UPLEX
understanding complex systems

Yet Another Log Reader, and Yet Another ...

VUG8 Berlin
November 28, 2013

Geoff Simmons
UPLEX

OTTO

VARNISH

- Otto and the Lhotse project
- varnishevent
  - General logging
  - Used to extract log data, forward to log management systems
- Tracking Log Reader (trackrdrd)
  - Aggregates VCL_log data in a special format by XID
  - For web analytics and data warehouse

Otto... find' ich gut.

LHOTSE

BERGE VERSETZEN

- **In-house re-implementation of the shop**
  - **Online since October 24th**
- **Varnish (3.0.3) as proxy for …**
  - **Frontend**
  - **Inter-backend (app-to-app requests via API)**
  - **CMS and product backend**

varnishevent

UPLEX
understanding complex systems

- By default ≈ varnishncsa
- Output formats for:
  - Client transactions
  - Backend transactions
  - "File descriptor 0" (e.g. backend health checks)
- Some additional formatters
  - esp. raw payload for an arbitrary log tag
- Implemented for high throughput

varnishevent

Formatting configuration

UPLEX
understanding complex systems

```
# Client output format
cformat=%t host=proxy-123 direction=c url=%{tag:RxURL}x method=%m rc=%s
bytes=%b req_start="%{tag:ReqStart}x" req_end="%{tag:ReqEnd}x"
fetch_error="%{tag:FetchError}x" cache_hit=%{VCL_Log:disp}x
backend="%{VCL_Log:backend}x" cache_hits=%{VCL_Log:hits}x %{VCL_Log:ttl}x
incomplete=%{incomplete:T:F}x

# Backend output format
bformat=%t host=proxy-123 direction=b url=%{tag:TxURL}x method=%m rc=%s
bytes=%b close=%{tag:BackendClose}x reuse=%{tag:BackendReuse}x
esi_level=%{X-ESI-Level}i incomplete=%{incomplete:T:F}x

# Health check output format
zformat=%t host=proxy-123 health="%{tag:Backend_health}x"
```

