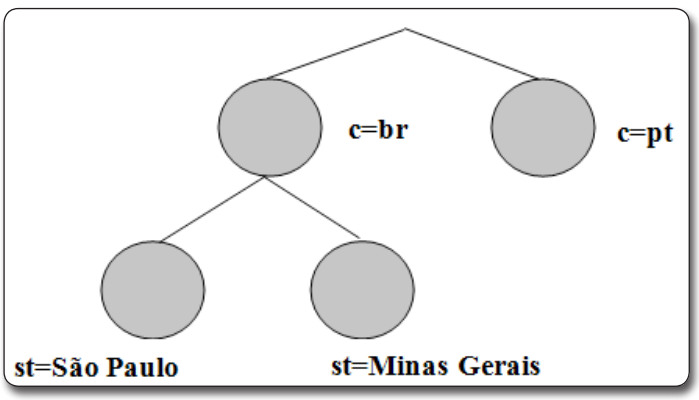


Figura 2. Exemplo de estrutura de Diretórios, com subníveis

Em seguida, o administrador poderá criar as unidades organizacionais que serão armazenadas no Diretório. Toda estrutura hierárquica criada pode representar unidades da empresa como matrizes, filiais e até mesmo divisões internas como usuários, grupos, impressoras, documentos, ou quaisquer outras entradas necessárias. Observa-se um exemplo de modelo de diretório na **Tabela 1**.

Entrada	Descrição
C	Country = pais
St	stateOrProvinceName = estado
L	localityName = cidade
Street	streetAddress = endereço
O	organizationName = organização
Ou	organizationalUnit = unidade organizacional
cn	Common name = Nome

Tabela 1. Modelo de Diretório

Na **Figura 3** tem-se um exemplo da representação de uma árvore de diretórios construída pela forma tradicional de nomes, considerando uma estrutura geográfica e organizacional.

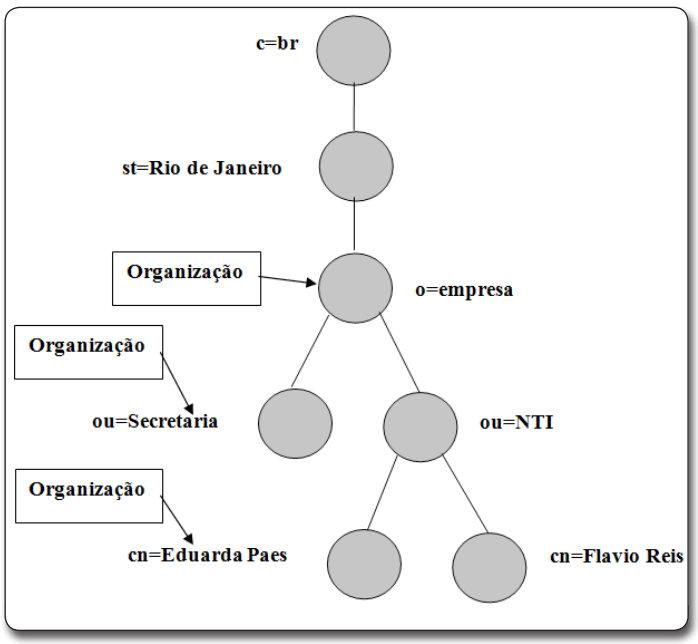


Figura 3. Modelo de Diretórios baseado na forma tradicional de nomes

A forma de armazenamento dos dados em um servidor de Diretórios segue a hierarquia dos objetos a que pertence a entrada. Na **Listagem 1**, pode ser observado que o objeto "Flavio Reis" pertence a "NTI", que por sua vez pertence a "Empresa", até que chegue ao objeto de maior valor na hierarquia.

Esse tipo de identificação é chamado de DN (*Distinguished Name*) ou nome único. O DN deverá indicar todos os ramos da árvore LDAP, desde a base (c=br) até a parte final, ou seja, a identificação do usuário. Outro ponto importante é que o DN deve sempre ser único em todo o Diretório.

Outra forma de construir uma árvore de Diretórios é utilizando componentes de domínio Internet (DC). Essa abordagem tem se tornado cada vez mais popular, pois permite que serviços de Diretórios sejam localizados utilizando um DNS. Um exemplo pode ser observado na **Figura 4**, onde são alterados os componentes inicias para construção da árvore, alterando os atributos (c=country) e (o=organization) para objetos (dc=domainComponent), replicando o formato DNS. Outra modificação é feita no usuário, inserindo a forma de identificação UID.

