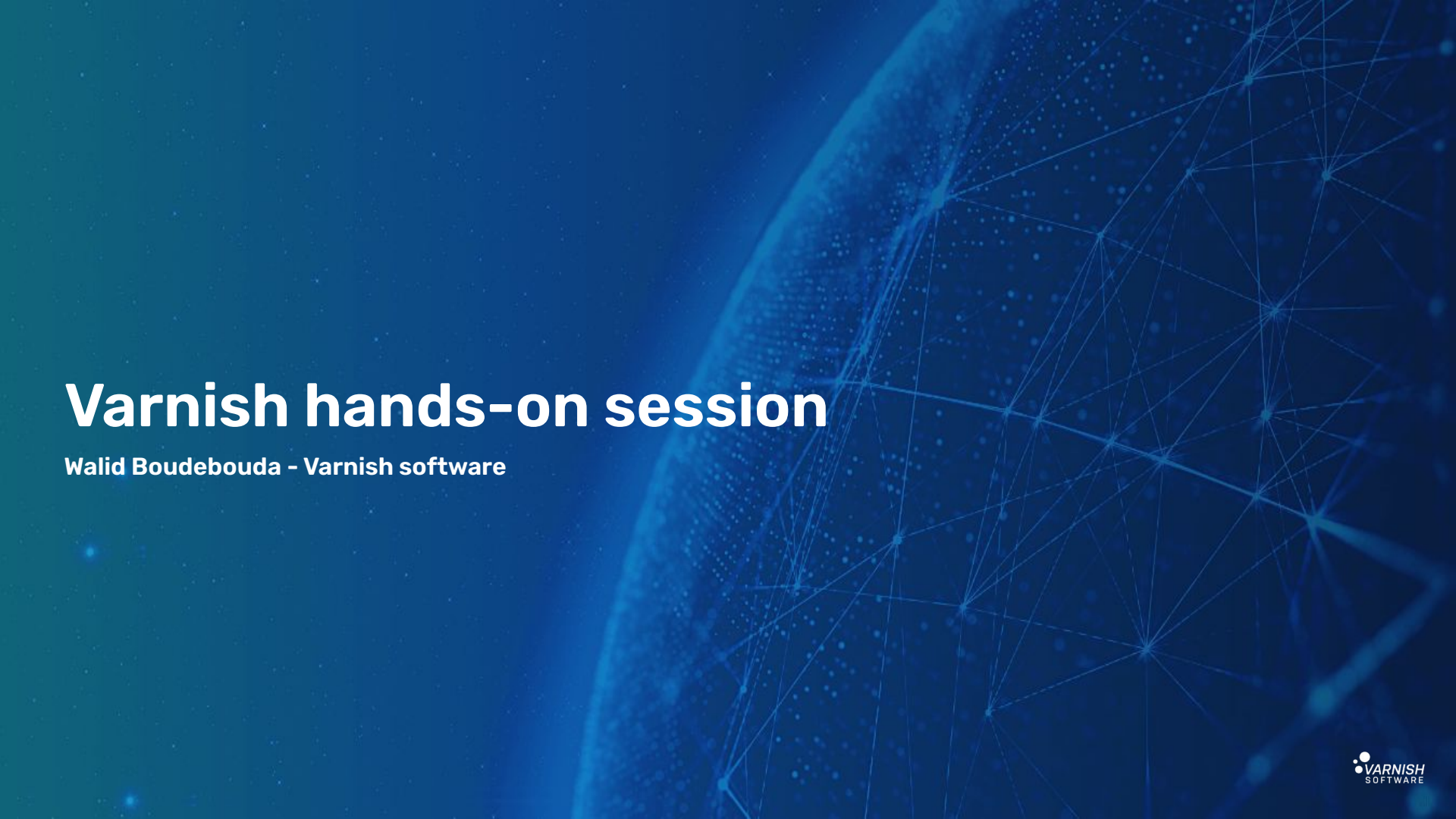
Varnish hands-on session

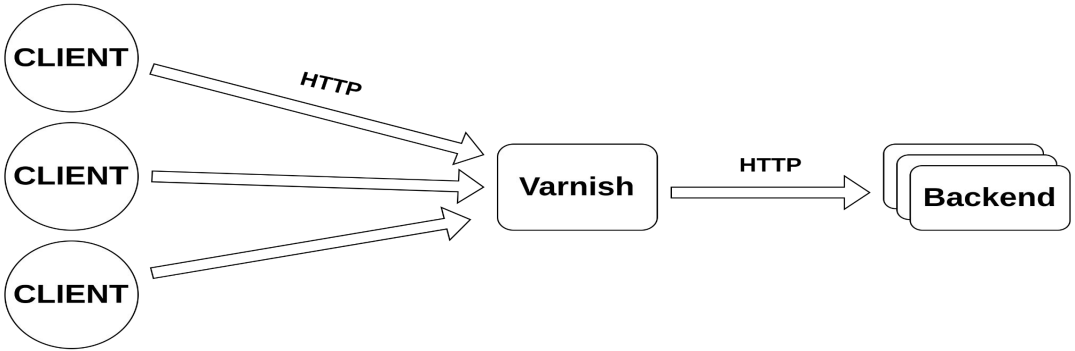
Walid Boudebouda - Varnish software

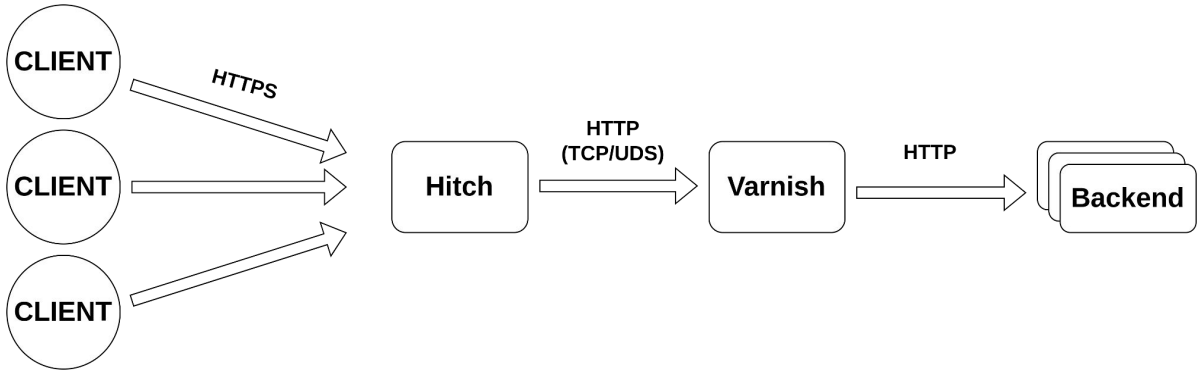


What is Varnish?

- **Reverse proxy**
- **HTTP Cache**
- **Load balancer**
- **More !**

Example architecture





Installing Varnish-cache

- **Set up repository: (other versions: https://packagecloud.io/varnishcache)**
  - **Deb**: `curl -s https://packagecloud.io/install/repositories/varnishcache/varnish75/script.deb.sh | sudo bash`
  - **Rpm**: `curl -s https://packagecloud.io/install/repositories/varnishcache/varnish75/script.rpm.sh | sudo bash`
- **Install:**
  - **apt:** `sudo apt update && sudo apt install varnish`
  - **yum:** `sudo yum install varnish`
- **Docker image:**
  - `docker pull varnish:latest`