varnishevent

Sample output

UPLEX
understanding complex systems

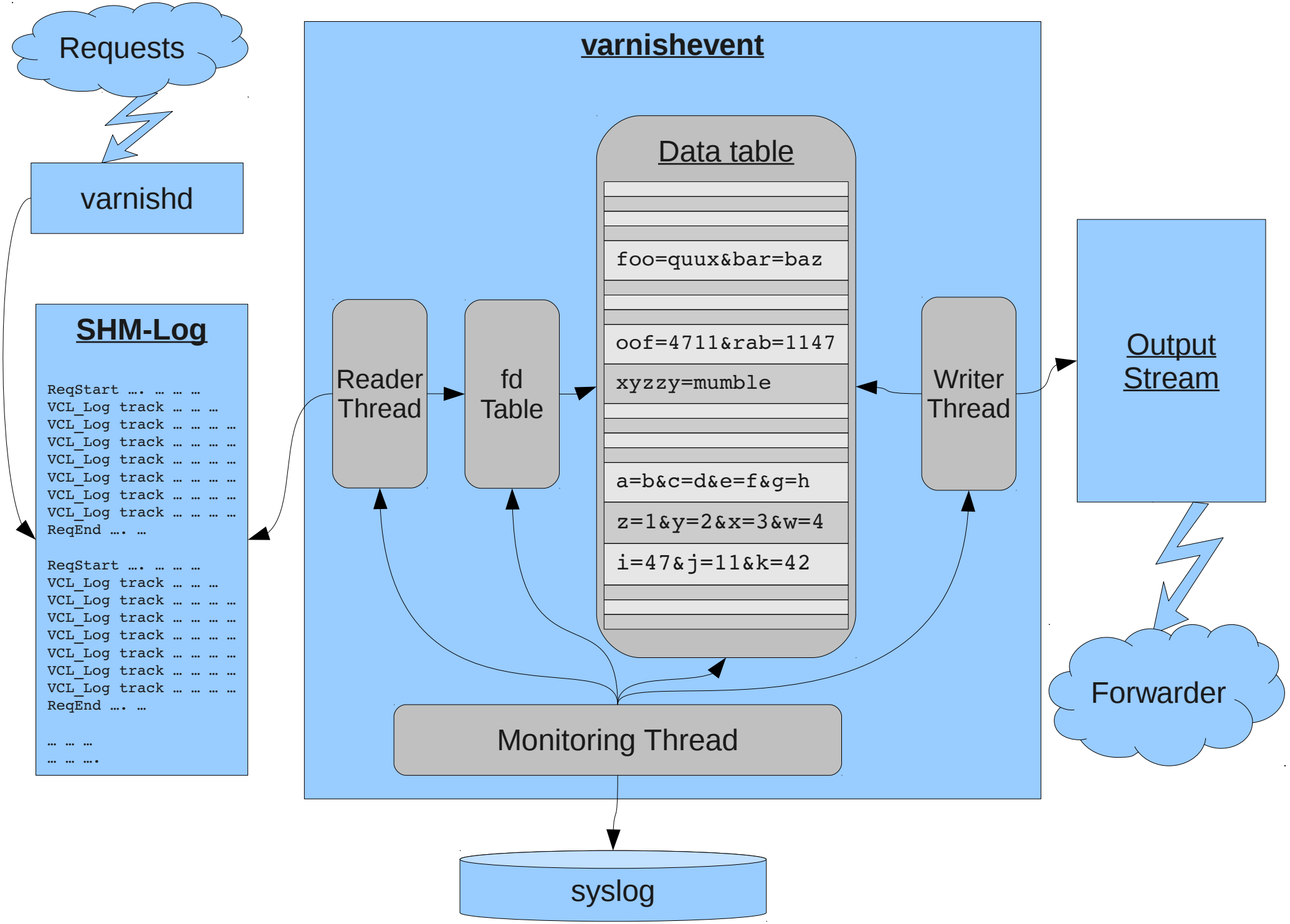
```
[27/Nov/2013:21:31:46 +0100] host=proxy-123 direction=c url=/foo/bar
method=GET rc=304 bytes=0 req_start="95.89.31.5 38665 1803698115"
req_end="1803698115 1385584306.012073040 1385584306.012546062 0.065220118
0.000354290 0.000118732" fetch_error="" cache_hit=HIT backend="app_4711"
cache_hits=90  incomplete=F

[27/Nov/2013:21:37:32 +0100] host=proxy-123 direction=b url=/baz/quux
method=GET rc=200 bytes=263 close= reuse=app_4711 esi_level= incomplete=F

[27/Nov/2013:21:41:54 +0100] host=proxy-123 health="app_4711 Still
healthy 4--X-RH 5 4 5 0.003220 0.002810 HTTP/1.1 200 OK"
```

- **Forward to log management via FIFO**

```
$ varnishevent -G config -w /named/pipe -D
```


Requests
varnishd
SHM-Log
ReqStart …. … … …
VCL_Log track … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
ReqEnd …. …
ReqStart …. … … …
VCL_Log track … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
ReqEnd …. …
… … …
… … ….
varnishevent
Data table
foo=quux&bar=baz
oof=4711&rab=1147
xyzzy=mumble
a=b&c=d&e=f&g=h
z=1&y=2&x=3&w=4
i=47&j=11&k=42
Reader Thread
fd Table
Writer Thread
Monitoring Thread
Output Stream
Forwarder
syslog

varnishevent

UPLEX
understanding complex systems

Results

- Peak throughput
  - 4500 messages/s, 4.5 MB/s (per host)
    - Varnish: 2500 requests/s (per host)
    - CPU load varnishevent 4% of varnishd load
  - 7000 messages/s per host in load tests
- 550 – 600 GB/day
- Applications for all that data
  - Arbitrary searches, error diagnosis
  - Daily caching & fetch error reports
  - Performance analyses
  - Graphing for monitoring

trackrdrd

UPLEX
understanding complex systems

Tracking Log Reader

- `std.log()` stores data in a fixed format:

```
track <XID> <payload>

std.log("track " + req.xid
        urlcode.encode(" foo=bar&baz=quux")

VCL_Log c track 12345678 foo=bar&baz=quux
```

