

いちから学んでみた HAProxy～入門編～

2025-02-21 OSC2025 Tokyo/Spring

Linux-HA Japan プロジェクト

高橋 由季菜



本日の流れ

- 自己紹介、コミュニティ紹介
- 本講座の背景
- HAProxyとは？
- HAProxyを試してみよう！
- HAProxyのポテンシャル！



自己紹介

- 名前

- 高橋 由季菜 (たかはし ゆきな)
- OSCは今回が初参加。宜しくお願い致します。

- 経歴

- 2024年4月より NTT OSSセンタ に配属
 - HAProxyとの初めての出会い



コミュニティ紹介



- Pacemakerの日本公式コミュニティ
 - <https://linux-ha-japan.github.io/> (※OSDNから移転済み)
 - メーリングリストは従来通りに活動中！
- Pacemakerを中心とした高可用クラスタに関する話題を日本語で情報交換しています！
- 過去OSC講演実績あり
 - 2024 Online/Spring :
試して覚えるPacemaker入門_ AzureでPacemakerを使ってみよう
 - 2022 Online/Fall :
VirtualBox と Rocky Linux 8 で始めるPacemaker

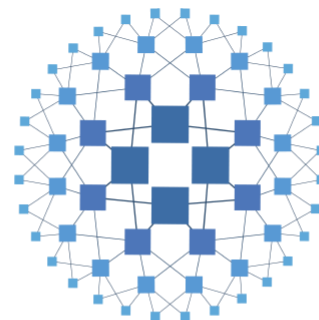
本講演の背景

- 昨年に初めてHAProxyと出会い、勉強を始めました
 - その経験(勉強)をふまえてHAProxyを紹介します
- 今回「**HAProxy**」を中心に扱います
 - HAProxyの冗長化については以下動画を参照ください
 - 2021 Online/Fall :
PacemakerとHAProxyではじめる高可用ロードバランサ入門
 - <https://www.youtube.com/watch?v=j2nsVqdx8qM>

HAProxyとは？

HAProxyとは？

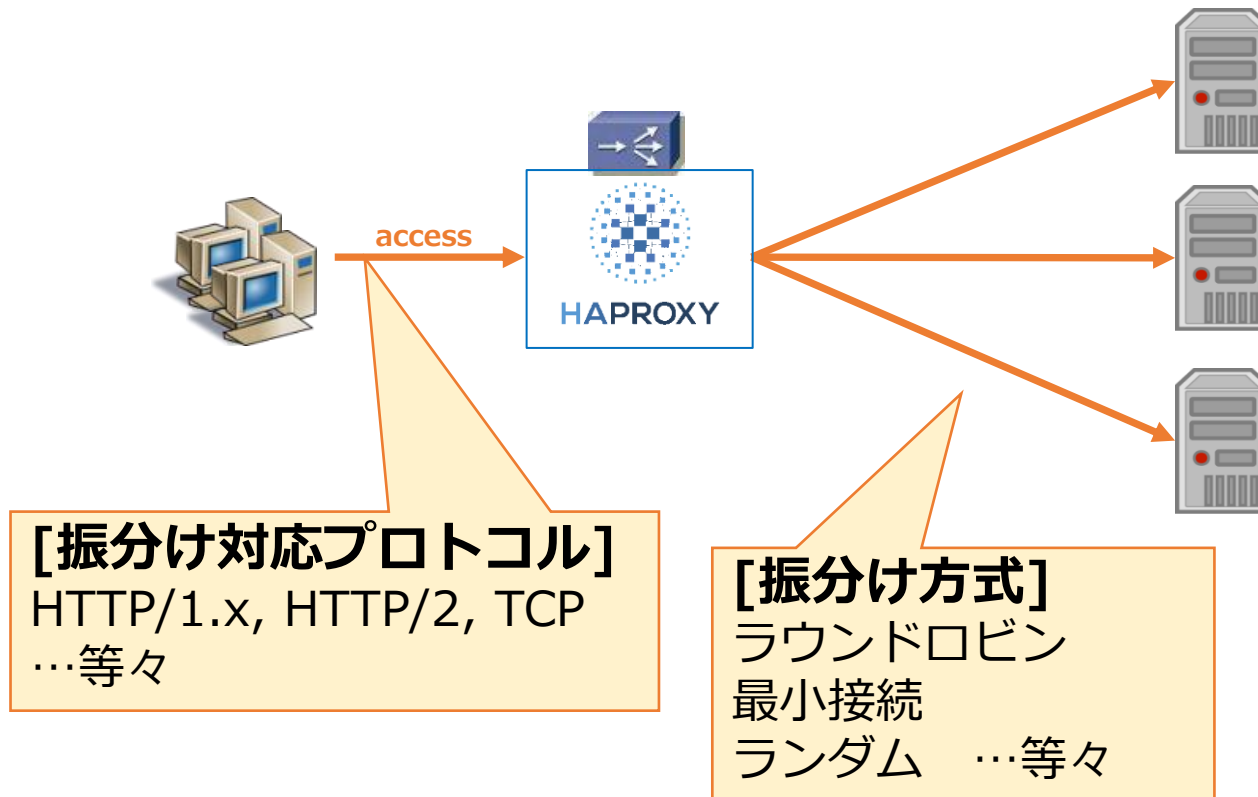
- 多機能なオープンソースのソフトウェアロードバランサ
 - L4/L7 ロードバランサとして利用可能
 - パーシステンス(セッション維持)、SSL暗号化・復号(※)が可能などロードバランサに必要な機能を具備
 - L4～L7の情報に基づく柔軟な振分け、通信拒否、書き換えが可能
 - プロトコルに準じたサーバ監視が可能
- コミュニティ版は無料！勉強しやすい！
 - URL : <https://www.haproxy.org/>



HAPROXY

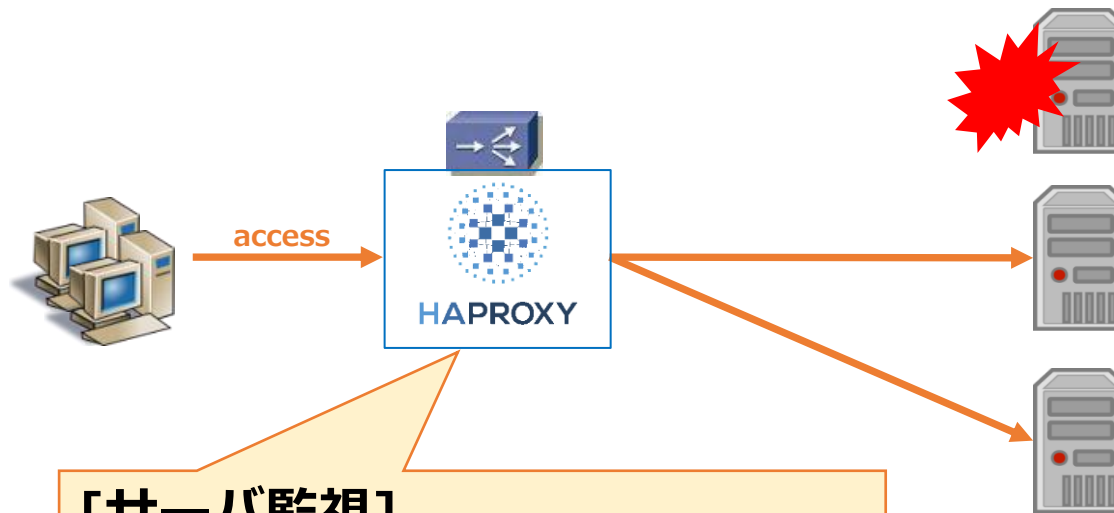
HAProxyができること [1/5]

- ロードバランズ(負荷分散)



HAProxyができること [2/5]