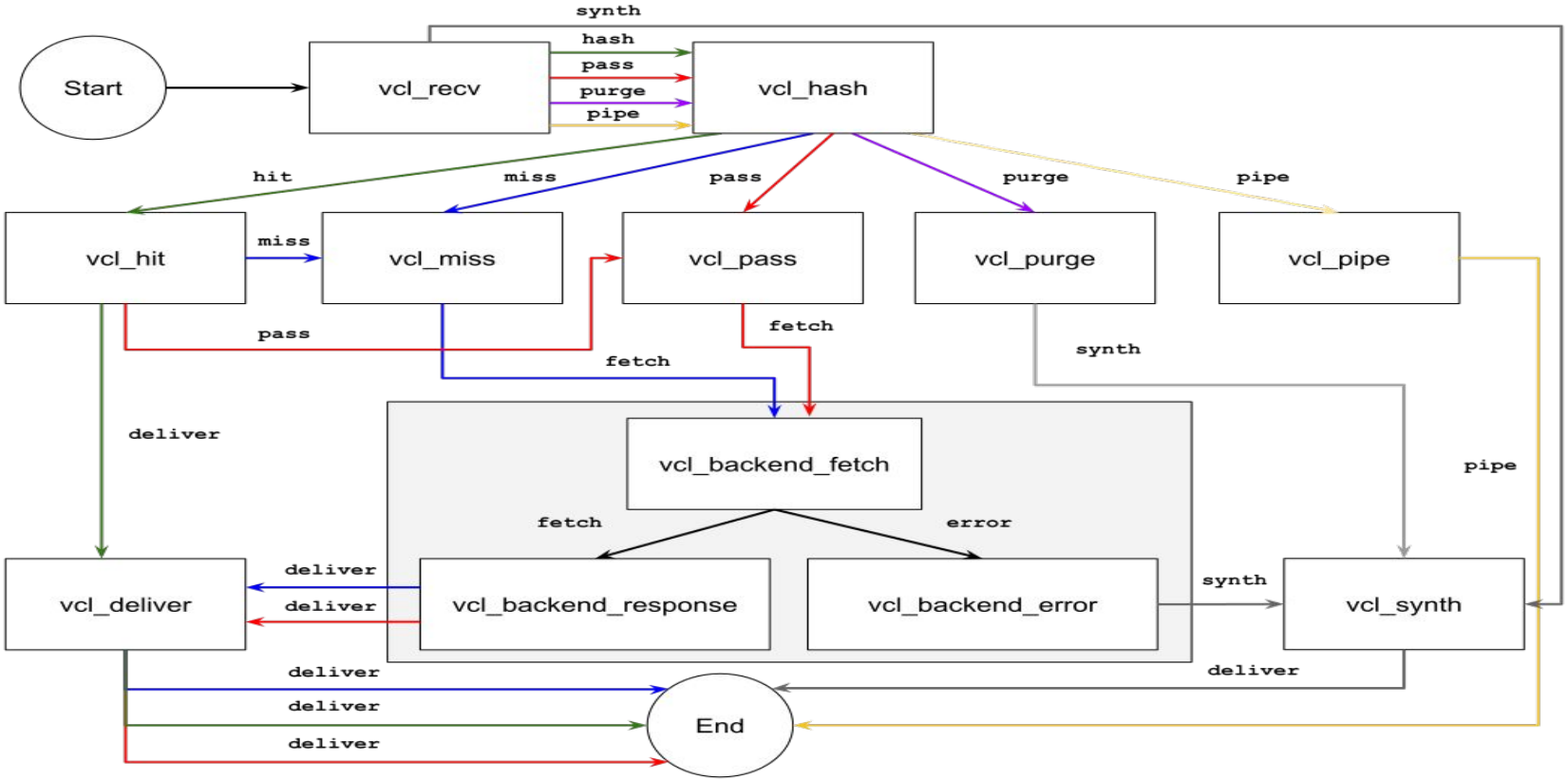
Varnish configuration

- **VCL (Varnish Config Language)**
- **Domain specific language (DSL)**
- **Control request handling, routing, caching, and other aspects**
- **See:** man vcl(7) vcl-backend(7) vcl-probe(7) vcl-step(7) vcl-var(7)

