SeVen - Módulo Apache

Manual de instalação, configuração e testes

RNP – Rede Nacional de Ensino e Pesquisa
GT – Actions
2015

Índice

- Seções

- Anexos

I - Instalação e configuração inicial do servidor Apache, versão 2.4.7

1. Instalando o Apache na máquina host

A instalação varia conforme o sistema operacional. Para uma variedade de opções e espelhos, visite a página de *download* no site oficial da Apache Foundation, ou procure a documentação de sua distribuição GNU/Linux ou BSD para saber o procedimento (o pacote comumente é nomeado *apache2* ou *httpd*). Os testes desse manual foram utilizando a distribuição Debian. Todos os comandos (tanto de instalação, como de configuração) foram utilizando o usuário 'root', para outros usuários utilize o 'sudo' na frente dos comandos.

1.1. O Apache possui alguns componentes chamados *Multi-Processing Module* (**MPM**). Os **MPM** gerenciam o tratamento das requisições pelos servidores virtuais, cada um se valendo de diferentes abordagens, tornando conveniente ou inconveniente a escolha de um deles, a depender do cenário. São 3: *mpm_event*, *mpm_worker* e *mpm_prefork*. Numa instalação a partir dos repositórios do Ubuntu, por exemplo, o *mpm_prefork* será o ativo por padrão. O *mpm_event* possui uma configuração mais complexa, aonde as diretivas são manipuladas através de objetos de baixo nível (*threads*), portanto é propenso a oferecer melhor performance. No entanto, para facilitar a configuração, iremos ativar o *mpm_prefork*.

2. Habilitando o *mpm_prefork*

2.1. Após a instalação do Apache, cheque qual MPM o ativo:

```
# apachectl -V | grep -i mpm
```

Caso o retorno já seja *mpm_prefork*, não há o que configurar.

2.2. Caso estejam ativados *mpm_worker* ou *mpm_event*, desabilite-os:

```
# a2dismod mpm_event
```

ou

```
# a2dismod mpm_worker
```

2.3. Habilite o *mpm_prefork*:

```
# a2enmod mpm_prefork
```

2.4. Reinicie o Apache:

```
# service apache2 restart
```

2.5. Cheque se mpm_prefork foi ativado corretamente:

```
# apachectl -V | grep -i mpm
```

3. Configurando o Apache

Na maioria dos casos, não se faz necessário alterar configuração alguma do Apache, pois o mesmo já vem adequado a um servidor de médio porte. Contudo, isto pode ser conveniente para realização de testes com as configurações do servidor. Neste cenário, o arquivo de configuração *apache2.conf*, situado geralmente no diretório /etc/apache2/ deve ser modificado para se adequar ao cenário dos testes, bem como o arquivo *mpm_prefork.conf* no diretório /etc/apache2/mods-available. Tal procedimento será descrito na seção de testes do presente texto.

4. Instalando o conjunto de ferramentas de desenvolvedor do Apache

Para compilar e consequentemente instalar o módulo Apache do SeVen, será necessário a instalação do conjunto de ferramentas de desenvolvedor do Apache. Então, será necessário a instalação do pacote chamado *apache2-dev*. Para isso:

```
# apt-get install apache2-dev
```

Após a instalação do pacote, estaremos aptos a compilar e instalar o módulo Apache do SeVen.

II – Instalação e execução do Módulo Apache - SeVen

1. Instalando o Módulo Apache - SeVen

A compilação e instalação do módulo é realizada de uma forma bastante simples, rodando um comando para referenciar a ferramenta específica de instalação e compilação de módulos (*apxs2*).

1.1. No diretório onde encontra-se o código fonte do módulo Apache (mod_seven.c), faça:

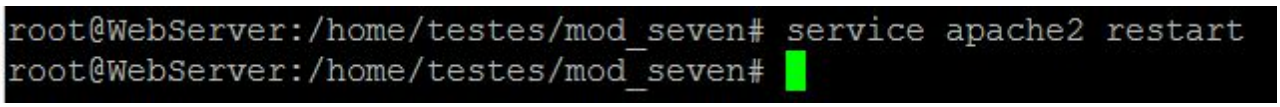
```
# apxs2 -i -c -a mod_seven.c
```

```
root@WebServer:/home/testes/mod_seven# apxs2 -c -i -a mod_seven.c
/usr/share/apr-1.0/build/libtool --silent --mode=compile --tag=disable-static x8
6_64-linux-gnu-gcc -std=gnu99 -prefer-pic -pipe -g -O2 -fstack-protector-strong
-Wformat -Werror=format-security  -D_FORTIFY_SOURCE=2   -DLINUX -D_REENTRANT -D_
GNU_SOURCE  -pthread  -I/usr/include/apache2  -I/usr/include/apr-1.0   -I/usr/in
clude/apr-1.0 -I/usr/include  -c -o mod_seven.lo mod_seven.c && touch mod_seven.
slo
/usr/share/apr-1.0/build/libtool --silent --mode=link --tag=disable-static x86_6
4-linux-gnu-gcc -std=gnu99 -Wl,--as-needed -Wl,-z,relro -Wl,-z,now    -o mod_sev
en.la  -rpath /usr/lib/apache2/modules -module -avoid-version    mod_seven.lo
/usr/share/apache2/build/instdso.sh SH_LIBTOOL='/usr/share/apr-1.0/build/libtool
' mod_seven.la /usr/lib/apache2/modules
/usr/share/apr-1.0/build/libtool --mode=install install mod_seven.la /usr/lib/ap
ache2/modules/
libtool: install: install .libs/mod_seven.so /usr/lib/apache2/modules/mod_seven.
so
libtool: install: install .libs/mod_seven.lai /usr/lib/apache2/modules/mod_seven
.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/
bin:/sbin" ldconfig -n /usr/lib/apache2/modules
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/lib/apache2/modules

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
chmod 644 /usr/lib/apache2/modules/mod_seven.so
[preparing module `seven' in /etc/apache2/mods-available/seven.load]
Enabling module seven.
To activate the new configuration, you need to run:
  service apache2 restart
root@WebServer:/home/testes/mod_seven#
```

Compilação e instalação do Módulo Apache - SeVen (mod_seven.c)

1.2. Como pode ser visto pela imagem acima, precisamos reiniciar o serviço do Apache a fim de efetuar a instalação do módulo. Observe que no diretório onde se encontra o código fonte (mod_seven.c), o Apache irá criar automaticamente três novos arquivos (.slo, .la e .lo) necessários para a execução e funcionamento correto do módulo e do Apache, não exclua-os. Para reiniciar o serviço do Apache, faça:

service apache2 restart



Reiniciando o Servidor Apache

É bastante recomendado, verificando o log dos comandos utilizados, que, tanto como a compilação e instalação do módulo e reinicialização do servidor Apache, foram realizadas com sucesso.