A forma de armazenamento dos dados em um servidor de Diretórios seguindo o formato DNS fica um pouco diferente, conforme pode ser observado no exemplo da **Listagem 2**.

Além da possibilidade de estruturação de uma rede de computadores como mostrado anteriormente, o LDAP ainda permite controlar quais atributos são necessários, ou quais são permitidos, para a entrada de dados por meio de um atributo especial chamado "objectClass". Dependendo do conteúdo desse atributo, um registro deverá obedecer às regras definidas em seu schema.

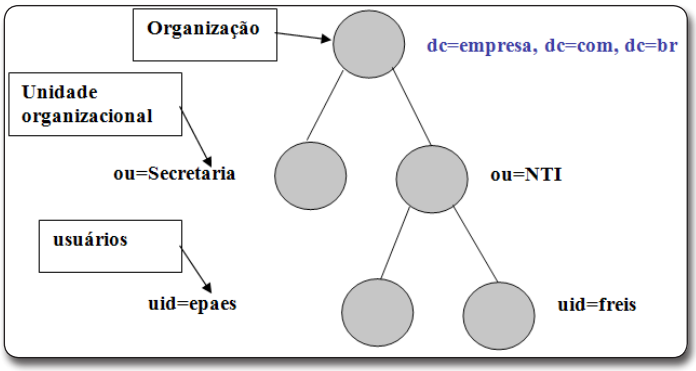


Figura 4. Modelo de Diretórios baseado em DNS

Listagem 1. Exemplo de resultado em uma estrutura hierárquica.

```
cn=Flavio Reis, ou=NTI, o=Empresa, st=Rio de Janeiro, c=br
```

Listagem 2. Identificando o usuário dentro da estrutura DNS.

```
uid=freis, ou=usuarios, dc=empresa, dc=com, dc=br
```

OpenLDAP

O *OpenLDAP* é uma aplicação multiplataforma, ou seja, independente de sistema operacional. Deste modo, várias distribuições GNU/Linux já incluem o pacote do aplicativo, que inicialmente foi desenvolvido pela Universidade de Michigan. O *OpenLDAP* trouxe as seguintes características como principais:

- Suporte a IPv4 e IPv6;
- Autenticação (Cyrus Sasl-Kerberos V, GSSAPI, Digest-MD5);
- Segurança no transporte – SSL e TLS;
- Controle de acessos;
- Escolha entre bancos de dados (DBD ou HDB);
- Capacidade de atender a múltiplos bancos de dados simultaneamente;
- Alta performance em múltiplas chamadas;
- Replicação de base, oferecendo redundância.

O projeto OpenLDAP é composto pelos seguintes elementos:

- SLAPD (*Stand Alone LDAP Daemon*), servidor que controla as requisições dos clientes;
- SLURPD (*Stand Alone LDAP Update Replication Daemon*), servidor utilizado para replicação da base de dados;
- Bibliotecas diversas de implementação do protocolo LDAP;
- Ferramentas, utilitários diversos e clientes de exemplo.

A aplicação possui basicamente dois arquivos de configuração, que são, respectivamente:

- *slapd.conf* - configuração do daemon;
- *ldap.conf* - configuração para acesso dos clientes à base.

O OpenLDAP possui alguns arquivos chamados de *schemas*. Um esquema é um conjunto de regras que define atributos, classes de objetos, e controles, indicando onde cada dado pode ser armazenado. Alguns autores abordam um *schema* como sendo uma estrutura de objetos que especifica a lista total de atributos permitidos e necessários para uma entrada de dados no serviço de diretório.

Os *schemas* permitem manter a consistência entre os dados. Uma importante característica desses arquivos é serem extensíveis e, assim, podem-se adicionar mais atributos ou classes em função de novas necessidades. Para usar um *schema* é necessário incluí-lo no arquivo de configuração *slapd.conf*.

Os *schemas* podem definir:

- Quais as classes de objetos (*object classes*) podem ser inseridas num diretório;
- Quais os atributos de uma determinada classe de objetos;
- Os valores possíveis para os atributos.

Se um objeto (entrada) não obedecer às regras do *schema*, esse não pode ser inserido no diretório. Portanto, cada entrada estará condicionada a uma hierarquia de armazenamento dos dados na base LDAP. Isto é especificado através do *Distinguished Name* (DN).