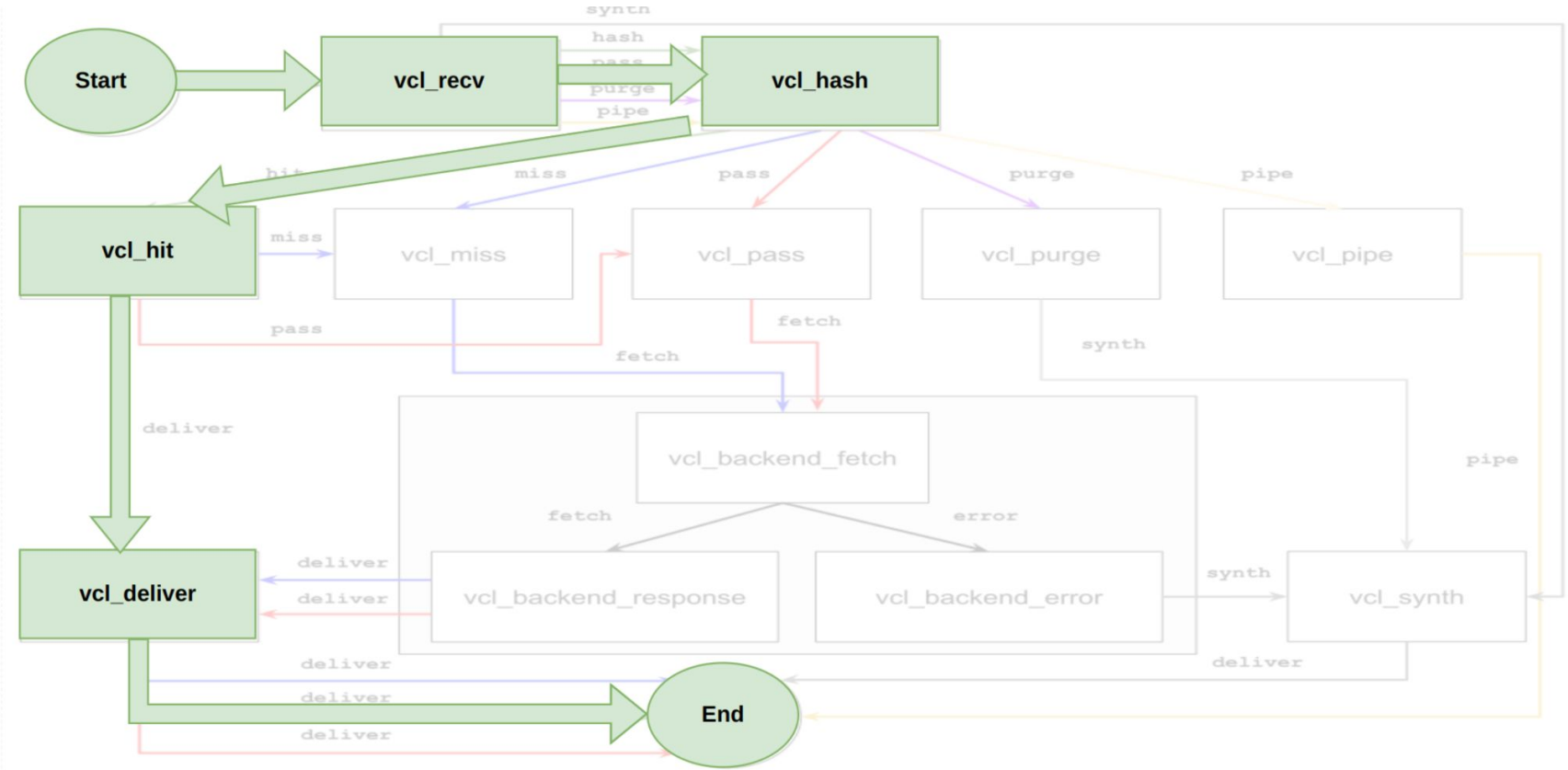
```
1  vcl 4.1;
2
3  backend default {
4          .host = "localhost";
5          .port = 8080;
6  }
7
8  sub vcl_recv {
9          if (req.http.Authorization != "SECRET_PASSWORD") {
10                 return (synth(403, "Wrong authorization !"));
11         }
12 }
```

