WAF-FLE: Guia de Implantação

Versão 0.6.3

Português - Brasil

Maio de 2014

WAF-FLE Guia de Implantação

Índice

Licença

Este obra está licenciado com uma Licença Creative Commons Atribuição-CompartilhaIgual 4.0 Internacional.

http://creativecommons.org/licenses/by-sa/4.0/

Introdução

Este “guia de implantação” guiará você na instalação e definições iniciais do WAF-FLE, a instalação é realmente simples, mas diferentes necessidades podem ser atendidas com cenários diferentes, como mostrado abaixo. Um processo detalhado de instalação é apresentado, e para configurações básicas e específicas, você poderá usar How-To específicos para os sistemas operacionais mais usados e atualmente suportados.

Cenários de implantação

O WAF-FLE como console para o ModSecurity pode ser implantado de várias maneiras, de acordo com as suas necessidades, como: laboratório/instalação pequena, instalação grande e instalação com volume muito grande de eventos, etc.

Neste guia, nos temos 3 cenários, mas você não está limitado a eles:

- Standalone: WAF-FLE no mesmo host do ModSecurity;
- Distribuído: WAF-FLE e banco de dados num mesmo host.
- Distribuído: servidor dedicado ao WAF-FLE, separado do banco de dados.

1. Standalone ou com o ModSecurity no mesmo

Para pequenas instalações ou laboratório, você pode instalar o WAF-FLE e o ModSecurity no mesmo host, is exige atenção e cuidados adicionais, para não fazer o ModSecurity bloquear os eventos enviados para o WAF-FLE. Isto é especialmente importante para evitar a amplificação de eventos (o ModSecurity envia um evento para o WAF-FLE, e ele mesmo bloqueia o envio do evento, gerando um novo evento, e assim indefinidamente). Abaixo um diagrama simples mostrando o que você precisa e como esta implantação é esperada.

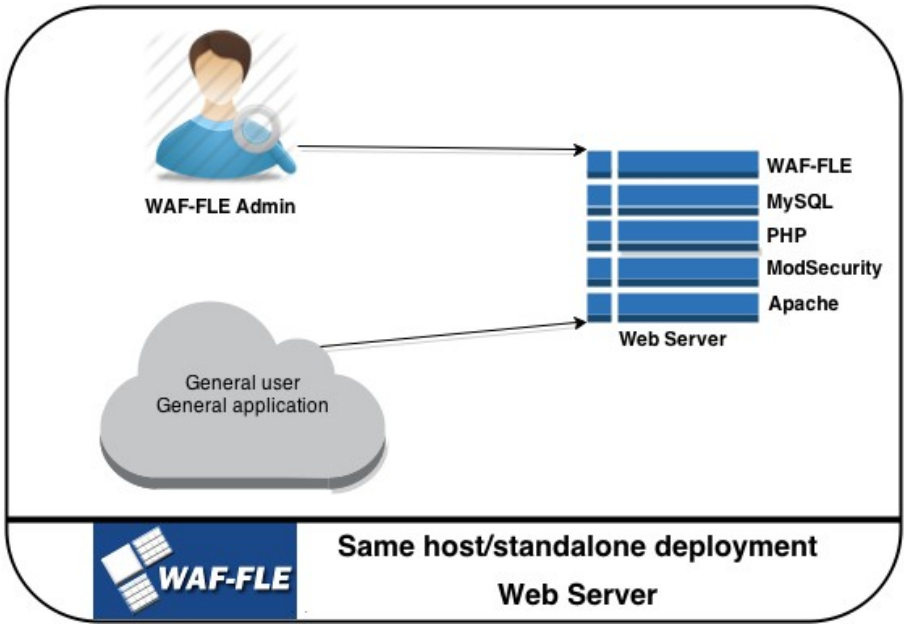


Figura 1: Standalone/Mesmo host do ModSecurity

2. Distribuído, WAF-FLE e banco de dados no mesmo host

Num ambiente com muitos sensores, com um alto volume de eventos, você pode preferir ou pode precisar usar um fazer uma implantação distribuída para consolidar os eventos de todos os servidores com ModSecurity. Isto irá requerer um servidor dedicado para o WAF-FLE, tendo seus eventos num ponto central, permitindo a você monitorar todos os eventos de forma consolidada.

Este cenário terá uma melhor performance quando comparado ao standalone, porque num alto volume de eventos a inserção ou consulta de eventos pode exigir muito da CPU ou do I/O, e é melhor evitar compartilhar estes recursos com a sua aplicação/servidor web.
Você pode mesmo ter eventos num site remoto (sobre uma WAN ou pela Internet), enviados em tempo real ou de forma agendada (você pode ver mais sobre isto na seção Configuração do Log Feeder).

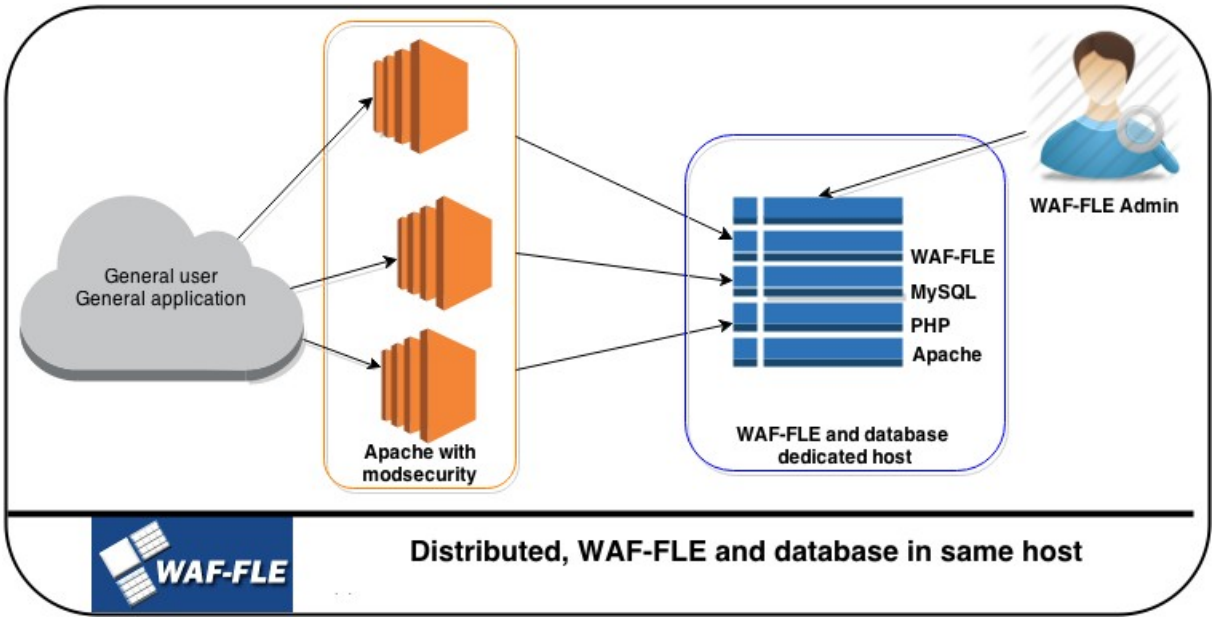
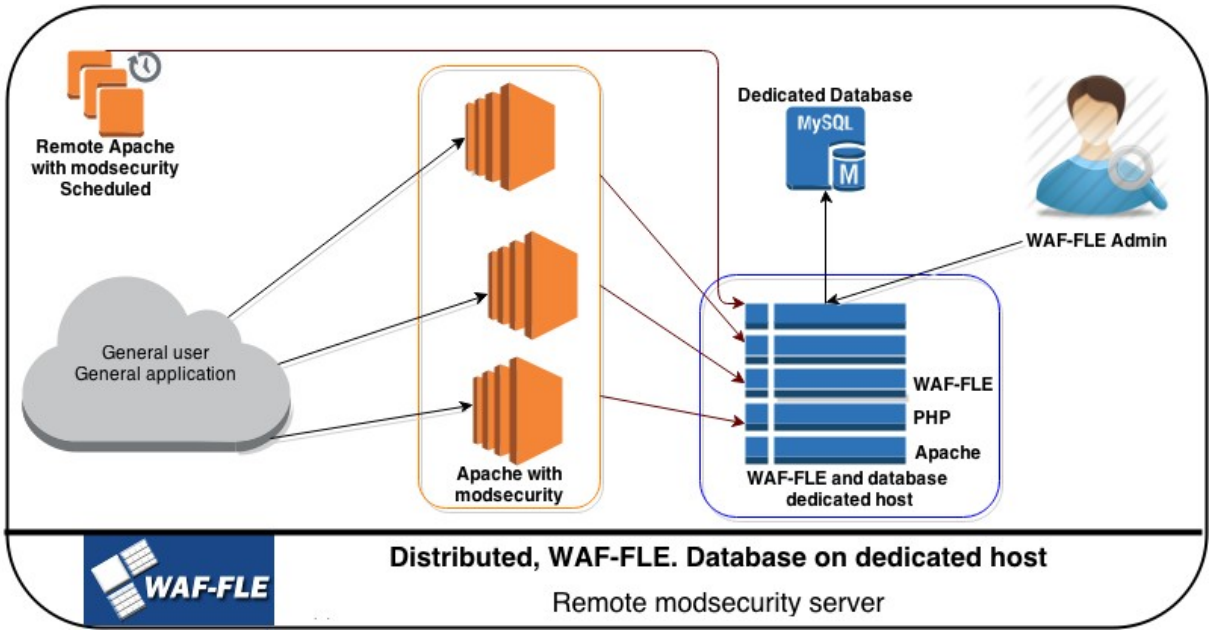


Figure 2: Distribuído, WAF-FLE e banco de dados no mesmo host

3. Distribuído, WAF-FLE dedicado, separado do banco de dados

Num grande volume de eventos, você pode precisar/preferir manter o banco de dados num servidor separador e dedicado. Mas para garantir boa performance, mantenha o WAF-FLE e o banco de dados próximos (tipicamente a latência de uma LAN estará OK)

Se você precisar, poderá ter dois servidores WAF-FLE, um dedicado para receber eventos, outro como console para visualizar eventos.



Comparando os tipos de implementação

	#1 - Standalone	#2 - Distribuído	#3 - Distribuído, com banco de dados dedicado
Simplicidade	++	+	-
Performance	-	+	++
Escalabilidade	-	+	++

DICA: Se você não tem certeza sobre o que é melhor para seu caso, então use o cenário #2, que tem um bom equilíbrio entre performance e uso de recursos.

