CONTEÚDO

RELACIONADO

Como instalar o nginx no CentOS 6 com o yum

Visão

Configuração inicial do servidor com o Ubuntu 12.04

Visão

// Tutorial //

Como proteger um servidor Nginx com Fail2Ban no Ubuntu 20.04

Publicado em 2 de agosto de 2022

Firewall Nginx Segurança Ubuntu Ubuntu 20.04

Por Alex Garnet
Escritor Técnico de DevOps Sênior

Não está usando o Ubuntu 20.04 ?
Escolha uma versão ou distribuição diferente.

Ubuntu 20.04

Introdução

Ao configurar um servidor da Web, geralmente há seções do site às quais você deseja restringir o acesso. Os aplicativos da Web geralmente fornecem seus próprios métodos de autenticação e autorização, mas o próprio servidor da Web pode ser usado para restringir o acesso se eles forem inadequados ou indisponíveis. No entanto, a autenticação do servidor web também representa uma *superfície* de ataque ou *vetor* de ataque muito previsível através do qual as pessoas podem tentar obter acesso.

Qualquer serviço exposto à rede é um alvo potencial dessa maneira. Se você revisar os logs de qualquer servidor da Web com tráfego intenso, muitas vezes verá tentativas de login repetidas e sistemáticas que representam ataques de força bruta por usuários e bots.

As implantações de produção em larga escala para as quais essa responsabilidade é completamente inaceitável geralmente implementarão uma VPN como o WireGuard na frente de quaisquer endpoints privados, de modo que seja impossível conectar-se diretamente a esses URLs da Internet externa sem abstração de software ou gateways adicionais. Essas soluções VPN são amplamente confiáveis, mas adicionarão complexidade e podem quebrar algumas automações ou outros pequenos ganchos de software.

Antes ou além de se comprometer com uma configuração completa de VPN, você pode implementar uma ferramenta chamada **Fail2ban** . O Fail2ban pode mitigar significativamente os ataques de força bruta criando regras que alteram automaticamente a configuração do firewall para banir IPs específicos após um certo número de tentativas de login malsucedidas. Isso permitirá que seu servidor se proteja contra essas tentativas de acesso sem a sua intervenção.

Neste guia, você aprenderá como instalar `fail2ban` em um servidor Ubuntu 20.04 e configurá-lo para monitorar seus logs Nginx para tentativas de intrusão.

Pré-requisitos

- Acesso a um ambiente de servidor Ubuntu 20.04 com um usuário não **root** com `sudo` privilégios para realizar tarefas administrativas. Para aprender como criar esse usuário, siga o guia de configuração inicial do servidor Ubuntu 20.04 .
- Nginx instalado em seu sistema, seguindo as **etapas 1 e 2** deste guia sobre como instalar o Nginx no Ubuntu 20.04 .
- Nginx instalado e configurado com autenticação de senha seguindo Como configurar a autenticação de senha com Nginx no Ubuntu 20.04 .

Etapa 1 – Instalando e configurando o Fail2ban

Fail2ban está disponível nos repositórios de software do Ubuntu. Comece executando os seguintes comandos como usuário não root para atualizar suas listagens de pacotes e instalar o Fail2ban:

```
$ sudo apt update
$ sudo apt install fail2ban
```

cópia de

O Fail2ban configurará automaticamente um serviço em segundo plano após a instalação. No entanto, está desabilitado por padrão, pois algumas de suas configurações padrão podem causar efeitos indesejados. Você pode verificar isso usando o `systemctl` comando:

```
$ systemctl status fail2ban.service
```

cópia de

```
Output
○ fail2ban.service - Fail2Ban Service
     Loaded: loaded (/lib/systemd/system/fail2ban.service; disabled; vendor preset: enabled
     Active: inactive (dead)
       Docs: man:fail2ban(1)
```

Você pode habilitar o Fail2ban imediatamente, mas primeiro, você revisará alguns de seus recursos.