VARNISH SOFTWARE

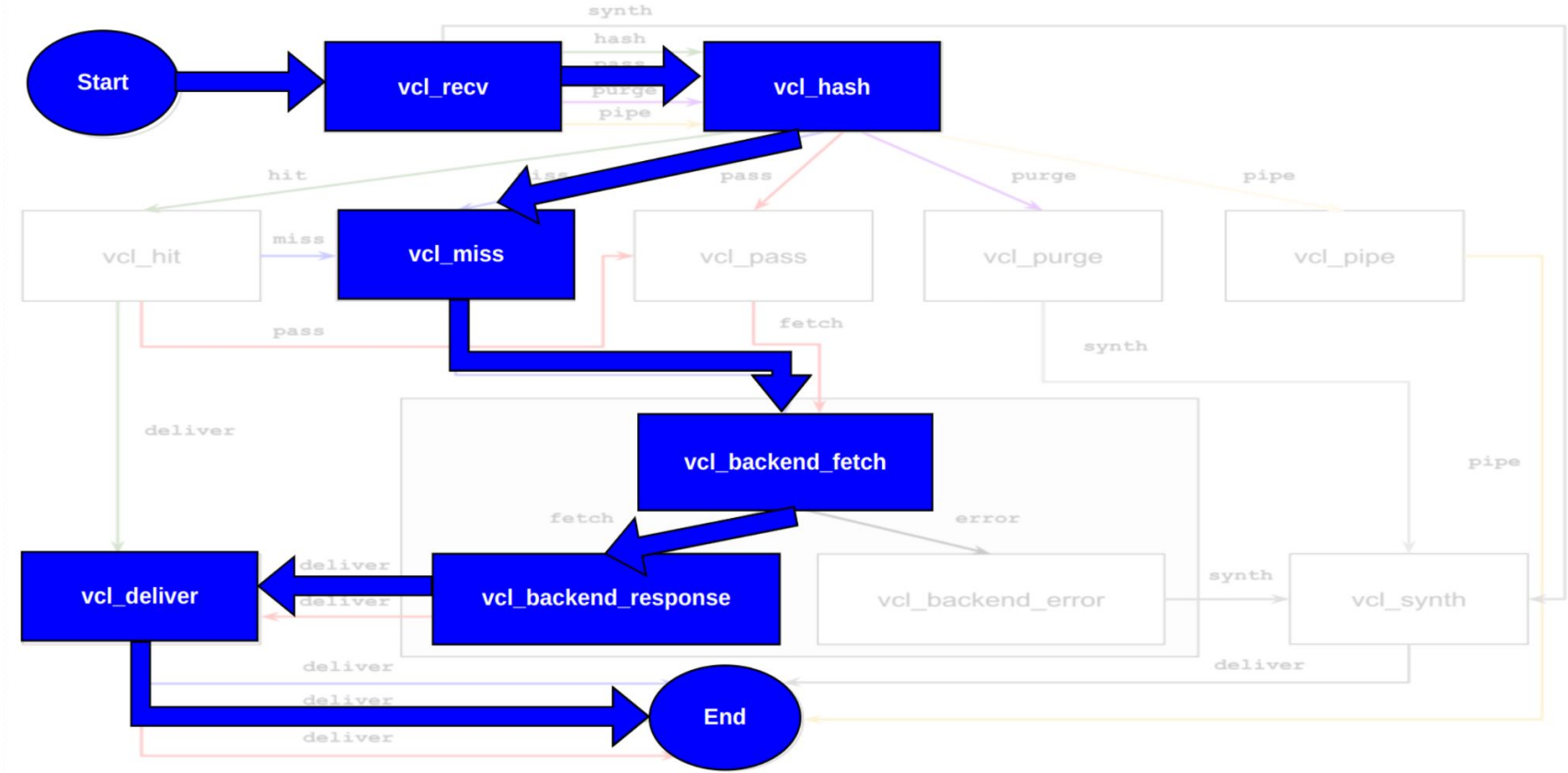
VCL state machine



VCL state machine (Hit)



VCL state machine (Miss)



Built-in VCL

- **Provides some safe-by-default logic.**
- **Runs after your VCL.**
- **Can be bypassed by using return statement in your VCL.**
- **Can be viewed with:**

  # varnishd -x builtin

```
38  sub vcl_recv {
39          call vcl_builtin_recv;
40          return (hash);
41  }
42
43  sub vcl_builtin_recv {
44          call vcl_req_host;
45          call vcl_req_method;
46          call vcl_req_authorization;
47          call vcl_req_cookie;
48  }
84  sub vcl_req_authorization {
85          if (req.http.Authorization) {
86                  # Not cacheable by default.
87                  return (pass);
88          }
89  }
90
91  sub vcl_req_cookie {
92          if (req.http.Cookie) {
93                  # Risky to cache by default.
94                  return (pass);
95          }
96  }
```

VARNISH SOFTWARE

Backend definition

- **Defined with at least one of** .host **or** .path **attribute**
- **Override global timeout parameters per backend**
- **Set max connections**
- **DNS is resolved at VCL load time and must at most resolve to 1 IPv4 and 1 IPv6 addresses**
- **See:** man vcl-backend(7)

```
1   vcl 4.1;
2
3   backend stratum1_backend1 {
4           .host = "backend1.stratum1.org";
5           .port = "8000";
6           .max_connections = 1000;
7           .probe = {
8                   .url = "/health-probe";
9                   .interval = 2 s;
10          }
11  }
12  backend stratum1_backend2 {
13          .path = "/var/run/http.sock";
14          .first_byte_timeout = 1s;
15  }
16
17  sub vcl_recv {
18          if (req.http.host  ~ "backend1") {
19                  set req.backend_hint = stratum1_backend1;
20          } else {
21                  set req.backend_hint = stratum1_backend2;
22          }
23  }
```

