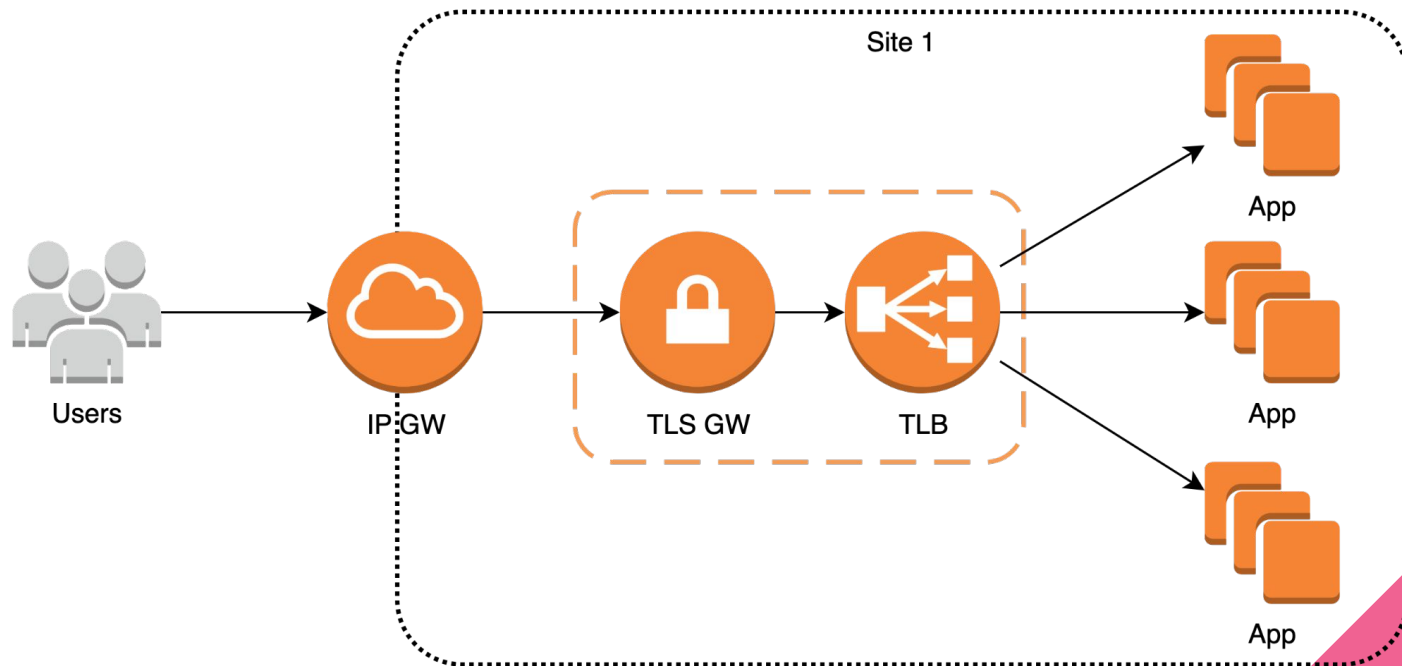


Использование HAProxy для балансировки нагрузки между локациями

Максим Куприянов



Внешний балансировщик



Почему как внешний балансировщик?

- Независимость от поставщика услуг
- Упрощение масштабирования
- Отказоустойчивость
- Безопасность

