

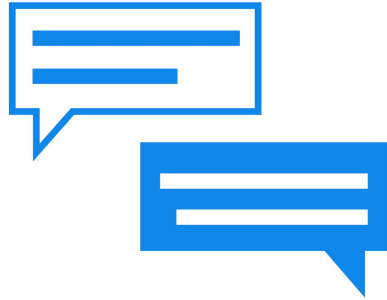
# HAProxy Data Plane API 101: Powering Interactions Across HAProxy



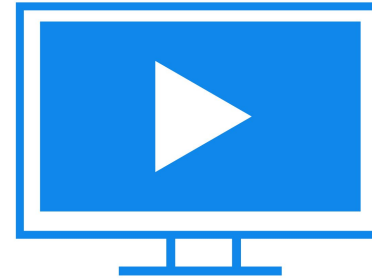
**Marko Juraga**  
Development Team Lead



# General Webinar Information



Use the Questions panel to **ask questions at any time.** They will be answered at the end of the webinar.



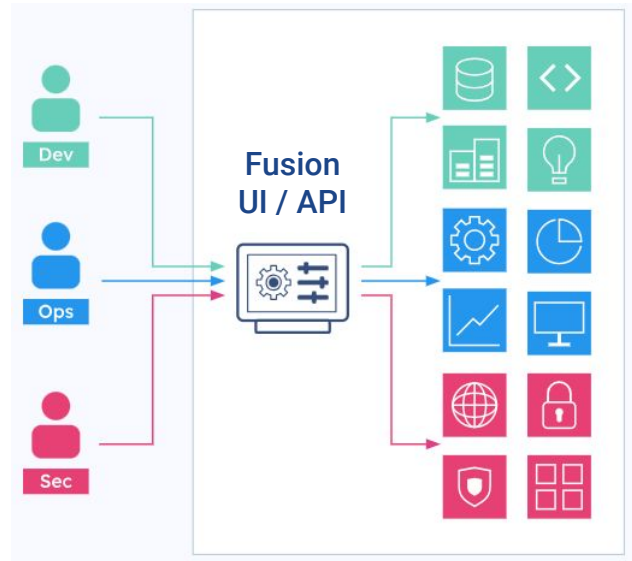
The **webinar recording** will be sent to all attendees 1-2 days after the webinar.

# Introducing HAProxy and Data Plane API

- HTTP Rest API for interacting with HAProxy process.
- It's a vital part of the [HAProxy One](#) platform.
  - HAProxy One is the world's fastest application delivery and security platform.
  - Platform consists of a flexible data plane ([HAProxy Enterprise](#) and [HAProxy ALOHA](#)), a scalable control plane ([HAProxy Fusion](#)) and a secure edge network ([HAProxy Edge](#)).
- Where does the Data Plane API fit here?
  - Included in our HAProxy ALOHA appliance, provided as a package with HAProxy Enterprise and in all the docker images to open an API access to those products.
  - HAProxy Fusion is the control plane that communicates with the Data Plane API to control and interact with each HAProxy in a fleet.

# HAProxy Fusion

- HAProxy Fusion is a an API-first control plane that also provides a modern UI.
- It makes it easier to **manage, monitor, and automate HAProxy Enterprise** deployments at scale, leveraging the flexibility of Data Plane API and the power of our high-performance control plane.
- This webinar will show some of what's possible with direct HAProxy automation, but everything we cover is easier at scale with HAProxy Fusion.



# HAProxy Data Plane API

- The HAProxy Data Plane API is a service that lets you configure the HAProxy load balancer using HTTP, RESTful commands, enabling dynamically-generated configurations.
- It runs as a standalone service or a program.
- The latest community version is 3.0, and the latest enterprise version is 2.9.
- Follows HAProxy versions and release cycle.
- The API is described using OpenAPI, you can find the API documentation here: <https://www.haproxy.com/documentation/dataplaneapi/>
- Provides a structured way to interact with HAProxy using well know formats and protocols.