III – Preparação para testes - Siege e slowloris

1. Instalando o a ferramenta de ataque *slowloris*

A instalação do *slowloris* é feita por meio da cópia do arquivo *slowloris.pl* para o diretório desejado da máquina atacante. Para isso vamos novamente utilizar o *scp*:

scp slowloris.pl usuario@ip:/"local onde vai ser copiado"

Onde usuario@ip:/"local onde vai ser copiado" é o caminho e o local onde você deseja salvar o *slowloris* na máquina atacante.

2. Instalando a ferramenta de simulação de clientes Siege

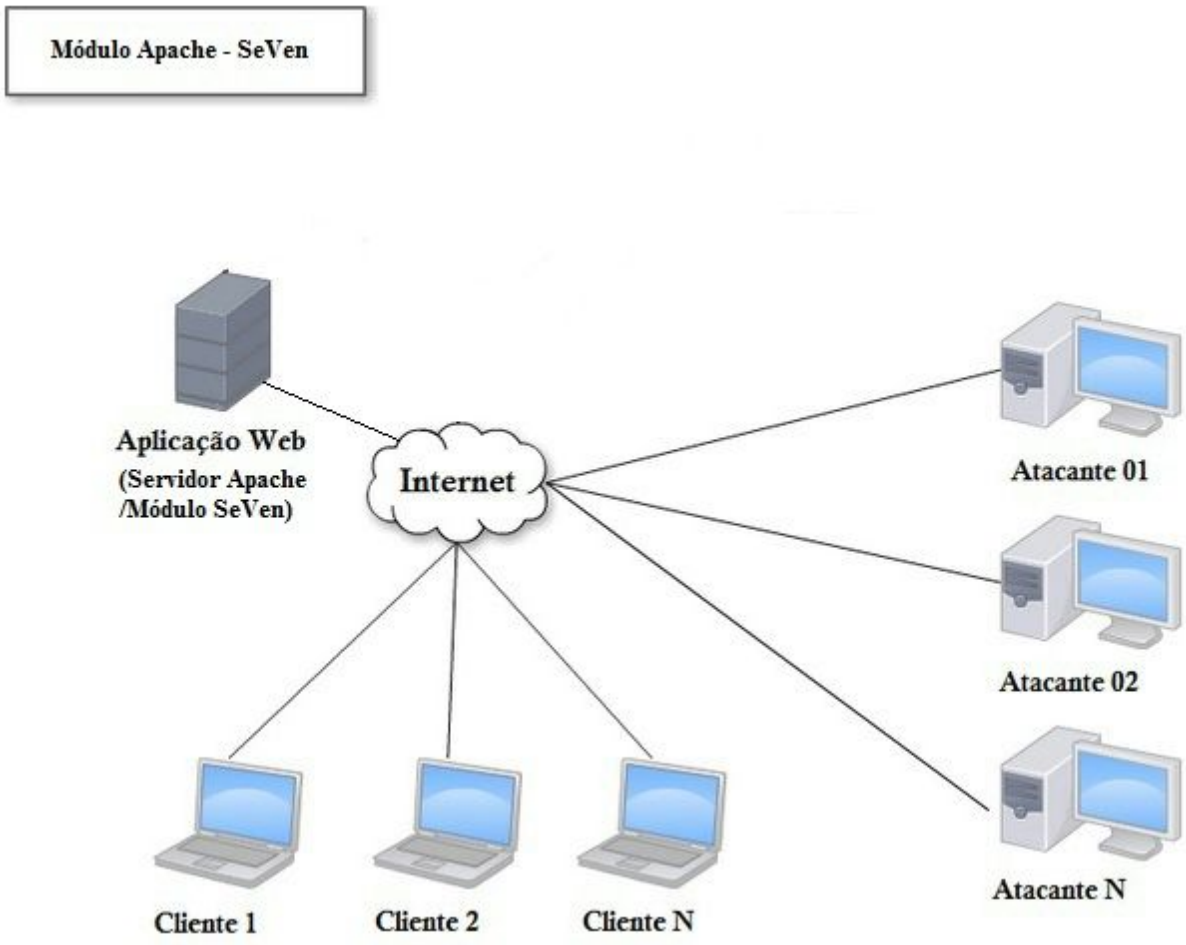
A instalação do S*iege* pode ser feita por meio dos repositórios oficiais, da mesma forma que ocorreu com o Apache. O comando abaixo pode ser utilizado (testado no Ubuntu) para instalar o S*iege* na máquina cliente:

apt-get install siege

3. Cenário padrão para testes

O módulo *SeVen*, quando ativado, é acoplado ao servidor Apache da aplicação Web, e será executado pelo *core* do Apache toda vez que uma requisição chegar no servidor.

Um cenário similar de testes encontra-se na imagem abaixo.



Cenário de testes do módulo Apache - SeVen

IV – Execução dos testes com o SeVen ativo

Para uma correta execução do procedimento de testes, é estritamente recomendado que os passos abaixo sejam seguidos conforme a ordem dos mesmos! O desrespeito dessa recomendação pode gerar resultados inconsistentes pelos quais este manual não se responsabilizará.

