Hardening do apache

Administração de Sistemas Operacionais de Rede
FATEC OURINHOS
Profº Paulo R. Galego H. Jr.
Última atualização: 30/05/2022

Hardening é um processo de mapeamento das ameaças, mitigação dos riscos e execução das atividades corretivas com foco na infraestrutura, com o objetivo principal de torná-la preparada para enfrentar tentativas de ataque.

Se o servidor vai ficar aberto na Internet, tem-se obrigação de mantê-lo seguro. Sem entrar em detalhes de configuração de firewall e de sistema, deve-se atentar para alguns pequenos detalhes e configurações que podem fazer a diferença no servidor.

A seguir serão listadas algumas opções de hardening que podem melhorar a segurança de um servidor Apache.

Restringindo acesso a informações sobre o Apache

Algumas alterações relativas à segurança podem ser feitas no arquivo security.conf, localizado no diretório /etc/apache2/conf-available/

Não é interessante que alguma pessoa mal intencionada consiga informações sobre o servidor facilmente. Para evitar que isso aconteça, alguns parâmetros podem ser alterados para restringir o nível de informação que é passado para os clientes. São eles:

• **ServerTokens OS**: Esta diretiva é responsável pelo nível de informação sobre o servidor que será retornado no cabeçalho HTTP. Os parâmetros existentes são: Full, OS, Minimal, Minor, Major e Prod. Onde Full retorna muitas informações como o Sistema Operacional e os módulos instalados e Prod retorna o mínimo de informações;

• **ServerSignature On**: Adiciona em páginas geradas pelo servidor (páginas de erro, listagem de arquivos, etc.) uma linha contendo a versão do servidor e o nome do Virtual Host;

• **TraceEnable On**: Este parâmetro é apenas utilizado para testes e diagnóstico do servidor.

Altere estes três parâmetros para:

```
ServerTokens Prod
ServerSignature Off
TraceEnable Off
```

Restringindo acesso a informações sobre o PHP

Para que requisições não consigam extrair informações sobre a versão do PHP instalado no servidor, deve-se alterar o arquivo **/etc/php/7.3/apache2/php.ini** e alterar a opção **expose_php** para **off**:

```
expose_php = Off
```

Depois é só restartar o apache para que as alterações entrem em vigor.

Habilitar o módulo rewrite

O **mod_rewrite** é um módulo do apache que utiliza uma regra de reescrita de URL's baseado em expressões regulares. Por padrão, o mod_rewrite mapeia uma URL para um caminho de sistema de arquivos, mas também pode ser utilizado para redirecionar uma URL para outra.

Para habilitar o módulo rewrite:

```
a2enmod rewrite
```

Agora vamos configurar o VirtualHost de acesso HTTP ao site segfatecou, forçando o redirecionamento para o HTTPS **acrescentando** ao VirtualHost da porta 80 as seguintes linhas em negrito:

```
<VirtualHost *:80>

#Habilite o redirect para sites no mesmo domínio
RewriteEngine On
RewriteRule ^(.*)$  https://%{SERVER_NAME}$1  [L,R]

</VirtualHost>
```

Isso fará com que quando um usuário digite o endereço sem o https, ele seja inserido na requisição do usuário de forma automática.

IMPORTANTE: Outra implicação é que a partir desse momento, todas as configurações do nosso site devem ser feitas na tag **<VirtualHost *:443>** pois as requisições só serão aceitas na porta 443 do nosso servidor.

Opção Indexes

Por padrão, em diretórios que não possuem nenhum arquivo index, o Apache lista todos os arquivos existentes nestes diretórios. Isto em alguns casos não é de interesse do administrador, seja porque o diretório contém arquivos privados, ou apenas para não permitir que os arquivos que ali se encontram sejam vistos. Esta configuração pode ser realizada em diretórios específicos mas, em algumas situações, a intenção pode ser a de bloquear a listagem de arquivos em todos os diretórios.