VARNISH SOFTWARE

Cache-policy

- **Follows the HTTP caching directives sent by the backend**
- **Can be overridden by VCL to set custom** TTL**,** grace **and** keep
- **Can send a different Cache-Control header to the clients**

```
102  sub vcl_backend_response {
103      if (beresp.status == 200) {
104       if (bereq.http.CVMFS-Mutable) {
105         set beresp.ttl = 2s;
106         set beresp.grace = 3s;
107         set beresp.keep = 1y;
108       } else if (bereq.http.CVMFS-Immutable
109             || bereq.http.CVMFS-External) {
110         set beresp.ttl = 30d;
111         set beresp.grace = 7d;
112         set beresp.keep = 1y;
113       }
114     } else {
115       set beresp.ttl = 1s;
116       set beresp.grace = 0s;
117     }
118  }
```

VARNISH SOFTWARE

VMODs

- Varnish modules
- Extend VCL capabilities
- Examples:
  - Vmod directors
  - Vmod curl
  - Vmod str
  - Others: www.varnish-cache.org/vmods/
- Write your own !

```
1   vcl 4.1;
2
3   import directors;
4
5   backend s1 {
6           .host = "s1.example.com";
7   }
8
9   backend s2 {
10          .host = "s2.example.com";
11  }
12
13  backend s3 {
14          .host = "s3.example.com";
15  }
16
17  sub vcl_init {
18          new rr = directors.round_robin();
19          rr.add_backend(s1);
20          rr.add_backend(s2);
21          rr.add_backend(s3);
22  }
23
24  sub vcl_backend_fetch {
25          set bereq.backend = rr.backend();
26  }
```

VARNISH SOFTWARE

varnishd parameters

- **Listen addresses**
- **Startup VCL file**
- **Stevedores (storage backend)**
- **Parameters: features, threads, timeouts..**
- **See:** man varnishd(7)

```
[Unit]
Description=Varnish Cache, a high-performance HTTP accelerator
After=network-online.target nss-lookup.target

[Service]
Type=forking
KillMode=mixed

# Maximum number of open files (for ulimit -n)
LimitNOFILE=131072

# Shared memory (VSM) segments are tentatively locked in memory. The
# default value for vsl_space (or shorthand varnishd -l option) is 80MB.
# There are other types of segments that would benefit from allowing
# more memory to be locked.
LimitMEMLOCK=100M

# Enable this to avoid "fork failed" on reload.
TasksMax=infinity

# Maximum size of the corefile.
LimitCORE=infinity

# A PID file makes the main process selection deterministic.
RuntimeDirectory=%N
PIDFile=%t/%N/varnishd.pid

ExecStart=/usr/sbin/varnishd \
          -a :6081 \
          -a localhost:8443,PROXY \
          -f /etc/varnish/default.vcl \
          -P %t/%N/varnishd.pid \
          -p feature=+http2 \
          -s malloc,256m
ExecReload=/usr/sbin/varnishreload

[Install]
WantedBy=multi-user.target
```