Primeiramente, temos que certificar que o módulo Apache do SeVen encontra-se ativado no servidor Apache. Para isso:

apachectl -M

```
root@WebServer:/home/testes/mod_seven# apachectl -M
AH00558: apache2: Could not reliably determine the server's fully qualified doma
in name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress th
is message
Loaded Modules:
 core_module (static)
 so_module (static)
 watchdog_module (static)
 http_module (static)
 log_config_module (static)
 logio_module (static)
 version_module (static)
 unixd_module (static)
 access_compat_module (shared)
 alias_module (shared)
 auth_basic_module (shared)
 authn_core_module (shared)
 authn_file_module (shared)
 authz_core_module (shared)
 authz_host_module (shared)
 authz_user_module (shared)
 autoindex_module (shared)
 deflate_module (shared)
 dir_module (shared)
 env_module (shared)
 filter_module (shared)
 mime_module (shared)
 mpm_prefork_module (shared)
 negotiation_module (shared)
 setenvif_module (shared)
 seven_module (shared)
 status_module (shared)
root@WebServer:/home/testes/mod_seven#
```

Lista de módulos ativos no servidor Apache

Caso você já tenha compilado e instalado o SeVen (estamos subentendendo que você já o fez), e não encontrou o mod_seven na lista, é porque ele foi desabilitado anteriormente, para habilitá-lo no servidor, execute:

a2enmod seven

```
root@WebServer:/home/testes/mod_seven# a2enmod seven
Enabling module seven.
To activate the new configuration, you need to run:
  service apache2 restart
root@WebServer:/home/testes/mod_seven# service apache2 restart
root@WebServer:/home/testes/mod_seven#
```

Habilitando o Módulo Apache - SeVen (mod_seven.c)

Após a confirmação que o módulo está habilitado do servidor, podemos iniciar a execução dos testes.

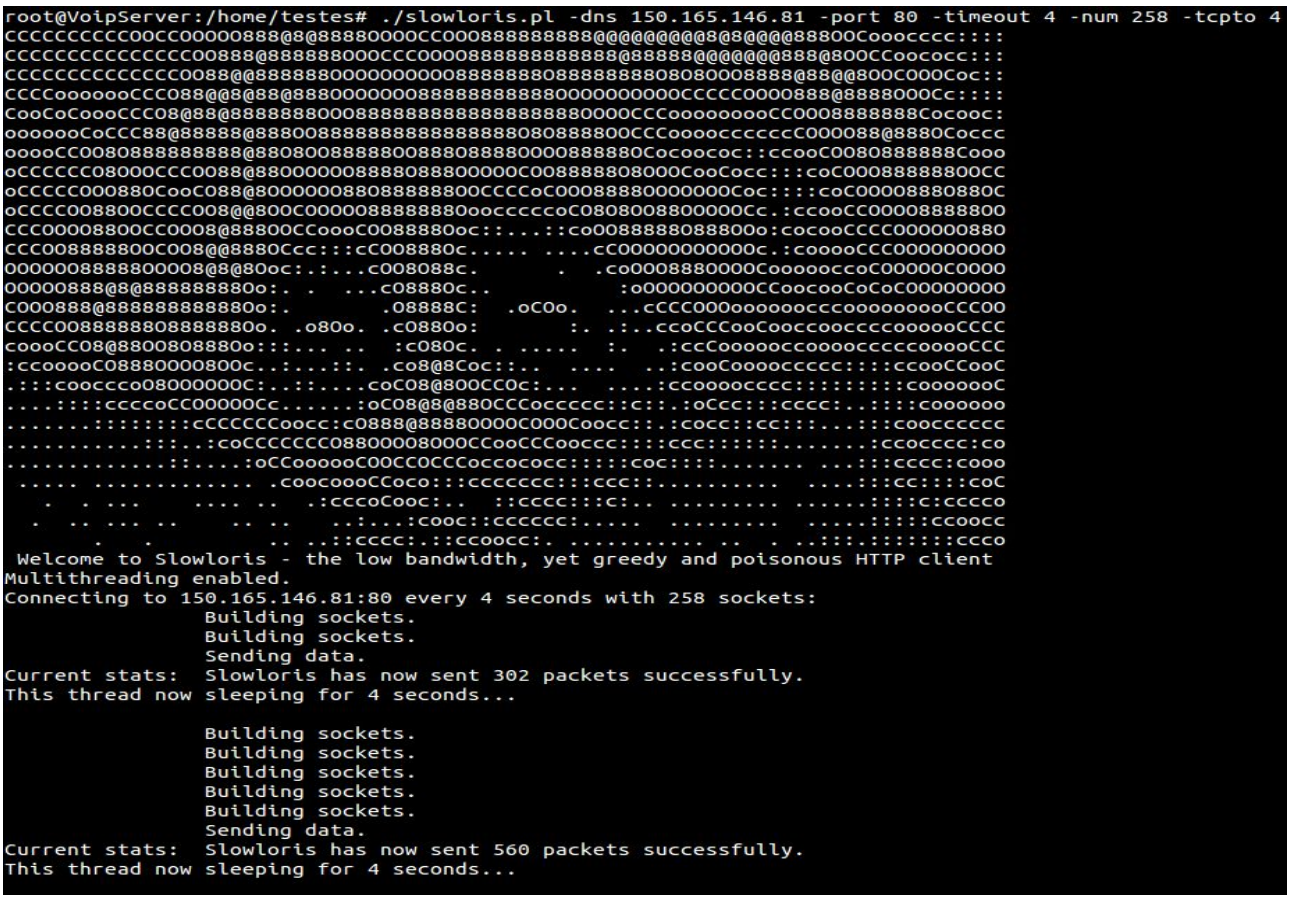
1. **Executa-se o slowloris na máquina atacante**

Para a execução do *slowloris*, acesse a máquina atacante (localmente ou via *ssh*), navegue até a pasta onde o *slowloris* foi salvo e execute o seguinte comando:

./slowloris.pl -dns [ip-a-atacar] -port [porta-a-atacar] -timeout [tempo-de-timeout] -num [numero-de-atacantes] -tcpto 4

Onde ip-a-atacar é o IP da máquina que está rodando o *SeVen* e que sofrerá o ataque, porta-a-atacar é a porta onde o *SeVen* está escutando, tempo-de-timeout é o timeout entre as requisições enviadas pelo *slowloris* e numero-de-atacantes é o número de atacantes que se deseja criar.