Instalação do Servidor OpenLDAP

Antes de instalar o sistema, deve-se efetuar um estudo aprofundado e projetar o servidor LDAP, observando quantos usuários farão uso do sistema, ou quantas filiais, por exemplo. Neste tópico serão apresentados três tipos de casos diferentes.

No primeiro caso, talvez o mais tradicional, assume-se como base que a implementação do servidor LDAP será feita em uma empresa simples, sem filiais, ou apenas em um servidor, sem acesso das filiais externas ao parque computacional.

Pode-se observar um exemplo na **Figura 5**, onde os clientes irão efetuar as pesquisas diretamente no servidor. O cliente solicita uma pesquisa e o servidor responde. É um tipo de estrutura simples, como mencionado.

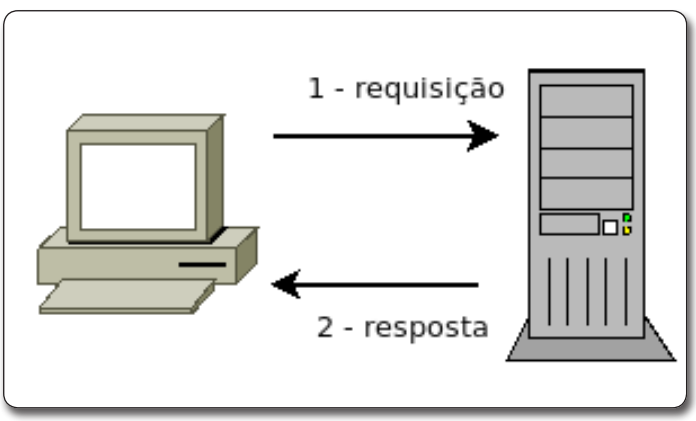


Figura 5. Estrutura de conexão LDAP – Primeiro Caso

No segundo caso, toda a organização usa o LDAP. Assim, quando os usuários da matriz precisam de alguma informação da filial, o servidor LDAP da matriz faz uma pesquisa no servidor da filial (pesquisa feita por referência) e retorna o valor para o cliente.

Para o exemplo, pode-se imaginar uma empresa que tem Matriz no Rio de Janeiro e Filial em Minas Gerais. Todos os servidores estão concentrados na matriz, e qualquer informação requerida pelos usuários das filiais deverá enviar uma solicitação ao servidor LDAP localizado na matriz. Deste modo ele irá responder à filial.

Ao analisar a situação, verifica-se que a filial sempre procura a mesma informação, que na maioria das vezes, está relacionada com a própria filial. Para resolver esse problema, pode-se criar um servidor LDAP em que uma parte da árvore fica na matriz (Rio de Janeiro) e outra parte da árvore fica na filial (Minas Gerais). Com essa estrutura, todas as informações importantes para a filial serão processadas no servidor local. Dessa forma, as pesquisas feitas pelos clientes da filial serão efetuadas localmente, obtendo ganho nas respostas. Caso a pesquisa solicitada não tenha resposta no servidor local, o próprio servidor se encarrega de efetuar uma busca no servidor da matriz e, em seguida, responder ao cliente. Observa-se esse tipo de conexão na **Figura 6.**