- サーバ監視

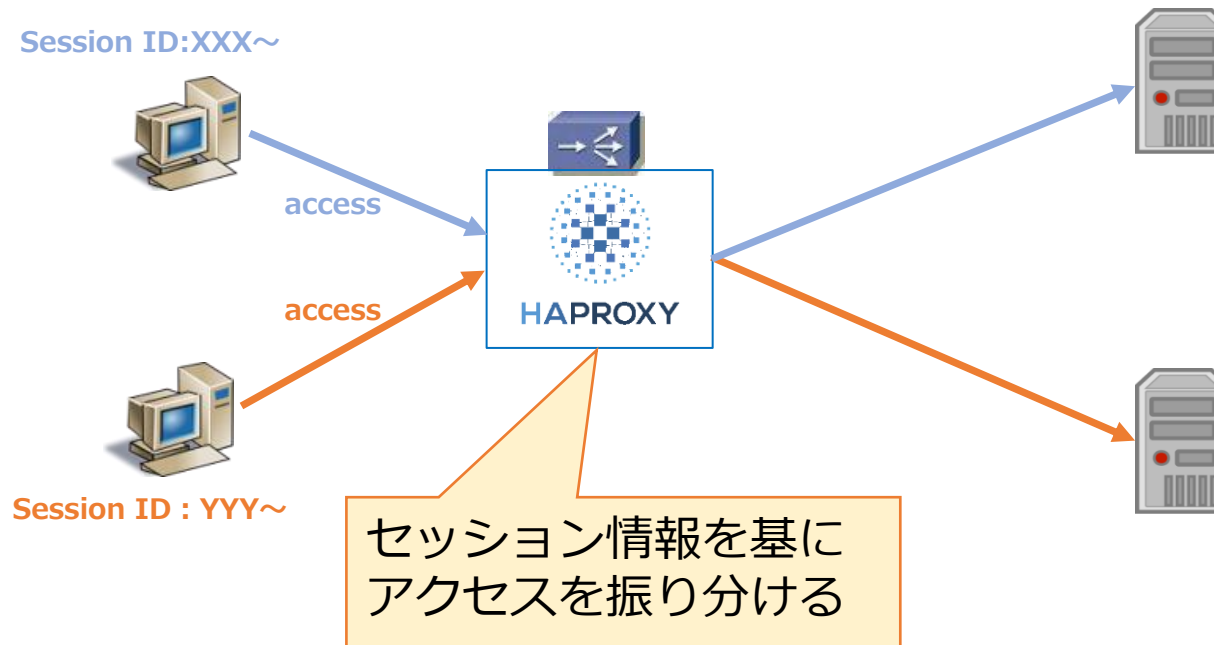


[サーバ監視]

- TCPポートチェック
 - プロトコルチェック
HTTP, LDAP, MySQL,
PostgreSQL, Redis, SMTP
- ※上記以外はユーザカスタマイズで
対応可能

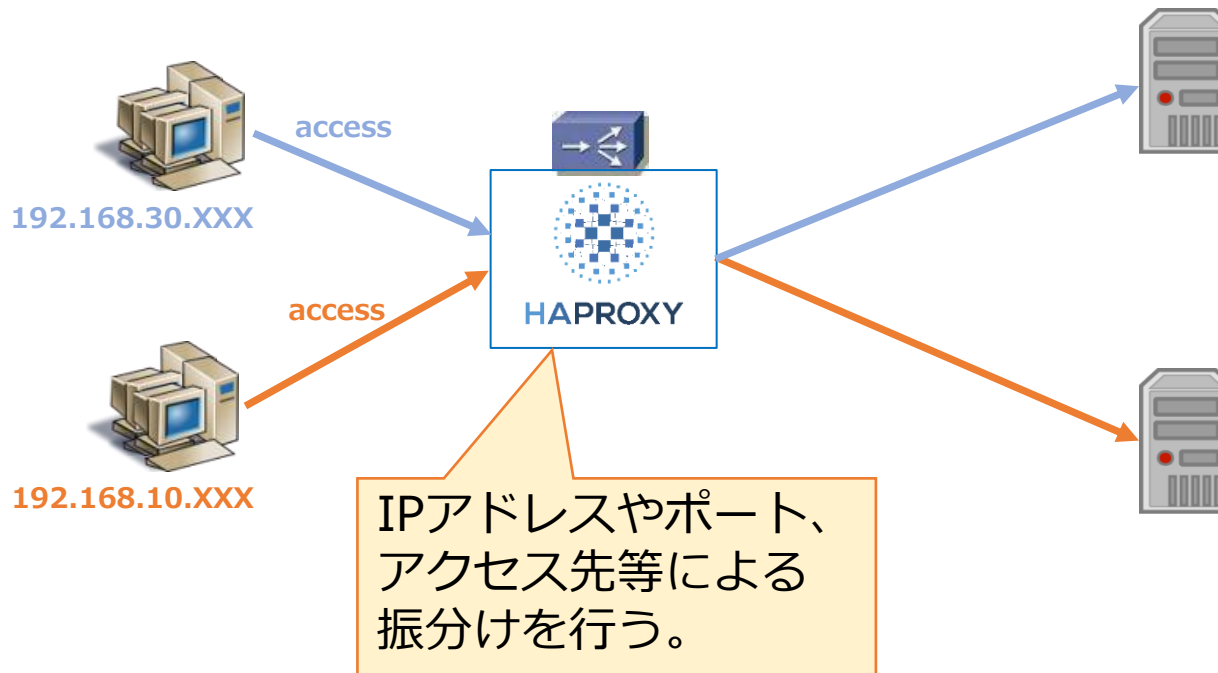
HAProxyができること [3/5]

- パーシステンス(セッション維持)



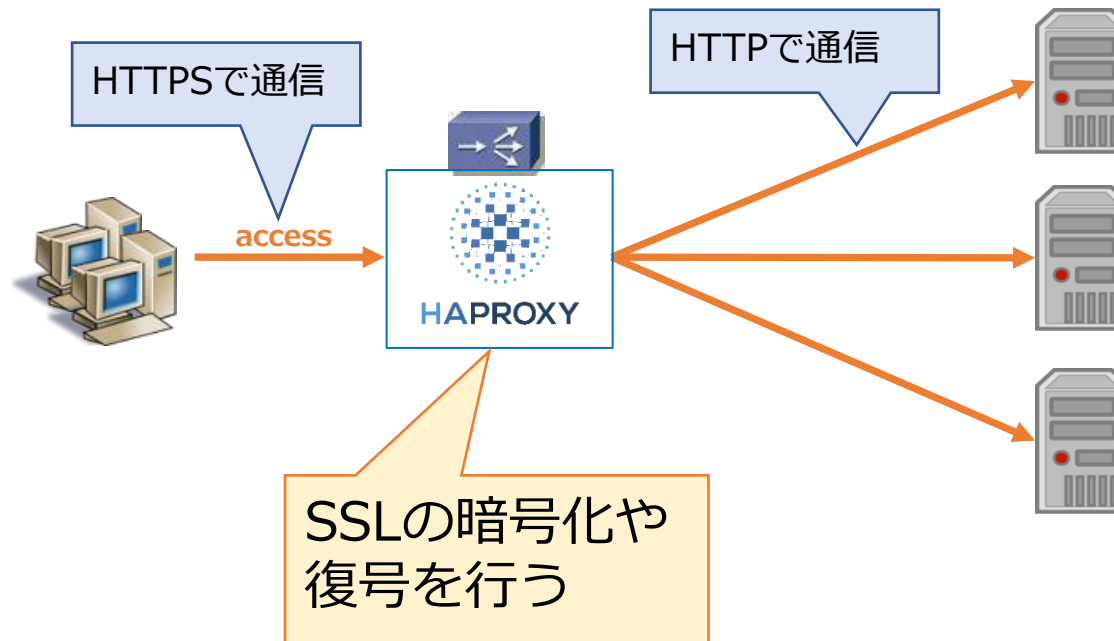
HAProxyができること [4/5]

- アクセス制御



HAProxyができること [5/5]

- SSLターミネーション



HAProxyとは？ ～まとめ～

- HAProxyは多機能な**ロードバランサ**！
 - ロードバランス(負荷分散)
 - サーバ監視
 - パーシステンス(セッション維持)
 - アクセス制御
 - SSLターミネーション