Nota: Independente da sua escolha, mantenha a segurança em mente, Você pode:

- Usar SSL para transportar os eventos para o WAF-FLE de forma criptografada, bem como para acessar e visualizar os eventos (opcional);
- Todos os feeds de eventos e os acessos à console são controlados por autenticação (padrão)
- Os feeds de eventos podem ser restritos a um endereço IP, ou bloco de rede (opcional);

Requisitos

Para ter o WAF-FLE funcionando, você precisa de alguns componentes tipicamente presentes nas distribuições *nix mais comuns. Entretanto, as vezes, você precisará providenciar estes componentes de algum pacote de terceiros ou do próprio fonte. Na seção How-To Rápido, você encontrará um passo a passo paras as principais distribuições usadas atualmente.

- **Requerido**
  - Apache 2.x server
    - Apache mod-rewrite
  - PHP 5.3 ou maior
    - Extensão PHP PDO Mysql
    - Extensão PHP GeoIP
  - MySQL 5.1 ou posterior
- **Opcional**
  - APC (alternative php cache), pode ser instalado, e você pode habilitar/desabilitar seu uso no config.php.
    DICA: Você deve considerar manter o APC habilitado para melhorar a performance do WAF-FLE.

Instalação Geral

Com o tempo, a instalação do WAF-FLE tem se tornado mais simples, no passado você mesmo precisa criar a base de dados e checar se os requisitos estavam disponíveis. Começando na versão 0.6.0-rc foi introduzido o script de setup que guiará você na verificação dos requisitos, na criação do banco de dados e nas permissões do acesso ao banco de dados.

Antes de começar a instalar o WAF-FLE, você precisa instalar todos os componentes requeridos, para tanto, use os comandos típicos do seu sistema para fazer isso. Você pode verificar o passo a passo para alguns SO's e distribuições na seção How-To Rápidos.

Requisitos para instalação

1. Instale o Apache
   1.1. Habilite o mod-rewrite
2. Instale o servidor MySQL (se você for usar o banco de dados no mesmo servidor)
3. Instale o PHP
   3.1. Instale o php-pdo
   3.2. Instale o php-mysql
   3.3. Instale o php-apc

3.4. Instale o php-geoip

NOTA: O módulo PHP-GeoIP precisa de um certo '*hack*' para ser usado pelo WAF-FLE, uma vez que o módulo não tem uma função para checar Autonomous System Number (ASN). Para resolver isso, você precisa seguir os passos abaixo (que incluem o download da base de dados GeoIP da MaxMind):

```
mkdir /usr/share/GeoIP/
cd /usr/share/GeoIP/

http://geolite.maxmind.com/download/geoip/database/GeoLiteCountry/GeoIP.dat.gz
http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz
http://geolite.maxmind.com/download/geoip/database/asnum/GeoIPASNum.dat.gz

gzip -d GeoIP.dat.gz
gzip -d GeoLiteCity.dat.gz
gzip -d GeoIPASNum.dat.gz
mv GeoLiteCity.dat GeoIPCity.dat
# Para fazer a extensão GeoIP do php funcionar com a base de dados ASNum
cp GeoIPASNum.dat GeoIPISP.dat
```

Instalação do WAF-FLE

1. Faça o download da última versão do WAF-FLE em http://waf-fle.org/download

2. Extraia o tarball do WAF-FLE num diretório, como "/usr/local/", fora do raiz do conteúdo web do Apache (o que irá criar um diretório /usr/local/waf-fle). Você pode usar outro diretório, mas neste caso você precisa mudar a configuração do Apache para apontar para este novo diretório;

```
cd /usr/local
tar -zxvf /tmp/waf-fle-0.6.3.tar.gz
```

3. Mude para o diretório /usr/local/waf-fle;

4. Copie o "extra/waf-fle.conf" (a configuração do WAF-FLE para funcionar no Apache) para o diretório de configuração do Apache (normalmente /etc/httpd/conf.d ou /etc/apache2/conf.d), e edite o arquivo para refletir as peculiaridades do seu ambiente;

```
cp extra/waf-fle.conf /etc/apache2/conf.d/
```

4.1. Se você usar um diretório diferente para o WAF-FLE, você precisará editar o *waf-fle.conf* (a configuração para o Apache), procurando pelas diretivas *alias* e *Directory*, mudando elas para refletir a localização do WAF-FLE, como mostrado abaixo:

```
alias /controller/ /usr/local/waf-fle/controller/
...
<Directory /usr/local/waf-fle/controller/>
...
alias /waf-fle /usr/local/waf-fle/dashboard/
...
<Directory /usr/local/waf-fle/dashboard/>
...
```

4.2. Verifique as permissões do diretório no Apache. Em algumas instalações do Apache (como o FreeBSD) a configuração padrão precisar ser alterada para dar permissão para o WAF-FLE, no Apache 2.0 e 2.2 você deve usar "A*llow from all*", no Apache 2.4 e posterior use "*Require all granted*", isto está explicado no arquivo de configuração waf-fle.conf, como mostrado abaixo:

```
<Directory /usr/local/waf-fle/controller/>
...
   # On some installation, like FreeBSD you need to adjust the
   # 'Allow from' directive bellow
   # For Apache 2.0/2.2 use "Allow", uncomment the line below
   # Order allow,deny
   # Allow from all

   # For Apache 2.4 and later "Require", uncomment the line below
   # Require all granted

   AddType application/x-httpd-php .php
</Directory>

...
<Directory /usr/local/waf-fle/dashboard/>
...
   # On some installation, like FreeBSD you need to adjust the
   #'Allow from' directive bellow
   # For Apache 2.0/2.2 use "Allow", uncomment the line below
   # Order allow,deny
   # Allow from all

   # For Apache 2.4 and later "Require", uncomment the line below
   # Require all granted

   AddType application/x-httpd-php .php
</Directory>
```

4.3. Se você preferir ou tiver um servidor dedicado para o WAF-FLE, você pode fazer o Apache redirecionar o raiz do servidor “/” para o “/waf-fle/”, como mostrado abaixo. No *waf-fle.conf* isto é comentado (desabilitado) por padrão.

```
# ATTENTION: If you deploy WAF-FLE as a dedicated virtual host/server
# you can uncomment the lines bellow to get a more simple access to
# web interface. You need mod_alias enabled to use this way.
#
<Location />
   RedirectMatch ^/$ /waf-fle/
</Location>
...
```

4.4. Para adicionar mais segurança à sua instalação, considere habilitar o SSL no seu servidor, para que o acesso seja criptografado tanto no envio de eventos, quanto no acesso à console;

4.5. Recarregue a configuração do Apache.

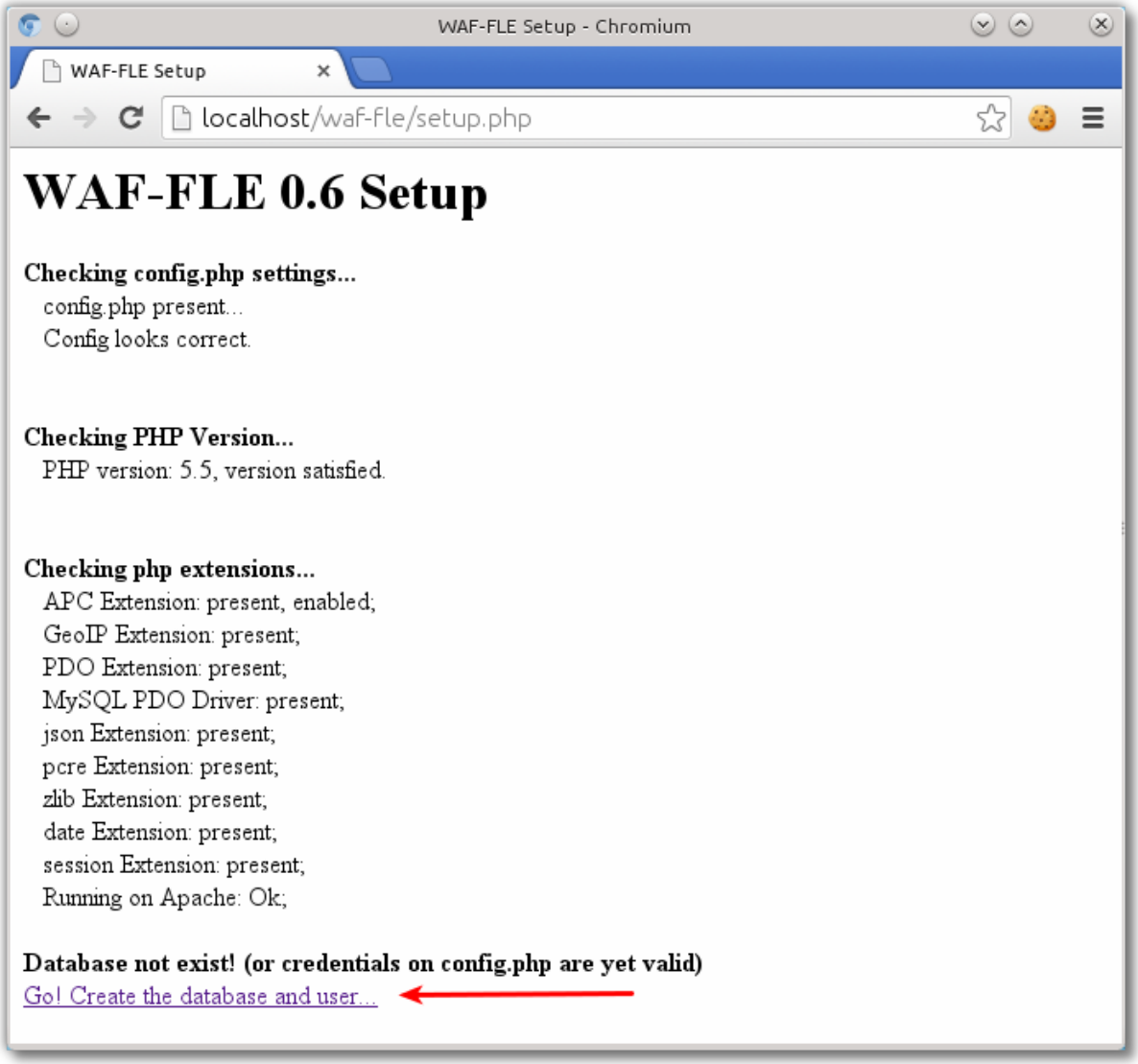
5. No diretório do WAF-FLE crie seu arquivo de configuração copiando o config.php.example para config.php:

```
cp config.php.example config.php
```

6. Edite o arquivo config.php para definir o servidor do banco de dados, usuário, senha e nome da base de dados. O banco de dados e as permissões do usuário serão criados pelo script de setup. Durante o setup, mantenha a diretiva $SETUP como *true*.