- `trackrdrd` ...
  - aggregates all payloads found for an XID
  - forwards the resulting record to message brokers (ActiveMQ)

```
XID=12345678&foo=bar&baz=quux&varnish=roolz
&otto=findichgut&ichbin=einberliner
&req_endt=1385589728.695621967
```

trackrdrd

UPLEX
understanding complex systems

Data extraction

- VCL for standard request/response data:

```
if (req.http.Referer) {
  std.log ("track " + req.xid + " referer="
          + urlcode.encode (req.http.Referer));
}

if (req.http.User-Agent) {
  std.log ("track " + req.xid + " ua="
        + urlcode.encode (req.http.User-Agent));
}
```

```
VCL_Log c track 987654321 referer=/foo/bar

VCL_Log c track 987654321 ua=Mozilla%2F5.0%20
```

trackrdrd

UPLEX
understanding complex systems

Data extraction

- Pseudo-URL with query string for business data:

```
/track?productID=4711


if (req.url ~ "^/track\?") {
  std.log ("track " + req.xid + " "
       + regsub (req.url, "^.+\?(.+)$", "\1"));
}
# /track/204 returns "204 No content" and
# is cached with TTL=forever
set req.url = "/track/204";
return(lookup);


VCL_Log c track 987654321 productID=4711
```