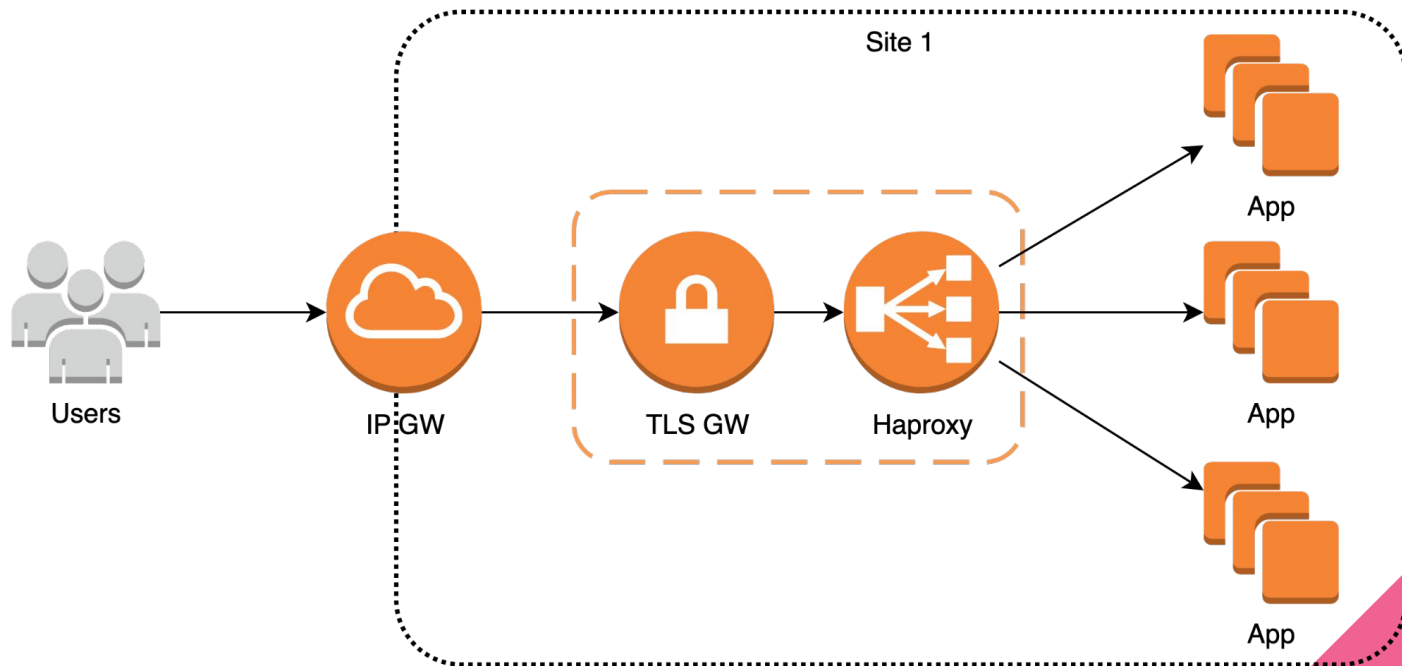
Нароуху?

- Высокопроизводительный L7-балансировщик трафика: до 2М RPS и до 100Гбит/с на одном сервере
- Прообраз (Zprox) появился в 1999-м, а Нароуху 1.0 вышел 16.12.2001 года. Текущая стабильная версия – 3.0. Написан на языке C
- Автор и мейнтейнер: [Willy Tarreau](#). Входит в Linux Kernel Stable Team и Linux Kernel Security Team
- Более миллиарда загрузок на Docker Hub
- Используется: GitHub, Stack Overflow, Reddit, X, Booking.com, Amazon Web Services и многими другими