O serviço fail2ban mantém seus arquivos de configuração no `/etc/fail2ban` diretório. Existe um arquivo com padrões chamado `jail.conf`. Vá para esse diretório e imprima as primeiras 20 linhas desse arquivo usando `head -20`:

```
$ cd /etc/fail2ban
$ head -20 jail.conf
```

cópia de

```
Output
#
# WARNING: heavily refactored in 0.9.0 release.  Please review and
#          customize settings for your setup.
#
# Changes:  in most of the cases you should not modify this
#           file, but provide customizations in jail.local file,
#           or separate .conf files under jail.d/ directory, e.g.:
#
# HOW TO ACTIVATE JAILS:
#
# YOU SHOULD NOT MODIFY THIS FILE.
#
# It will probably be overwritten or improved in a distribution update.
#
# Provide customizations in a jail.local file or a jail.d/customisation.local.
# For example to change the default bantime for all jails and to enable the
# ssh-iptables jail the following (uncommented) would appear in the .local file.
# See man 5 jail.conf for details.
#
# [DEFAULT]
```

As you'll see, the first several lines of this file are **commented out** – they begin with `#` characters indicating that they are to be read as documentation rather than as settings. As you'll also see, these comments are directing you not to modify this file directly. Instead, you have two options: either create individual profiles for Fail2ban in multiple files within the `jail.d/` directory, or create and collect all of your local settings in a `jail.local` file. The `jail.conf` file will be periodically updated as Fail2ban itself is updated, and will be used as a source of default settings for which you have not created any overrides.

In this tutorial, you'll create `jail.local`. You can do that by copying `jail.conf`:

```
$ sudo cp jail.conf jail.local
```

Copy

Now you can begin making configuration changes. Open the file in `nano` or your favorite text editor:

```
$ sudo nano jail.local
```

Copy

Changing Defaults

You'll start by evaluating the defaults set within the file. These will be found under the `[DEFAULT]` section within the file. These items set the general policy and can be overridden on a per-application basis. If you are using `nano`, you can search within the file by pressing `Ctrl+W`, entering a search string, then pressing enter.

One of the first items to look at is the list of clients that are not subject to the `fail2ban` policies. This is set by the `ignoreip` directive. It is sometimes a good idea to add your own IP address or network to the list of exceptions to avoid locking yourself out. This is less of an issue with web server logins than SSH, since if you are able to maintain shell access you can always reverse a ban. You can uncomment this line and add additional IP addresses or networks delimited by a space, to the existing list:

/etc/fail2ban/jail.local

```
[DEFAULT]

. . .
#ignoreip = 127.0.0.1/8 your_home_IP
```

Another item that you may want to adjust is the `bantime`, which controls how many seconds an offending member is banned for. It is ideal to set this to a long enough time to be disruptive to malicious, automated efforts, while short enough to allow users to correct mistakes. By default, this is set to 10 minutes. You can increase or decrease this value:

/etc/fail2ban/jail.local

```
[DEFAULT]

. . .
bantime = 10m
```

The next two items determine the scope of log lines used to determine an offending client. The `findtime` specifies an amount of time in seconds and the `maxretry` directive indicates the number of attempts to be tolerated within that time. If a client makes more than `maxretry` attempts within the amount of time set by `findtime`, they will be banned:

/etc/fail2ban/jail.local

```
[DEFAULT]

. . .
findtime = 10m
maxretry = 5
```

Setting Up Mail Notifications (Optional)

You can optionally enable email notifications to receive mail whenever a ban takes place. You will have to first set up an MTA on your server so that it can send out email. To learn how to use Postfix for this task, follow How to Install and Configure Postfix on ubuntu 22.04.