Para isso vamos alterar nosso VirtualHost, colocando as opções abaixo. É importante notar, que se o mod_rewrite tiver sido habilitado, a opção Indexes deve ser realizada no VirtualHost da conexão SSL (Porta 443):

```
<VirtualHost *:443>

<Directory /var/www/http/segfatecou>
    Options -Indexes
</Directory>
```

```
</VirtualHost >
```

Restringindo o acesso a uma pasta por usuário e senha

As vezes precisamos restringir certos arquivos e diretórios apenas para certos usuários. Isso é válido por exemplo, quando você tem um provedor, e só se quer permitir acessar algumas páginas de administração, os usuários autorizados do provedor. Para isso podem-se aplicar algumas regras usando arquivos .htaccess ou fazendo as configurações direto no VirtualHost. Vamos falar sobre cada uma dessas opções.

Arquivos .htaccess

Os arquivos **.htaccess** (ou "arquivos de configuração distribuída") oferecem um meio de fazer mudanças nas configurações por diretório. Um arquivo, contendo uma ou mais diretrizes de configurações, é colocado em um diretório em particular, e as configurações se aplicam para aquele diretório e todos os seu subdiretórios subseqüentes.

No geral, arquivos **.htaccess** usam a mesma sintaxe que os arquivos de configuração principal. O que você pode colocar nesses arquivos é determinado pela diretriz **AllowOverride**. Essa diretriz especifica, em categorias, quais diretrizes serão aceitas caso sejam encontradas em um arquivo .htaccess. Se uma diretriz for permitida em um arquivo .htaccess, a documentação para essa diretriz irá conter uma seção Override, especificando que valor precisa estar em AllowOverride para que esta diretriz seja permitida.

Em geral, no Debian configura-se a diretiva **AllowOverride** no arquivo **/etc/apache2/apache2.conf**. Lembre-se de se atentar aos diretórios corretos, no nosso caso deve-se alterar na tag **<Directory /var/www/>**.

Quando (não) usar arquivos .htaccess

No geral, você nunca deve usar arquivos .htaccess a não ser que você não tenha acesso ao arquivo de configuração principal do servidor. Existe, por exemplo, um erro de concepção que dita que a autenticação de usuários sempre deve ser feita usando os arquivos .htaccess. Esse simplesmente não é o caso. Você pode usar as configurações de autenticação de usuário no arquivo de configuração principal do servidor, e isso é, de fato, a maneira mais adequada de se fazer as coisas.

Arquivos .htaccess devem ser usados em casos onde os provedores de conteúdo do site precisem fazer mudanças na configuração do servidor por diretório, mas não tem acesso root ao sistema do servidor. Caso o administrador do servidor não esteja disposto a fazer mudanças freqüentes nas configurações do servidor, é desejável permitir que os usuários possam fazer essas mudanças através de arquivos .htaccess eles mesmos. Isso é particularmente verdade, por exemplo, em casos onde provedores estão fornecendo múltiplos sites para usuários em apenas uma máquina, e querem que seus usuários possam alterar suas configurações.

No entanto, de modo geral, o uso de arquivos .htaccess deve ser evitado

quando possível. Quaisquer configurações que você considerar acrescentar em um arquivo .htaccess, podem ser efetivamente colocadas em uma seção <Directory> no arquivo principal de configuração de seu servidor.

Existem duas razões principais para evitar o uso de arquivos .htaccess.

A primeira delas é a performance. Quando AllowOverride é configurado para permitir o uso de arquivos .htaccess, o Apache procura em todos diretórios por arquivos .htaccess. Logo, permitir arquivos .htaccess, causa um impacto na performance, mesmo sem você usá-los de fato! Além disso, o arquivo .htaccess é carregado toda vez que um documento é requerido.

Além disso, note que o Apache precisa procurar pelos arquivos .htaccess em todos os diretórios superiores, para ter o complemento total de todas as diretivas que devem ser aplicadas. Então, se um arquivo de um diretório /www/htdocs/example é requerido, o Apache precisa procurar pelos seguintes arquivos:

/.htaccess
/www/.htaccess
/www/htdocs/.htaccess
/www/htdocs/example/.htaccess

Assim, para cada acesso de arquivo fora desse diretório, existem 4 acessos ao sistema de arquivos adicionais, mesmo que nenhum desses arquivos estejam presentes. (Note que esse só será o caso se os arquivos .htaccess estiverem habilitados para /, o que normalmente não é verdade.)

