Implementasi Terminasi SSL Dengan HAProxy di Ubuntu 14.04

Pendahuluan

HAProxy yang merupakan kependekan bagi *High Availability Proxy*, adalah software load balancer TCP/HTTP open source yang terkenal dan dijadikan solusi proxying yang dapat berjalan di Linux, Solaris dan FreeBSD. Pemanfaatan utamanya adalah untuk meningkatkan kinerja dan reliabilitas dari suatu lingkungan server dengan mendistribusikan beban kerja (workload) kepada banyak server (misalnya: web, aplikasi dan database). Banyak lingkungan dengan profil-tinggi menggunakannya, termasuk GitHub, Imgur, Instagram dan Twitter.

Pada tutorial ini, kita akan belajar menggunakan HAProxy sebagai terminasi SSL, mengenskripsi lalu-lintas data dan menyeimbangkan beban dari beberapa web server. Bagaimana menggunakan HAProxy untuk mengarahkan lalu-lintas HTTP ke HTTPS juga dibahas.

Dukungan SSL natif telah terimplementasi dengan baik mulai dari HAProxy 1.5.x yang versi stabilnya telah dirilis ke publik pada Juni 2014.

Apa yang diperlukan?

Agar tutorial ini dapat diikuti secara lengkap, berikut ini adalah hal (terkait langsung) yang harus disiapkan:

- Setidaknya satu web server yang berjalan baik di dalam suatu jaringan lokal (*private*) dan mendengar koneksi HTTP (port 80)
- Akses root yang diperlukan untuk menginstall HAProxy dan beberapa konfigurasi. **(Bagaimana cara meng-setup akses root dapat dilihat pada tutorial *Initial Server Setup with Ubuntu 14.04* *(https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-14-04)*, langkah 3 dan 4).**
- Pasangan sertifikat SSL dan kunci lokal (*private key*) dengan suatu "*common name*" yang sesuai dengan nama domain atau IP Address yang dikelola.

Jika belum mempunyai pasangan seritifikat SSL dan private key, maka silakan membeli atau membuat sertifikat SSL sendiri.

Membuat File PEM (Gabungan Sertifikat dan Key SSL)

Dalam implementasi terminasi SSL dengan HAProxy, kita harus memastikan bahwa pasangan sertifikat dan key SSL mempunyai format yang tepat, PEM. Pada banyak kasus, kita dapat dengan mudah menggabungkan sertfikat SSL (file .crt atau .cer yang diberikan oleh Otoritas Sertifikat) dan key privat-nya (file .key, dibuat sendiri). Seandainya file sertifikat yang diperoleh bernama example.com.crt, dan file kuncinya bernama example.com.key, maka penggabungan dua file ini dapat dilakukan dengan perintah berikut:

```
cat example.com.crt example.com.key > example.com.pem
```

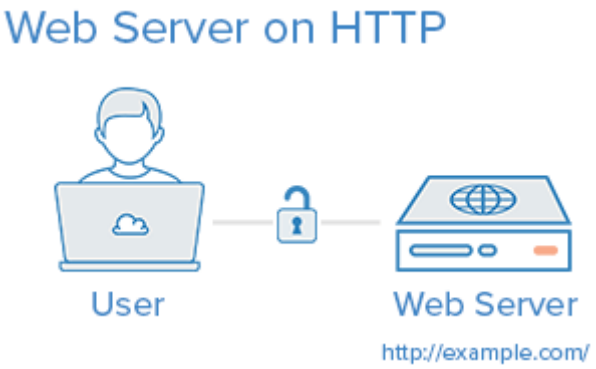
```
sudo cp example.com.pem /etc/ssl/private/
```

Ini membuat file PEM gabungan bernama example.com.pem dan menyalinkan ke dalam /etc/ssl/private. Seperti biasa, pastikan anda telah mengamankan salinan dari file *private key*, termasuk file PEM (yang tentu saja mengandung *private key*).

Sering terjadi, kita harus menyalin sertifikat *CA root* dan *CA intermediate* ke dalam file PEM.