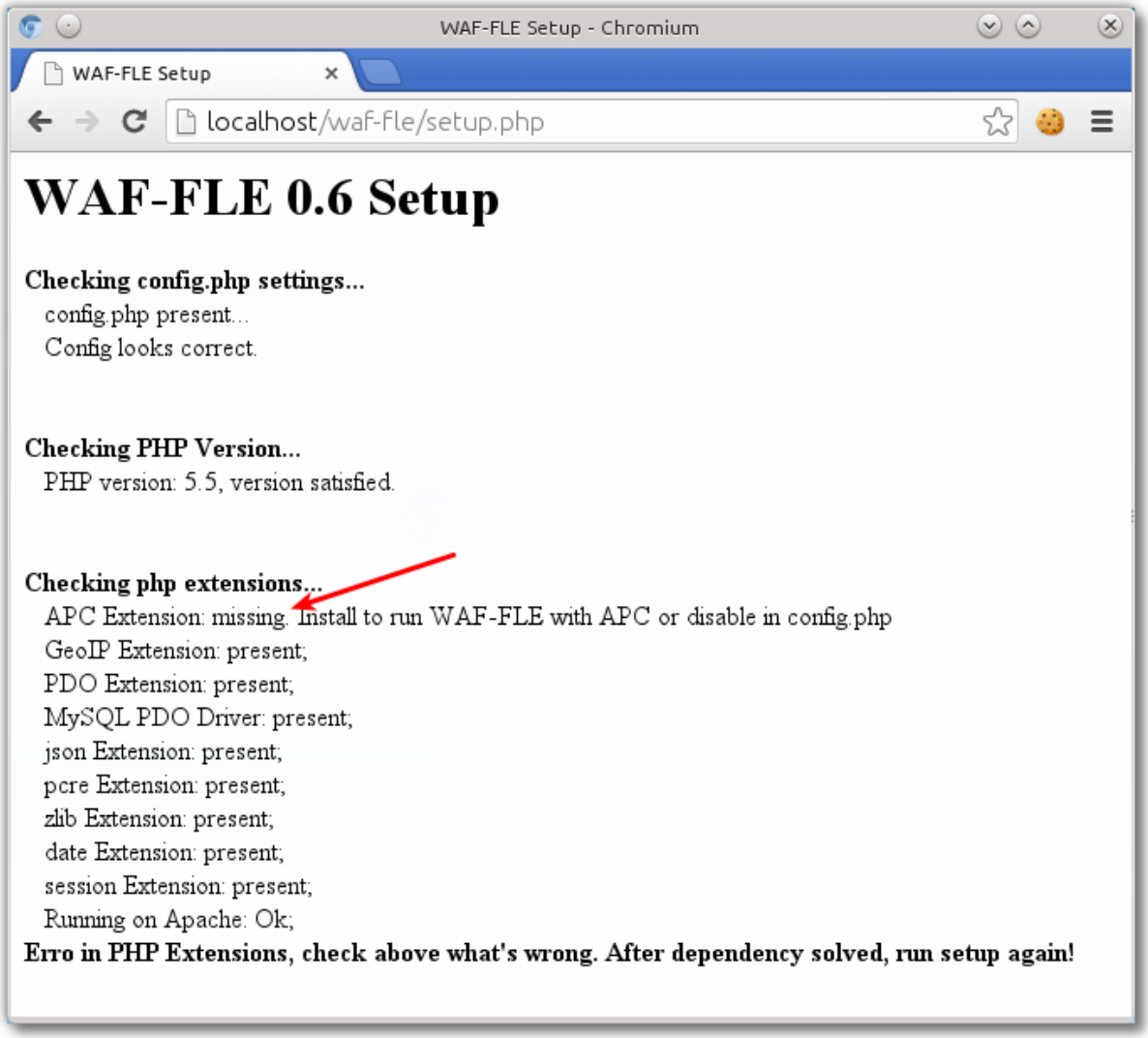
```
$DB_HOST  = "localhost";
$DB_USER  = "waffle_user";
$DB_PASS  = "<FILL_User_Password>";
$DATABASE = "waffle";
...
$SETUP = true;
```

7. Com seu navegador, acesse http://<your server>/waf-fle/ para iniciar no script do setup;

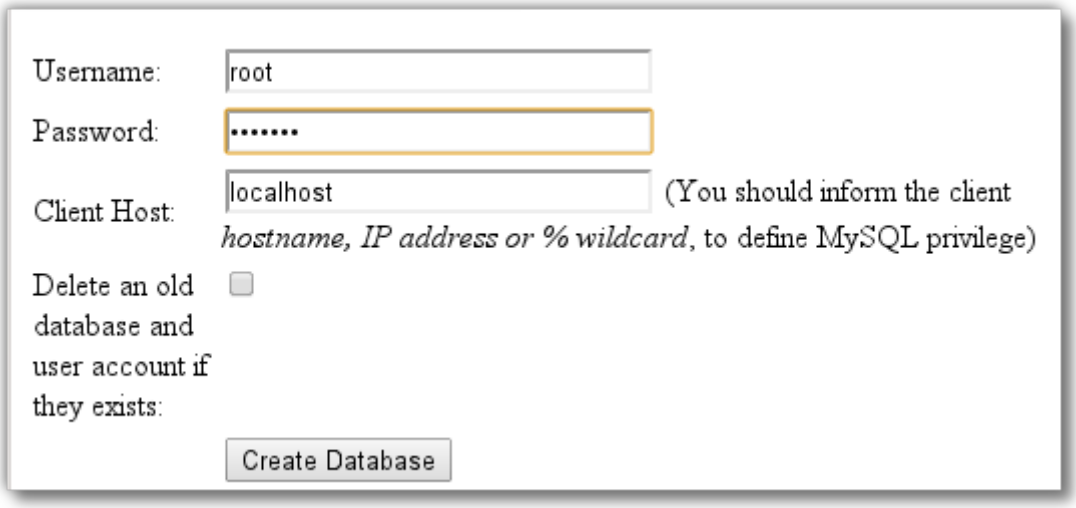


8. O script de instalação irá primeiro checar e o servidor tem todos os componentes requeridos instalados e corretamente configurados. O script de instalação verificará se a base de dados já existe (para evitar sobrescrevê-la), como mostrado abaixo.

8.1. Se algum componente requerido estiver faltando, você receberá uma mensagem de erro, como mostrado abaixo:



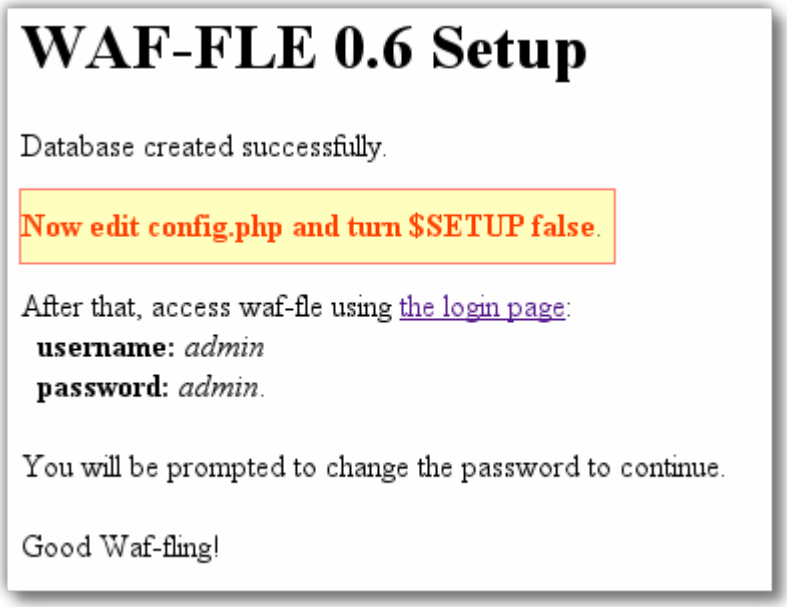
9. Vá para o próximo passo clicando em "*Go! Create the database and user...*", e informe as credenciais para acessar seu banco de dados MySQL como administrador (Username, Password), o servidor do banco de dados (que pode ser o localhost, para o MySQL e WAF-FLE no mesmo host, ou outro nome de servidor para o MySQL e o WAF-FLE em hosts diferentes). Então clique em "*Create Database*".



10. Se tudo correr bem, você deverá receber a mensagem abaixo, mostrando que a base de dados foi criada com sucesso. Preste atenção à enfase "*Now edit config.php and turn $SETUP false*".

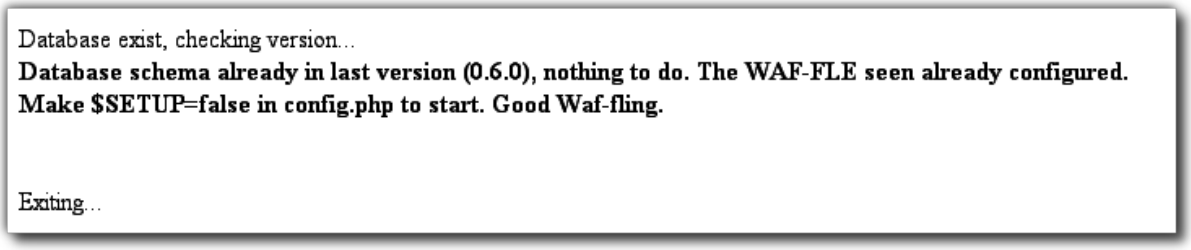
```
$SETUP = false;
```

10.1. Após mudar a variável $SETUP, você pode clicar em "login page" para acesso a página de login do WAF-FLE;

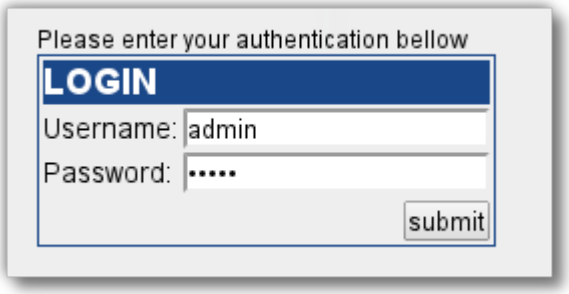


10.2. Se você não alterar $SETUP para *false* no arquivo config.php, o

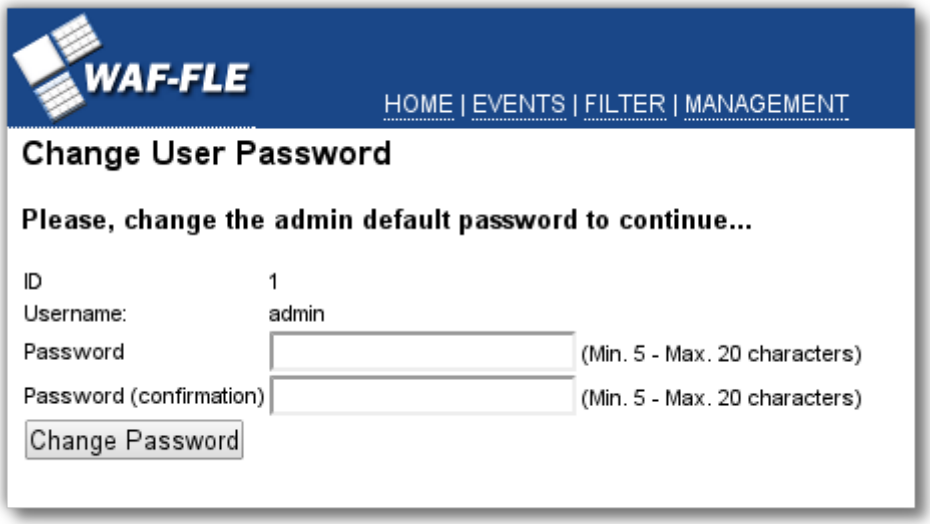
WAF-FLE irá redirecionar você novamente para o script de instalação, e irá exibir a mensagem de erro abaixo:



11. Faça login no WAF-FLE. (**Username** admin, **Password**: admin)



12. Você será forçado a alterar a senha do usuário *admin,* escolha uma senha forte.



13. Vá para o menu *Management*, para definir seu primeiro sensor.

14. **Nota para o usuários do CRS**: se você usar o WAF-FLE numa máquina com o ModSecurity, usando as regras Core Rule Set (CRS), você pode experimentar problemas com algumas regras que rodam na fase 1. Um exemplo é a regra "allowed method" (id 960032)", outras regras que

inspecionam a fase 1 podem causar problemas também. A exceção feita no waf-fle.conf não funciona com estas regras. Desta forma, você é aconselhado a criar e usar uma regra de *bypass*, coloque-a na ordem apropriada na sua estrutura do CRS.
Por exemplo, você pode criar um arquivo "modsecurity_crs_11_waffle.conf" com a regra abaixo:

```
SecRule REQUEST_FILENAME '^/controller/$' \
"phase:1,msg:'Match',id:99999,nolog,noauditlog, \
allow,ctl:RuleEngine=On"
```

Não se preocupe, esta regra liga o *engine* do ModSecurity apenas para o caminha do *controller* (para evitar logs não desejados no modo *DetectionOnly*), e mantém as demais regras com o *Engine Status* definido na configuração do ModSecurity.

Atualização do WAF-FLE

O processo de upgrade permite que você inicie o uso de uma nova versão, aproveite as novas características e correções de bugs. O processo pode ser muito simples ou, as vezes, ele pode ser mais complexo e lento. Preste atenção às notas abaixo, para cada versão, o que também é explicado no arquivo README em cada nova versão.

Atenção: cada vez que uma nova versão altera o esquema da base de dados, o sistema de arquivos da sua base de dados deverá ter mais de 50% de espaço livre para proceder com as modificações. Se você não tiver os espaço disponível, será preciso excluir eventos antigos, de outra forma você não será capaz de fazer o upgrade do banco de dados, e por consequência do WAF-FLE.

Da versão	Para a versão	Alteração no esquema da base de dados	Notas e procedimento para atualização
0.6.0	0.6.3	Não	**WAF-FLE**: Copie os novos arquivos sobre os antigos. **Mlog2Waffle**: Copie os novos arquivos sobre os antigos. **Process**: Atualização muito simples/rápido
0.6.0-rcX	0.6.0	Não	**WAF-FLE:** Apenas copie os novos arquivos sobre os antigos. **Process**: Atualização muito simples/rápido
0.5x	0.6.0-rc1	Sim	**WAF-FLE:** Copie os novos arquivos sobre os antigos. **Mlog2Waffle**: primeiro release do mlog2waffle. **Process:** um processo mais lento, porque há mudanças no esquema da base de dados. Para fazer a atualização do esquema e migrar os dados você deve editar o config.php e definir todos os parâmetros necessários como servidor da base de dados, nome do usuário e senha, e definir a variável $SETUP para *"true"* no config.php. Depois disso, você deve acessar com seu navegador o http(s)://webserver/waf-fle/setup.php, seguindo os passos do script de instalação/upgrade.

			Se você já tem muitos eventos na base de dados, a migração poderá levar um longo tempo, seja paciente. O script de atualização não comprimirá eventos antigos, e não processará dados GeoIP a partir do IP de origem.
0.5	0.5.1	Não	**WAF-FLE:** Apenas copie os novos arquivos sobre os antigos. **Process**: Atualização muito simples/rápido

Definição de Sensor

Enquanto a instalação do WAF-FLE é feita num único processo, a configuração de sensores pode ser feitas muitas vezes, uma vez para cada novo sensor. Siga o procedimento abaixo para cada sensor na sua rede.
Nota: você pode usar um sensor definido no WAF-FLE para agregar um cluster de servidores rodando ModSecurity, como se fosse um sensor só, mas com muitos servidores. Cabe a você, usar um sensor para cada servidor ou um sensor para todos os servidores do cluster.
Nota: Sem definir um sensor no WAF-FLE, você não será capaz de receber novos eventos. Então, esta passo é muito importante.

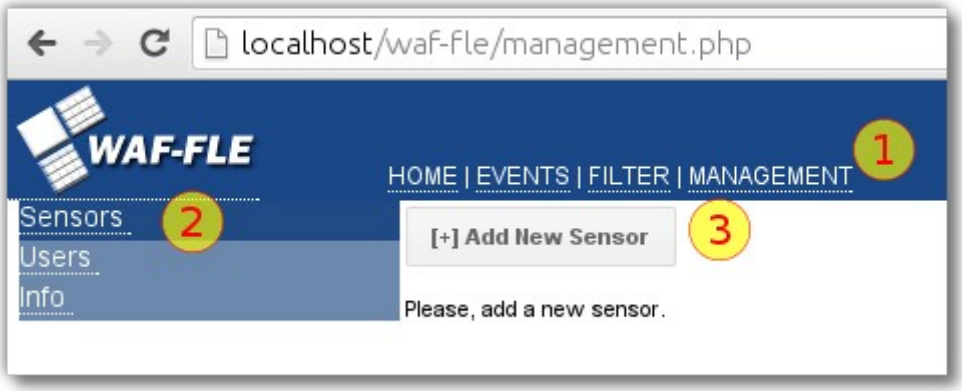
A configuração de um Sensor é feita em duas etapas, ambas discutidas aqui:

1. Definição do sensor no WAF-FLE,
2. Configuração do Event Feeder (processo que envia os logs), feito em cada servidor rodando ModSecurity.

Definição do Sensor

Para criar um novo sensor, siga os passos abaixo:

1. Vá ao menu *Management;*
2. Clique em *Sensors;*
3. Clique no botão "*Add New Sensor*"



1. Preencha os campos para identificar o novo sensor
   a. **Sensor**: é o nome do sensor, este é um campo obrigatório, e é mostrado na lista de eventos e no dashboard como o sensor que originador do evento. Ele também é usado como *username* para a autenticação do *event feeder* do sensor.
   b. **Password**: é usado para a autenticação do event feeder. É um campo obrigatório. É requerido que a senha tenha no mínimo 5 e no máximo 20

caracteres.

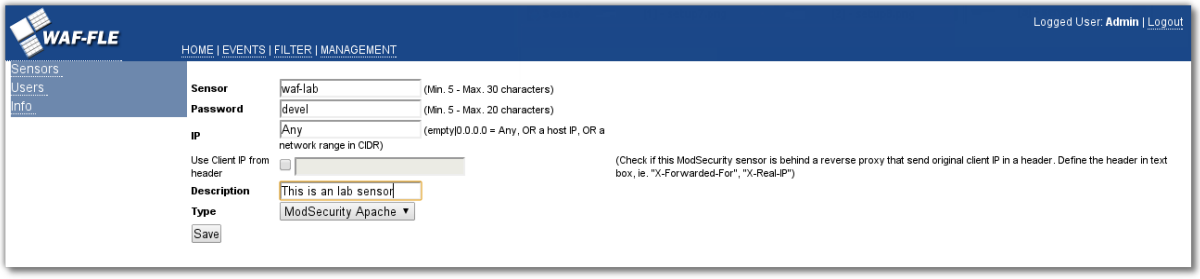
c. **IP**: Você deve informar um endereço IP para o sensor, que pode ser:
   1. Any/0.0.0.0/Empty para aceitar eventos de qualquer endereço de origem;
   2. Um endereço IP, para aceitar eventos somente de um endereço IP de origem;
   3. Um endereço de rede, especificado na notação CIDR (exemplo, 192.168.1.0/24), onde o sensor pode ser qualquer endereço daquela rede;

d. **Use Client IP from header**: As vezes você precisa usar o ModSecurity atrás de um Proxy Reverso (como Varnish, Nginx etc), neste caso pode ser útil (ou mesmo mandatório) informar qual cabeçalho HTTP é usado pelo proxy reverso para registrar o endereço IP do cliente remoto. O típico é "X-Forwarded-For", mas você pode usar qualquer outro, basta informar no campo apropriado. Este é um campo opcional.

e. **Description**: Uma descrição, opcional, do sensor.

f. **Type**: Que tipo de sensor é esse. Atualmente somente "ModSecurity Apache" está disponível;

2. Clique em *save,* após preencher todos os campos necessários;



3. Após receber alguns eventos, a definição do sensor mostrará algumas informações úteis sobre os eventos gerados pelo sensor, detalhados abaixo:

a. **Event's total**: Quantos eventos este sensor enviou (em ainda permanecem) na base de dados;

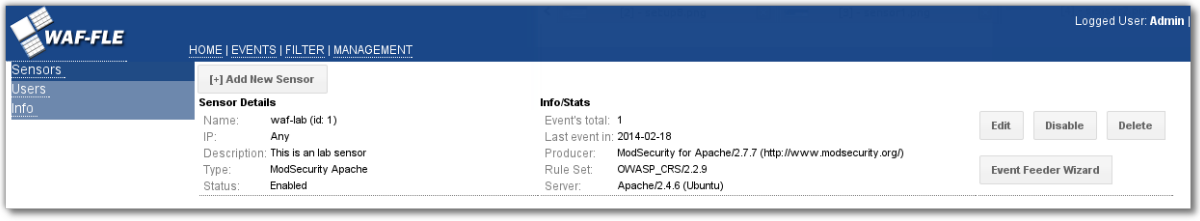
b. **Last event in**: Quando o último evento chegou;

c. **Producer**: Qual versão do ModSecurity enviou o último evento;

d. **Rule Set**: Qual conjunto de regras (rule set) gerou o último evento;

e. **Server**: Qual servidor está rodando o sensor;

f. **Status**: Informa se o sensor está habilitado ou desabilitado no WAF-FLE;



4. Você pode também, realizar outras operações nesta interface:

a. **Edit**: Permite que você altere parâmetros do sensor, todos os parâmetros podem ser alterados;
b. **Disable**: Isto evita que novos eventos sejam aceitos no WAF-FLE, preservando todos os eventos da base de dados. Isto é útil quando você precisa fazer alguma manutenção no banco de dados;
c. **Delete**: Exclui o sensor, E TODOS OS SEUS EVENTOS;
d. **Event Feeder Wizard**: Cria um template útil para configurar sensores e seus event feeder nos servidores do ModSecurity. Mais detalhado abaixo:

Configuração de Event Feeder

O event feeder é localizado no lado do cliente, é um agente que irá enviar os eventos do ModSecurity (logs) para o WAF-FLE. Atualmente existem duas formas de conseguir isto, primeiro usando o *mlog2waffle* e segundo o *mlogc*. Cada feeder ter duas formas de trabalhar, como destacado abaixo.