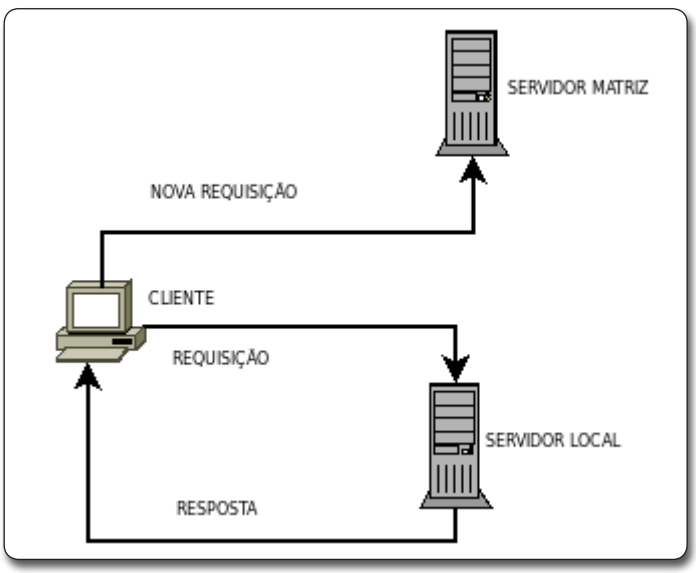


Figura 6. Estrutura de conexão LDAP – Segundo Caso

No terceiro caso, utilizam-se servidores com redundância. Isto é, para todos os dados cadastrados no servidor, serão feitas réplicas em outro servidor LDAP. Deste modo, em caso de problemas do servidor principal, é possível iniciar o servidor secundário, mantendo as informações disponíveis aos usuários. Vale ressaltar que os casos 2 e 3 que foram apresentados podem ser unificados, mantendo redundância de dados. Seguem algumas vantagens de se trabalhar com servidores com redundância:

- Fazer cópia parcial ou total do diretório;
- Melhorar o desempenho e a confiabilidade;
- Aproximar os dados dos clientes em caso de filiais;
- Distribuir melhor a carga de processamento dos servidores;
- Garantir redundância do serviço, minimizando os efeitos de falhas no servidor.

Na **Figura 7** pode-se observar um exemplo desse caso.

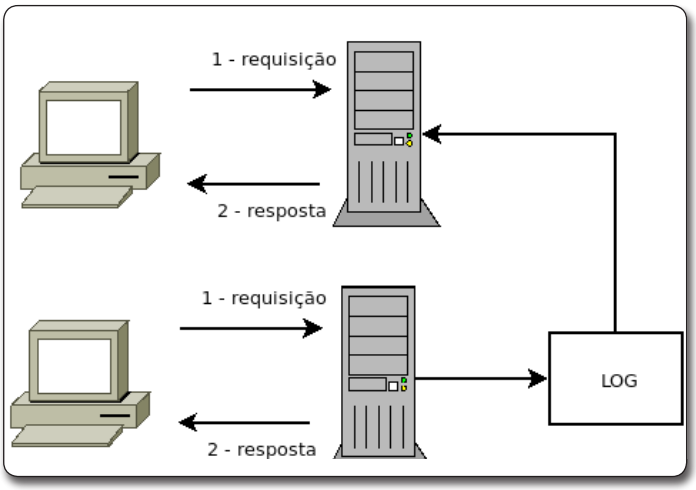


Figura 7. Estrutura de conexão LDAP – Terceiro Caso

Após essas análises, pode-se iniciar o processo de instalação do OpenLDAP. Para os exemplos demonstrados neste artigo será utilizado o Sistema Operacional GNU/Linux Debian Squeeze. Os pacotes necessários para sua instalação são:

- libldap: biblioteca ldap;
- slapd: daemon do servidor ldap;
- ldap-utils: utilitários para o OpenLDAP (comandos de busca, inserção, atualização);
- migrationtools: conjunto de scripts em *Perl* utilizados para migrar a base de usuários e grupos existentes no sistema GNU/Linux para a base LDAP;
- phpldapadmin, apache2 e php5: utilizados para gerência via Web.

Na **Listagem 3** pode-se verificar como efetuar a instalação do OpenLDAP nos sistemas GNU/Linux Debian ou derivados.

BerkeleyDB

Todas as informações do diretório devem ser armazenadas em algum tipo de base. E para isso, o LDAP possui dois bancos de dados nativos, LDBM e BerkeleyDB. Alguns autores apontam o segundo como melhor quando analisado o desempenho. Em distribuições baseadas em GNU/Linux Debian, a instalação do BerkeleyDB pode ser feita conforme apresentado na **Listagem 4**.

OpenSSL e Cyrus

Na maioria das vezes, o LDAP é utilizado para armazenar dados de usuários, inclusive a senha de autenticação. Com isso, tem-se a necessidade de um nível de segurança extra. E para aumentar a segurança no transporte dos dados, é possível criptografá-los de forma que, mesmo sendo interceptados por terceiros, não seja possível ver o que está sendo transmitido.

Para implementar esse nível de segurança, será configurado o *OpenSSL*, sendo sua instalação também simples, pois a maioria das distribuições trazem o pacote em seu repositório oficial. Para aumentar ainda mais a segurança dos dados pode-se instalar o cyrus (libsasl), que é uma opção de autenticação e criptografia para o *OpenSSL*. Feito a instalação, pode iniciar a migração dos dados. A **Listagem 5** apresenta como instalar o *OpenSSL* no GNU/Linux Debian.