Kondisi Awal

Secara umum, komunikasi antara pengguna (pengunjung aplikasi web) dan web server terjadi secara langsung, tanpa pengamanan (enkripsi) dan tanpa perantara. Kondisi yang tidak aman ini diperlihatkan pada gambar di bawah:

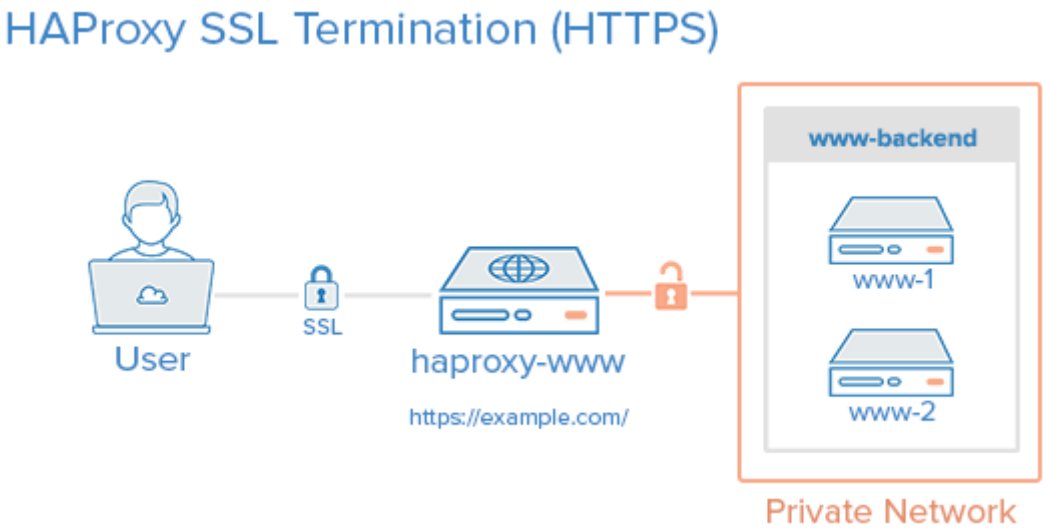


Jika lingkungan server anda berbeda dengan contoh di atas, misalnya sudah menggunakan SSL pada web server atau adanya database server yang terpisah, anda tetap dapat mengadopsi tutorial ini secara lengkap.

Jika anda belum akrab dengan konsep atau terminologi load-balancing dasar, seperti load balancing layer 7 atau backends atau ACL, artikel berjudul **An Introduction to HAProxy and Load Balancing Concepts** (https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts) dapat dipelajari lebih dahulu. Artikel lain yang sangat menarik **adalah How To Use HAProxy As A Layer 7 Load Balancer For WordPress and Nginx On Ubuntu 14.04** (https://www.digitalocean.com/community/tutorials/how-to-use-haproxy-as-a-layer-7-load-balancer-for-wordpress-and-nginx-on-ubuntu-14-04).

Tujuan Implementasi

Setelah menerapkan apa yang dibahas dalam tutorial ini, diharapkan kita akan mempunyai lingkungan server sebagai berikut:



Pengunjung aplikasi web akan mengakses website kita dengan mengubungi server HAProxy via protokol khusus HTTPS yang akan mendekripsi sesi SSL dan meneruskan request tidak-terenkripsi ke web server yang berfungsi sebagai back-end (yaitu server-server yang di letakkan di dalam bagian www-backend) melalui antarmuka jaringan lokal (private) pada port 80. Web server kemudian akan mengirimkan responnya ke server HAProxy yang akan melakukan enkripsi respon tersebut dan selanjutnya mengirimkannya kepada pengunjung yang membuat request asalnya.

Kita dapat men-setup www-backend berisi web server sebanyak yang diinginkan, selama mereka melayani *content* yang identik. Dengan kata lain, kita dapat memulainya dengan menyiapkan satu web server dan memperluasnya di kemudian waktu dengan menambahkan server-server web lan yang menyediakan layanan sama tersebut. Perlu diingat, saat lalu-lintas naik, server HAProxy mungkin mengalami bottleneck kinerje (performance) karena tidak tersedia cukup sumber daya dari sistem untuk menangani lalu-lintas yang tinggi dari pengunjung.

Catatan: Tutorial ini tidak meng-cover bagaimana memastikan bahwa server aplikasi/web menyediakan content yang sama karena itu seringnya tergantung pada server aplikasi atau web, bukan pada proxy atau load balancer.

Instalasi HAProxy 1.5.x

Instalasi haproxy dapat dilakukan dengan mudah. Repository ubuntu 14.04 telah menyiapkan semuanya. Namun, versi terbaru biasanya belum tersedia pada repo tersebut. Jadi kita harus mengupdate informasi sumber dari program apt atau dpkg. Caranya instalasi HAProxy terbaru adalah:

Tambahkan repository haproxy terbaru ke dalam sistem Linux:

```
sudo add-apt-repository ppa:vbernat/haproxy-1.5
```

Aupdate cache dari program apt:

```
sudo apt-get update
```

Install HAProxy 1.5 dengan perintah apt-get:

```
sudo apt-get install haproxy
```

Sekarang HAProxy 1.5 telah terinstal dan siap dikonfigurasi!