Once you have your MTA set up, you will have to adjust some additional settings within the `[DEFAULT]` section of the `/etc/fail2ban/jail.local` file. Start by setting the `mta` directive. If you set up Postfix, like the above tutorial demonstrates, change this value to “mail”:

/etc/fail2ban/jail.local

```
[DEFAULT]

. . .
mta = mail
```

Provide the email address that will receive mail in the `destemail` field. The `sender` option configures the address the mail will be sent from, and needs to be compatible with your Postfix configuration:

/etc/fail2ban/jail.local

```
[DEFAULT]

. . .
destemail = youraccount@email.com
sendername = root@<fq-hostname>
```

The `action` parameter configures the action that Fail2ban takes when it wants to institute a ban. The value `action_` is defined in the file shortly before this parameter. The default action is to update your firewall configuration to reject traffic from the offending host until the ban time elapses.

/etc/fail2ban/jail.local

```
[DEFAULT]
. . .
action = $(action_)s
. . .
```

There are other `action_` scripts provided by default which you can replace `$(action_)` with above:

/etc/fail2ban/jail.local

```
…
# ban & send an e-mail with whois report to the destemail.
action_mw = %(action_)s
            %(mta)s-whois[sender="%(sender)s", dest="%(destemail)s", protocol="%(protocol)s",

# ban & send an e-mail with whois report and relevant log lines
# to the destemail.
action_mwl = %(action_)s
```

```
%(mta)s-whois-lines[sender="%(sender)s", dest="%(destemail)s", logpath="%(logpath
…
```

Both `action_mw` and `action_mwl` will handle sending email using the configuration you provided. In the next step, you'll move on to Nginx-specific configuration.

Step 2 – Configuring Fail2Ban to Monitor Nginx Logs

Now that you have some of the general `fail2ban` settings in place, you can enable some Nginx-specific jails that will monitor your web server logs for specific patterns.

Cada jaula dentro do arquivo de configuração é marcada por um cabeçalho contendo o nome da jaula entre colchetes – cada seção, exceto a `[DEFAULT]` seção, indica a configuração de uma jaula específica. Por padrão, apenas a `[ssh]` jail está habilitada.

Para habilitar o monitoramento de log para tentativas de login do Nginx, habilite o `[nginx-http-auth]` jail. Adicione uma `enabled = true` diretiva nesta seção:

/etc/fail2ban/jail.local

```
…
[nginx-http-auth]

enabled  = true
port     = http,https
logpath = %(nginx_error_log)s
. . .
```

Quando terminar de fazer as modificações, salve e feche o arquivo. Se estiver usando `nano`, pressione `Ctrl+X`, quando solicitado `Y` e, em seguida, Enter. Em seguida, você revisará a configuração do filtro para `nginx-http-auth`.

Passo 3 – Revisando Filtros para Nginx Jails

Você deve ter notado que o `[nginx-http-auth]` bloqueio `jail.local` não contém nenhuma regra específica do Nginx. Essas regras não são codificadas automaticamente dentro do Fail2ban – na verdade, o `[nginx-http-auth]` cabeçalho corresponde diretamente a um nome de arquivo dentro do `filter.d` diretório de filtros pré-empacotados do Fail2ban. Se você listar o conteúdo desse diretório, poderá ver os outros filtros pré-empacotados disponíveis, caso precise usá-los:

```
$ ls /etc/fail2ban/filter.d
```

cópia de

```
Output
3proxy.conf                freeswitch.conf         proftpd.conf
apache-auth.conf           froxlor-auth.conf       pure-ftpd.conf
apache-badbots.conf        gitlab.conf             qmail.conf
apache-botsearch.conf      grafana.conf            recidive.conf
apache-common.conf         groupoffice.conf        roundcube-auth.conf
apache-fakegooglebot.conf  gssftpd.conf            scanlogd.conf
apache-modsecurity.conf    guacamole.conf          screensharingd.conf
apache-nohome.conf         haproxy-http-auth.conf  selinux-common.conf
apache-noscript.conf       horde.conf              selinux-ssh.conf
apache-overflows.conf      ignorecommands          sendmail-auth.conf
apache-pass.conf           kerio.conf              sendmail-reject.conf
apache-shellshock.conf     lighttpd-auth.conf      sieve.conf
assp.conf                  mongodb-auth.conf       slapd.conf
asterisk.conf              monit.conf              softethervpn.conf
bitwarden.conf             murmur.conf             sogo-auth.conf
…
```