No entanto, antes de iniciar a migração dos dados de usuários e grupos do sistema, deve-se ajustar algumas configurações no *OpenLDAP*. O primeiro passo é gerar a senha de administrador da base LDAP. Assim, digite o comando *ldappasswd* e copie o valor retornado, pois ele será usado posteriormente. A **Listagem 6** apresenta como fica o *shell* quando executado o comando para gerar a senha.

Listagem 3. Instalando os pacotes necessários.

```
#aptitude install libldap-2.4-2 slapd ldap-utils
#aptitude install migrationtools phpldapadmin apache2 php5
```

Listagem 4. Instalando com BerkeleyDB.

```
#aptitude install libdb4.8++-dev
```

Listagem 5. Instalando o OpenSSL no Debian.

```
#aptitude install openssl libsasl2-2
```

Listagem 6. Gerando a senha de admin do OpenLDAP.

```
#ldappasswd
#ldappasswd:
{SSHA}lt/pthgyeFA4NhGdzJ+qlqMuAGi267dq
```

Nas versões anteriores, o LDAP criava um arquivo chamado *slapd.conf* em */etc/init.d/ldap*, mas nas versões mais atuais ele é localizado em */usr/share/slapd*. Com isso, faça uma cópia do arquivo para */etc/ldap*, alterando o arquivo de acordo com a necessidade. Esse é um arquivo genérico, usado como exemplo de configuração. A senha gerada anteriormente deverá ser inserida na linha *rootpw*. Na **Listagem 7** observa-se um exemplo do arquivo *slapd.conf* já com o valor do *rootpw* alterado.

Para gerar uma nova estrutura de configuração do OpenLDAP, devem ser seguidos os passos apresentados na **Listagem 8**.

Para que o *migrationtools* possa efetuar as alterações de forma correta, o arquivo *migrate_common.ph* deverá ser modificado. Dessa forma, edite o arquivo através de um editor de textos, como exemplo o *vim*, e localize as linhas **$DEFAULT_MAIL_DOMAIN** e **$DEFAULT_BASE**, que devem ter seus respectivos valores modificados conforme o *slapd.conf*. Observe um exemplo na **Listagem 9**.

LDIF

Alguns administradores de sistemas LDAP preferem usar arquivos de texto puro para informar dados de configuração dos servidores em vez de alguns *bits* de armazenamento na base de dados. O formato de troca de dados LDAP é conhecido como LDIF (*LDAP Data Interchange Format*), definida pela RFC 2849,

Listagem 7. Exemplo do arquivo slapd.conf.

```
include     /etc/ldap/schema/core.schema
include     /etc/ldap/schema/cosine.schema
include     /etc/ldap/schema/nis.schema
include     /etc/ldap/schema/inetorgperson.schema
allow          bind_v2
pidfile      /var/run/slapd/slapd.pid
argsfile    /var/run/slapd/slapd.args
loglevel    none
modulepath  /usr/lib/ldap
moduleload  back_bdb
sizelimit 500
tool-threads 1
backend     bdb
database   bdb
suffix     "dc=reisfa,dc=eti,dc=br"
rootdn     "cn=admin,dc=reisfa,dc=eti,dc=br"
rootpw     {SSHA}lt/pthgyeFA4NhGdzJ+qlqMuAGi267dq
directory   "/var/lib/ldap"
dbconfig set_cachesize 0 2097152 0
dbconfig set_lk_max_objects 1500
dbconfig set_lk_max_locks 1500
dbconfig set_lk_max_lockers 1500
index      objectClass eq
lastmod      on
checkpoint   512 30
access   to   attrs=userPassword,shadowLastChange
   by dn="cn=admin,dc=reisfa,dc=eti,dc=br" write
   by anonymous auth
   by self write
   by * none
access to dn.base="" by * read
access to *
  by dn="cn=admin,dc=reisfa,dc=eti,dc=br" write
  by * read
```

Listagem 8. Gerando uma nova estrutura para o OpenLDAP.

```
# /etc/init.d/slapd stop
# cd /etc/ldap
# mv slapd.d slapd.d.backup
# mkdir slapd.d
# slaptest -f slapd.conf -F slapd.d
# chown -R openldap:openldap slapd.d
# /etc/init.d/slapd start
```

Listagem 9. Exemplo de configuração do arquivo migrate_common.ph.

```
# vim /usr/share/migrationtools/migrate_common.ph
$DEFAULT_MAIL_DOMAIN = "reisfa.eti.br";
$DEFAULT_BASE = "dc=reisfa,dc=eti,dc=br";
```