Mlog2waffle

O mlog2waffle é um componente multi-thread do WAF-FLE, escrito em Perl, para enviar eventos do ModSecurity para o WAF-FLE. Ele é um substituto para o mlogc. O Mlog2waffle lê o arquivo de índice de eventos gerado pelo ModSecurity e envia os eventos para o WAF-FLE, em tempo real (usando o modo 'tail') ou periodicamente (em modo batch). Ele não é encadeado com os logs do ModSecurity, o que evita que o log *feeder* atrapalhe o servidor web.

- **Características**:
  - Roda em tempo real, seguindo o final (tail) do índice de logs do ModSecurity;
  - Roda agendadamente, no crontab;
  - Suporta o envio de eventos com HTTPS (SSL/TLS);
  - Suporte a Multi-thread para melhoria na performance;
  - Usa keep-alive do HTTP para economizar recursos e aumentar a performance;
- **Requisitos**:
  - Perl
  - libwww (6.0 ou mais recente para aceitar certificados autoassinados)
  - File::Pid
  - File::Tail
  - LWP::UserAgent
- **Modos de operação:**
  - *Daemon de serviço ou modo tail*: significa que o arquivo de log do ModSecurity será gravado no disco, e o mlog2waffle lerá esse log, tão logo ele seja gerado, processando todas as entradas. O log de auditoria é

armazenado no disco, até que cada entrada seja processada e enviada para o WAF-FLE. Isto faz os logs serem enviados em tempo real.

- *Agendado no crontab ou modo batch*: significa que o arquivo de log do ModSecurity será gravado no disco, e uma tarefa agenda no *crontab* rodará o mlog2waffle que lerá e processará o arquivo de log. O log de auditoria é armazenado no disco até que o mlog2waffle processe cada entrada e a envia para o WAF-FLE. Isto faz com que os logs sejam enviados periodicamente (dependendo da frequência programada no crontab), mas não imediatamente. Tipicamente, os logs são enviados a cada 5 minutos (uma vez por hora, uma vez por dia, você escolhe).

Mlogc

O Mlogc é um componente multi-thread do ModSecurity, escrito em C, para enviar eventos do ModSecurity para consoles como o WAF-FLE ou outras, ele é a ferramenta original para o envio de logs para a console. Mlogc lê os eventos gerados pelo ModSecurity e envia para a console (ex. WAF-FLE), em tempo real (quando usado no modo "piped") ou periodicamente (em modo batch). O modo *piped* pode causar algum impacto no servidor web, no caso do mlogc se comportar de forma anormal, como muitas vezes citado no mailing-list e na ferramenta de bug-tracking do ModSecurity.

- **Características**:
  - Roda em tempo real, encadeado (piped) com os logs do ModSecurity;
  - Roda em modo agendado, no crontab (usando um script Perl como apoio);
  - Suporta o envio com HTTPS (SSL/TLS);
  - Suporte a multi-thread para incremento da performance;
- **Requisitos**:
  - Mesmos requisitos para compilar o ModSecurity, mais
  - libcurl
- **Modos de operação:**
  - *Modo Piped*: significa que o arquivo de índice dos logs do ModSecurity irá alimentar o Mlogc diretamente e não será escrito no disco. O logs de auditoria são mantidos armazenados no disco, até que o programa processe cada entrada e as envie para o WAF-FLE. Isto faz os logs serem assim que são gerados, em tempo real.

  - *Agendados no crontab ou modo batch*: significa que o arquivo de índice do log do ModSecurity será gravado no disco, e uma tarefa agendada no crontab irá ler e processar o arquivo de log. O log de auditoria é armazenado no disco até que a tarefa processe cada entrada, e as envie para o WAF-FLE. Isto faz os logs serem enviados periodicamente (dependendo da frequência programada no crontab), mas não imediatamente. Tipicamente, os logs são enviados a cada 5 minutos (uma

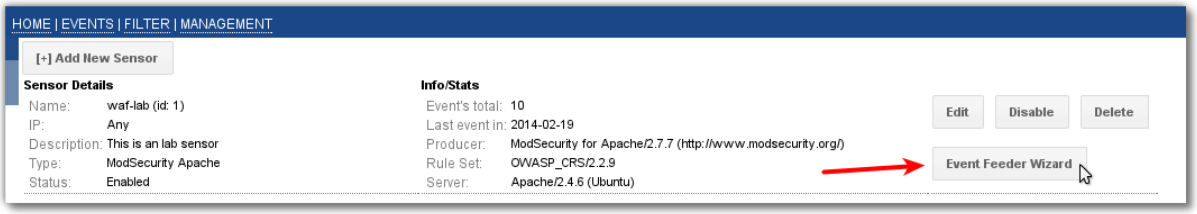
vez por hora, uma vez por dia, você escolhe).

Event Feeder Wizard

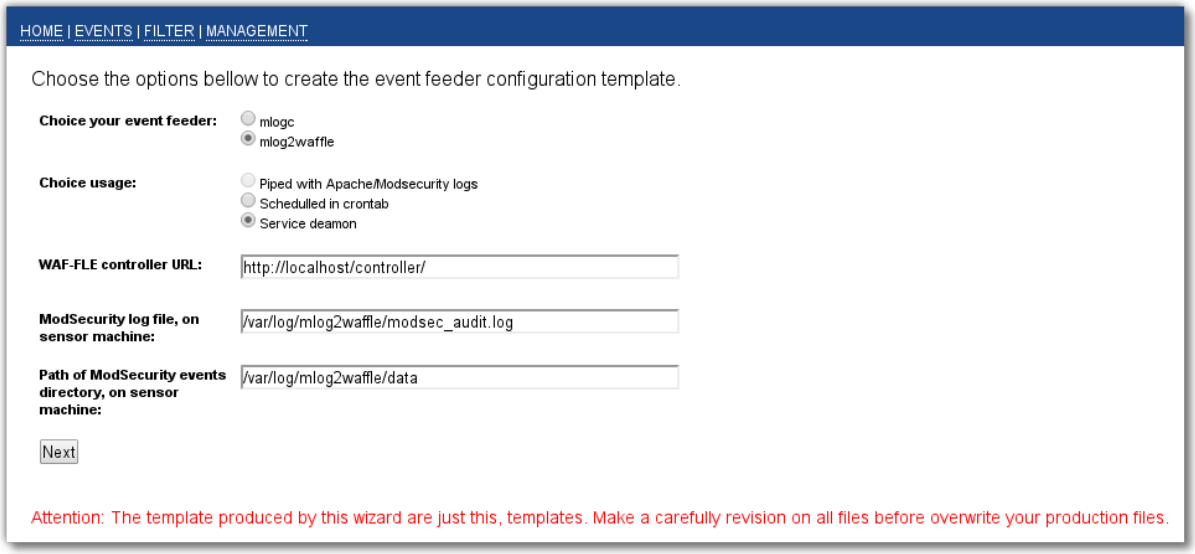
O Event Feeder Wizard irá auxiliar na configuração dos sensores, usando o mlog2waffle ou o mlogc, como serviço, ou modos batch e pided, provendo todos os arquivos de configuração e alterações necessárias para configurar o *feeder* de sua escolha.
O wizard fornece templates para configurar as diretivas “Log Directives” do modsecurity.conf e as configurações do mlog2waffle/mlogc, bem como instruções para a criação de diretórios e suas permissões.

Note: Todos os arquivos e comandos mostrados no Wizard devem ser configurados/executados em cada sensor que você instalar.

1. Para acessar o **Event Feeder Wizard**, vá para Management/Sensors e clique no botão Event Feeder Wizard do sensor que você quer configurar.



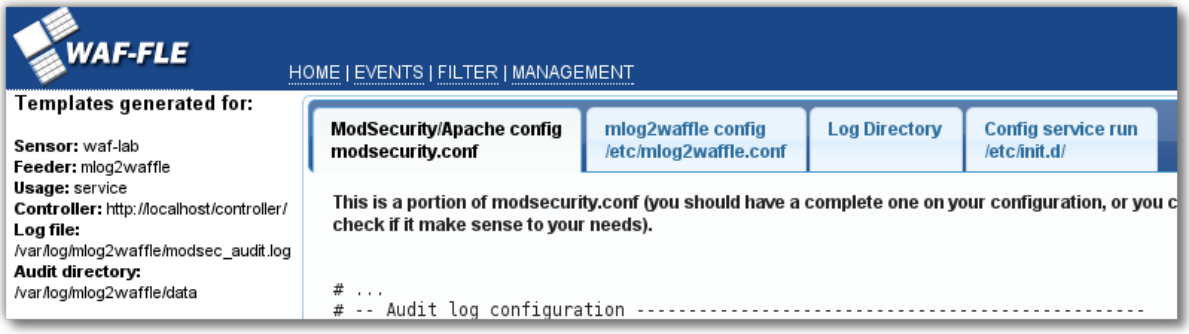
2. Faça as seleções apropriadas:
   2.1. Selecione o event feeder de sua preferência (mlog2waffle ou mlogc);
   2.2. Selecione o modo de operação: serviço (disponível apenas com o mlog2waffle), piped (disponível apenas com o mlogc) ou batch;
   2.3. Confirme ou edite a URL do controller. Esta é a URL para a qual o event feeder irá enviar os eventos para o WAF-FLE, tipicamente ela tem o forma http://<WAF-FLE>/controller.
   2.4. Verifique o arquivo de log do ModSecurity (na máquina do sensor) sugerido, necessário apenas nos modos serviço e batch. Isto está definido na diretiva SecAuditLog do modsecurity.conf.
   2.5. Verifique o caminho sugerido para o diretório de eventos do ModSecurity, na máquina do sensor. Isto está definido na diretiva *SecAuditLogStorageDir do modsecurity.conf.*