Abaixo mostraremos um exemplo do que ocorre ao se executar o *slowloris*, novamente vale frisar a importância da execução destes passos na ordem apresentada por este manual, evitando assim possíveis resultados inconsistentes.



Exemplo de execução do *slowloris*

2. **Executa-se o Siege na máquina cliente e coleta-se os resultados**

Atenção, para uma melhor execução dos testes, é recomendado esperar 10 segundos antes da execução do *Siege*. Quando executados (*slowloris* e Siege) quase ao mesmo tempo (*Siege* após logo após o *slowloris*), o Apache pode atender requisições do *Siege* antes de estar completamente ocupado pelas requisições do *slowloris*, assim, "maquiando" alguns resultados do *Siege*.

Para executar o *Siege*, acesse a máquina cliente (localmente ou via *ssh*) e execute o seguinte comando:

siege [ip] -c[número de clientes] -t[tempo do teste]s -d[delay das requisições]

Onde [ip] é o IP da máquina de defesa, [número de clientes] é o número desejado de clientes simulados pelo *Siege* e [tempo do teste] é a duração em segundos do teste.

```
root@VoIPClient2:/home/testes# siege -d3 -c50 -t5M 150.165.146.131
** SIEGE 3.0.8
** Preparing 50 concurrent users for battle.
The server is now under siege...
Lifting the server siege...      done.

Transactions:                   9764 hits
Availability:                 100.00 %
Elapsed time:                 299.08 secs
Data transferred:              28.66 MB
Response time:                  0.01 secs
Transaction rate:              32.65 trans/sec
Throughput:                     0.10 MB/sec
Concurrency:                    0.36
Successful transactions:        9764
Failed transactions:               0
Longest transaction:            3.01
Shortest transaction:           0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@VoIPClient2:/home/testes#
```

Exemplo de execução do *Siege*

3. **Paramos o *slowloris* quando o *Siege* para**

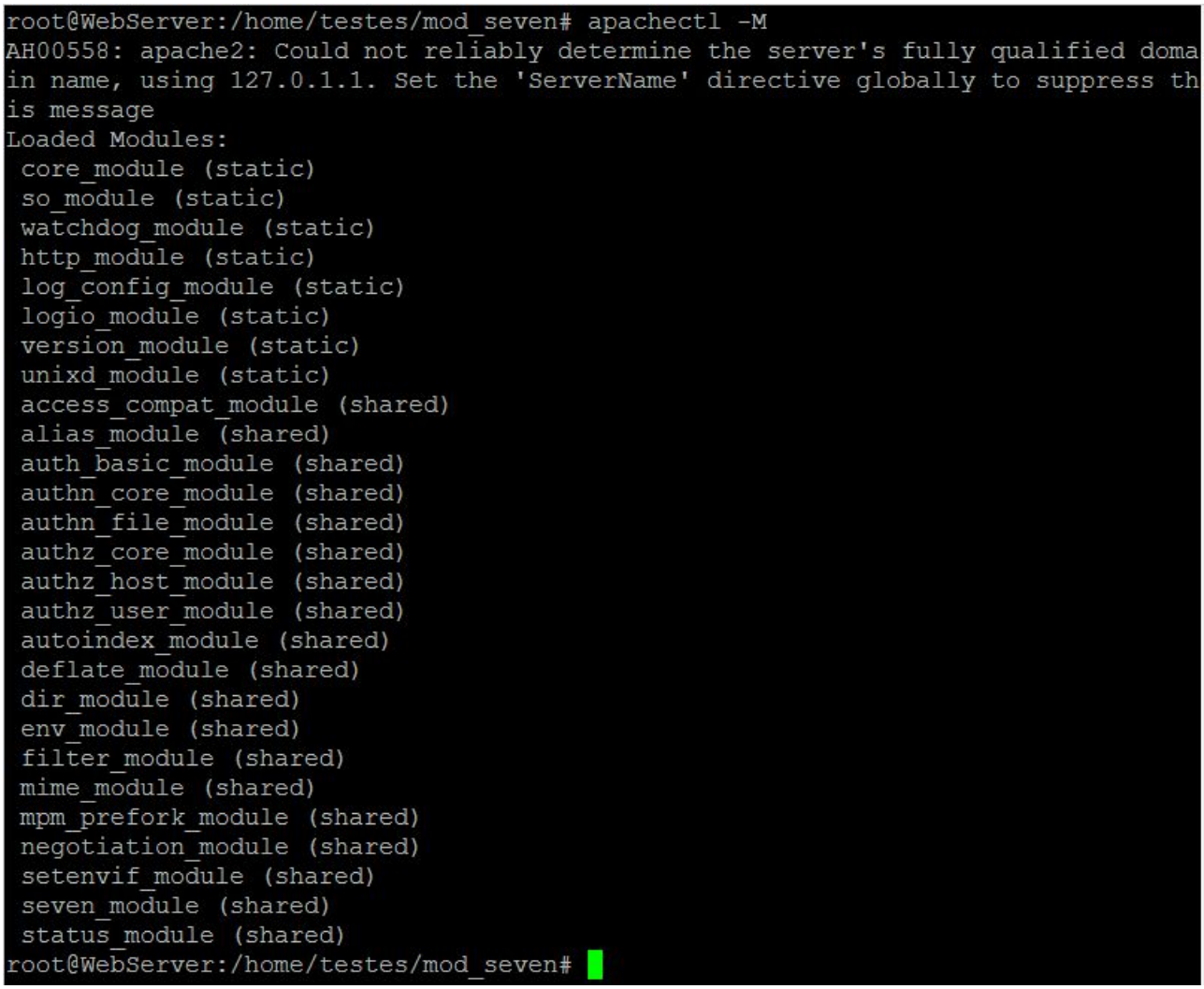
Para tal, basta apertar CTRL+C ou CTRL+4 (distribuições Linux) na máquina atacante.

V – Execução dos testes com o Módulo Apache - SeVen inativo (sem SeVen)

Para uma correta execução do procedimento de testes, é estritamente recomendado que os passos abaixo sejam seguidos conforme a ordem dos mesmos! O desrespeito dessa recomendação pode gerar resultados inconsistentes pelos quais este manual não se responsabilizará.