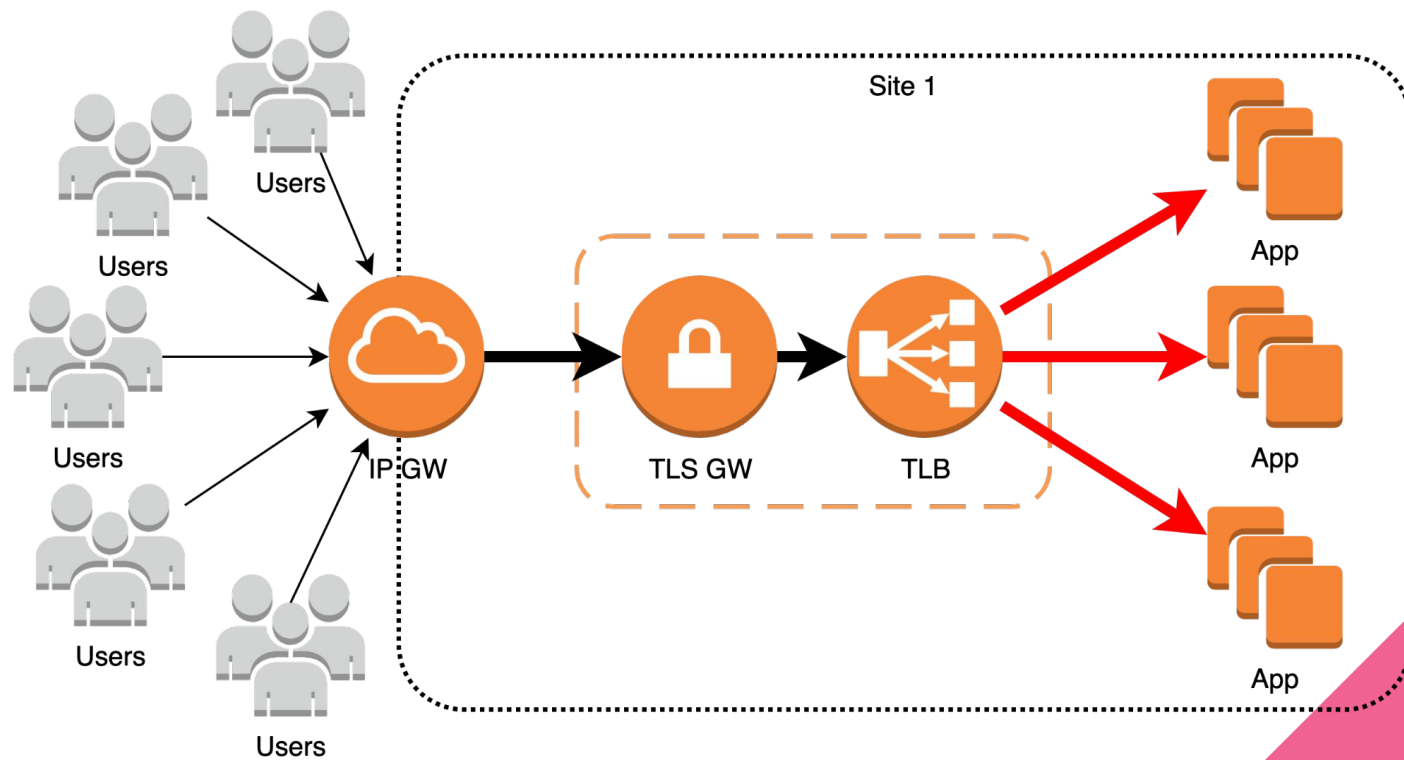
Учимся падать



Наш сайт



Пиковая нагрузка



Главный показатель нагрузки

Requests Per Second (RPS) – количество запросов в секунду



Простейший способ защиты от перегрузки

```
frontend fe_site1
  bind :80
  stick-table type ip size 1m expire 10s store http_req_rate(10s)
  http-request track-sc0 src
  http-request deny if { sc_http_req_cnt(0) gt 100 }
  default_backend be_site1
```

Composite Address B Haproxy

```
/****** Composite address manipulation *****/  
* Composite addresses are simply unsigned long data in which the higher bits  
* represent a pointer, and the two lower bits are flags. There are several  
* places where we just want to associate one or two flags to a pointer (eg,  
* to type it), and these functions permit this. The pointer is necessarily a  
* 32-bit aligned pointer, as its two lower bits will be cleared and replaced  
* with the flags.  
*****/
```

Способ защиты от перегрузки #2

```
frontend fe_site1
  bind :80
  stick-table type integer size 1 expire 1s store http_req_rate(1s)
  http-request track-sc0 always_true
  http-request set-var(req.rps) sc_http_req_cnt(0)
  http-request set-var(req.rnd) rand,mod(req.rps)
  acl overloaded var(req.rps) -m int gt 100
  acl unlucky_one var(req.rnd) -m int ge 100
  use_backend be_error_429 if overloaded unlucky_one
  default_backend be_site1

backend be_error_429
  http-request deny deny_status 429
```

Основа метода:

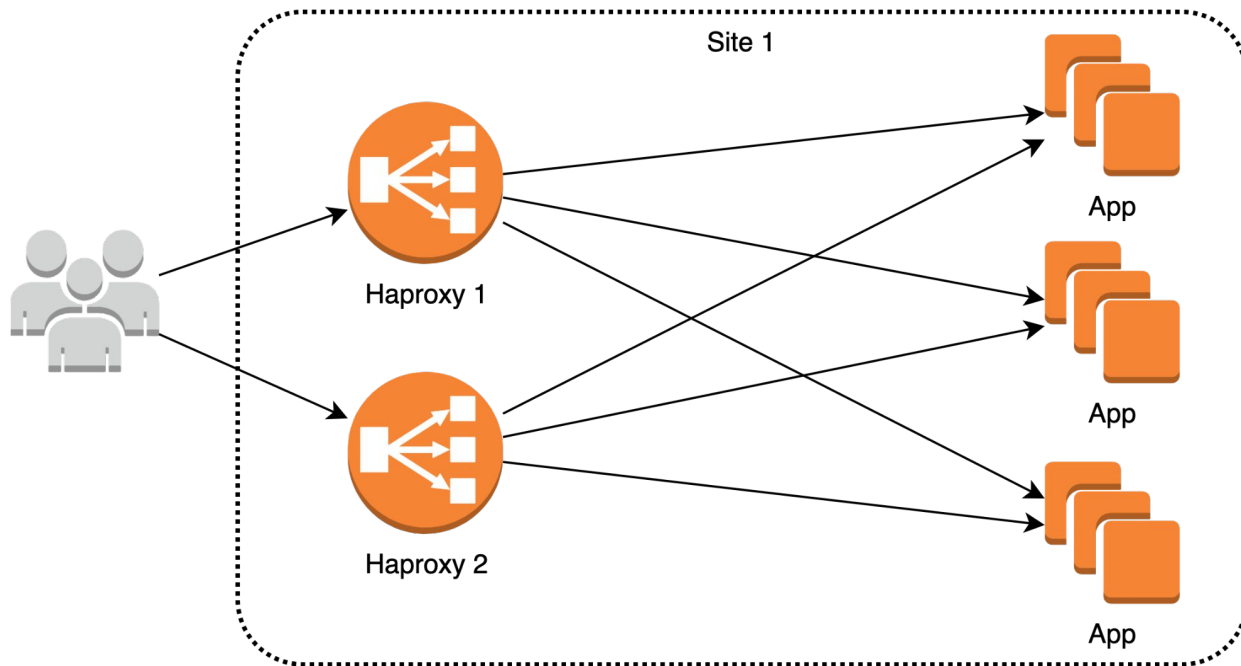
- Stick-tables & Sticky-counters
- ACL
- Переменные



Командная работа



Теперь у нас два прокси



Peers-секция

```
global
  localpeer tlb1s1

peers tlb_peers
  bind :10000
  server tlb1s1
  server tlb2s1 10.0.0.2:10000
  table tlb1s1 type integer size 1 expire 5s store http_req_rate(5s)
  table tlb2s1 type integer size 1 expire 5s store http_req_rate(5s)
```

Frontend-секция

```
frontend fe_site1
  bind :80
  http-request track-sc0 always_true table tlb_peers/tlb1s1
  http-request set-var(req.rps1s1) sc0_http_req_rate(tlb_peers/tlb1s1),div(5)
  http-request set-var(req.rps2s1) sc0_http_req_rate(tlb_peers/tlb2s1),div(5)
  http-request set-var(req.rps) var(req.rps1s1),add(req.rps2s1)
  http-request set-var(req.rnd) rand,mod(req.rps)
  acl overloaded var(req.rps) -m int gt 100
  acl unlucky_one var(req.rnd) -m int ge 100
  use_backend be_error_429 if overloaded unlucky_one
  default_backend be_site1
```


Важно помнить:

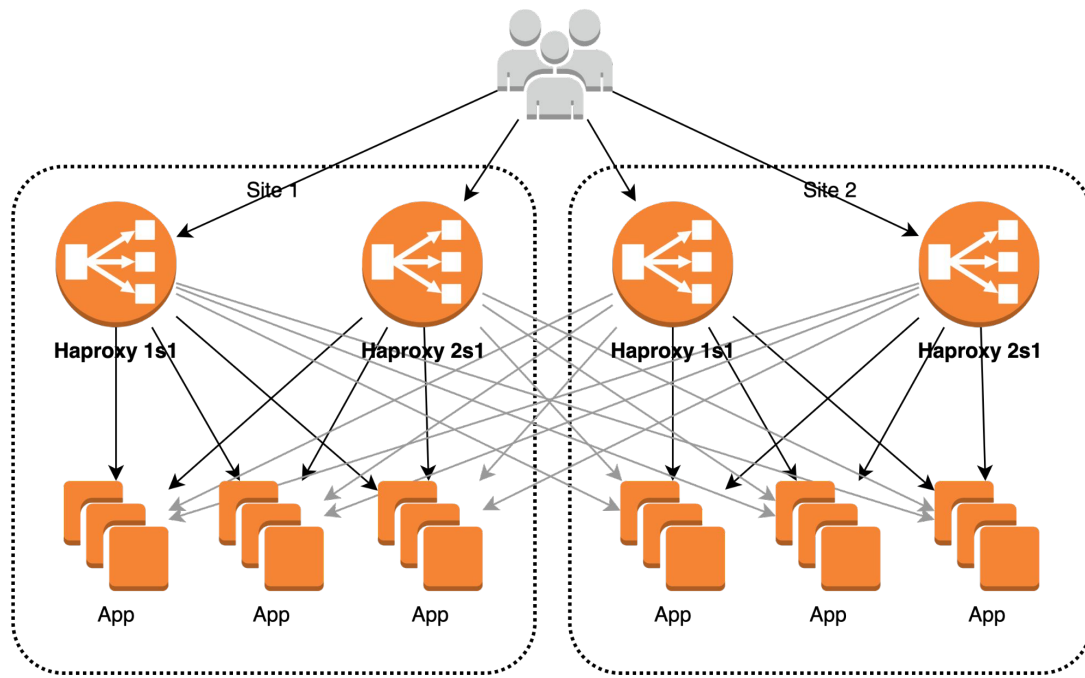
- Явно задавайте `localpeer` в `global`-секции
- Между балансерами можно включить TLS
- Балансеры не синхронизируют, а перезаписывают таблицы



Fallback на второй сайт



Растём, второй сайт



Peers-секция

```
global
  localpeer tlb1s1

peers tlb_peers
  bind :10000
  server tlb1s1
  server tlb2s1 10.0.0.2:10000
  server tlb1s2 10.0.1.1:10000
  server tlb2s2 10.0.1.2:10000
  table tlb1s1 type integer size 1 expire 5s store http_req_rate(5s)
  table tlb2s1 type integer size 1 expire 5s store http_req_rate(5s)
  table tlb1s2 type integer size 1 expire 5s store http_req_rate(5s)
  table tlb2s2 type integer size 1 expire 5s store http_req_rate(5s)
```

Backend-секции

```
backend be_local_site
    server cluster1s1 track be_all_sites/cluster1s1
    server cluster2s1 track be_all_sites/cluster2s1
```

```
backend be_all_sites
    default-server check
    server cluster1s1 10.0.0.11:80
    server cluster2s1 10.0.0.12:80
    server cluster1s2 10.0.1.11:80
    server cluster2s2 10.0.1.12:80
```

Frontend-секция

```
frontend fe_site1
  default_backend be_all_sites
  http-request track-sc0 always_true table tlb_peers/tlb1s1
  http-request set-var(req.rps1s1) sc0_http_req_rate(tlb_peers/tlb1s1),div(5)
  http-request set-var(req.rps2s1) sc0_http_req_rate(tlb_peers/tlb2s1),div(5)
  http-request set-var(req.rps1s2) sc0_http_req_rate(tlb_peers/tlb1s2),div(5)
  http-request set-var(req.rps2s1) sc0_http_req_rate(tlb_peers/tlb2s2),div(5)
  http-request set-var(req.rps1) var(req.rps1s1),add(req.rps2s1)
  http-request set-var(req.rps) var(req.rps1),add(req.rps1s2),add(req.rps2s2)
  http-request set-var(req.local_srv_count) nbsrv(be_local_site)
  http-request set-var(req.avg_local_srv_rps) var(req.rps1),div(req.local_srv_count)
  http-request set-header X-LB-RPS %[var(req.rps)]
  acl local_srv_alive var(req.local_srv_count) -m int gt 0
  acl local_rps_ok var(req.avg_local_srv_rps) -m int le 50
  use_backend be_local_site if local_srv_alive local_rps_ok
```

Полезно знать:

- Может быть несколько backend-секций с различной настройкой весов для серверов
- Заголовки очень важны и нужны для отладки
- Все данные можно отправить в логи



Вопросы?