Por enquanto, dê uma olhada em `nginx-http-auth.conf`:

```
$ cat /etc/fail2ban/filter.d/nginx-http-auth.conf
```

cópia de

```
Output
# fail2ban filter configuration for nginx
```

```
[Definition]

failregex = ^ \[error\] \d+#\d+: \*\d+ user "(?:[^"]+|.*?)":? (?:password mismatch|was not fou

ignoreregex =

datepattern = {^LN-BEG}
…
```

Esses arquivos contêm expressões regulares (uma abreviação comum para análise de texto) que determinam se uma linha no log é uma tentativa de autenticação com falha. Eles podem ser modificados diretamente conforme necessário.

Nas próximas etapas, você habilitará e testará o Fail2ban.

Passo 4 – Ativando suas Jails Nginx

Neste ponto, você pode habilitar seu serviço Fail2ban para que ele seja executado automaticamente a partir de agora. Primeiro, execute `systemctl enable`:

```
$ sudo systemctl enable fail2ban
```

cópia de

Em seguida, inicie-o manualmente pela primeira vez com `systemctl start`:

```
$ sudo systemctl start fail2ban
```

cópia de

Você pode verificar se ele está sendo executado com `systemctl status`:

```
$ sudo systemctl status fail2ban
```

cópia de

```
Output
● fail2ban.service - Fail2Ban Service
     Loaded: loaded (/lib/systemd/system/fail2ban.service; enabled; vendor preset: enabled
     Active: active (running) since Fri 2022-07-08 17:19:38 UTC; 7s ago
       Docs: man:fail2ban(1)
   Main PID: 5962 (fail2ban-server)
      Tasks: 7 (limit: 2327)
     Memory: 12.6M
        CPU: 195ms
     CGroup: /system.slice/fail2ban.service
             └─5962 /usr/bin/python3 /usr/bin/fail2ban-server -xf start
```

Observação: para implementar quaisquer alterações de configuração futuras, você precisará reiniciar o `fail2ban` serviço. Você pode fazer isso usando `sudo systemctl restart fail2ban`

Obtendo informações sobre prisões habilitadas

Você pode ver todas as suas jails habilitadas usando o `fail2ban-client` comando:

```
$ sudo fail2ban-client status
```

cópia de

Você deve ver uma lista das jails habilitadas:

```
Output
Status
|- Number of jail:      2
`- Jail list:           nginx-http-auth, sshd
```

Se você quiser ver os detalhes das proibições impostas por qualquer prisão, use o `fail2ban-client` comando novamente:

```
$ sudo fail2ban-client status nginx-http-auth
```

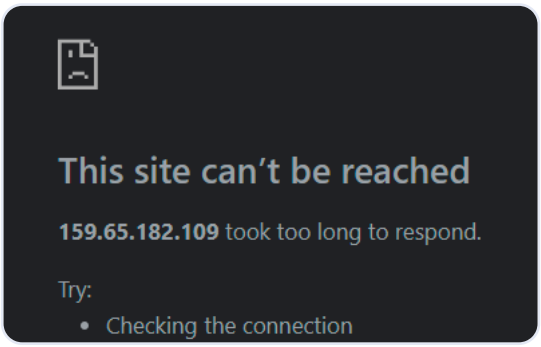
cópia de

```
Output
Status for the jail: nginx-http-auth
|- filter
|  |- File list:        /var/log/nginx/error.log
|  |- Currently failed: 0
|  `- Total failed:     0
`- action
   |- Currently banned: 0
   |  `- IP list:
   `- Total banned:     0
```

Na última etapa deste tutorial, você testará deliberadamente o banimento para verificar se a configuração do Fail2ban está funcionando.