Listagem 10. Parte de um arquivo LDIF.

```
dn: uid=reisfa,ou=People,dc=reisfa,dc=eti,dc=br
uid: reisfa
cn: Flavio Reis
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
userPassword: {crypt}$6$fBWxv7O9$LWFyYlfCk79LiCdi9cVU48YOEULTnDwpuKmq
VTrmGa3i/4hcOnWyIiStCb/JZkbVmvdL0ZwR8BHr28TWYRQt51
shadowLastChange: 15167
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/bash
uidNumber: 1000
gidNumber: 1000
homeDirectory: /home/reisfa
gecos: Flavio Reis,,,
```

sendo um formato padrão de arquivos de texto para armazenar informações de configuração do LDAP e conteúdo dos diretórios. Em sua forma básica, um LDIF é:

- Um conjunto de entradas separadas umas das outras por linhas em branco;
- Um mapeamento de nomes de atributos e valores;
- Um conjunto de diretivas que instruem o analisador sobre como processar a informação.

As duas primeiras características fornecem exatamente o que é preciso para descrever o conteúdo de um diretório DAP. Arquivos LDIF são frequentemente utilizados para importar novos dados para seu diretório ou fazer mudanças de dados existentes, facilitando assim a manutenção dos mesmos. Os dados no arquivo de LDIF deverão obedecer às regras de *schemas* de seu diretório LDAP. Cada item que é adicionado ou alterado no diretório é verificado segundo o esquema para correção. Uma violação do esquema ocorre quando os dados não correspondem às regras existentes. Observa-se o exemplo de uma parte de um arquivo LDIF na **Listagem 10**.

Após uma breve descrição sobre LDIF, podem-se gerar os arquivos que serão usados para importar os dados para a base de usuários do sistema. Para isso, serão adotados os scripts do *migrationtools*. Os passos descritos na **Listagem 11** criam as LDIFs de usuários, grupos e a base utilizada pelo OpenLDAP.

Listagem 11. Criando as LDIFs.

```
# cd /usr/share/migrationtools/
# ./migrate_passwd.pl /etc/passwd /etc/ldap/users.ldif
# ./migrate_group.pl /etc/group /etc/ldap/groups.ldif
# ./migrate_base.pl > /etc/ldap/base.ldif
```

A **Listagem 12** apresenta um exemplo de como deverá ficar o arquivo *base.ldif*.

Após ter criado as LDIFs de usuários, grupos e base, deve-se inseri-las na base OpenLDAP. Lembre-se que já haverá os usuários e grupos antes criados no GNU/Linux, pois toda a base foi importada de lá. A **Listagem 13** apresenta a sequência de comandos utilizados para essa operação.

Onde:

- **ldapadd** – é o comando para inserir novos dados;
- **x** – é utilizado para autenticação simples;
- **D** – especifica o domínio;
- **f** – especifica o arquivo que será incluído;
- **W** – chama um prompt para digitar a senha.

Após executar os comandos, pode ser feita uma pesquisa para confirmar se os dados foram inseridos com sucesso. Observa-se na **Listagem 14** a utilização do comando *ldapsearch*.

Autenticando serviços

Os tópicos apresentados a seguir mostram a configuração para se autenticar em uma base *OpenLDAP*, não sendo abordada a

Listagem 12. Exemplo do arquivo base.ldif.

```
dn: dc=reisfa,dc=eti,dc=br
dc: reisfa
objectClass: top
objectClass: domain

dn: ou=Hosts,dc=reisfa,dc=eti,dc=br
ou: Hosts
objectClass: top
objectClass: organizationalUnit

dn: ou=Rpc,dc=reisfa,dc=eti,dc=br
ou: Rpc
objectClass: top
objectClass: organizationalUnit

dn: ou=Services,dc=reisfa,dc=eti,dc=br
ou: Services
objectClass: top
objectClass: organizationalUnit

dn: nisMapName=netgroup.byuser,dc=reisfa,dc=eti,dc=br
nismapname: netgroup.byuser
objectClass: top
objectClass: nisMap

dn: ou=Mounts,dc=reisfa,dc=eti,dc=br
ou: Mounts
objectClass: top
objectClass: organizationalUnit

dn: ou=Networks,dc=reisfa,dc=eti,dc=br
ou: Networks
objectClass: top
objectClass: organizationalUnit

dn: ou=People,dc=reisfa,dc=eti,dc=br
ou: People
objectClass: top
objectClass: organizationalUnit

dn: ou=Group,dc=reisfa,dc=eti,dc=br
ou: Group
objectClass: top
objectClass: organizationalUnit

dn: ou=Netgroup,dc=reisfa,dc=eti,dc=br
ou: Netgroup
objectClass: top
objectClass: organizationalUnit

dn: ou=Protocols,dc=reisfa,dc=eti,dc=br
ou: Protocols
objectClass: top
objectClass: organizationalUnit

dn: ou=Aliases,dc=reisfa,dc=eti,dc=br
ou: Aliases
objectClass: top
objectClass: organizationalUnit

dn: nisMapName=netgroup.byhost,dc=reisfa,dc=eti,dc=br
nismapname: netgroup.byhost
objectClass: top
objectClass: nisMap
```