Primeiramente, temos que certificar que o módulo Apache do SeVen não encontra-se ativado no servidor Apache. Para isso:

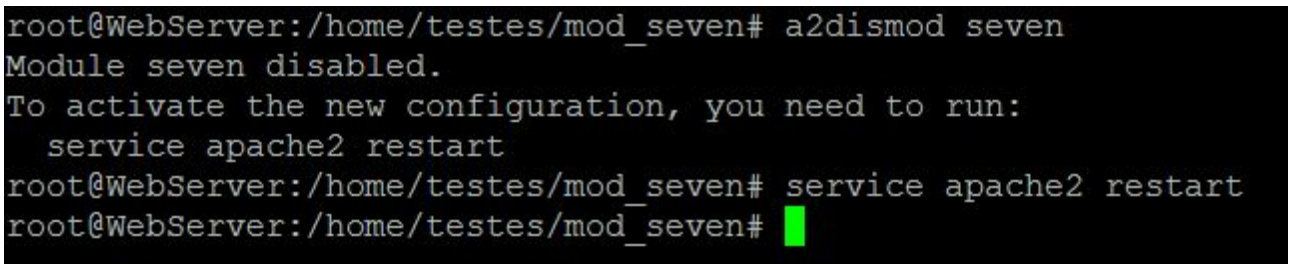
```
# apachectl -M
```



Lista de módulos ativos no servidor Apache

Irá ser gerada uma lista contendo todos os módulos instalados e rodando no servidor atualmente. Verifique se o SeVen não está presente nesta lista. Se estiver, precisaremos desabilitá-lo:

```
# a2dismod seven
```



Desabilitando Módulo Apache - SeVen (mod_seven.c)

Após a confirmação que o módulo está desabilitado do servidor, podemos iniciar a execução dos testes.

1. **Executa-se o *slowloris* na máquina atacante**

Para a execução do *slowloris*, acesse a máquina atacante (localmente ou via *ssh*), navegue até a pasta onde o *slowloris* foi salvo e execute o seguinte comando:

./slowloris.pl -dns [ip-a-atacar] -port [porta-a-atacar] -timeout [tempo-de-timeout] -num [numero-de-atacantes] -tcpto 4

Onde ip-a-atacar é o IP da máquina que está rodando o Apache e que sofrerá o ataque, porta-a-atacar é a porta onde o Apache está escutando, tempo-de-timeout é o timeout entre as requisições enviadas pelo *slowloris* e numero-de-atacantes é o número de atacantes que se deseja criar.

Abaixo mostraremos um exemplo do que ocorre ao se executar o *slowloris*, novamente vale frisar a importância da execução destes passos na ordem apresentada por este manual, evitando assim possíveis resultados inconsistentes.

```
root@VoipServer:/home/testes# ./slowloris.pl -dns 150.165.146.81 -port 80 -timeout 4 -num 258 -tcpto 4
CCCCCCCCCCOOCCOOOOO888@8@8888OOOOCCOOO88888888888@@@@@@@@@@8@8@@@@88800Cooocccc::::
CCCCCCCCCCCCCCCCOO888@888888OOOCCCOOOO88888888888888@88888@@@@@@@888@80OCCoococc:::
CCCCCCCCCCCCCCCOO88@@88888800000000008888888088888888080800008888@88@@@8OOCOOOCoc::
CCCCooooooCCCO88@@8@88@88800000008888888888800000000CCCCCOOOO888@8888OOOCc::::
CooCoCoooCCCO8@88@88888880008888888888888888880000CCCooooooooCCOOO8888888Cocooc:
ooooooCoCCC88@88888@888008888888888888888080888800CCCooooccccccCOOOO88@8880Coccc
ooooCCOO8O888888888@88808008888800888O888880000888880Cocoococ::ccooCO0808888888Cooo
oCCCCCCO8OOOCCCOO88@88000000888808880800000CO088888808000CooCocc:::coCOOO8888880OCC
oCCCCCOOO880CooCO88@80000008808888888OCCCCoCOOO888880000000Coc::::coCOOOO8880880C
oCCCCOO8800CCCCOO8@@8OOCOOOOO88888888OoocccccoCO8O8OO88000000Cc.:ccooCCOOOO8888800
CCCOOOO8800CCOOO8@88800CCoooCO088880oc::...::coO0888880888OOo:cocooCCCCOOOOO8880
CCCO088888800CO08@@8880Ccc:::cCO08880c..... ....cCOOOOOOOOOOc.:cooooCCCOOOOOOOO
OOOOOO88888800008@8@8Ooc:.:...cO08O88c.      .  .coOOO8880000CooooocooCOOOOCOOOO
OOOOO888@8@888888880o:. .  ...cO8880c..          :oOOOOOOOOCCoocooCoCoCOOOOOOOO
COOO888@88888888880o:.       .O8888C:  .oCOo.  ...cCCCOOOooooooccooooooooCCCOO
CCCCO088888808888880o. .o8Oo. .cO880o:       :. .:..ccoCCCooCooccooccccooooCCCC
cooocCO8@88008088800:::... ..  :cO8Oc. . .....  :.  .:ccCooooocooooocccccooooCCC
:ccooooCO8880000800c..:...::. .co8@8Coc::..  ....  ..:cooCooooccccc::::ccooCCooC
.:::coocccoO8OOOOOOC:..::....coCO8@8OOCCOc:...  ....:ccooooccccc:::::::::coooooC
....::::ccccoCCOOOOOCc......:oCO8@8@88OCCCocccccc::c::.:oCcc:::cccc:..::::coooooo
.......::::::::cCCCCCCCoocc:cO888@88880000CO00Coocc::.:cocc::cc:::...:::coocccccc
...........:::..:coCCCCCCCO88000008000CCooCCCooccc::::ccc:::::::.......:ccooccc:co
.............::....:oCCooooooCOOCCOCCCoccocococ:::::coc:::::..... ...:::cccc:cooo
 ..... ............. .coocoooCCoco:::ccccccc:::ccc::.........  ....:::cc::::coC
   .  . ...    .... ..  .:cccoCooc:..  ::cccc:::c:.. ......... .....::::c:ccccc
  .  .. ... ..    .. ..   ..:...:cooc::cccccc:.....  .........  .....:::::ccoocc
      .   .       .. ..::cccc:.::ccoocc:. ........... ..  . ..:::.:::::::ccco
 Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client
Multithreading enabled.
Connecting to 150.165.146.81:80 every 4 seconds with 258 sockets:
               Building sockets.
               Building sockets.
               Sending data.
Current stats:  Slowloris has now sent 302 packets successfully.
This thread now sleeping for 4 seconds...

               Building sockets.
               Building sockets.
               Building sockets.
               Building sockets.
               Building sockets.
               Sending data.
Current stats:  Slowloris has now sent 560 packets successfully.
This thread now sleeping for 4 seconds...
```