VARNISH SOFTWARE

Varnish CLI

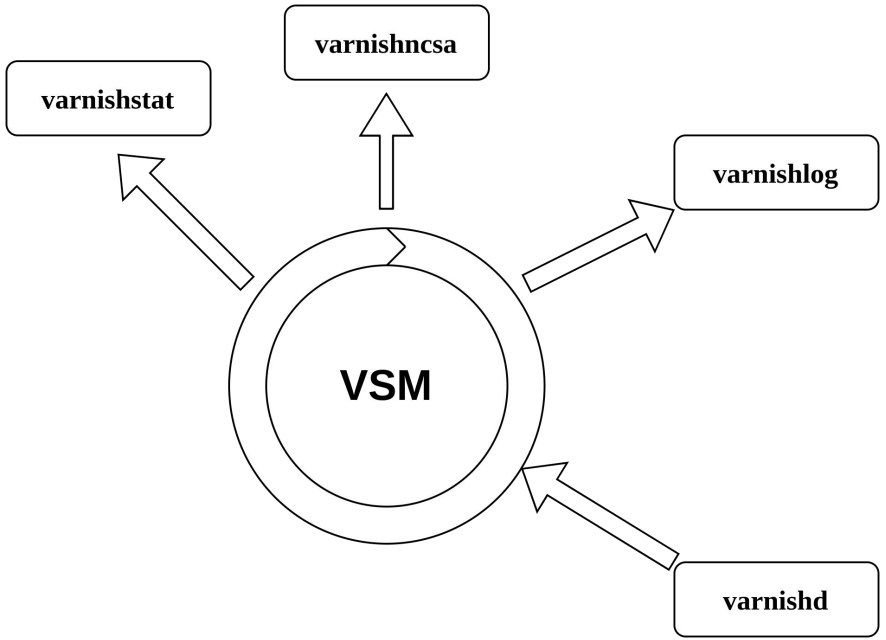
- **Control operational parameters without interrupting service**
- **Load and use VCLs**
- **Invalidate objects with bans**
- **start/stop child process**
- **Inspect/set backend health**
- **See:** man varnish-cli(7)

```
root@walid-XPS-9320:/home/walid# varnishadm
200
-----------------------------
Varnish Cache CLI 1.0
-----------------------------
Linux,6.8.0-40-generic,x86_64,-junix,-sdefault,-sdefault,-hcritbit
varnish-7.5.0 revision eef25264e5ca5f96a77129308edb83ccf84cb1b1

Type 'help' for command list.
Type 'quit' to close CLI session.

varnish> param.set feature +http2
200

varnish> param.show thread_pool_max
200
thread_pool_max
        Value is: 5000 [threads] (default)
        Minimum is: 100

        The maximum number of worker threads in each pool.

        Do not set this higher than you have to, since excess worker
        threads soak up RAM and CPU and generally just get in the way
        of getting work done.

        NB: This parameter may take quite some time to take (full)
        effect.


varnish> param.set thread_pool_max 10000
200
```

VARNISH SOFTWARE

Logging & Monitoring

- **Varnishd publishes data to a shared memory (circular buffer)**
- **Extremely fast and doesn't slow down the main process**
- **Multiple utility programs provided to exploit the data**
- **See:** man varnishlog(1) varnishncsa(1) varnishstat(1) vsl(1)