# HAProxy Data Plane API - Where to find it?

## Community

- Github project [here](#)
- Downloadable packages, archives and binaries in releases pages
- Build from source running `make` (prerequisite: install go 1.23), binary gets built in build/ directory
- [Docker images](#): Data Plane API not running by default

```
$ docker cp dataplaneapi.yaml haproxy:/etc/haproxy/dataplaneapi.yaml
$ docker exec -d haproxy sh -c dataplaneapi -f
/etc/haproxy/dataplaneapi.yaml
```

## Enterprise

- Packages in our repo:  
hapee-extras-dataplaneapi29
- Docker images:  
hapee-registry.haproxy.com/haproxy-enterprise
- Data Plane API is running by default on  
0.0.0.0:5555 under s6-supervisor

# HAProxy Data Plane API - Let's start!

- Installing HAProxy and Data Plane API
- Configure and start HAProxy
- Configure and start HAProxy Data Plane API

- Make a first call:

```
$ curl -s -u admin:admin http://localhost:5555/v3/info | jq
```

```
{  
  "api": {  
    "build_date": "2024-10-15T10:52:55.000Z",  
    "version": "v3.0.3 f1d4189a"  
  },  
  "system": {}  
}
```

- Let's follow the steps in a popular blog post but with Data Plane API driving the changes:  
<https://www.haproxy.com/blog/haproxy-configuration-basics-load-balance-your-servers>

# HAProxy Data Plane API - Configuration 1

```
# _md5hash=053c8606b5d70a6fcbcab4317af259f
# _version=10
# Dataplaneapi managed File
# changing file directly can cause a conflict if dataplaneapi is running
```

```
frontend myfrontend from defaults
  bind 127.0.0.1:80 name 127.0.0.1:80
  bind 127.0.0.1:81 name 127.0.0.1:81
  use_backend special if { dst_port 81 }
  default_backend myservers
```

```
http://localhost:5555/v3/services/haproxy/configuration/frontends?full_section=true
[
  {
    "default_backend": "myservers",
    "from": "defaults",
    "name": "myfrontend",
    "backend_switching_rule_list": [
      {
        "cond": "if",
        "cond_test": "{ dst_port 81 }",
        "name": "special"
      }
    ],
    "binds": {
      "127.0.0.1:80": {
        "name": "127.0.0.1:80",
        "address": "127.0.0.1",
        "port": 80
      },
      "127.0.0.1:81": {
        "name": "127.0.0.1:81",
        "address": "127.0.0.1",
        "port": 81
      }
    }
  }
]
```



# HAProxy Data Plane API - Configuration 2

```
backend myservers from defaults
  server server1 127.0.0.1:8000
  server server2 127.0.0.1:8001
```

```
backend special from defaults
  server server3 127.0.0.1:8002
```

```
http://localhost:5555/v3/services/haproxy/configuration/backends?full_section=true
[
  {
    "from": "defaults",
    "name": "myservers",
    "servers": {
      "server1": {
        "address": "127.0.0.1",
        "name": "server1",
        "port": 8000
      },
      "server2": {
        "address": "127.0.0.1",
        "name": "server2",
        "port": 8001
      }
    }
  },
  {
    "from": "defaults",
    "name": "special",
    "servers": {
      "server3": {
        "address": "127.0.0.1",
        "name": "server3",
        "port": 8002
      }
    }
  }
]
]
```

# HAProxy Data Plane API - Stats HAProxy

HAProxy version 3.0.5-1ppa1~jammy, released 2024/09/21

Statistics Report for pid 96944

> General process information

pid = 96944 (process #1, nbproc = 1, nbthread = 8)  
 uptime = 0d 0h0m22s, warnings = 0  
 system limits: memmax = unlimited, ulimit-n = 2074  
 maxsock = 2074, maxconn = 1000, reached = 0, maxpipes = 0  
 current conn = 1, current pipes = 0/0, conn rate = 1/sec, bit rate = 0.000 kbps  
 Running tasks: 0/31 (0 nice), tile = 100 %

active UP  
 active UP, going down  
 active DOWN, going up  
 active or backup DOWN  
 active or backup SOFT STOPPED for maintenance  
 Note: "NO LB"/"DRAIN" = UP with load-balancing disabled.

backup UP  
 backup UP, going down  
 backup DOWN, going up  
 not checked

Display option:

- Scope:
- Hide DOWN servers
- Disable refresh
- Refresh page
- CSV export
- JSON export (schema)

External resources:

- Cherry site
- Updates (v3.0)
- Online manual

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

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

myservers		Queue		Session rate			Sessions				LbTot		Bytes		Denied		Errors		Warnings		Status		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total		Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis		LastChk	Wght	Act	Bck	Chk	Dwn	Dvntme	Thrtle	
<input type="checkbox"/>	server1	0	0	-	0	0	0	0		0	0	?	0	0	0	0	0	0	0	0	0	no check		1/1	Y	-					
<input type="checkbox"/>	server2	0	0	-	0	0	0	0		0	0	?	0	0	0	0	0	0	0	0	0	no check		1/1	Y	-					
<input type="checkbox"/>	Backend	0	0	-	0	0	0	0	100	0	0	?	0	0	0	0	0	0	0	0	0	23s UP		2/2	2	0		0	0s		

Choose the action to perform on the checked servers:

special		Queue		Session rate			Sessions				LbTot		Bytes		Denied		Errors		Warnings		Status		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total		Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis		LastChk	Wght	Act	Bck	Chk	Dwn	Dvntme	Thrtle	
<input type="checkbox"/>	server3	0	0	-	0	0	0	0		0	0	?	0	0	0	0	0	0	0	0	0	no check		1/1	Y	-					
<input type="checkbox"/>	Backend	0	0	-	0	0	0	0	100	0	0	?	0	0	0	0	0	0	0	0	0	23s UP		1/1	1	0		0	0s		

Choose the action to perform on the checked servers:

# HAProxy Data Plane API - Stats Data Plane API

```
{ "runtimeAPI": "/var/run/haproxy/master.sock@1", "stats": [{"name": "myfrontend", "stats": {"bin": 0, "bout": 0, "cli_abrt": 0, "comp_byp": 0, "comp_in": 0, "comp_out": 0, "comp_rsp": 0, "conn_rate": 0, "conn_rate_max": 0, "conn_tot": 0, "dcon": 0, "dreq": 0, "dresp": 0, "dses": 0, "ereq": 0, "hrsp_1xx": 0, "hrsp_2xx": 0, "hrsp_3xx": 0, "hrsp_4xx": 0, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 2, "intercepted": 0, "lbtot": 0, "mode": "http", "pid": 1, "rate": 0, "rate_lim": 0, "rate_max": 0, "req_rate": 0, "req_rate_max": 0, "req_tot": 0, "scur": 0, "sid": 0, "slim": 1000, "smax": 5, "srv_abrt": 0, "status": "OPEN", "stot": 0}, "type": "frontend"}, {"name": "stats", "stats": {"bin": 595, "bout": 29234, "cli_abrt": 5, "comp_byp": 0, "comp_in": 0, "comp_out": 0, "comp_rsp": 0, "conn_rate": 0, "conn_rate_max": 2, "conn_tot": 2, "dcon": 0, "dreq": 0, "dresp": 0, "dses": 0, "ereq": 1, "hrsp_1xx": 0, "hrsp_2xx": 1, "hrsp_3xx": 0, "hrsp_4xx": 1, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 3, "intercepted": 1, "lbtot": 5, "mode": "http", "pid": 1, "rate": 0, "rate_lim": 0, "rate_max": 2, "req_rate": 0, "req_rate_max": 1, "req_tot": 2, "scur": 0, "sid": 0, "slim": 1000, "smax": 5, "srv_abrt": 5, "status": "OPEN", "stot": 2}, "type": "frontend"}, {"backend_name": "myservers", "name": "server1", "stats": {"act": 1, "addr": "127.0.0.1:8000", "bck": 0, "bin": 0, "bout": 0, "cli_abrt": 0, "ctime": 0, "dresp": 0, "econ": 0, "eresp": 0, "hrsp_1xx": 0, "hrsp_2xx": 0, "hrsp_3xx": 0, "hrsp_4xx": 0, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 4, "lastchg": 61, "lastsess": -1, "lbtot": 0, "mode": "http", "pid": 1, "qcur": 0, "qmax": 0, "qtime": 0, "rate": 0, "rate_max": 0, "req_tot": 0, "rtime": 0, "scur": 0, "sid": 1, "smax": 0, "srv_abrt": 0, "status": "no check", "stot": 0, "ttime": 0, "weight": 1, "wredis": 0, "wretr": 0}, "type": "server"}, {"backend_name": "myservers", "name": "server2", "stats": {"act": 1, "addr": "127.0.0.1:8001", "bck": 0, "bin": 0, "bout": 0, "cli_abrt": 0, "ctime": 0, "dresp": 0, "econ": 0, "eresp": 0, "hrsp_1xx": 0, "hrsp_2xx": 0, "hrsp_3xx": 0, "hrsp_4xx": 0, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 4, "lastchg": 61, "lastsess": -1, "lbtot": 0, "mode": "http", "pid": 1, "qcur": 0, "qmax": 0, "qtime": 0, "rate": 0, "rate_max": 0, "req_tot": 0, "rtime": 0, "scur": 0, "sid": 2, "smax": 0, "srv_abrt": 0, "status": "no check", "stot": 0, "ttime": 0, "weight": 1, "wredis": 0, "wretr": 0}, "type": "server"}, {"act": 2, "algo": "roundrobin", "bck": 0, "bin": 0, "bout": 0, "chkdown": 0, "cli_abrt": 0, "comp_byp": 0, "comp_in": 0, "comp_out": 0, "comp_rsp": 0, "ctime": 0, "downtime": 0, "dreq": 0, "dresp": 0, "econ": 0, "eresp": 0, "hrsp_1xx": 0, "hrsp_2xx": 0, "hrsp_3xx": 0, "hrsp_4xx": 0, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 4, "lastchg": 61, "lastsess": -1, "lbtot": 0, "mode": "http", "pid": 1, "qcur": 0, "qmax": 0, "qtime": 0, "rate": 0, "rate_max": 0, "req_tot": 0, "rtime": 0, "scur": 0, "sid": 0, "slim": 100, "smax": 0, "srv_abrt": 0, "status": "UP", "stot": 0, "ttime": 0, "weight": 2, "wredis": 0, "wretr": 0}, "type": "backend"}, {"backend_name": "special", "name": "server3", "stats": {"act": 1, "addr": "127.0.0.1:8002", "bck": 0, "bin": 0, "bout": 0, "cli_abrt": 0, "ctime": 0, "dresp": 0, "econ": 0, "eresp": 0, "hrsp_1xx": 0, "hrsp_2xx": 0, "hrsp_3xx": 0, "hrsp_4xx": 0, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 5, "lastchg": 61, "lastsess": -1, "lbtot": 0, "mode": "http", "pid": 1, "qcur": 0, "qmax": 0, "qtime": 0, "rate": 0, "rate_max": 0, "req_tot": 0, "rtime": 0, "scur": 0, "sid": 1, "smax": 0, "srv_abrt": 0, "status": "no check", "stot": 0, "ttime": 0, "weight": 1, "wredis": 0, "wretr": 0}, "type": "server"}, {"act": 1, "algo": "roundrobin", "bck": 0, "bin": 0, "bout": 0, "chkdown": 0, "cli_abrt": 0, "comp_byp": 0, "comp_in": 0, "comp_out": 0, "comp_rsp": 0, "ctime": 0, "downtime": 0, "dreq": 0, "dresp": 0, "econ": 0, "eresp": 0, "hrsp_1xx": 0, "hrsp_2xx": 0, "hrsp_3xx": 0, "hrsp_4xx": 0, "hrsp_5xx": 0, "hrsp_other": 0, "iid": 5, "lastchg": 61, "lastsess": -1, "lbtot": 0, "mode": "http", "pid": 1, "qcur": 0, "qmax": 0, "qtime": 0, "rate": 0, "rate_max": 0, "req_tot": 0, "rtime": 0, "scur": 0, "sid": 0, "slim": 100, "smax": 0, "srv_abrt": 0, "status": "UP", "stot": 0, "ttime": 0, "weight": 1, "wredis": 0, "wretr": 0}, "type": "backend"}]}
```

# HAProxy Data Plane API - What's new in 3.0?

- This release makes key improvements to the API:
  - Users can now fetch a parent resource with all children embedded.
  - We've added a PUT operation on index-based resources.
  - Timeouts are now handled with your preferred time suffix.
  - The API's state data is removed from the configuration file. User-defined settings are now the configuration's sole focus.
- Support has been added for every new keyword introduced in HAProxy 3.0.
- API has been moved to `/v3` introducing a lot of breaking changes, more here: <https://www.haproxy.com/blog/announcing-haproxy-data-plane-api-3-0>

# HAProxy Data Plane API - Enterprise

- As we've covered here you can do a lot with our Data Plane API community version, but with enterprise version we have a few extra features:
  - Provides support for configuring all the HAProxy Enterprise modules such as Bot Management, CAPTCHA, WAF, SAML and many more
  - Provides extra endpoints (facts, health, summary, VRRP)
  - Structured configuration
  - Works with HAProxy ALOHA appliance
  - Enables you to fetch, parse and forward HAProxy logs
  - Works with HAProxy Fusion for centralized management, monitoring, and automation.