A segunda consideração é relativa à segurança. Você está permitindo que os usuários modifiquem as configurações do servidor, o que pode resultar em mudanças que podem fugir ao seu controle. Considere com cuidado se você quer ou não dar aos seus usuários esses privilégios. Note também que dar aos usuários menos privilégios que eles precisam, acarreta em pedidos de suporte técnico adicionais. Tenha certeza que você comunicou aos usuários que nível de privilégios você os deu. Especificar exatamente o que você configurou na diretriz AllowOverride, e direcioná-los para a documentação relevante, irá poupá-lo de muita confusão depois.

O uso de arquivos .htaccess pode ser totalmente desabilitado ajustando a diretriz **AllowOverride** para **None**:

AllowOverride None

Na versão 2.4 do apache essa é a configuração padrão do servidor.

Criando o arquivo contendo usuários e senhas

Você terá que usar o utilitário "htpasswd", que serve para criar um arquivo de senhas criptografadas. Para isso vamos entrar na pasta do apache e criar lá esse arquivo com os comandos:

```
cd /etc/apache2/
```

```
htpasswd -c acesso.pwd paulo
```

A linha acima cria o usuário paulo e também cria o arquivo acesso.pwd contendo o usuário e sua senha criptografada. Para adicionar outro usuário, por exemplo o usuário joao, usa-se o htpasswd sem a opção -c:

```
htpasswd acesso.pwd joao
```

Agora, seguindo a linha de não usar um arquivo .htaccess no servidor, criaremos a tag <Directory > dentro do VirtualHost do nosso site segfatecou. Lembre-se de criar dentro do VirtualHost correto, ou seja, no nosso caso, o da porta 443! Adicione dentro dele as seguintes linhas:

```
<Directory /var/www/http/segfatecou/admin>
    AuthName “Acesso Restrito a pessoas autorizadas”
    AuthType Basic
    AuthUserFile /etc/apache2/acesso.pwd
    require valid-user
</Directory>
```

As configurações dentro dessa tag vão ter efeito para o diretório do nosso site, e também para os seus sub-diretórios.

Vamos explicar as opções acima:

AuthName: O nome que aparece como mensagem de Login. Pode usar algo como "Entre com Login e Senha", ou coisa deste tipo.

AuthType: Tipo de autenticação. Atualmente o Basic é o tipo mais comum.

AuthUserFile: Onde está o arquivo de usuários e senhas que nós criamos.

require valid-user: O que o Apache precisa para validar o acesso. Neste caso foi indicado que é necessário um usuário válido para acessar a página, ou seja, alguém que digite um usuário e senha e confira com o que está no arquivo de senhas. Pode-se restringir para apenas alguns usuários do arquivo de senhas. Por exemplo, se quiséssemos restringir apenas para os usuários paulo e joao, em vez de "**require valid-user**", ficaria "**require user paulo joao**".

Usando o arquivo .htaccess

Se a opção fosse pela criação do arquivo .htaccess, a configuração seria a seguinte:

Crie o arquivo **.htaccess** dentro da pasta que terá acesso restrito. Por exemplo para a pasta /var/www/http/segfatecou/admin/ seria criado lá dentro o arquivo **.htaccess** com o seguinte conteúdo:

```
AuthName “Acesso Restrito a pessoas autorizadas”
AuthType Basic
AuthUserFile /etc/apache2/acesso.pwd
require valid-user
```

Salve o arquivo. Agora, quando um usuário acessar a URL, o servidor verificará este arquivo .htaccess e solicitará ao cliente que informe um usuário e senha válidos.

Note que **AllowOverride AuthConfig** precisa estar habilitado para que estas diretrizes tenham efeito.

Referências para mais informações:

http://httpd.apache.org/docs/current/
https://httpd.apache.org/docs/current/howto/htaccess.html
https://pt.wikibooks.org/wiki/Guia_do_Linux/Avançado/Apache/Especificando_opções/permissões_para_as_páginas
http://www.tecmint.com/apache-security-tips/
http://www.ibliss.com.br/blog/hardening-apachenginx-ssl/
http://henriquecorrea.com/news/URLs_amigaveis_com_Mod_Rewrite
http://www.devmedia.com.br/configuracoes-basicas-no-apache2-artigo-revista-infra-magazine-1/26395
http://ask.xmodulo.com/turn-off-server-signature-apache-web-server.html

Comandos para testes:

```
nmap -sV 192.168.0.1
curl --head 192.168.0.1
curl --head www.segfatecou.edu.br/info.php
```