varnishlog

- Inspect requests/responses
- Very verbose multi-line output, can be limited
- Supports fine grained querying for specific requests
- Supports output rate limiting, writing to files, ..etc

```
root@walid-XPS-9320:/home/walid# varnishlog -g session -q "Timestamp:Resp[2] < 0.1 and RespStatus == 200" -d
*   << Session  >> 1
-   Begin          sess 0 HTTP/1
-   SessOpen       127.0.0.1 40650 a0 127.0.0.1 1234 1726070940.842462 18
-   Link           req 2 rxreq
-   SessClose      REM_CLOSE 0.016
-   End
**  << Request  >> 2
--  Begin          req 1 rxreq
--  Timestamp      Start: 1726070940.842507 0.000000 0.000000
--  Timestamp      Req: 1726070940.842507 0.000000 0.000000
--  VCL_use        boot
--  ReqStart       127.0.0.1 40650 a0
--  ReqMethod      GET
--  ReqURL         /
--  ReqProtocol    HTTP/1.1
--  ReqHeader      Host: localhost:1234
--  ReqHeader      User-Agent: curl/7.81.0
--  ReqHeader      Accept: */*
--  ReqHeader      X-Forwarded-For: 127.0.0.1
--  ReqHeader      Via: 1.1 walid-XPS-9320 (Varnish/7.5)
--  VCL_call       RECV
--  VCL_return     hash
--  VCL_call       HASH
--  VCL_return     lookup
--  VCL_call       MISS
--  VCL_return     fetch
--  Link           bereq 3 fetch
--  Timestamp      Fetch: 1726070940.849464 0.006956 0.006956
--  RespProtocol   HTTP/1.0
--  RespStatus     200
--  RespReason     OK
--  RespHeader     Server: SimpleHTTP/0.6 Python/3.10.12
--  RespHeader     Date: Wed, 11 Sep 2024 16:09:00 GMT
--  RespHeader     Content-type: text/html
--  RespHeader     Content-Length: 14
--  RespHeader     Last-Modified: Mon, 18 Mar 2024 12:10:38 GMT
--  RespProtocol   HTTP/1.1
--  RespHeader     X-Varnish: 2
--  RespHeader     Age: 0
--  RespHeader     Via: 1.1 walid-XPS-9320 (Varnish/7.5)
--  RespHeader     Accept-Ranges: bytes
--  VCL_call       DELIVER
--  RespHeader     default: default
--  VCL_return     deliver
--  Timestamp      Process: 1726070940.849501 0.006993 0.000037
--  Filters
--  RespHeader     Connection: keep-alive
--  Timestamp      Resp: 1726070940.857911 0.015403 0.008409
--  ReqAcct        78 0 78 311 14 325
--  End
*** << BeReq    >> 3
---  Begin          bereq 2 fetch
---  VCL_use        boot
---  Timestamp      Start: 1726070940.842612 0.000000 0.000000
---  BereqMethod    GET
---  BereqURL       /
---  BereqProtocol  HTTP/1.1
---  BereqHeader    Host: localhost:1234
---  BereqHeader    User-Agent: curl/7.81.0
---  BereqHeader    Accept: */*
---  BereqHeader    X-Forwarded-For: 127.0.0.1
---  BereqHeader    Via: 1.1 walid-XPS-9320 (Varnish/7.5)
---  BereqHeader    Accept-Encoding: gzip
---  BereqHeader    X-Varnish: 3
---  VCL_call       BACKEND_FETCH
---  VCL_return     fetch
---  Timestamp      Fetch: 1726070940.842625 0.000012 0.000012
---  Timestamp      Connected: 1726070940.842736 0.000123 0.000111
---  BackendOpen    20 default 127.0.0.1 8080 127.0.0.1 50716 connect
---  Timestamp      Bereq: 1726070940.842769 0.000157 0.000033
---  BerespProtocol HTTP/1.0
---  BerespStatus   200
---  BerespReason   OK
---  BerespHeader   Server: SimpleHTTP/0.6 Python/3.10.12
---  BerespHeader   Date: Wed, 11 Sep 2024 16:09:00 GMT
---  BerespHeader   Content-type: text/html
---  BerespHeader   Content-Length: 14
---  BerespHeader   Last-Modified: Mon, 18 Mar 2024 12:10:38 GMT
---  Timestamp      Beresp: 1726070940.849147 0.006534 0.006377
---  TTL            RFC 120 10 0 1726070941 1726070941 1726070940 0 0 cacheable
---  VCL_call       BACKEND_RESPONSE
---  VCL_return     deliver
---  Timestamp      Process: 1726070940.849257 0.006644 0.000109
---  Filters
---  Storage        malloc s0
---  Fetch_Body     3 length stream
---  BackendClose   20 default close REQ_HTTP10
---  Timestamp      BerespBody: 1726070940.857815 0.015202 0.008558
---  Length         14
---  BereqAcct      182 0 182 186 14 200
---  End
```

VARNISH SOFTWARE

varnishncsa

- Logs in a more standardized format
- Format can be customized
- Supports fine grained querying for specific requests
- Supports output rate limiting, writing to files, ..etc

```
root@walid-XPS-9320:/home/walid# varnishncsa -c -F "%{Varnish:hitmiss}x %s %r"
miss 200 GET http://localhost:1234/ HTTP/1.1
miss 200 GET http://localhost:1234/ HTTP/1.1
hit 200 GET http://localhost:1234/ HTTP/1.1
hit 200 GET http://localhost:1234/ HTTP/1.1
hit 200 GET http://localhost:1234/ HTTP/1.1
hit 200 GET http://localhost:1234/ HTTP/1.1
miss 301 GET http://localhost:1234/tmp HTTP/1.1
hit 301 GET http://localhost:1234/tmp HTTP/1.1
hit 301 GET http://localhost:1234/tmp HTTP/1.1
hit 200 GET http://localhost:1234/ HTTP/1.1
```

Varnish-cache community

- **Mailing lists**
  - `varnish-misc@varnish-cache.org: Miscellaneous discussions`
  - `varnish-bugs@varnish-cache.org: Bug reports`
  - `Others: https://varnish-cache.org/lists/mailman/listinfo`
- **Github**
  - `Issues for bug reports`
  - `Pull request to contribute code`
- **irc: `irc.redpill-linpro.com`**
  - **#varnish:** `Get help from other varnish users`
  - **#varnish-hacking:** `Weekly "bug-wash" Monday at 15:00-16:00 (EU time)`
- **Discord**
  - `https://discord.com/invite/EuwdvbZR6d`

VARNISH SOFTWARE

DEMO
(CVMFS Use case)