3. Clique Next para criar os templates dos arquivos de configuração. Estes templates devem ser copiados ou editados por você na máquina do sensor;
4. Dependendo da sua seleção, você terá alguns arquivos para editar/criar.

	mlog2waffle	**mlogc**
Serviço	- modsecurity.conf - /etc/mlog2waffle.conf - Diretório para dados de eventos - Habilitar script de inicialização do mlog2waffle	—
Batch	- modsecurity.conf - /etc/mlog2waffle.conf - Diretório para dados de eventos - Adiciona o mlog2waffle ao crontab	- modsecurity.conf - /etc/mlogc.conf - Diretório para dados de eventos - Adiciona o push-mlogc.sh ao crontab
Piped	—	- modsecurity.conf - /etc/mlogc.conf - Diretório para dados de eventos

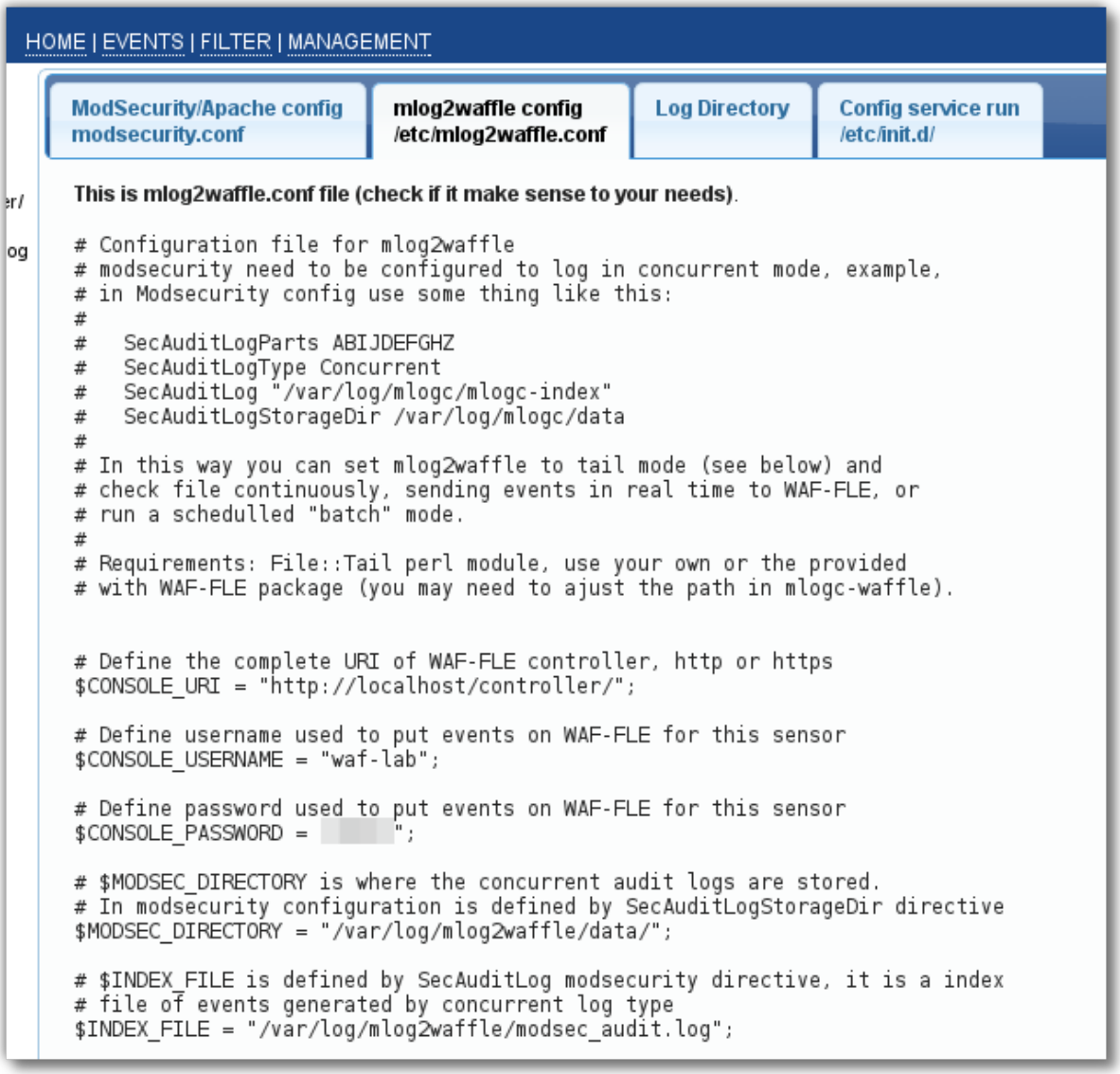
4.1. No lado esquerdo, você pode revisar as escolhas que fez:



4.2. A primeira aba apresenta o que precisa ser alterado no arquivo modsecurity.conf para usar o log apropriadamente com o event feeder que você escolheu. O modsecurity.conf é um arquivo que define as configurações básicas para o ModSecurity, incluindo como se comportará o log;

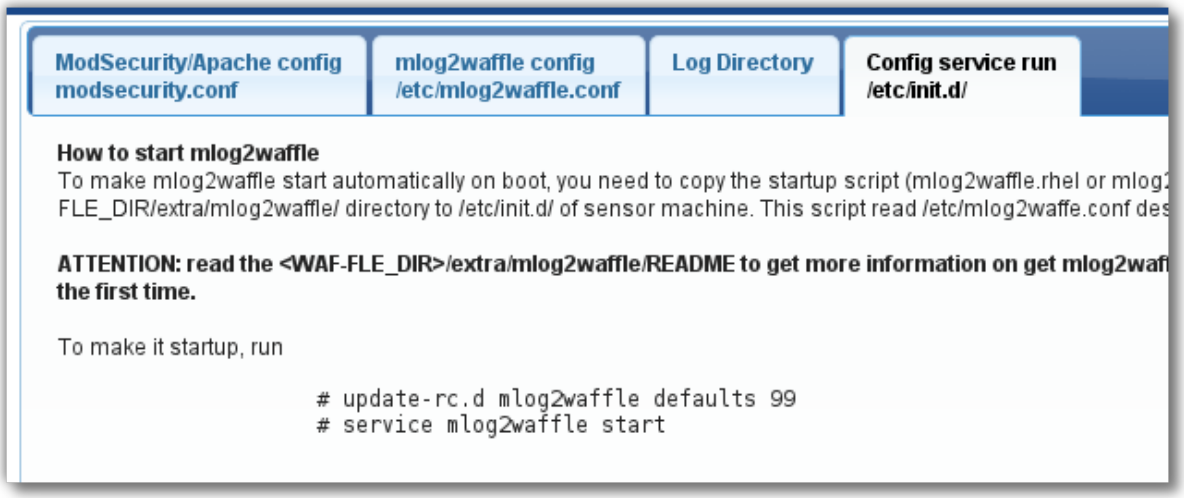
4.3. A segunda aba mostra o template do mlog2waffle.conf ou o mlogc.conf. Este é um arquivo de configuração completo, você pode substituir seu

arquivo com este novo, que inclui a URL do WAF-FLE, nome de usuário e senha necessários para autenticar o sensor no WAF-FLE.
Outros detalhes do mlog2waffle.conf ou do mlogc.conf estão fora do escopo deste guida, mas eles estão documentados no arquivo de configuração padrão de ambos log feeders.



4.4. A terceira aba se refere aos diretórios necessários para armazenar os logs antes deles serem enviados para o WAF-FLE.
Nota: Preste atenção do comando *chown*, ele precisar ser executado com o usuário apropriado, o usuário que roda o Apache.

4.5. A quarta aba (se estiver presente), é o init script para o serviço mlog2waffle, ou a entrada crontab para rodar o mlog2waffle ou o mlogc em modo batch.



5. Após editar todos os arquivos, você precisa:
   5.1. Recarregar o Apache para ativar as mudanças do modsecurity.conf;
   5.2. Para o mlog2waffle em modo serviço, iniciar o serviço:
   (/etc/init.d/mlog2waffle start)
6. Testar o sistema. Faça uma requisição que deve disparar um evento no ModSecurity, e verifique se o evento foi enviado ao WAF-FLE.

Configurando o mlog2waffle.conf como um Service Daemon (modo tail)

Como mostrado no *Event Feeder Wizard, você pode usar o mlog2waffle numa sensor ModSecurity como um Daemon de Serviço (modo tail)* usando os passos abaixo (use o *Event Feeder Wizard* como referência):

1. Configure o m*odsecurity.conf* para ajustar as definições de log:
   Todas as configurações desta caixa são relevantes para a configuração do log, mas a parte em negrito é fundamental. Para entender mais sobre o log do ModSecurity acesse:

https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual#wiki-secauditlog

```
# ...
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4(?!04))"

# Log everything we know about a transaction.
SecAuditLogParts ABIDEFGHZ

SecAuditLogType Concurrent
# Specify the log index
SecAuditLog /var/log/mlog2waffle/modsec_audit.log
# Specify the path for concurrent audit logging.
SecAuditLogStorageDir /var/log/mlog2waffle/data
# ...
```

2. Copie os arquivos do mlog2waffle do seu servidor WAF-FLE para o servidor do sensor e instale os pacotes requeridos (veja mais detalhes nos How-To abaixo ou no arquivo README do mlog2waffle).

```
cd waf-fle/extra/mlog2waffle
cp mlog2waffle /usr/sbin
cp mlog2waffle.conf /etc
# for RedHat based dist
cp mlog2waffle.rhel /etc/init.d/mlog2waffle
# for Debian/Ubuntu based dist
cp mlog2waffle.ubuntu /etc/init.d
```

3. Crie um diretório de log para o ModSecurity e dê a permissão apropriada (a localização e a permissão devem ser revisadas, pois são dependentes do sistema). A propriedade do /var/log/mlog2waffle/data precisa ser dada ao usuário que executa o Apache (ex. nobody, www-data, apache):

```
mkdir -p /var/log/mlog2waffle/data
chown -R nobody /var/log/mlog2waffle/data
```

4. Edite o arquivo de configura (mlog2waffle.conf) e ajuste de acordo com suas necessidades. Em destaque as diretivas mais relevantes (outras estão comentadas no arquivo):

```
vi /etc/mlog2waffle.conf
```

```
# ...
# Define the complete URI of WAF-FLE controller, http or https
$CONSOLE_URI = "https://<host>/controller/";

# Define username used to put events on WAF-FLE for this sensor
$CONSOLE_USERNAME = "<sensor-name>";

# Define password used to put events on WAF-FLE for this sensor
$CONSOLE_PASSWORD = "<password>";

# $MODSEC_DIRECTORY is where the concurrent audit logs are stored.
#  In modsecurity configuration is defined by SecAuditLogStorageDir
# directive
$MODSEC_DIRECTORY = "/var/log/mlog2waffle/data/";

# $INDEX_FILE is defined by SecAuditLog modsecurity directive, it is
# a index file of events generated by concurrent log type
$INDEX_FILE = "/var/log/mlog2waffle/modsec_audit.log";
# ...
# Define the execution mode:
#  "tail": for run continuously, waiting for new entries on log
# file;
$MODE = "tail";
# ...
```