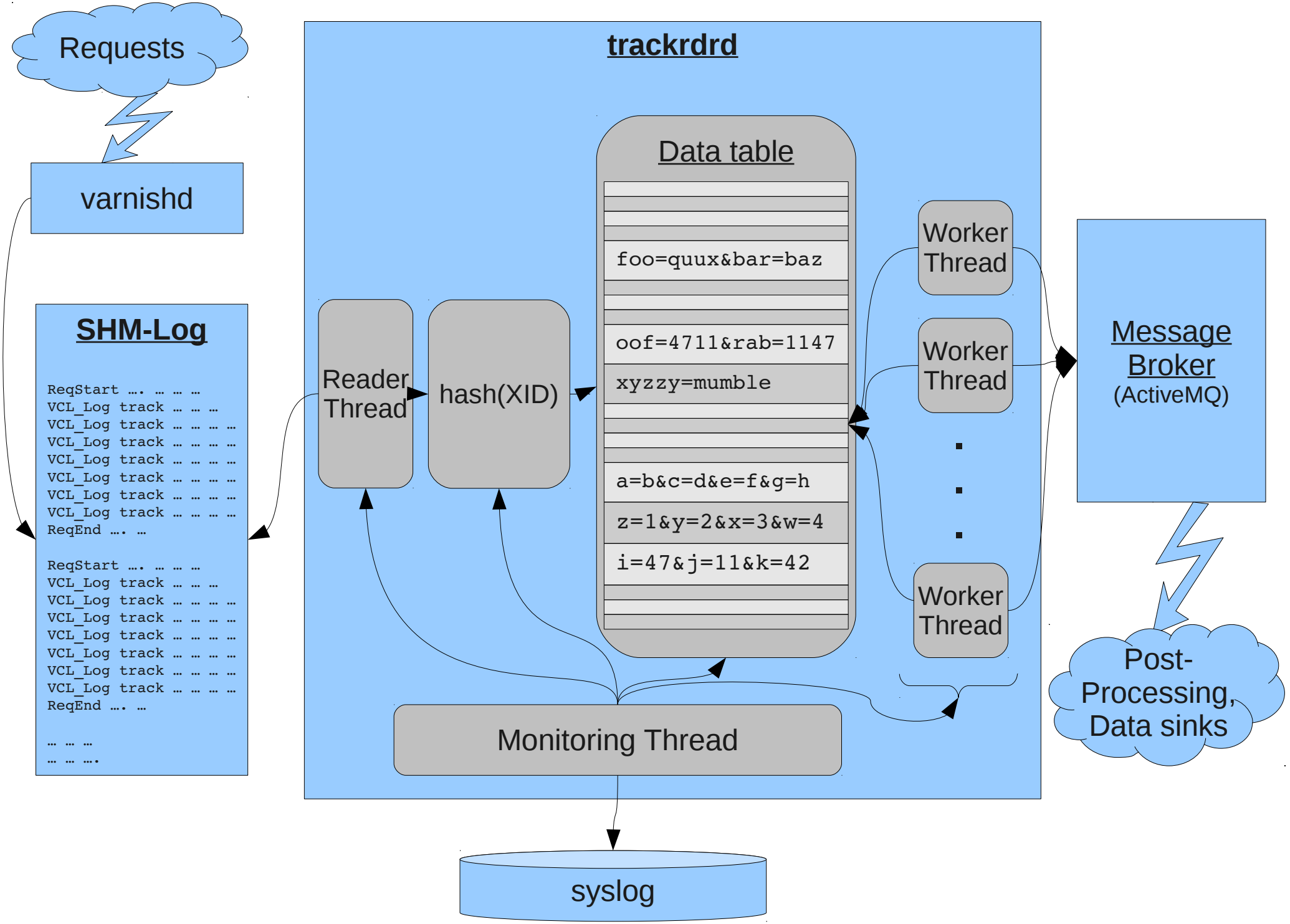
trackrdrd

UPLEX
understanding complex systems

ESI includes for aggregation

```
<esi:include src="/track?productID=4711"/>
<div class="foo"><p>mumble</p></div>
<esi:include src="/track?variationID=0815"/>
```

- **`<esi:include/>` replaced with the cached empty response (no round trip)**
- **Business data stored in Varnish log and read asynchronously by `trackrdrd`**
- **(Varnish 3) Requests and all ESI includes have the same XID**
  - **Aggregation by XID partially aggregates data per request to data per page impression**


Requests
varnishd
SHM-Log
ReqStart …. … … …
VCL_Log track … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
ReqEnd …. …
ReqStart …. … … …
VCL_Log track … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
VCL_Log track … … … …
ReqEnd …. …
… … …
… … ….
trackrdrd
Reader Thread
hash(XID)
Data table
foo=quux&bar=baz
oof=4711&rab=1147
xyzzy=mumble
a=b&c=d&e=f&g=h
z=1&y=2&x=3&w=4
i=47&j=11&k=42
Worker Thread
Worker Thread
Worker Thread
Monitoring Thread
syslog
Message Broker
(ActiveMQ)
Post-Processing, Data sinks

trackrdrd

UPLEX
understanding complex systems

Results

- Peak throughput
  - 1750 records/s sent (per host)
  - 2700 records/s in load tests (per host)
  - max record length 4.8 KB
  - 70 – 75 Mrecords/day sent (per host)
- Applications
  - Web analytics
  - Data warehouse

To Do

UPLEX
understanding complex systems

- **Logging API in Varnish 4**
  - **varnishncsa v4 may make varnishevent obsolete**
  - **More powerful querying & filtering**
  - **Less memory copying?**
    Smaller memory footprint?
  - **Requests & ESI includes do not share VXIDs**
    - **But request grouping will do the same job (Martin will explain later today)**
- **`trackrdrd`: pluggable message broker interface**
  - **Currently hard-wired for ActiveMQ**

Grab the code!

UPLEX
understanding complex systems

- `https://code.uplex.de/uplex-varnish/trackrdrd`
- Coming soon!
  - `varnishevent` also available at `code.uplex.de`
  - Both tools also available at:
    - https://github.com/otto-de
      Open source projects from the Lhotse project
    - https://www.varnish-cache.org/utilities
      Varnish Utilities Directory

UPLEX
understanding complex systems

Danke!

Questions?

- geoff.simmons@uplex.de