Exemplo de execução do *slowloris*

2. **Executa-se o *Siege* na máquina cliente e coleta-se os resultados**

Atenção, para uma melhor execução dos testes, é recomendado esperar 10 segundos antes da execução do *Siege*. Quando executados (*slowloris* e Siege) quase ao mesmo tempo (*Siege* após logo após o *slowloris*), o Apache pode atender requisições do *Siege* antes de estar completamente ocupado pelas requisições do *slowloris*, assim, "maquiando" alguns resultados do *Siege*.

Para executar o *Siege*, acesse a máquina cliente (localmente ou via *ssh*) e execute o seguinte comando:

siege [ip] -c[número de clientes] -t[tempo do teste]s -d[delay das requisições]

Onde [ip] é o IP da máquina de defesa, [número de clientes] é o número desejado de clientes simulados pelo *Siege* e [tempo do teste] é a duração em segundos do teste.

```
Lifting the server siege...      done.

Transactions:                       0 hits
Availability:                    0.00 %
Elapsed time:                  299.70 secs
Data transferred:                0.00 MB
Response time:                   0.00 secs
Transaction rate:                0.00 trans/sec
Throughput:                      0.00 MB/sec
Concurrency:                     0.00
Successful transactions:            0
Failed transactions:              486
Longest transaction:             0.00
Shortest transaction:            0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@VoIPClient2:/home/testes#
```

Exemplo de execução do *Siege*

3. **Paramos o *slowloris* quando o *Siege* para**

1. Para tal, basta apertar CTRL+C ou CRTL+4 (distribuições Linux) na máquina atacante.

VI – Alguns exemplos de testes e seus resultados

1. Sem SeVen, sem ataque

Apenas o *Siege* mandando pacotes diretamente para a máquina hospedando o servidor *Apache.* Os parâmetros utilizados no *Siege* foram:

```
# siege -d3 -c50 -t5M [ip]
```

```
root@VoIPClient2:/home/testes# siege -d3 -c50 -t5M 150.165.146.131
** SIEGE 3.0.8
** Preparing 50 concurrent users for battle.
The server is now under siege...
Lifting the server siege...      done.

Transactions:                   9764 hits
Availability:                 100.00 %
Elapsed time:                 299.08 secs
Data transferred:              28.66 MB
Response time:                  0.01 secs
Transaction rate:              32.65 trans/sec
Throughput:                     0.10 MB/sec
Concurrency:                    0.36
Successful transactions:        9764
Failed transactions:               0
Longest transaction:            3.01
Shortest transaction:           0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@VoIPClient2:/home/testes#
```

Execução do teste Sem SeVen, sem ataque'

Observamos 100% de disponibilidade.