5. Define a inicialização automática do mlog2waffle. Inicie manualmente na primeira vez:

```
ln -s /etc/init.d/mlog2waffle /etc/rc3.d/S99mlog2waffle
/usr/sbin/mlog2waffle
```

6. Recarregue o Apache para efetivar as mudanças feitas no arquivo modsecurity.conf;

Configurando o mlog2waffle.conf agendado no crontab (modo batch)

Como mostrado no Event Feeder Wizard, você pode usar o mlog2waffle num sensor ModSecurity em modo agendado no crontab (modo batch) usando os passos abaixo (use o Event Feeder Wizard como referência):

1. Configure o modsecurity.conf para ajustar as definições de log:
   Todas as configurações desta caixa são relevantes para a configuração do log, mas a parte em negrito é fundamental. Para entender mais sobre o log do ModSecurity acesse:
   https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual#wiki-secauditlog

```
# ...
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4(?!04))"

# Log everything we know about a transaction.
SecAuditLogParts ABIDEFGHZ

SecAuditLogType Concurrent
# Specify the log index
SecAuditLog /var/log/mlog2waffle/modsec_audit.log
# Specify the path for concurrent audit logging.
SecAuditLogStorageDir /var/log/mlog2waffle/data
# ...
```

2. Copie os arquivos do mlog2waffle do seu servidor WAF-FLE para o servidor do sensor e instale os pacotes requeridos (veja mais detalhes nos How-To abaixo ou no arquivo README do mlog2waffle).

```
cd waf-fle/extra/mlog2waffle
cp mlog2waffle /usr/sbin
cp mlog2waffle.conf /etc
cp mlog2waffle.cron /etc/cron.d/mlog2waffle
```

3. Crie um diretório de log para o ModSecurity e dê a permissão apropriada (a localização e a permissão devem ser revisadas, pois são dependentes do sistema). A propriedade do /var/log/mlog2waffle/data precisa ser dada ao usuário que executa o Apache (ex. nobody, www-data, apache):

```
mkdir -p /var/log/mlog2waffle/data
chown -R nobody /var/log/mlog2waffle/data
```

4. Edite o arquivo de configura (mlog2waffle.conf) e ajuste de acordo com suas necessidades. Em destaque as diretivas mais relevantes (outras estão comentadas no arquivo):

```
vi /etc/mlog2waffle.conf
```

```
# ...
# Define the complete URI of WAF-FLE controller, http or https
$CONSOLE_URI = "https://<host>/controller/";

# Define username used to put events on WAF-FLE for this sensor
$CONSOLE_USERNAME = "<sensor-name>";

# Define password used to put events on WAF-FLE for this sensor
$CONSOLE_PASSWORD = "<password>";

# $MODSEC_DIRECTORY is where the concurrent audit logs are stored.
#  In modsecurity configuration is defined by SecAuditLogStorageDir
# directive
$MODSEC_DIRECTORY = "/var/log/mlog2waffle/data/";

# $INDEX_FILE is defined by SecAuditLog modsecurity directive, it is
# a index file of events generated by concurrent log type
$INDEX_FILE = "/var/log/mlog2waffle/modsec_audit.log";
# ...
# Define the execution mode:
# "batch": for run and exit at end, but recording (offset file) the
# position in the last run, speeding up next execution. You can
#  schedulle the mlog2waffle in crontab to run periocally (for
# example, each 5min).
$MODE = "batch";
# ...
```

5. Edite a entrada no crontab (copiado no passo 2) para executar o mlog2waffle com a periodicidade necessária:

```
vi /etc/cron.d/mlog2waffle
```

```
PATH=/sbin:/usr/sbin:/bin:/usr/bin

# start mlog2waffle periodically, in this case 5 minutes
*/5 * * * * root mlog2waffle
```

6. Recarregue o Apache para efetivar as mudanças feitas no arquivo modsecurity.conf;

Configurando o mlogc agendado no crontab (modo batch)

Como mostrado no Event Feeder Wizard, você pode usar o mlogc num sensor ModSecurity em modo agendado no crontab (modo batch) usando os passos abaixo (use o Event Feeder Wizard como referência):

1. Configure o modsecurity.conf para ajustar as definições de log:
   Todas as configurações desta caixa são relevantes para a configuração do log, mas a parte em negrito é fundamental. Para entender mais sobre o log do ModSecurity acesse:
   https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual#wiki-secauditlog

```
# ...
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4(?!04))"

# Log everything we know about a transaction.
SecAuditLogParts ABIDEFGHZ

SecAuditLogType Concurrent
# Specify the log index
SecAuditLog /var/log/mlogc/modsec_audit.log
# Specify the path for concurrent audit logging.
SecAuditLogStorageDir /var/log/mlogc/data
# ...
```

2. Verifique se o mlogc está instalado no /usr/local/modsecurity/bin/, se você compilou o ModSecurity na máquina ele deve estar lá (ex. /usr/local/modsecurity/bin/mlogc).

3. Crie um diretório de log para o ModSecurity e dê a permissão apropriada (a localização e a permissão devem ser revisadas, pois são dependentes do sistema). A propriedade do /var/log/mlogc/data precisa ser dada ao usuário que executa o Apache (ex. nobody, www-data, apache):

```
mkdir -p /var/log/mlogc/data
chown nobody /var/log/mlogc/data
```

4. Edite o arquivo de configura (mlogc.conf) e ajuste de acordo com suas necessidades. Em destaque as diretivas mais relevantes (outras estão comentadas no arquivo):

```
vi /etc/mlogc.conf
```

```
# Points to the root of the installation. All relative
# paths will be resolved with the help of this path.
CollectorRoot        "/var/log/mlogc"

# ModSecurity Console receiving URI. You can change the host
# and the port parts but leave everything else as is.
ConsoleURI           "http://<host>/controller/"

# Sensor credentials
SensorUsername       "<sensor-name>"
SensorPassword       "<password>"

# Base directory where the audit logs are stored.  This can be
  specified
# as a path relative to the CollectorRoot, or a full path.
LogStorageDir        "data"

# Transaction log will contain the information on all log collector
# activities that happen between checkpoints. The transaction log
# is used to recover data in case of a crash (or if Apache kills
# the process).
TransactionLog       "mlogc-transaction.log"

# The file where the pending audit log entry data is kept. This file
# is updated on every checkpoint.
QueuePath            "mlogc-queue.log"

# The location of the error log.
ErrorLog             "mlogc-error.log"

# ...
```

5. Verifique se o script mlogc-batch-load.pl (instalado por padrão) está em /usr/local/modsecurity/bin/. Ele lerá e processará as entradas do log, e usará o mlogc para enviar os eventos para o WAF-FLE. O mlogc-batch-load.pl vem com o código fonte do ModSecurity.

6. Crie um script, com nome push-mlogc.sh e coloque em /usr/local/sbin/. Use as linhas como mostrado abaixo. Torne o arquivo executável.

```
vi /usr/local/modsecurity/bin/push-mlogc.sh
```

```
#!/bin/bash

# Check if a old execution still running, and kill it
Status=0;
while [ $Status -eq 0 ]; do
  PmlogcBatch=`/sbin/pidof -x /usr/local/modsecurity/bin/mlogc-batch-load.pl`
  PplStatus=$?
  Pmlogc=`/sbin/pidof -x /usr/sbin/mlogc`
  PmlogcStatus=$?

  if [ $PplStatus -eq 0 ]; then
     kill -9 $PmlogcBatch
     echo "Killing $PmlogcBatch"
  fi
  if [ $PmlogcStatus -eq 0 ]; then
     kill -9 $Pmlogc
     echo "Killing $Pmlogc"
  fi

  if [ $PplStatus -ne 0 -a $PmlogcStatus -ne 0 ]; then
     Status=1;
  fi
done

# Start mlogc push
echo "Sending logs to WAF-FLE";
date
/usr/local/modsecurity/bin/mlogc-batch-load.pl    /var/log/mlogc/data    \
/usr/local/modsecurity/bin/mlogc /etc/mlogc.conf

find  /var/log/mlogc/data -type d -empty -delete
```

```
chmod +x /usr/local/modsecurity/bin/push-mlogc.sh
```

7. Crie uma entrada no crontab para rodar o push-mlogc.sh a cada 5 minutos;

```
PATH=/sbin:/usr/sbin:/bin:/usr/bin

# start push-mlogc periodically, in this case 5 minutes
*/5 * * * * root /usr/local/modsecurity/bin/push-mlogc.sh
```

8. Recarregue o Apache para efetivar as mudanças feitas no arquivo modsecurity.conf;

Configurando o mlogc Piped com log do Apache/ModSecurity

Como mostrado no Event Feeder Wizard, você pode usar o mlogc num sensor ModSecurity em modo piped (que é a configuração original) usando os passos abaixo (use o Event Feeder Wizard como referência):

1. Configure o modsecurity.conf para ajustar as definições:
   Todas as configurações desta caixa são relevantes para a configuração do log, mas a parte em negrito é fundamental. Para entender mais sobre o log do ModSecurity acesse:
   https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual#wiki-secauditlog

```
# ...
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4(?!04))"

# Log everything we know about a transaction.
SecAuditLogParts ABIDEFGHZ

SecAuditLogType Concurrent
# Specify the log index
SecAuditLog "|/usr/local/bin/mlogc /etc/mlogc.conf"
# Specify the path for concurrent audit logging.
SecAuditLogStorageDir /var/log/mlogc/data
# ...
```

2. Verifique se o mlogc está instalado em /usr/local/modsecurity/bin/, se você compilou o ModSecurity na máquina, ele deverá estar lá (ex. /usr/local/modsecurity/bin/mlogc).