Listagem 13. Inserindo os dados na base OpenLDAP.

```
# ldapadd -x -D cn=admin,dc=reisfa,dc=eti,dc=br -f /etc/ldap/base.ldif -W
# ldapadd -x -D cn=admin,dc= reisfa,dc= eti,dc=br -f /etc/ldap/groups.ldif -W
# ldapadd -x -D cn=admin,dc= reisfa,dc= eti,dc=br -f /etc/ldap/users.ldif -W
```

Listagem 14. Exemplo de uso do comando ldapsearch.

```
# ldapsearch -x -b dc=reisfa,dc=eti,dc=br uidNumber=1000

dn: uid=reisfa,ou=People,dc=reisfa,dc=eti,dc=br
uid: reisfa
cn: Flavio Reis
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/bash
uidNumber: 1000
gidNumber: 1000
homeDirectory: /home/reisfa
gecos: Flavio Reis,,,
```

instalação e configuração de cada serviço. Será apresentado o que deve ser alterado para que uma aplicação possa utilizar a autenticação em conjunto com a base de Diretórios.

Samba

O Samba é uma aplicação para GNU/Linux (e outros sistemas baseados em Unix) que permite o gerenciamento e compartilhamento de recursos em redes formadas por computadores com o Windows. Assim, é possível usar o Linux como servidor de arquivos, servidor de impressão, entre outros, como se a rede utilizasse servidores Windows (NT, 2000, XP, Server 2003).

Com o servidor Samba, é possível compartilhar arquivos, compartilhar impressoras e controlar o acesso a determinados recursos de rede com igual ou maior eficiência que servidores baseados em sistemas operacionais da Microsoft.

Todo trabalho feito pelo Samba é provido de grande segurança, uma vez que há grande rigor nos controles dos recursos oferecidos. Tanto é que existem empresas que usam o Samba como solução para conflitos existentes entre diferentes versões do Windows.

Pode-se usar o Samba como um controlador primário de domínio (PDC), onde ele passa a funcionar como um servidor de autenticação para os clientes Windows e, opcionalmente, armazenar os perfis dos usuários, permitindo que eles tenham acesso a seus arquivos e configurações a partir de qualquer máquina onde façam *logon*.

Ao cadastrar um novo usuário no servidor Samba, ele automaticamente pode fazer *logon* em qualquer uma das estações configuradas. Ao remover ou bloquear uma conta de acesso, o usuário é automaticamente bloqueado em todas as estações. Isso elimina o problema de sincronismo entre as senhas no servidor e nas estações, além de centralizar a administração de usuários e permissões de acesso no servidor, simplificando bastante seu trabalho de administração.

Neste contexto, para que o Samba utilize a base de dados do *OpenLDAP*, algumas configurações devem ser efetuadas, alterando a diretiva *passdb backend = tdbsam guest* para o que está na **Listagem 15**.

Listagem 15. Configuração do smb.conf a ser inserido para autenticação via OpenLDAP.

```
passdb backend = ldapsam:ldap://127.0.0.1
ldap suffix = dc=reisfa,dc=eti,dc=br
ldap machine suffix = ou=machines
ldap user suffix = ou=users
ldap group suffix = ou=groups
ldap admin dn = cn=admin,dc=reisfa,dc=eti,dc=br
ldap delete dn = no
domain logons = yes
enable privileges = yes
```

Após essas alterações, deve-se reiniciar o Samba, através do comando *service samba restart*, e os usuários cadastrados no OpenLDAP estarão aptos a utilizar os compartilhamentos disponíveis pelo Samba.