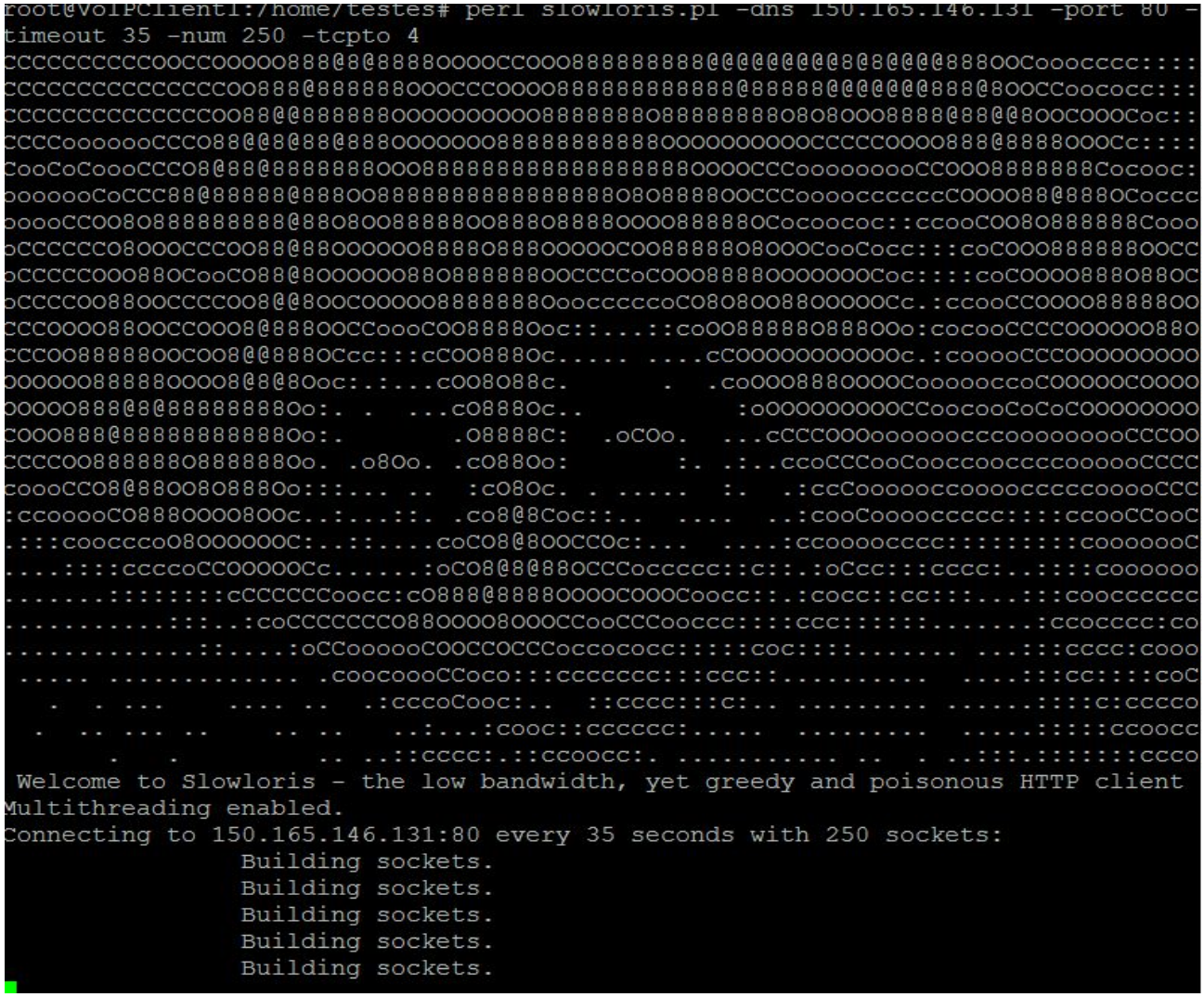
2. Sem SeVen, com ataque

O *Siege* mandando pacotes diretamente para a máquina hospedando o servidor *Apache.* Os parâmetros utilizados no *Siege* foram:

```
# siege -d3 -c50 -t5M [ip]
```

Junto ao *slowloris* atacando a máquina que hospeda o apache com os seguintes parâmetros

./slowloris.pl -dns [ip] -port 80 -timeout 35 -num 250 -tcpto 4



```
Lifting the server siege...      done.

Transactions:                       0 hits
Availability:                    0.00 %
Elapsed time:                  299.70 secs
Data transferred:                0.00 MB
Response time:                   0.00 secs
Transaction rate:                0.00 trans/sec
Throughput:                      0.00 MB/sec
Concurrency:                     0.00
Successful transactions:            0
Failed transactions:              486
Longest transaction:             0.00
Shortest transaction:            0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@VoIPClient2:/home/testes#
```

Observamos 0% de disponibilidade, devido ao *slowlois* negando o serviço.

3. **Com SeVen, sem ataque**

O *Siege* mandando pacotes diretamente para a máquina hospedando o *SeVen.* Os parâmetros utilizados no *Siege* foram:

siege -d3 -c50 -t5M [ip]

```
root@WebServer:/home/testes/mod_seven# a2enmod seven
Enabling module seven.
To activate the new configuration, you need to run:
  service apache2 restart
root@WebServer:/home/testes/mod_seven# service apache2 restart
root@WebServer:/home/testes/mod_seven#
```

```
root@VoIPClient2:/home/testes# siege -d3 -c50 -t5M 150.165.146.131
** SIEGE 3.0.8
** Preparing 50 concurrent users for battle.
The server is now under siege...
Lifting the server siege...      done.

Transactions:                   9764 hits
Availability:                 100.00 %
Elapsed time:                 299.08 secs
Data transferred:              28.66 MB
Response time:                  0.01 secs
Transaction rate:              32.65 trans/sec
Throughput:                     0.10 MB/sec
Concurrency:                    0.36
Successful transactions:        9764
Failed transactions:               0
Longest transaction:            3.01
Shortest transaction:           0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@VoIPClient2:/home/testes#
```

Observamos que o *SeVen* não interfere na disponibilidade sem ataque.