Konfigurasi HAProxy

Konfigrasi HAProxy adalah di dalam file /etc/haproxy/haproxy.cfg dan dibagi ke dalam dua bagian utama, yaitu:

- Global: berisi parameter-parameter *process-wide*
- Proxies: mengandung sesi defaults, listen, frontend dan backend

Sekali lagi, jika anda merasa belum akrab dengan HAProxy atau konsep dasar atau terminologi *load-balancing*, silakan membaca artikel **An Introduction to HAProxy and Load Balancing Concepts**.

Konfigurasi HAProxy: Global

Semua konfigurasi HAProxy ini dilakukan pada mesin di mana HAProxy diinstal, bukan pada web server.

Buka file haproxy.cfg menggunakan text editor kesuakaan anda, misalnya sublime text:

```
sudo subl /etc/haproxy/haproxy.cfg
```

Akan kita lihat bahwa di sana ada dua bagian yang telah terdefinisi, yaitu: **global** dan **defaults**.

Hal pertama yang perlu diset adalah maxconn dengan nilai (numerik) yang wajar dan beralasan. Seting ini mengakibatkan berapa banyak koneksi concurrent yang akan dibolehkan oleh HAProxy, ini dapat mempengaruhi QoS dan mencegah crashnya web server dalam menangani terlalu banyak request. Tambahkan baris berikut ke bagian global dari konfigurasi:

```
maxconn 2048
```

Berikutnya, dalam bagian defaults, tambahkan baris-baris berikut di bawah baris `mode http`:

```
 option forwardfor
option http-server-close
```

Opsi **forwardfor** digunakan untuk mengatur HAProxy untuk menambahkan header **X-Forwarded-For** untuk setiap request, dan opsi **http-server-close** akan mengurangi *latency* antara HAProxy dan pengguna dengan menutup koneksi tetapi memelihara yang **keep-alives**.

Konfigurasi HAProxy: Stats

HAProxy menyediakan fitur statistik yang sangat berguna dalam menentukan bagaimana HAProxy menangani lalu-lintas masuk (*incoming traffic*). Kita dapat mengaktifkan halaman statistik (stats) dari HAProxy dengan menambahkan baris-baris berikut ke dalam bagian defaults (sesuaikan user dan password dengan nilai yang aman):

```
   stats enable
   stats uri /stats
   stats realm Haproxy\ Statistics
   stats auth user:password
```

Ini berati kita dapat mengakses halaman statistik HAProxy dengan mengunjungi web http://nama_domain/stats (misalnya: https://example.com/stats).

Ada beberapa poin lagi yang perlu dikonfigurasi, jangan tutup dulu sublime text anda!

Konfigurasi HAProxy: Proxies

Konfigurasi Frontend

Hal pertama yang perlu ditambahkan suatu **frontend** untuk menangani koneksi HTTP yang masuk. Pada akhir file, tambahkan fronted bernama www-http. Pastikan untuk mengganti haproxy_www_public_IP dengan IP Address publik dari komputer dimana HAProxy berjalan:

```
frontend www-http
   bind haproxy_www_public_IP:80
   reqadd X-Forwarded-Proto:\ http
   default_backend www-backend
```

Di bawah ini adalah penjelasan baris-baris di dalam fronted www-http tersebut:

- **frontend www-http**: menentukan suatu frontend bernama "www-http"
- **bind haproxy_www_public_IP:80**: silakan ganti haproxy_www_public_IP dengan IP Address publik dari server HAProxy. Ini memberitahukan HAProxy bahwa fronted ini akan menangani lalu-lintas jaringan yang masuk pada IP Adress ini dan port 80 (HTTP)
- **reqadd X-Forwarded-Proto:\ http**: Menambahkan header http pada ujung dari akhir request HTTP
- **default_backend www-backend**: Ini menentukan bahwa suatu lalu-lintas yang diterima fronted ini akan diteruskan ke **www-backend** yang akan didefnisikan pada langkah berikutnya.

Selanjutnya, kita akan menambahkan suatu frontend untuk menangani koneksi HTTPS yang masuk. Pada ujung file, tambahkan frontend bernama www-https. Pastikan untuk mengganti haproxy_www_public_IP dengan IP Address publik dari server HAProxy:

```
frontend www-https
   bind haproxy_www_public_IP:443 ssl crt /etc/ssl/private/example.com.pem
   reqadd X-Forwarded-Proto:\ https
   default_backend www-backend
```

Di bawah ini adalah penjelasan baris-baris di dalam fronted www-https tersebut:

- **frontend www-https**: menetapkan suatu frontend bernama "www-https"
- **bind haproxy_www_public_IP:443 ssl crt ...**: Ganti haproxy_www_public_IP dengan IP Address dari server HAProxy dan example.com.pem dengan pasangan sertifikat SSL dan key-nya yang digabung ke dalam format pem. Ini memberitahukan HAProxy bahwa frontend ini akan menangani lalu-lintas jaringan masuk pada IP Address ini dan port 443 (HTTPS).
- **reqadd X-Forwarded-Proto:\ https**: Menambahkan header https pada ujung dari akhir request HTTPS.
- **default_backend www-backend**: Ini menetapkan bahwa lalu-lintas yang diterima oleh frontend ini juga akan diteruskan ke **www-backend**.