Squid

Com a Internet cada vez mais acessível a pequenas e médias empresas, um número imenso de usuários está se interligando a Internet. Além de todos os benefícios fornecidos, como informação em tempo real, comunicação mundial a baixo custo, contato com possíveis clientes e fornecedores por todo o mundo, a mesma trouxe alguns problemas.

Os usuários tendem a passar cada vez mais tempo navegando por *sites* não relativos ao seu trabalho primário, acessam *sites* que não condizem com a política da empresa, utilizam a banda de Internet destinada a serviços como WEB ou VPN e podem, em muitos casos, acabar infectando toda a rede da empresa com vírus e *worms* que são adquiridos em *sites* impróprios. Isso sem contar com a ameaça sempre presente da propagação de *downloads* de softwares piratas e músicas, fatores que podem complicar a vida de uma empresa durante fiscalizações.

O Squid concede a possibilidade de autenticação de usuários. Dessa forma, sempre que o usuário acessar um navegador serão solicitadas suas credenciais. A **Listagem 16** mostra a configuração que deve ser inserida no *squid.conf*.

Listagem 16. Configuração a ser inserida no proxy para autenticação via OpenLDAP.

```
auth_param basic program /usr/lib/squid/ldap_auth -v 3 -b
"dc=reisfa,dc=eti,dc=br" -f "uid=%s" -h serverldap.reisfa.eti.br
auth_param basic children 8
auth_param basic realm Acesso Restrito - Informe usuário e senha para acesso a
Internet.
auth_param basic credentialsttl 20 minutes
acl usuarios proxy_auth REQUIRED
```

Dessa forma, o administrador da rede poderá identificar os acessos de cada usuário visualizando os *logs* através do *sarg*, utilitário para análise de *log* de acesso do *squid*.

Conclusão

A concentração da base de dados de autenticação de usuários visa facilitar a administração de um parque computacional, independente do tamanho e complexidade. Com o uso do *OpenLDAP*, pode-se centralizar toda essa administração.

Com tudo o que foi relatado neste artigo, pode-se concluir que a centralização da informação, bem como o processo de autenticação seguro e centralizado, são ferramentas essenciais para uma organização competitiva. O acesso à informação deve, sempre que possível, ser monitorado e logado para a realização de auditorias futuras.

A utilização de um processo centralizado e criptografado de autenticação proporciona segurança ao ambiente organizacional, focando na precisão do controle da informação, garantindo que essas sejam acessadas somente pelos usuários que realmente têm direito. Sistemas com autenticação robusta elevam a confiabilidade de sua aplicação, porém vale ressaltar que uma aplicação é tão segura quanto seu ponto mais vulnerável, logo é necessário aplicar segurança em todas as partes.

Para concluir, vale destacar que é considerada uma boa prática a utilização de criptografia dos dados não só durante o processo de autenticação, mas também nas trocas de informação por meio de ambientes públicos como a Internet. É importante também fazer uma prévia análise de ameaças no sistema, para que sejam mapeadas e classificadas as possíveis vulnerabilidades de cada funcionalidade, evitando assim que falhas venham a comprometer toda a base de usuários.



Flávio Alexandre dos Reis
reis.falexandre@gmail.com – www.reisfa.eti.br
Especialista em Segurança da Informação – FMG/JF, Pós Graduando em Redes de Computadores – CES/JF. Certificado LPIC-1. Atualmente professor em Instituições de Ensino Superior da Região de Juiz de Fora em disciplinas ligadas a Redes de Computadores.



Eduardo Pagani Julio
epagani@gmail.com
Doutorando e Mestre em Computação pela Universidade Federal Fluminense. Atualmente professor do curso de Ciência da Computação da Universidade Federal de Juiz de Fora em disciplinas ligadas a redes.

Viva o Linux
www.vivaolinux.com.br

Squid
www.squid-cache.org

Samba
www.samba.org

Under Linux
http://under-linux.org

OpenLDAP
http://www.openldap.org/

Seu **conhecimento vale muito** na DevMedia

Publique artigos e vídeos gratuitamente no site e participe do programa **PayContent.**

Sua participação pode ser premiada: quanto mais você postar, mais chances de ganhar.



Para saber mais acesse:
www.devmedia.com.br/paycontent

DevMedia