Etapa 5 – Testando as políticas Fail2Ban

É importante testar suas políticas Fail2ban para garantir que elas bloqueiem o tráfego conforme o esperado. Para fazer isso, navegue até seu servidor em um navegador da Web local. No prompt de autenticação do Nginx, insira repetidamente credenciais incorretas. Após várias tentativas, o servidor deve parar de responder a você completamente, como se sua conexão estivesse inativa:



Se você observar o status da `nginx-http-auth` configuração com `fail2ban-client`, verá seu endereço IP sendo banido do site:

```
$ sudo fail2ban-client status nginx-http-auth
```

cópia de

```
Output
Status for the jail: nginx-http-auth
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     5
|  `- File list:        /var/log/nginx/error.log
`- Actions
   |- Currently banned: 1
   |- Total banned:     1
   `- Banned IP list:   108.172.85.62
```

Você também pode ver a nova regra verificando sua `iptables` saída. `iptables` é um comando para interagir com regras de porta e firewall de baixo nível em seu servidor. Se você seguiu o guia da DigitalOcean para a configuração inicial do servidor, você usará `ufw` para gerenciar as regras de firewall em um nível superior. A execução `iptables -S` mostrará todas as regras de firewall que `ufw` já foram criadas:

```
$ sudo iptables -S
```

cópia de

```
Output
-P INPUT DROP
-P FORWARD DROP
-P OUTPUT ACCEPT
-N f2b-nginx-http-auth
-N ufw-after-forward
-N ufw-after-input
-N ufw-after-logging-forward
-N ufw-after-logging-input
-N ufw-after-logging-output
-N ufw-after-output
```

```
-N ufw-before-forward
-N ufw-before-input
-N ufw-before-logging-forward
-N ufw-before-logging-input
-N ufw-before-logging-output
…
```

Se você canalizar a saída de `iptables -S` to `grep` para pesquisar dentro dessas regras pela string `f2b`, poderá ver as regras que foram adicionadas por fail2ban:

```
$ sudo iptables -S | grep f2b
```

cópia de

```
Output
-N f2b-nginx-http-auth
-A INPUT -p tcp -m multiport --dports 80,443 -j f2b-nginx-http-auth
-A f2b-nginx-http-auth -s 108.172.85.62/32 -j REJECT --reject-with icmp-port-unreachable
-A f2b-nginx-http-auth -j RETURN
```

A linha que contém `REJECT --reject-with icmp-port-unreachable` foi adicionada pelo Fail2ban e deve refletir o endereço IP da sua máquina local. Quando estiver convencido de que suas regras estão funcionando, você pode desbanir manualmente seu endereço IP `fail2ban-client` digitando:

```
$ sudo fail2ban-client set nginx-http-auth unbanip 108.172.85.62
```

cópia de

Agora você deve poder tentar a autenticação novamente.

Conclusão

O Fail2ban oferece muita flexibilidade para construir políticas que atendam às suas necessidades específicas de segurança. Dando uma olhada nas variáveis e padrões dentro do `/etc/fail2ban/jail.local` arquivo, e os arquivos dos quais ele depende dentro dos diretórios `/etc/fail2ban/filter.d` e `/etc/fail2ban/action.d`, você pode encontrar muitas partes para ajustar e mudar conforme suas necessidades evoluem. Proteger seu servidor `fail2ban` pode fornecer uma linha de base de segurança útil.

Para descobrir mais maneiras de usar `fail2ban`, confira Como o Fail2Ban funciona para proteger os serviços em um servidor Linux e Como proteger o SSH com o Fail2Ban no Ubuntu 20.04.

Sobre os autores

Alex Garnett Autor

Escritor Técnico de DevOps Sênior