早速HAProxyを
試してみよう！



HAProxyを試してみよう！

HAProxyを試してみよう！

以下の流れで試してみました

1. サーバを用意する
2. HAProxyをインストールする
3. HAProxyを設定する
4. HAProxyを起動する
5. HAProxyを試してみる

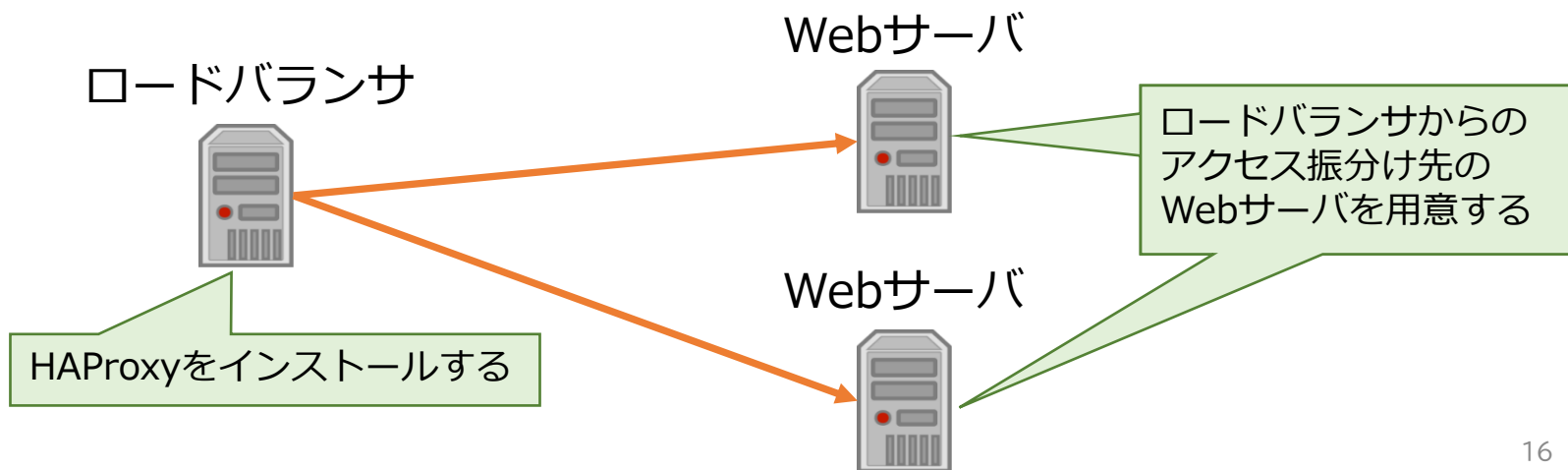
サーバを用意する

- 必要なサーバ

1. HAProxyをインストールするロードバランサ
2. ロードバランサからのアクセス振分け先のWebサーバ

…ロードバランサ内にWebサーバを構築してもOK

…Webサーバの構築については本講演では扱いません

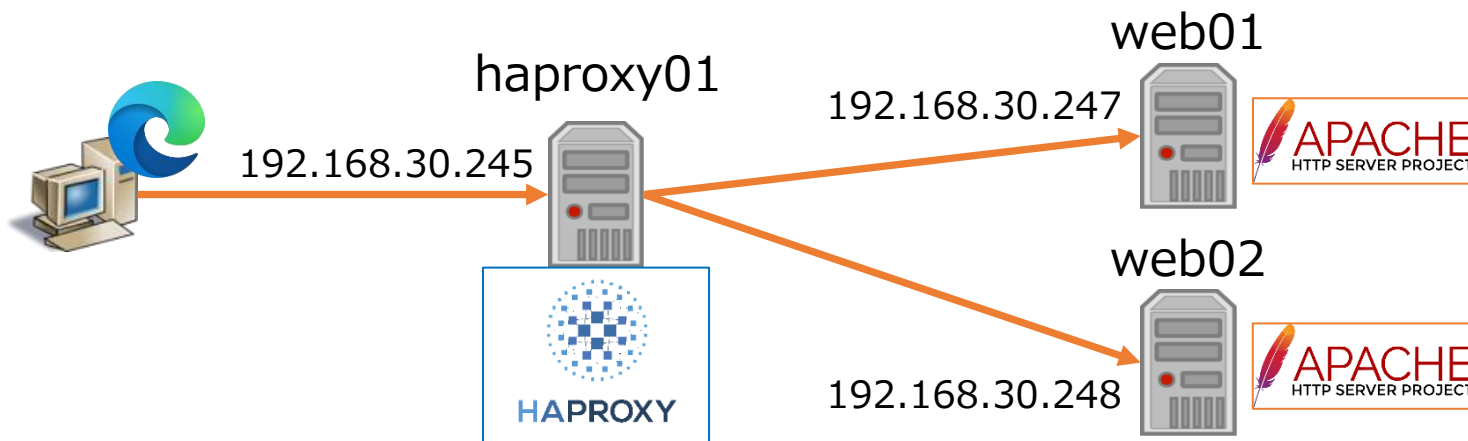


HAProxyをインストールする [1/4]

- 本講演資料のサーバ構成

サーバ種別 (ホスト名)	OS	スペック	インストールするパッケージ
ロードバランサ (haproxy01)	RHEL 9.2	CPU : 2 メモリ : 4 GiB	haproxy-2.4.17-6.el9 socat-1.7.4.1-5.el9
Webサーバ#1 (web01)	RHEL 9.2	CPU : 2 メモリ : 4 GiB	httpd-2.4.53-11.el9_2.5
Webサーバ#2 (web02)	RHEL 9.2	CPU : 2 メモリ : 4 GiB	httpd-2.4.53-11.el9_2.5

- 構成イメージ



HAProxyをインストールする [2/4]

HAProxyに使用するパッケージをインストールする。

HAProxyパッケージのインストール ■ロードバランサ(haproxy01)で

```
# dnf install -y haproxy
```

socatパッケージのインストール ■ロードバランサ(haproxy01)で

```
# dnf install -y socat
```

HAProxyをインストールする [3/4]

HAProxyがインストールされていることを確認する。

HAProxyパッケージのインストール確認 ■ロードバランサ(haproxy01)で

```
# haproxy -v
HAProxy version 2.4.22-f8e3218 2023/02/14 - https://haproxy.org/
Status: long-term supported branch - will stop receiving fixes
around Q2 2026.
Known bugs: http://www.haproxy.org/bugs/bugs-2.4.22.html
Running on: Linux 5.14.0-284.71.1.el9_2.x86_64 #1 SMP
PREEMPT_DYNAMIC Mon Jun 17 10:33:25 EST 2024 x86_64
```

HAProxyをインストールする [4/4]

参考：コミュニティ版をソースからビルドする場合

1. コミュニティからソースファイルをダウンロード
 - <https://www.haproxy.org/>
2. インストールガイドに従ってビルドする
 - <https://github.com/haproxy/haproxy/blob/master/INSTALL>

HAProxyを設定する [1/6]

- HAProxyの設定ファイル
 - 設定ファイル名 : haproxy.cfg
 - ファイル配置先 : /etc/haproxy
 - 配置先は、起動オプション-fで、指定することも可能。

HAProxyを設定する [2/6]

• HAProxyの設定ファイル構成

セクション名		説明
global		HAProxy全体の動作に関わるパラメータを設定するセクション
proxies	defaults	listen、frontend、backend セクションにおけるデフォルト値を設定するセクション
	frontend	クライアントとの接続に関するパラメータを設定するセクション
	backend	バックエンドサーバとの接続に関するパラメータを設定するセクション
	listen	frontend、backendセクションの設定を統合して設定するセクション

※その他は割愛。気になる方はコミュニティのマニュアルをご参照ください

<https://docs.haproxy.org/>

HAProxyを設定する [3/6]

• HAProxyの設定変更(1/4)

HAProxyの設定変更 ■ロードバランサ(haproxy01)で

```
# vi /etc/haproxy/haproxy.cfg
global
  log      127.0.0.1 local2 info
  chroot   /var/lib/haproxy
  pidfile  /var/run/haproxy.pid
  user     haproxy
  group    haproxy
  maxconn  2010
  daemon

  stats socket /var/lib/haproxy/stats mode 660 level admin
  userlist stats-auth
  group admin users admin
  group admin users admin
  user  admin insecure-password passwd
```

global の maxconn (最大同時接続数) は後述のlisten/frontend の maxconn の合計値より大きい値を設定する

管理画面のアクセス用ユーザを設定する

HAProxyを設定する [4/6]