# Conclusion

- After this webinar you should be able to install and use HAProxy Data Plane API to **configure** your HAProxy instance, **fetch key statistics** from your HAProxy process and explore it to configure your **dynamic use case**.
- Once you master this, if you have a cluster of HAProxys running, migrating your scripts and configurations to **HAProxy Fusion** should be a piece of cake.
- HAProxy Fusion shares API philosophy of the Data Plane API so it's **easy to migrate to it**.
- In addition to this, it covers your fleet of HAProxys, allows you to cluster them in **multiple clusters** and **cluster groups**, aggregates their statistics and logs and all of that through a simple centralized API and modern UI.

# Questions & Answers

The background is a dark blue gradient. On the right side, there are several overlapping geometric shapes, including squares and rectangles, in various shades of blue and cyan. A prominent cyan line runs diagonally across these shapes. At the bottom of the slide, there is a horizontal grid of small, light blue dots.

# Appendix 1 - Data Plane API configuration

```
config_version: 2
name: mj-laptop
dataplaneapi:
  host: 0.0.0.0
  port: 5555
  advertised:
    api_address: ""
    api_port: 0
  scheme:
  - http
  transaction:
    transaction_dir: /etc/haproxy/transactions
    backups_number: 10
    backups_dir: /etc/haproxy/backups
  resources:
    maps_dir: /etc/haproxy/storage/maps
    ssl_certs_dir: /etc/haproxy/storage/ssl
    general_storage_dir: /etc/haproxy/storage/general
    dataplane_storage_dir: /etc/haproxy/storage/dataplaneapi
    spoe_dir: /etc/haproxy/storage/spoe
    spoe_transaction_dir: /etc/haproxy/storage/spoe/transactions
  user:
  - name: admin
    insecure: true
    password: admin
haproxy:
  config_file: /etc/haproxy/haproxy.cfg
  haproxy_bin: /usr/sbin/haproxy
  master_runtime: /var/run/haproxy/master.sock
  master_worker_mode: true
  reload:
    reload_strategy: systemd
log_targets:
- log_to: stdout
  log_level: debug
  log_format: text
  log_types:
  - app
  - access
```



# Appendix 2 - Starting HAProxy Configuration

```
global
  daemon
  master-worker
  maxconn 1000
  stats socket /var/run/haproxy/haproxy.sock
  stats timeout 10m
```

```
defaults defaults
  mode http
  timeout http-request 10s
  timeout connect 5s
  timeout client 10s
  timeout server 10s
```

# Appendix 3 - Final HAProxy Configuration

```
global
  daemon
  master-worker
  maxconn 1000
  stats socket /var/run/haproxy/haproxy.sock
  stats timeout 10m

defaults defaults
  mode http
  timeout http-request 10s
  timeout connect 5s
  timeout client 10s
  timeout server 10s

frontend myfrontend from defaults
  bind 127.0.0.1:81 name 127.0.0.1:81
  bind 127.0.0.1:80 name 127.0.0.1:80
  use_backend special if { dst_port 81 }
  default_backend myservers

frontend stats from defaults
  mode http
  bind 127.0.0.1:8404 name stats_bind
  stats enable
  stats refresh 10000
  stats uri /stats
  stats admin if LOCALHOST

backend myservers from defaults
  balance leastconn
  server server1 127.0.0.1:8000
  server server2 127.0.0.1:8001
  server server3 127.0.0.1:8002

backend special from defaults
  server server3 127.0.0.1:8002
```