Konfigurasi Backend

Setelah menyelesaikan konfigurasi pada bagian frontends, sekarang saatnya mengatur backend dengan menambahkan beberapa baris berikut. Pastikan untuk mengganti teks tertentu sesuai dengan kondisi dan kebutuhan, terutama IP Address dari web server yang dijadikan backend:

```
backend www-backend
   redirect scheme https if !{ ssl_fc }
   server www-1 www_1_private_IP:80 check
   server www-2 www_2_private_IP:80 check
```

Berikut ini adalah penjelasan dari baris-baris di atas:

- **backend www-backend**: menetapkan backend bernama www-backend
- **redirect scheme https if !{ ssl_fc }**: Baris ini meredirect request HTTP ke HTTPS, mengakibatkan situs web kita menjadi HTTPS-only. Jika anda membolehkan request HTTP dan HTTPS, silakan hapus baris ini
- **server www-1 ...**: menentukan server backend bernama www-1, IP Address private (yang perlu disesuaikan) dan port dimana server www_1 mendengar koneksi, 80. Opsi check mengakibatkan load balancer secara berkala memeriksa kesehatan dari server ini.
- **server www-2 ...**: sama dengan baris sebelumnya. www_2 adalah server lain yang menyediakanlayanan indentik dengan www_1. Server web lain dapat diikutkan dengan menambahkan baris-baris berikutnya yang mirip.

Sekarang simpan dan keluar dari sublime text (haproxy.cfg). HAProxy telah siap untuk dijalankan, tetapi kita akan mengaktifkan fitur logging terlebih dahulu.

Mengaktifkan Fitur Logging di HAProxy

Fitur logging dari HAProxy dapat diaktifkan dengan mudah. Pertama, edit file rsyslog.conf:

```
sudo subl /etc/rsyslog.conf
```

Kemudian temukan dua baris berikut dan hilangkan tanda komentar (*uncomment*) untuk mengaktifkan resepsi syslog UDP. Hasilnya seperti di bawah ini:

```
$ModLoad imudp
$UDPServerRun 514
$UDPServerAddress 127.0.0.1
```

Berikutnya silakan restart rsyslog agar konfigurasi ini berfungsi baik:

```
sudo service rsyslog restart
```

Logging HAProxy sudah aktif! File log akan dibuat di /var/log/haproxy.log setelah HAProxy dijalankan (dimuat ulang).

Menjalankan HAProxy

Pada mesin dimana HAProxy berada, jalankan HAProxy dengan konfigurasi yang telah kita atur di atas:

```
sudo service haproxy restart
```

Sekarang HAProxy telah mengerjakan terminasi SSL dan melakukan *load balancing* terhadap beberapa web server.

Perlu Diperhatikan

HAProxy telah berjalan tetapi ada beberapa hal yang masih perlu diperhatikan, di antaranya:

- Jika server web yang dijadikan backend belum dapat diakses, silakan update konfigurasi nameserver untuk mengarahkan nama domain dari web server ke IP Address dari server HAProxy.
- Jika anda ingin server-server menggunakan hanya HTTPS, perlu dipastikan bahwa semua web server (misalnya: www-1, www-2, dst.) hanya mendengar pada **IP Address privat** dan port 80. Jika tidak, pengunjung akan dapat mengakses server web via HTTP (tidak terenkripsi) pada IP Address publiknya.
- Kunjungi IP Address server HAProxy (haproxy-www) via HTTPS dan pastikan backend ditampilkan sebagaimana yang diharapkan.
- Kunjungi haproxy-www via HTTP dan pastikan bahwa koneksi tersebut diarahkan ke HTTPS (kecuali HAProxy dikonfigurasi untuk membolehkan kedua request HTTP dan HTTPS)

Kesimpulan

Solusi load balancer dan reverse-proxy telah digunakan untuk menangani koneksi SSL dan dapat digunakan untuk memperluas lingkungan server (secara horisontal). Ada banyak poin yang dapat ditambahkan ke dalam konfigurasi HAProxy untuk meningkatkan availability layanan web. Insya Allah akan terus hadir tutorial berikutnya tentang HAProxy.

Referensi:

- **How To Implement SSL Termination With HAProxy on Ubuntu 14.04** (https://www.digitalocean.com /community/tutorials/how-to-implement-ssl-termination-with-haproxy-on-ubuntu-14-04)