• HAProxyの設定変更(2/4)

```
defaults
mode                http
log                 global
option             httpclose
option forwardfor  except 127.0.0.0/8
log-format         "%ci:%cp [%tr] %b/%s %ST %hr %hs %Qr"
retries            3
timeout connect    10s
timeout client     60s
timeout server     60s
timeout queue      60s
timeout http-request 10s
timeout check      10s
```

X-Forwarded-For(XFF)ヘッダを付与したい場合に設定する。

タイムアウト値はシステム要件や振分け先にあわせて調整が必要。

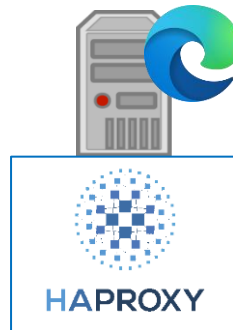
HAProxyを設定する [4/6]

- HAProxyの設定変更(3/4)

```
listen statspage
  bind 127.0.0.1:80
  maxconn 10
  stats enable
  stats uri /statspage
  stats admin if TRUE
```

管理画面関連の設定

haproxy01



ローカルからアクセスすることを想定

HAProxyを設定する [5/6]

• HAProxyの設定変更(4/4)

```
frontend main
```

```
bind 192.168.30.245:80
```

クライアントからの接続先を指定

```
maxconn 2000
```

```
default_backend web
```

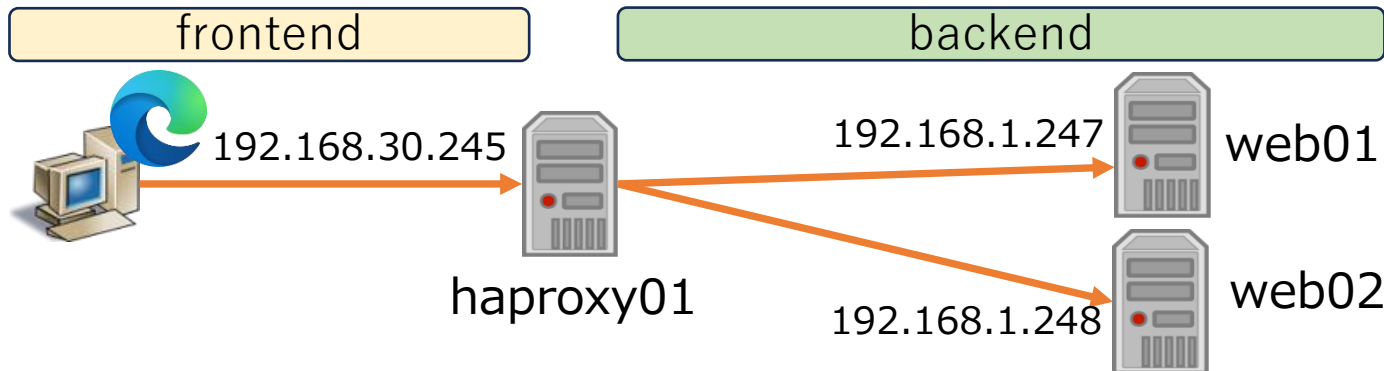
```
backend web
```

振り分け先のWebサーバを指定

```
balance roundrobin
```

```
server web01 192.168.1.247:80 check maxconn 1000
```

```
server web02 192.168.1.248:80 check maxconn 1000
```



HAProxyを設定する [6/6]

- rsyslogの設定変更

HAProxyの設定変更 ■ロードバランサ(haproxy01)で

```
# vi /etc/rsyslog.conf
～略～
module(load="imudp")
Input(type="imudp" port="514")
local2.*    /var/log/haproxy.log

# systemctl restart rsyslog.service
```

HAProxyを起動する

- HAProxyの起動(systemdサービス起動)

HAProxyの起動 ■ロードバランサ(haproxy01)で

```
# systemctl start haproxy.service
# systemctl status haproxy.service
● haproxy.service - HAProxy Load Balancer
~略~

Feb 03 09:54:10 haproxy01 systemd[1]: Starting HAProxy Load Balancer...
Feb 03 09:54:10 haproxy01 haproxy[3486]: [NOTICE] (3486) : New worker
#1 (3488) forked
Feb 03 09:54:10 haproxy01 systemd[1]: Started HAProxy Load Balancer.
```

HAProxyを起動する

参考：HAProxyの起動(コマンド起動)

HAProxyの起動 ■ロードバランサ(haproxy01)で

```
# haproxy -f /etc/haproxy/haproxy.cfg
```

HAProxyを試してみよう！

以下の流れで試してみました

1. サーバを用意する
2. HAProxyをインストールする
3. HAProxyを設定する
4. HAProxyを起動する
5. HAProxyを試してみる

試してみよう！



HAProxyを試してみよう！

試したことリスト

1. HAProxy管理画面へのアクセス
2. 負荷分散の確認
3. HAProxy管理画面の操作
4. HAProxyの状態確認

デモもあります！

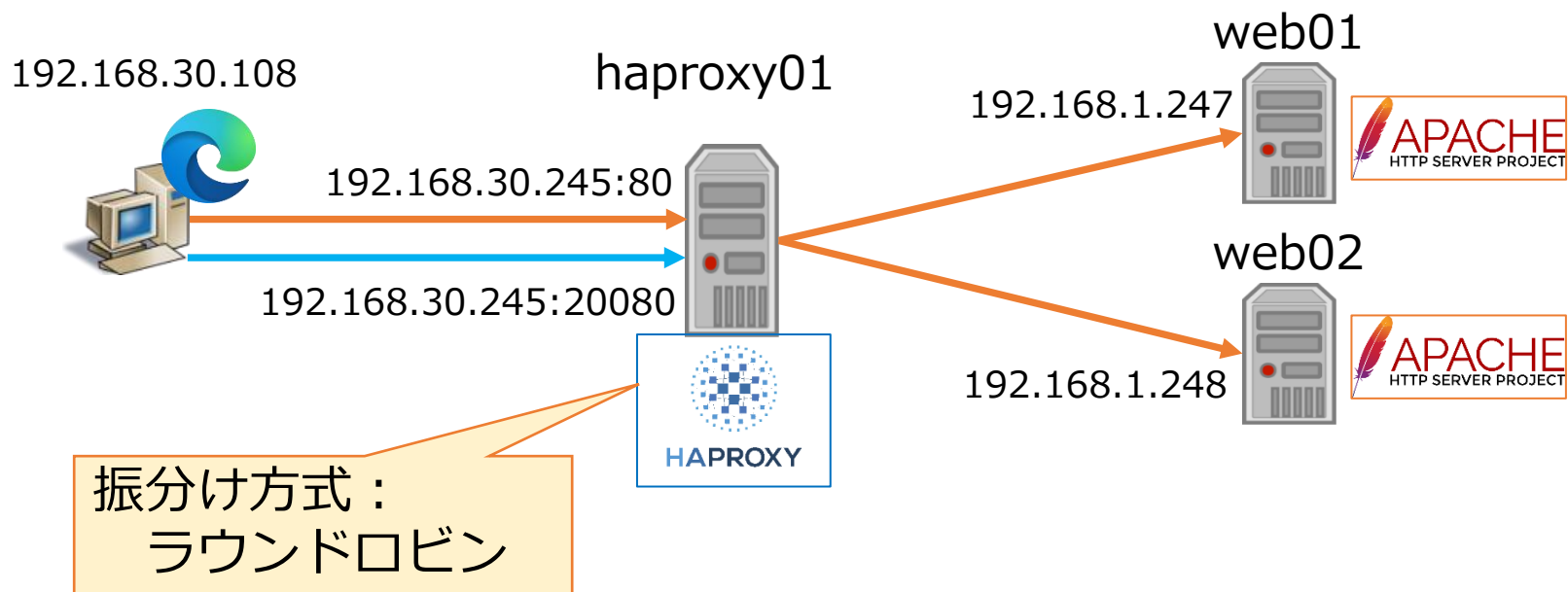


HAProxyを試してみよう！

デモ環境の補足

【凡例】

- Webサーバへのアクセス
- HAProxy管理画面へのアクセス



HAProxy管理画面へのアクセス [1/2]