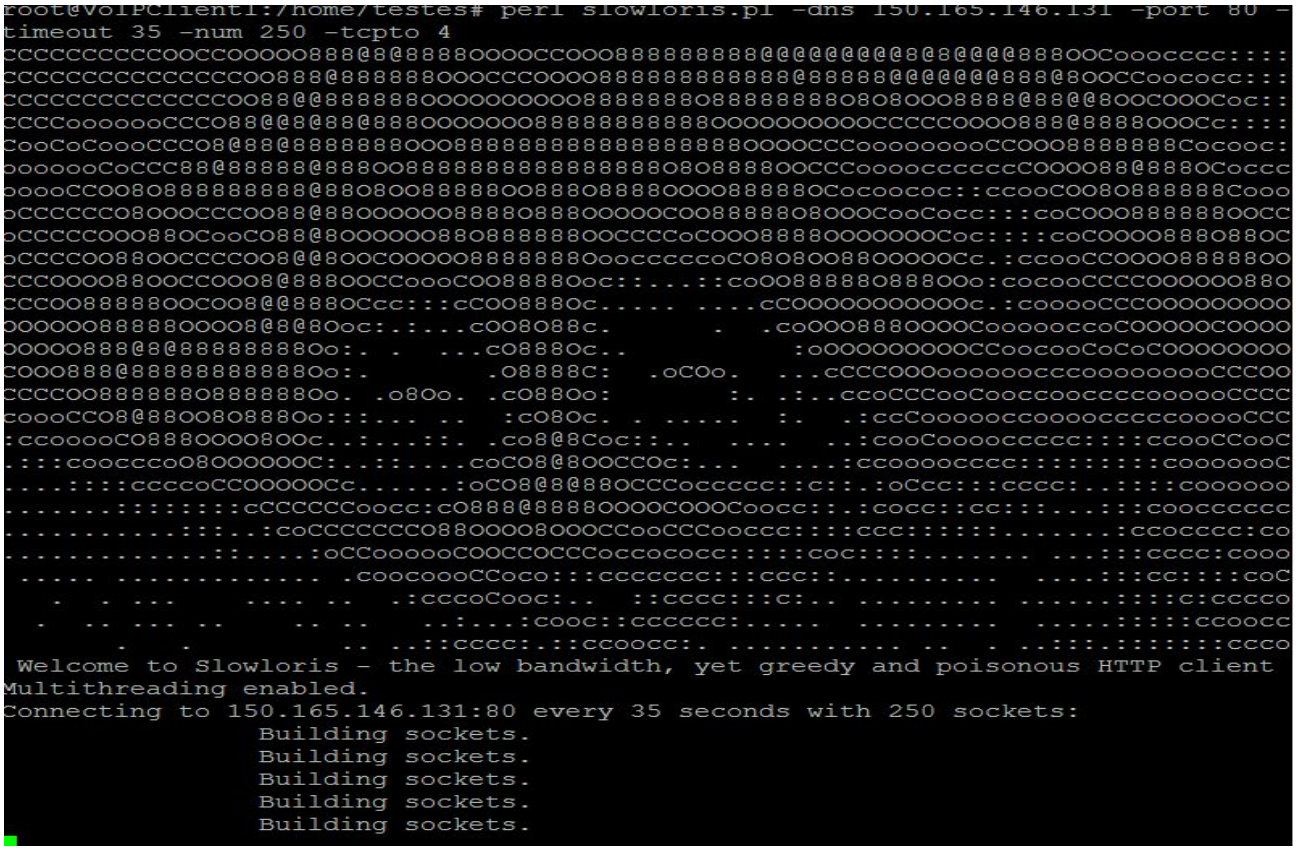
4. **Com SeVen, com ataque**

O *Siege* mandando pacotes diretamente para a máquina hospedando o servidor *Apache.* Os parâmetros utilizados no *Siege* foram:

siege -d3 -c50 -t5M [ip]

Junto ao *slowloris* atacando a máquina que hospeda o apache com os seguintes parâmetros

$ ./slowloris.pl -dns [ip] -port 80 -timeout 35 -num 250 -tcpto 4




```
Lifting the server siege...      done.

Transactions:                    8811 hits
Availability:                   96.98 %
Elapsed time:                  299.94 secs
Data transferred:               25.86 MB
Response time:                   0.13 secs
Transaction rate:               29.38 trans/sec
Throughput:                      0.09 MB/sec
Concurrency:                     3.76
Successful transactions:         8811
Failed transactions:              274
Longest transaction:            15.23
Shortest transaction:            0.00

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
root@VoIPClient2:/home/testes#
```

Observamos 96,98% de disponibilidade, mostrando a efetividade do *SeVen*

5. **Configuração utilizada no servidor Apache dos testes acima**

Todos os testes acima foram realizados fazendo o uso de um servidor Apache com a seguinte configuração no *mpm_prefork.conf*:

StartServers	8
MinSpareServers	5
MaxSpareServers	15
MaxRequestWorkers	200
MaxConnectionsPerChild	256
ServerLimit	456
ListenBackLog	1

As configurações do *apache2.conf* também tinham sido alteradas e eram as seguintes:

Timeout	40
KeepAlive	On
MaxKeepAliveRequests	100
KeepAliveTimeout	5

Todo o resto de ambos os dois arquivos de configuração foi deixado da forma *default*, sendo alteradas apenas as linhas citadas acima.

Anexo A – Informações sobre as diretivas de configuração do Apache 2

Este pequeno manual oferece sucintas abstrações relativas às configurações do Apache, especificadas em diretivas nos arquivos *apache2.conf* no diretório /etc/apache2/ e *mpm_prefork.con*f em /etc/apache2/mods-available.

Diretivas no arquivo *apache2.conf*

1. **Timeout**
   1.1. Oferta de tempo (em segundos) até uma possível desconexão com o cliente, considerando a ausência de envio de pacotes. Após o timeout, a conexão é perdida.
2. **KeepAlive**
   2.1. Continuar conectado entre requisições subsequentes;
   2.2. Para simulações mais precisas, configurar como off, uma vez que cada requisição - num cenário realístico - partirá de um endereço IP diferente e construirá novos *streams* a cada conexão.
3. **MaxKeepAliveRequest**
   3.1. Oferta de requisições a um cliente, antes deste ser desconectado, num cenário de conexão persistente.
4. **KeepAliveTimeout**

4.1. Oferta de tempo de espera (em segundos) entre requisições subsequentes, preservando a *thread* do cliente em questão.

5. **HostnameLookup**
   5.1. Habilitar as negociações com o servidor de nomes (DNS).
   5.2. Caso desativado, não haverá negociações desta espécie, o que possibilita um ganho de performance.
6. **ErrorLog**
   6.1. Diretório para armazenar registros de erros.
7. **LogLevel**
   7.1. Nível de detalhamento dos registros.

8. **Include**
   8.1. Anexa o conteúdo de outros arquivos contendo configurações.
   8.2. Caso essas configurações não existam, um erro será provocado.

9. **IncludeOptional**
   9.1. Anexa o conteúdo de outros arquivos contendo configurações.
   9.2. Caso essas configurações não existam, nenhum erro será provocado.
10. **AllowOverride**
    10.1. Permitir que a configuração especificada num *.htaccess* sobrescreva a do apache2.conf.
11. **LogFormat**
    11.1. Formatação do nome de arquivo de registros.
12. **SitesEnabled**
    12.1. Configurações dos sites hospedados no Apache.

Diretivas no arquivo *mpm_prefork.conf*

- **StartServers**
  - Número de processos a iniciar por padrão.
- **MinSpareServers**
  - Quantidade **miníma** de processos alocados dinamicamente;
- **MaxSpareServers**
  - Quantidade **máxima** de processo alocados dinamicamente;

Conclui-se então, que:

Quantidade mínima de processos = StartServers + MinSpareServers
Quantidade máxima de processos = StartServers + MaxSpareServers

- **MaxRequestWorkers**

- Quantidade total de clientes que podem ser atentidos simultaneamente.

- **MaxConnectionsPerChild**
  - Quantidade de requisições que um processo pode receber antes de morrer (ser descartado na reciclagem de processos).
  - Independe do número total de usuários, no entanto é relevante para o consumo de memória, visto que quando configurada como 0 (zero), não provocará reciclagem alguma; isto é, o processo continuará vivo, malgastando recursos até definhar ou acarretar um vazamento de memória. Mesmo num cenário que esta diretiva esteja com um número pequeno, processos culminarão em frequente reciclagem; o que em cenários de estresse pode ocasionar um déficit no tempo de resposta do servidor.
- **ServerLimit**
  - Quantidade de processos simultâneos que um servidor pode executar.
- **ListenBackLog**
  - Quantidade de lugares na fila de espera, quando os clientes extrapolar o valor da diretiva MaxSpareServers.