# Appendix 4 - Frontend json files

```
$ cat frondend1.json
```

```
{
  "from": "defaults",
  "name": "myfrontend",
  "binds": {
    "127.0.0.1:80": {
      "name": "127.0.0.1:80",
      "address": "127.0.0.1",
      "port": 80
    }
  }
}
```

```
$ cat frondend2.json
```

```
{
  "from": "defaults",
  "name": "myfrontend",
  "default_backend": "myservers",
  "binds": {
    "127.0.0.1:80": {
      "name": "127.0.0.1:80",
      "address": "127.0.0.1",
      "port": 80
    }
  }
}
```

```
$ cat frondend_put.json
```

```
{
  "from": "defaults",
  "name": "myfrontend",
  "default_backend": "myservers",
  "backend_switching_rule_list": [
    {
      "name": "special",
      "cond": "if",
      "cond_test": "{ dst_port 81 }"
    }
  ],
  "binds": {
    "127.0.0.1:80": {
      "name": "127.0.0.1:80",
      "address": "127.0.0.1",
      "port": 80
    },
    "127.0.0.1:81": {
      "name": "127.0.0.1:81",
      "address": "127.0.0.1",
      "port": 81
    }
  }
}
```

```
$ cat frondend_stats.json
```

```
{
  "name": "stats",
  "from": "defaults",
  "mode": "http",
  "binds": {
    "stats_bind": {
      "name": "stats_bind",
      "address": "127.0.0.1",
      "port": 8404
    }
  },
  "stats_options": {
    "stats_enable": true,
    "stats_uri_prefix": "/stats",
    "stats_refresh_delay": 10000,
    "stats_admin": true,
    "stats_admin_cond": "if",
    "stats_admin_cond_test":
"LOCALHOST"
  }
}
```

# Appendix 5 - Backend json files

```
$ cat backend_post.json
```

```
{
  "name": "myservers",
  "from": "defaults",
  "servers": {
    "server1":
    {
      "name": "server1",
      "address": "127.0.0.1",
      "port": 8000
    }
  }
}
```

```
$ cat backend_put.json
```

```
{
  "name": "myservers",
  "from": "defaults",
  "balance": {
    "algorithm": "leastconn"
  },
  "servers": {
    "server1":
    {
      "name": "server1",
      "address": "127.0.0.1",
      "port": 8000
    },
    "server2":
    {
      "name": "server2",
      "address": "127.0.0.1",
      "port": 8001
    },
    "server3":
    {
      "name": "server3",
      "address": "127.0.0.1",
      "port": 8002
    }
  }
}
```

```
$ cat backend_special.json
```

```
{
  "name": "special",
  "from": "defaults",
  "servers": {
    "server1":
    {
      "name": "server3",
      "address": "127.0.0.1",
      "port": 8002
    }
  }
}
```

# Appendix 6 - API calls in order of presentation

- `curl -s -H "Content-Type: application/json" -u admin:admin -XPOST`  
`"http://localhost:5555/v3/services/haproxy/configuration/frontends?full\_section=true&version=1" --data "@frontend1.json"`
- `curl -s -H "Content-Type: application/json" -u admin:admin -XPOST`  
`"http://localhost:5555/v3/services/haproxy/configuration/backends?full\_section=true&version=2" --data "@backend_post.json"`
- `curl -s -H "Content-Type: application/json" -u admin:admin -XPUT`  
`"http://localhost:5555/v3/services/haproxy/configuration/frontends/myfrontend?full\_section=true&version=3" --data "@frontend2.json"`
- `curl -s -H "Content-Type: application/json" -u admin:admin -XPUT`  
`"http://localhost:5555/v3/services/haproxy/configuration/backends/myserver?full\_section=true&version=4" --data "@backend_put.json"`
- `curl -s -H "Content-Type: application/json" -u admin:admin`  
`"http://localhost:5555/v3/services/haproxy/configuration/backends/myserver/server3?version=5" -XDELETE`
- `curl -s -H "Content-Type: application/json" -u admin:admin -XPOST`  
`"http://localhost:5555/v3/services/haproxy/configuration/backends?full\_section=true&version=6" --data "@backend_special.json"`
- `curl -s -H "Content-Type: application/json" -u admin:admin -XPUT`  
`"http://localhost:5555/v3/services/haproxy/configuration/frontends/myfrontend?full\_section=true&version=7" --data "@frontend_put.json"`
- `curl -s -H "Content-Type: application/json" -u admin:admin -XPOST`  
`"http://localhost:5555/v3/services/haproxy/configuration/frontends?full\_section=true&version=8" --data "@frontend_stats.json"`
- `curl -v -u admin:admin http://localhost:5555/v3/services/haproxy/configuration/raw`
- `curl -s -u admin:admin http://localhost:5555/v3/info`
- `curl -s -u admin:admin http://localhost:5555/v3/specification`