- HAProxy管理画面にアクセスする

本デモ環境の場合：http://192.168.30.245:20080/statspage



HAProxy管理画面へのアクセス [2/2]

- HAProxy管理画面にログインできたことを確認する

HAProxy version 2.4.22-f8e3218, released 2023/02/14
Statistics Report for pid 6218

> General process information

pid = 6218 (process #1, nproc = 1, nthread = 2)
uptime = 0d 0h26m35s
system limits: memmax = unlimited; ulimit-n = 4052
maxsock = 4052; maxconn = 2010; maxpipes = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec; bit rate = 0.000 kbps
Running tasks: 0/15; idle = 100 %

active UP backup UP
active UP, going down backup UP, going down
active DOWN, going up backup DOWN, going up
active or backup DOWN not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
External resources:
• [Primary site](#)
• [Updates \(v2.4\)](#)
• [Online manual](#)
• [Scope](#)
• [Hide 'DOWN' servers](#)
• [Refresh now](#)
• [CSV export](#)
• [JSON export \(schema\)](#)

statpage		Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				1	1	-	1	2	10	6			2 470	23 808	0	0	0					OPEN									
Backend	0	0		0	1		0	1	1	1	0	0s	2 470	23 808	0	0	0	1	0	0	0	26m35s UP		0/0	0	0		0			

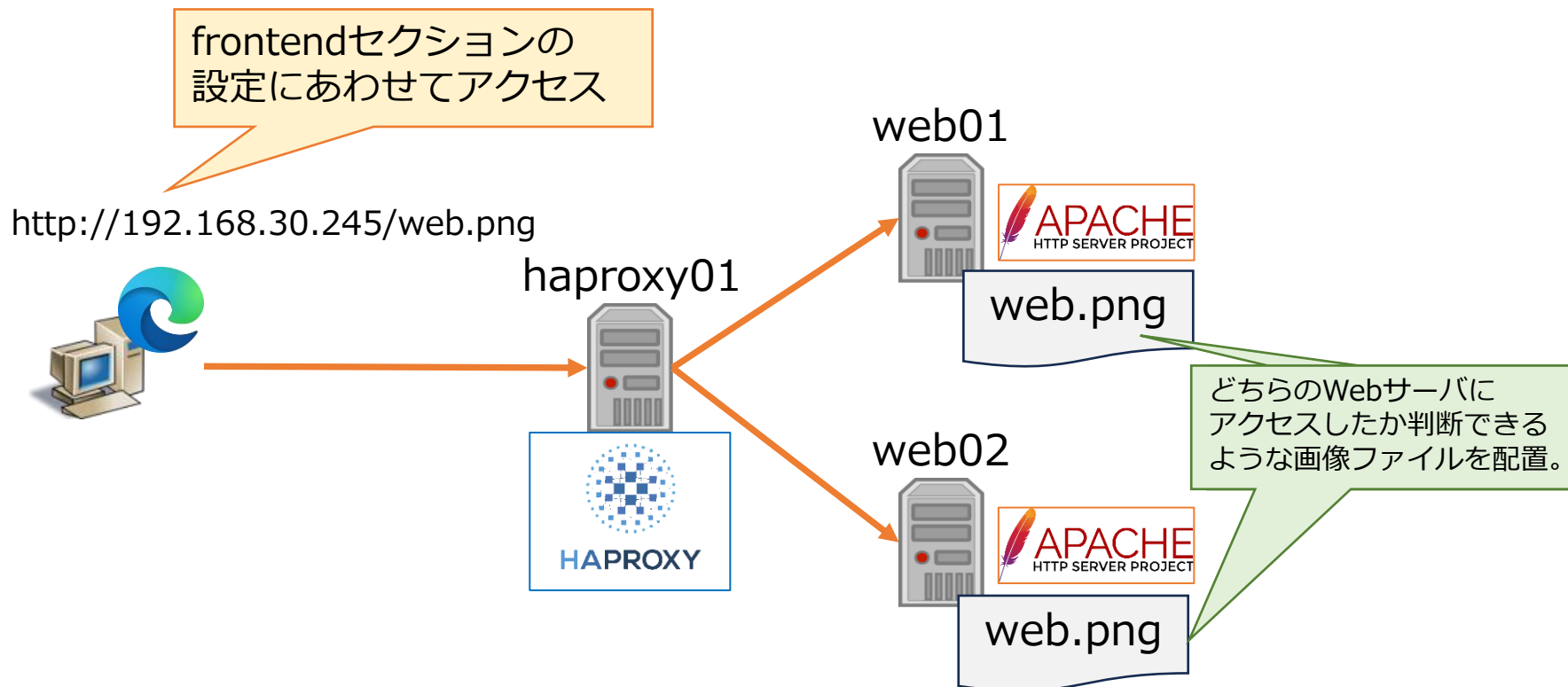
main		Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend	0			0	1	-	0	2	2 000	3			1 402	1 122	0	0	0					OPEN									

web		Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
<input type="checkbox"/>	web01	0	0	-	0	1	0	1	1000	2	2	8s	878	748	0	0	0	0	0	0	0	26m35s UP	L4OK in 1ms	1/1	Y	-	0	0	0s	-
<input type="checkbox"/>	web02	0	0	-	0	1	0	1	1000	1	1	25m40s	524	374	0	0	0	0	0	0	0	26m35s UP	L4OK in 1ms	1/1	Y	-	0	0	0s	-
	Backend	0	0		0	1	0	1	200	3	3	8s	1 402	1 122	0	0	0	0	0	0	0	26m35s UP		2/2	2	0		0	0s	

Choose the action to perform on the checked servers: Apply

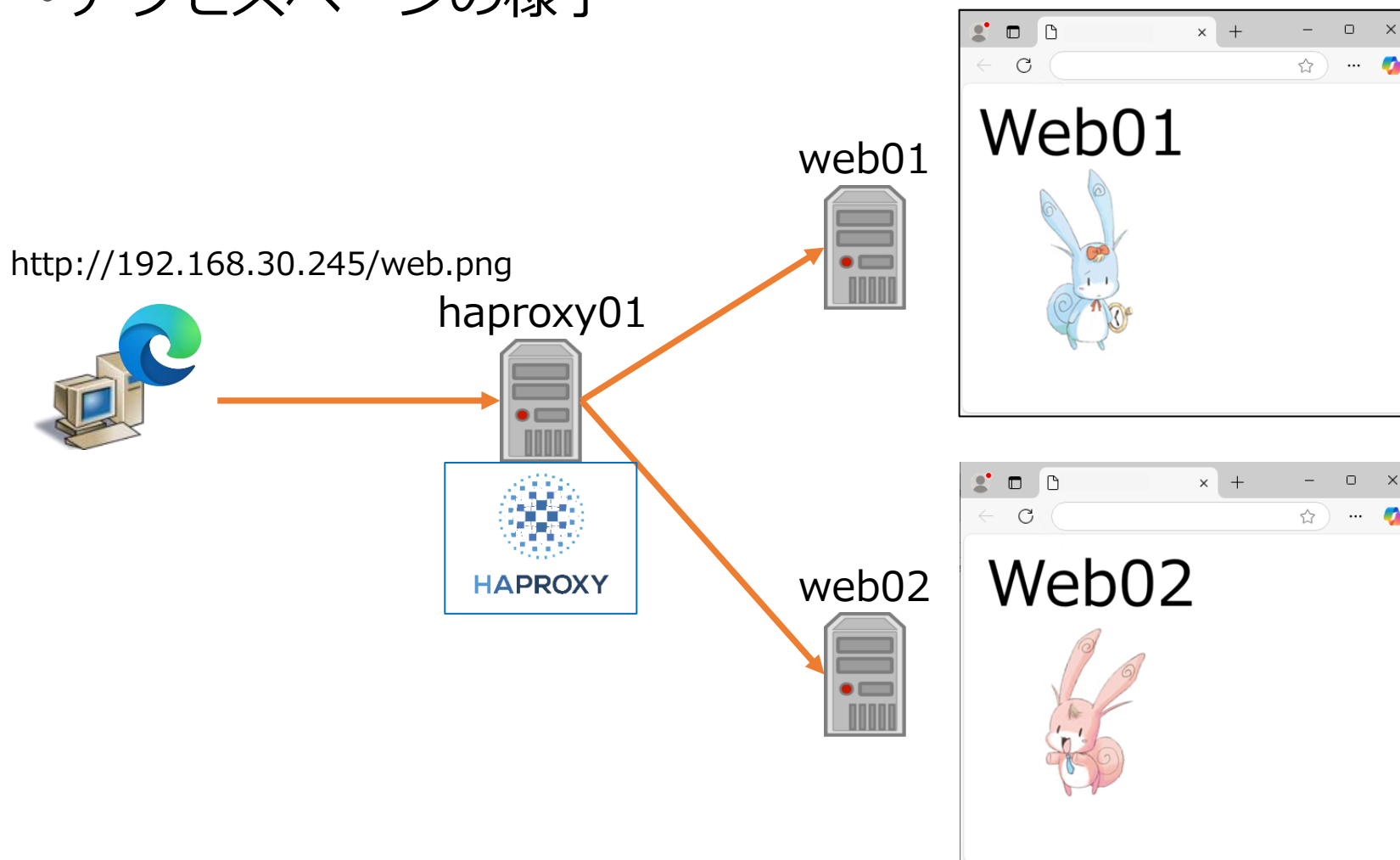
負荷分散の確認 [1/2]