3. Crie o diretório de log do ModSecurity e dê as permissões apropriadas (a localização e as permissões devem ser revisadas, pois são dependentes do sistema). A propriedade do /var/log/mlogc/data precisa ser data ao usuário que executa o Apache (ex. nobody, www-data, apache):

```
mkdir -p /var/log/mlogc/data
chown nobody /var/log/mlogc/data
```

4. Edite o arquivo de configuração (mlogc.conf) e ajuste de acordo com a sua necessidade. Em destaque as diretivas mais relevantes (outras estão comentadas no arquivo):

```
vi /etc/mlogc.conf
```

```
# Points to the root of the installation. All relative
# paths will be resolved with the help of this path.
CollectorRoot       "/var/log/mlogc"

# ModSecurity Console receiving URI. You can change the host
# and the port parts but leave everything else as is.
ConsoleURI          "http://<host>/controller/"

# Sensor credentials
SensorUsername      "<sensor-name>"
SensorPassword      "<password>"

# Base directory where the audit logs are stored.  This can be
  specified
# as a path relative to the CollectorRoot, or a full path.
LogStorageDir       "data"

# Transaction log will contain the information on all log collector
# activities that happen between checkpoints. The transaction log
# is used to recover data in case of a crash (or if Apache kills
# the process).
TransactionLog      "mlogc-transaction.log"

# The file where the pending audit log entry data is kept. This file
# is updated on every checkpoint.
QueuePath           "mlogc-queue.log"

# The location of the error log.
ErrorLog            "mlogc-error.log"

# ...
```

5. Recarregue o Apache para efetivar as mudanças feitas no arquivo modsecurity.conf;

How-To Rápido

Este How-To foi criado para ajudar você na configuração do WAF-FLE com sistemas operacionais e distribuições populares mais rapidamente. Com o tempo, outros sistemas operacionais e distribuições serão adicionados. O foco é prover os passos, nos comandos e pacotes específicos do SO/distribuição, para alcançar os requisitos para executar o WAF-FLE em cada um desses sistemas.

NOTA: Estas instruções são básicas, podem ser incompletas e são somente o ponto de partida para criar um sistema capaz de rodar o WAF-FLE.

Maiores explicações sobre a instalação do WAF-FLE é fornecida na primeira parte deste guia.

CentOS/RedHat 6.5

Requisitos do WAF-FLE

```
yum install httpd
yum install mysql-server
yum install php php-pdo php-mysql php-pecl-apc

# Install GeoIP and GeoIP for PHP[1], download from EPEL
http://pkgs.org/download/GeoIP and http://pkgs.org/download/php-pecl-geoip
yum localinstall php-pecl-geoip-1.0.8-3.el6.x86_64.rpm
yum localinstall GeoIP-1.4.8-1.el6.x86_64.rpm

# After GeoIP install, download all MaxMind GeoIP Database, as follow:
cd /usr/share/GeoIP/
wget
http://geolite.maxmind.com/download/geoip/database/GeoLiteCountry/GeoIP.dat
.gz
wget http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz
wget
http://geolite.maxmind.com/download/geoip/database/asnum/GeoIPASNum.dat.gz

gzip -d GeoIP.dat.gz
gzip -d GeoLiteCity.dat.gz
gzip -d GeoIPASNum.dat.gz
mv GeoLiteCity.dat GeoIPCity.dat
# To make php GeoIP extension works with ASNum database
cp GeoIPASNum.dat GeoIPISP.dat

/etc/init.d/httpd start
# check if Apache is working properly
# check if IPTables rules allow your client connect to Apache
/etc/init.d/mysqld start
# define a password for root user in MySQL
/usr/bin/mysql_secure_installation
```

Siga os passos descritos no capítulo: **Instalação do WAF-FLE**

1 Ao invés de instalar do EPEL, você pode seguir as instruções de: http://blog.thecodingmachine.com/fr/content/installing-php-geolocalization-extension-centos

Debian 7 (wheezy) /Ubuntu 12.04 LTS (precise)

Requisitos do WAF-FLE

```
apt-get install apache2
apt-get install mysql-server
apt-get install php5 php5-mysql php-apc php5-geoip

# After GeoIP install, download all MaxMind GeoIP Database, as follow:
cd /usr/share/GeoIP/
wget
http://geolite.maxmind.com/download/geoip/database/GeoLiteCountry/GeoIP.dat
.gz
wget http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz
wget
http://geolite.maxmind.com/download/geoip/database/asnum/GeoIPASNum.dat.gz

gzip -d GeoIP.dat.gz
gzip -d GeoLiteCity.dat.gz
gzip -d GeoIPASNum.dat.gz
mv GeoLiteCity.dat GeoIPCity.dat
# To make php GeoIP extension works with ASNum database
cp GeoIPASNum.dat GeoIPISP.dat

# enable mod_rewrite, needed to waf-fle
a2enmod rewrite
service apache2 restart

# check if Apache is working properly
# check if IPTables rules allow your client connect to Apache
```

Siga os passos descritos no capítulo: Instalação do WAF-FLE

FreeBSD 10

```
pkg install apache24
# To run apache www server from startup, add
# apache24_enable="yes" in your /etc/rc.conf.
# Uncomment the line
# "LoadModule rewrite_module libexec/apache24/mod_rewrite.so" in
# /usr/local/etc/apache24/httpd.conf

pkg install mysql55-server
# To run MySql from startup, add mysql_enable="YES" in your /etc/rc.conf.


# to install php you will need to use Ports[2]
# by now, you should use php 5.4 ( php5) as APC is not
# available in PHP 5.5

portsnap fetch
cd /usr/ports/lang/php55
make config
# select "Build Apache Module"
make install

cp /usr/local/etc/php.ini-production /usr/local/etc/php.ini

# edit /usr/local/etc/apache24/httpd.conf to include
AddType application/x-httpd-php .php
# and
<IfModule dir_module>
 DirectoryIndex index.php index.html
</IfModule>

cd /usr/ports/lang/php5-extensions
make config

# select the following extensions (keep the already selected as is)
      JSON
      PDO
      PDO_MYSQL
      ZLIB
      SESSION

make install

cd /usr/ports/www/pecl-APC/
make installation
vi /usr/local/etc/php/extensions.ini:

   apc.enabled=1
   apc.shm_size=32M # or other value appropriated to your setup
```

2 Passos adaptados de http://fosskb.wordpress.com/2013/07/15/famp-installing-apache2-2-mysql-php-on-freebsd-9-1/

```
cd /usr/ports/net/pecl-geoip
make install
# After GeoIP install, download all MaxMind GeoIP Database, follow:
cd /usr/local/share/GeoIP

wget
http://geolite.maxmind.com/download/geoip/database/GeoLiteCountry/GeoIP.dat
.gz
wget http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz
wget
http://geolite.maxmind.com/download/geoip/database/asnum/GeoIPASNum.dat.gz

gzip -d GeoIP.dat.gz
gzip -d GeoLiteCity.dat.gz
gzip -d GeoIPASNum.dat.gz
mv GeoLiteCity.dat GeoIPCity.dat
# To make php GeoIP extension work with ASNum database
cp GeoIPASNum.dat GeoIPISP.dat

/usr/local/sbin/apachectl restart
```

Siga os passos descritos no capítulo: Instalação do WAF-FLE

Dimensionamento

O WAF-FLE não demanda muito recurso da máquina, mas isto pode variar conforme a sua necessidade.
Os recursos para o WAF-FLE são diretamente dependente dos eventos recebidos/processados por segundo e pelo uso da console. Mais eventos: mais recursos necessários (CPU, memória e armazenamento).

Cenários de referência:

- # 1 – Standalone ou com ModSecurity no mesmo host (laboratório):
  Inicie com 1G de memória;
- # 1 – Standalone ou com ModSecurity no mesmo host (produção):
  Considere as necessidades da sua aplicação e adicione memória extra. Lembre-se que neste caso você já tem ou ainda terá a instalação do ModSecurity, que tem seus próprios requisitos.
- #2 – Distribuído, WAF-FLE e banco de dados no mesmo host:
  Inicie com 4G, ao menos 2 (V)CPU Core, e armazenamento suficiente para manter seus eventos, considere também espaço disponível para fazer a manutenção do MySQL. Num servidor físico, considere o uso de uma boa solução RAID para não ter o I/O como um problema.
- #3 – Distribuído, WAF-FLE dedicado, separado do banco de dados:
  Neste caso, espera-se muito mais eventos por segundo.
  - Para o WAF-FLE, inicie com 4GB de memória, e ao menos 4 (V)CPU Cores, e ajuste a configuração do Apache para suportar uma maior concorrência de conexões;
  - Para o MySQL, inicie com 8GB de memória, otimize o armazenamento, e mantenha espaço suficiente para armazenar os eventos.

Quanto espaço em disco é necessário para o banco de dados dependerá de quantos eventos você espera manter num período. Por exemplo, considere um evento (como armazenado no banco de dados, compactado) entre 5kB e 10kB, recebendo 10.000 eventos por sensor por dia. Neste caso, você terá 300.000 eventos por mês, por sensor. O que consumirá 2.9GB por sensor, por mês.

Use a formula:

$$\text{Database Size (in GB)} = \frac{\text{event size (in KB)} * \textit{number of events} * \textit{Days do keep} * \textit{Number of Sensors}}{1024 * 1024}$$

exemplo:

$$2{,}86 = \frac{10 * 10000 * 30 * 1}{1024 * 1024}$$

Nota: O tamanho dos eventos é variável, dependem das seções do log gravadas pelo ModSecurity e do tamanho do corpo da resposta (se for logado).

Lembre-se: O total de eventos é impactado por três fatores:

1. Regras: Se você fizer/usar uma regra que é disparada por qualquer coisa (gerando falso positivo), você terá muitos eventos, não importando se está no modo blocking ou detection only.
2. Requisições disparando eventos: mesmo regras bem ajustadas, podem produzir muitos eventos se você receber uma grande quantidade de requisições que disparam as regras. Scanners, worms, DoS e outros tipos de ataque são exemplos disso.
3. Log de auditoria do Relevant Only/Relevant Status (SecAuditEngine RelevantOnly/ SecAuditLogRelevantStatus): se você configurar o "Relevant status" de forma muito abrangente, você poderá receber milhares (ou milhões) de eventos no caso de um erro na aplicação.

Ajuste fino do MySQL

Para instalações crescendo, você pode precisar ajustar a sua instalação do MySQL, para o que você pode consultar referências mais especializadas abaixo:

Entendendo caches e buffers do MySQL, uma boa explicação sobre isso em: http://www.mysqlperformanceblog.com/2006/09/29/what-to-tune-in-mysql-server-after-installation/

Para uma revisão mais automática do seu servidor MySQL você pode usar ferramentas que irão analisar a sugerir ajustes específicos para a sua necessidade, um bom artigo sumarizando excelentes ferramentas é: http://www.askapache.com/mysql/performance-tuning-mysql.html

O Percona Tools pode ajudar a configurar o seu arquivo *my.cnf* fazendo perguntas sobre a sua necessidade: https://tools.percona.com/wizard (grátis, mas necessita cadastro).

O usuário do WAF-FLE, Fábio Miranda, compartilhou uma dica de ajuste do MySQL, com grande melhoria no tempo de resposta (de uma base de dados com 70GB, rodando numa VM com 6GB de RAM):

No /etc/my.cnf :

```
#tuning
query_cache_size=64M
thread_cache_size=4
table_cache=256
key_buffer_size=1300M
```