- 構成図



負荷分散の確認 [2/2]

- アクセスページの様子



HAProxyの管理画面の操作 [2/4]

- web01への振分けを停止した様子

HAProxy version 2.4.22-f8e3218, released 2023/02/14

Statistics Report for pid 650500

> General process information

pid = 650500 (process #1, nbproc = 1, nbthread = 2)
uptime = 0d 0h28m34s
system limits: memmax = unlimited; ulimit-n = 4052
maxsock = 4052; maxconn = 2010; maxpipes = 0
current conns = 2; current pipes = 0/0; conn rate = 3/sec; bit rate = 1.359 kbps
Running tasks: 0/16; idle = 100 %

active UP backup UP
active UP, going down backup UP, going down
active DOWN, going up backup DOWN, going up
active or backup DOWN not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option: External resources:
• [Scope](#)
• [Hide 'DOWN' servers](#)
• [Refresh now](#)
• [CSV export](#)
• [JSON export \(schema\)](#)
• [Primary site](#)
• [Updates \(v2.4\)](#)
• [Online manual](#)

[X] Action processed successfully.

statspage

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Status	LastChk	Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn			Resp	Retr	Redis	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				3	3	-	2	2	10	22			6 840	162 056	0	0	8					OPEN								
Backend	0	0		0	0		0	0	1	0	0	0s	6 840	162 056	0	0	0	0	0	0	0	28m34s UP		0/0	0	0		0		

main

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Status	LastChk	Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn			Resp	Retr	Redis	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	3	-	0	2	2 000	11			3 234	171 603	0	0	5					OPEN								

web

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Status	LastChk	Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn			Resp	Retr	Redis	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
<input checked="" type="checkbox"/>	web01	0	0	-	0	1	0	1	1000	3	3	2m25s	1 617	102 284	0	0	0	0	0	0	0	0s MAINT								
<input type="checkbox"/>	web02	0	0	-	0	1	0	1	1000	3	3	2m26s	1 617	69 339	0	0	0	0	0	0	0	28m34s	1.40K in 0ms	1/1	Y	-	0	0	0s	-
<input type="checkbox"/>	Backend	0	0	-	0	2	0	1	200	6	6	2m25s	3 234	171 603	0	0	0	0	0	0	0	28m34s		1/1	1	0		0	0s	-

Choose the action to perform on the checked servers : Apply

MAINT に変更

HAProxyの管理画面の操作 [3/4]

- web01への振分け再開

HAProxy version 2.4.22-f8e3218, released 2023/02/14

Statistics Report for pid 650500

> General process information

pid = 650500 (process #1, nbproc = 1, nbthread = 2)
uptime = 0d 0h28m34s
system limits: memmax = unlimited; ulimit-n = 4052
maxsock = 4052; maxconn = 2010; maxpipes = 0
current conns = 2, current pipes = 0/0; conn rate = 3/sec; bit rate = 1.359 kbps
Running tasks: 0/19; idle = 100 %

Legend:
active UP (green), active UP, going down (yellow), active DOWN, going up (orange), active or backup DOWN (red), active or backup DOWN for maintenance (MAINT) (blue), active or backup SOFT STOPPED for maintenance (purple), backup UP (light blue), backup UP, going down (dark blue), backup DOWN, going up (pink), not checked (grey), Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option: Scope: []
External resources:
• Primary site
• Updates (v2.4)
• Online manual
• Hide DOWN servers
• Refresh now
• CSV export
• JSON export (schema)

[X] Action processed successfully.

statspage

	Queue			Session rate			Sessions				Bytes		Denied		Errors			Warnings		Server											
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				3	3	-	2	2	10	22			6 840	182 056	0	0	8					OPEN									
Backend	0	0		0	0		0	0	1	0	0	0s	6 840	182 056	0	0	0	0	0	0	0	28m34s UP		0/0	0	0	0	0			

main

	Queue			Session rate			Sessions				Bytes		Denied		Errors			Warnings		Server											
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				0	3	-	0	2	2 000	11			3 234	171 603	0	0	5					OPEN									

web

	Queue			Session rate			Sessions				Bytes		Denied		Errors			Warnings		Server										
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
<input checked="" type="checkbox"/>	web01	0	0	-	0	1	0	1	1000	3	3	2m25s	1 617	102 264	0	0	0	0	0	0	0	0s MAINT		1/1	Y	-	0	3	7s	-
<input type="checkbox"/>	web02	0	0	-	0	1	0	1	1000	3	3	2m26s	1 617	69 339	0	0	0	0	0	0	0	28m34s UP	L4OK in 0ms	1/1	Y	-	0	0	0s	-
<input type="checkbox"/>	Backend	0	0	-	0	2	0	1	200	8	8	2m26s	3 234	171 603	0	0	0	0	0	0	0	28m34s UP		1/1	1	0	0	0s	-	

Choose the action to perform on the checked servers: [Set state to READY] [Apply]

Set state to READY
Set state to DRAIN
Set state to MAINT
Health: disable checks
Health: enable checks
Health: force UP
Health: force NOLB
Health: force DOWN
Agent: disable checks
Agent: enable checks
Agent: force UP
Agent: force DOWN
Kill Sessions

①web01にチェック

②Set state to READY を選択

③Apply を選択

HAProxyの管理画面の操作 [4/4]

- web01への振分け再開した様子

HAProxy version 2.4.22-f8e3218, released 2023/02/14

Statistics Report for pid 650500

> General process information

pid = 650500 (process #1, nbproc = 1, nbthread = 2)
uptime = 0d 0h32m39s
system limits: memmax = unlimited; ulimit-n = 4052
maxsock = 4052; maxconn = 2010; maxpipes = 0
current conns = 2; current pipes = 0/0; conn rate = 0/sec; bit rate = 0.000 kbps
Running tasks: 0/16; idle = 100 %

active UP backup UP
active UP, going down backup UP, going down
active DOWN, going up backup DOWN, going up
active or backup DOWN not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
External resources:
• Primary site
• Updates (v2.4)
• Online manual

• Scope:
• Hide 'DOWN' servers
• Refresh now
• CSV export
• JSON export (schema)

[X] Action processed successfully.

statspage

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server										
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend	0	0	-	0	3	-	1	3	10	25			8 546	208 587	0	0	0					OPEN									
Backend	0	0	-	0	0	-	0	0	1	0			8 546	208 587	0	0	0	0	0	0	0	32m39s UP		0/0	0	0	0		0		

main

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server										
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend	0	0	-	0	4	-	1	2	2 000	20			6 468	207 051	0	0	7					OPEN									

web

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
<input type="checkbox"/>	web01	0	0	-	0	1	0	1	1000	3	3	6m30s	1 617	102 264	0	0	0	0	0	0	0	0s UP	L4OK in 0ms	1/1	Y	-	0	3	7s	-
<input type="checkbox"/>	web02	0	0	-	0	2	0	1	1000	9	9	3m19s	4 851	104 787	0	0	0	0	0	0	0	39s UP	L4OK in 0ms	1/1	Y	-	0	0	0s	-
	Backend	0	0	-	0	2	0	1	200	12	12	3m19s	6 468	207 051	0	0	0	0	0	0	0	39s UP		2/2	2	0	0	0	0s	-

Choose the action to perform on the checked servers : Apply

UPに変更

HAProxyの状態確認 [1/2]

- ログファイルを確認する(出力例)

本講演資料の設定 : /var/log/haproxy.log

```
# tail -2 /var/log/haproxy.log
Feb 21 13:00:00 localhost <local2.info> haproxy[651970]:
192.168.30.108:65032 [21/Feb/2025:13:00:00.00] web/web01 200
“GET” /web.png HTTP/1.1”
Feb 21 13:00:00 localhost <local2.info> haproxy[651970]:
192.168.30.108:65032 [21/Feb/2025:13:00:00.00] web/web01 200
“GET” /web.png HTTP/1.1”
```

【項目説明】

192.168.30.108:65032	…クライアントのIPアドレス:ポート
[21/Feb/2025:13:00:00.00]	…クライアントからのTCP接続を HAProxyが受信した時刻
web	…バックエンドサービスの識別子
web01	…受信したWebサーバの識別子
200	…HTTPステータスコード
“GET” /web.png HTTP/1.1”	…受信したHTTPリクエスト

HAProxyの状態確認 [2/2]

- socatコマンドを使って確認する(出力例)

web01の動作状況を確認する (例 : web01が起動中のとき)

```
# echo "show stat" | socat stdio /var/lib/haproxy/stats | ¥  
awk -F, '$2=="web01"{print $18}'  
UP
```

HAProxyを試してみよう！～まとめ～

以下の流れで試してみました

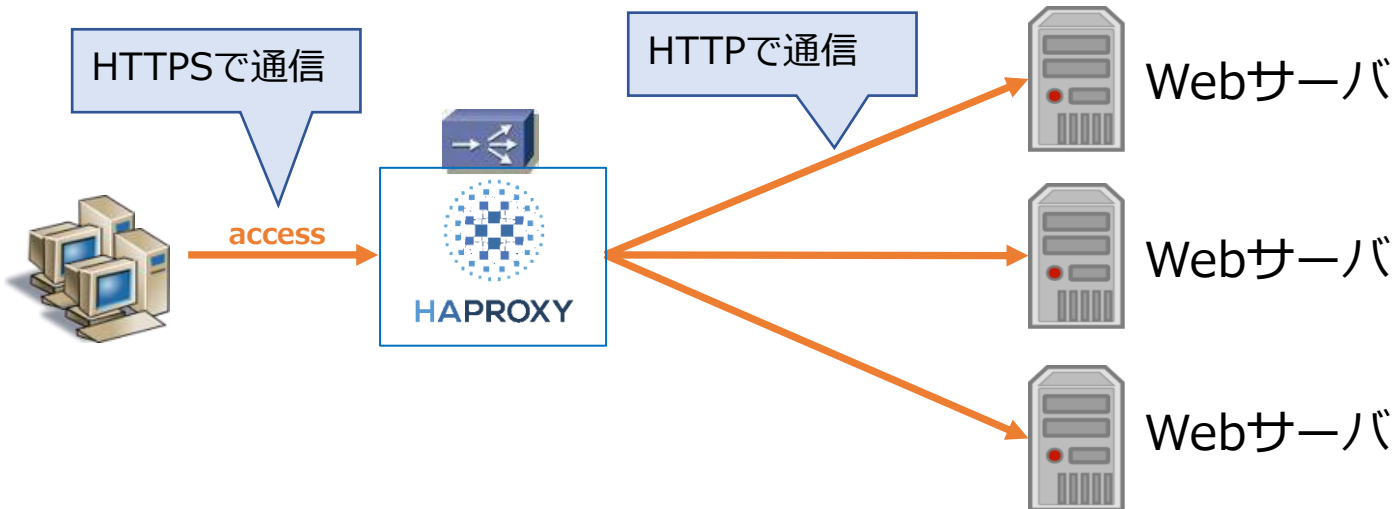
1. サーバを用意する
2. HAProxyをインストールする
3. HAProxyを設定する
4. HAProxyを起動する
5. HAProxyを試してみる

HAProxyが
動かしました！



HAProxyのポテンシャル！

HAProxyの構成例① [1/3]



HAProxyの構成例① [2/3]

- HAProxyの設定ポイント(SSL関連)

```
global
  ssl-default-bind-options ssl-min-ver TLSv1.2
  ssl-default-bind-ciphers PROFILE=SYSTEM:!PSK
  ssl-default-bind-ciphersuites TLS_AES_128_GCM_SHA256
  tune.ssl.cachesize 102400
  tune.ssl.lifetime 300
  tune.ssl.default-dh-param 2048
```

SSL/TLSバージョンや
暗号スイートを指定可能

～省略～

```
frontend vs_https01
  bind *:443 ssl crt /etc/haproxy/ssl/server-crtkey.pem
```

～省略～

サーバ証明書・秘密鍵を指定

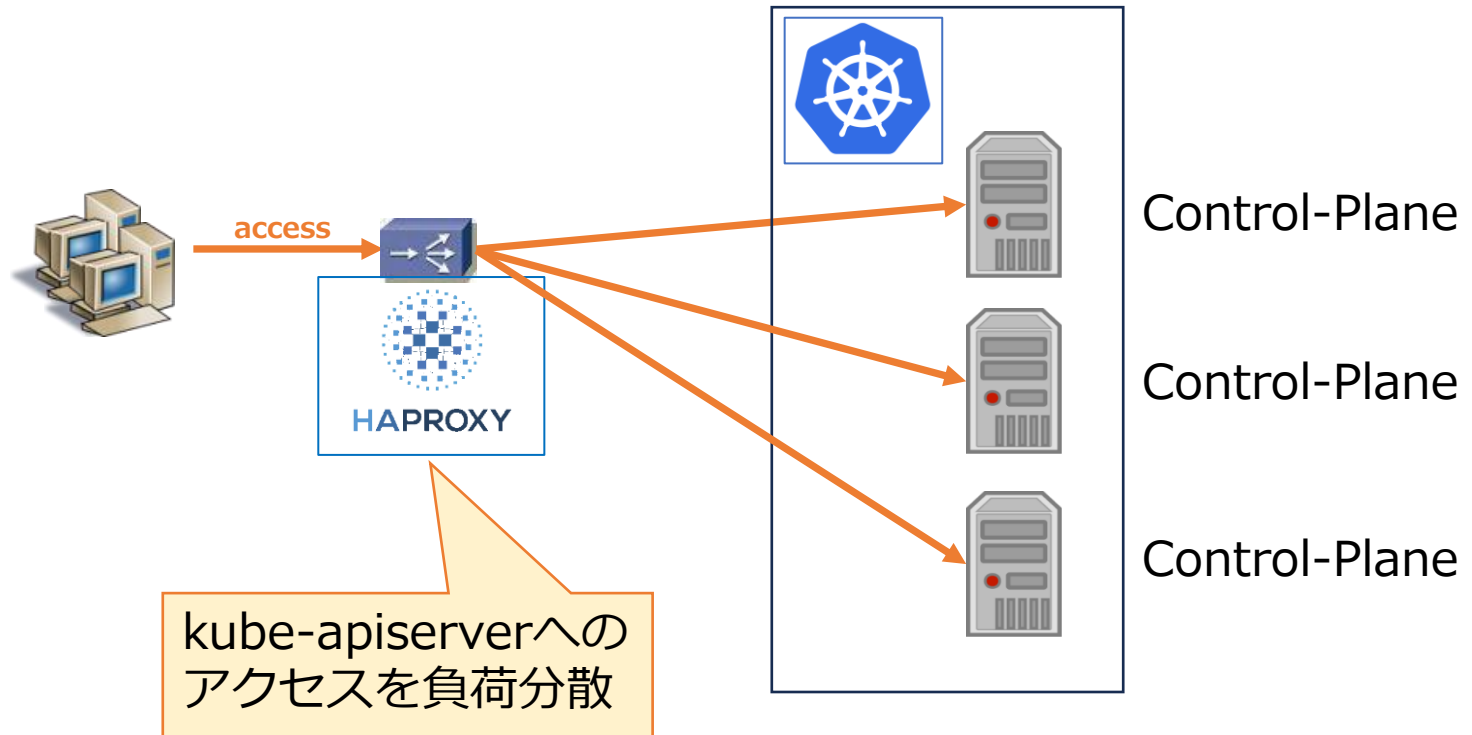
HAProxyの構成例① [3/3]

• HAProxyの設定ポイント(エラーページ)

```
defaults
  mode http
  ~省略~
  log-format "%ci:%cp [%tr] %b/%s %ST %hr %hs %Q}r %[var(txn.ostatus)]"
  errorfile 400 /etc/haproxy/errorfiles/400.http
  errorfile 500 /etc/haproxy/errorfiles/500.http
  ~省略~
frontend main
  ~省略~
  http-response allow if { status lt 400 }
  http-response return status 400 default-errorfiles if { status eq 400 }
  http-response return status 500 default-errorfiles if { status eq 500 }
  http-response set-var(txn.ostatus) status
  http-response deny
  ~省略~
```

HTTPレスポンスコードに
対応するエラーページの指定が可能

HAProxyの構成例② [1/2]



HAProxyの構成例② [2/2]

- HAProxyの設定ポイント

```
frontend mycluster_apiserver
  bind *:6443
  mode tcp
  option tcplog
  default_backend mycluster_backend


backend mycluster_backend
  mode tcp
  balance roundrobin
  option tcp-check
  server c11 192.168.1.1:6443 check
  server c12 192.168.2.2:6443 check
  server c13 192.168.3.3:6443 check
  option log-health-checks
```


tcpモードを指定

HAProxyのチューニング項目！

- チューニング項目(一例)
 - 最大接続数
 - タイムアウト値（クライアント、サーバ等々）
 - SSL/TLSバージョン、暗号スイート
 - スレッドの設定（マルチスレッド数、CPUの指定）

HAProxyConf 2025 - [Registration & Call for Papers are Open!](#)

 HAPROXY HAProxy config tutorials ▼

Search CTRL K 

Client IP preservation ▼ ▲

- Forwarded header
- X-Forwarded-For header
- Enable the Proxy Protocol

DNS resolution

HTTP redirects

HTTP rewrites

Load balancing ▼

- FastCGI
- gRPC
- HTTP
- Passive FTP
- Syslog
- TCP
- WebSocket

Network performance ▼

- Caching
- Compression
- Traffic shaping

Programs

Service reliability >

[Home](#) > [HAProxy config tutorials](#)

[HAProxy config tutorials](#)

HAProxy config tutorials

See examples of configuring the load balancer for common use cases. Please choose a topic from the navigation menu.

Core concepts

Alerts and monitoring

AuthN / authZ

Client IP preservation

DNS resolution
Add DNS nameservers to resolve hostnames.

HTTP redirects
Redirect a client to a different destination.

HTTP rewrites
Change the properties of a request or a response on the fly.

Load balancing

Network performance

Programs
Use the process manager to run external programs.

Service reliability

Session persistence
Route clients to the same backend server with session persistence.

[Feedback?](#)

HAProxyのポテンシャル！～まとめ～

1. HAProxyの構成例
2. チューニング項目
3. マニュアル紹介

HAProxyは
色々なシーンで使えそう！



～おわりに～

- 昨年に初めてHAProxyと出会い、勉強した経験をふまえてHAProxyを紹介しました
- HAProxyに関心がある方の参考になれば幸いです

今回取り扱わなかったPacemakerについて もっと知りたい方はこちらまで

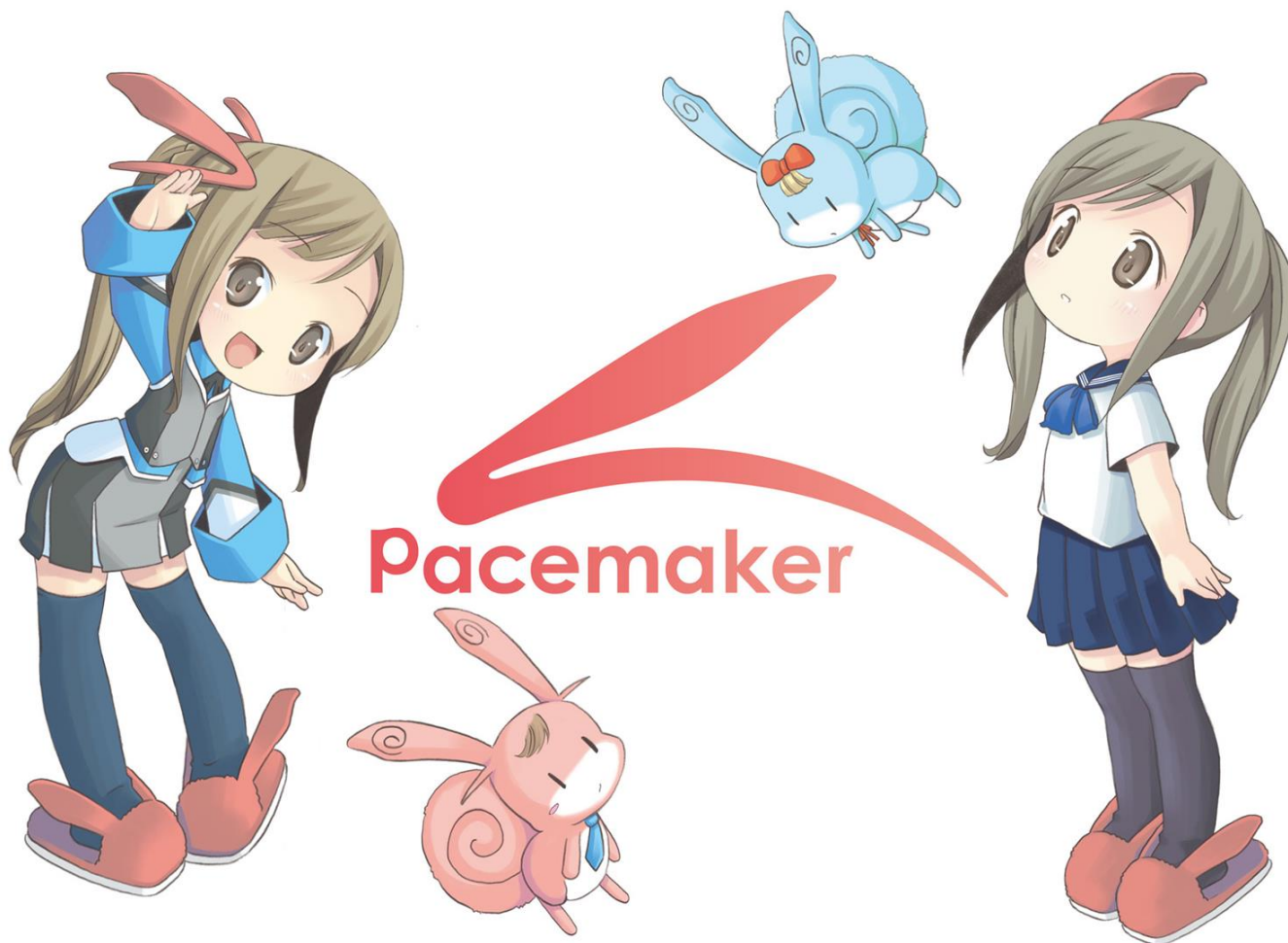
<https://linux-ha-japan.github.io/>



The screenshot shows the homepage of the Linux-HA Japan project. The browser address bar displays 'https://linux-ha-japan.github.io/'. The page features the Linux-HA Japan logo and the text 'High-Availability Clustering on Linux'. Navigation links include 'TOP', 'ダウンロード', 'メーリングリスト', 'ドキュメント', '各種コンテンツ', and '投稿記事'. The main heading is 'Linux-HA Japan プロジェクト'. Below this, there is a paragraph in Japanese describing the project's goals and a list of supported technologies: Pacemaker, Corosync, and DRBD. A notice dated 2024/05/30 mentions the restoration of content from an old website. At the bottom, there are social media links for GitHub and Twitter, and a Pacemaker logo with the text 'Linux-HA Japan Project'.



今後もLinux-HA Japanを よろしくお願いします



ご清聴ありがとうございました