

Poul-Henning Kamp

<phk@FreeBSD.org>

Getting the content out

from

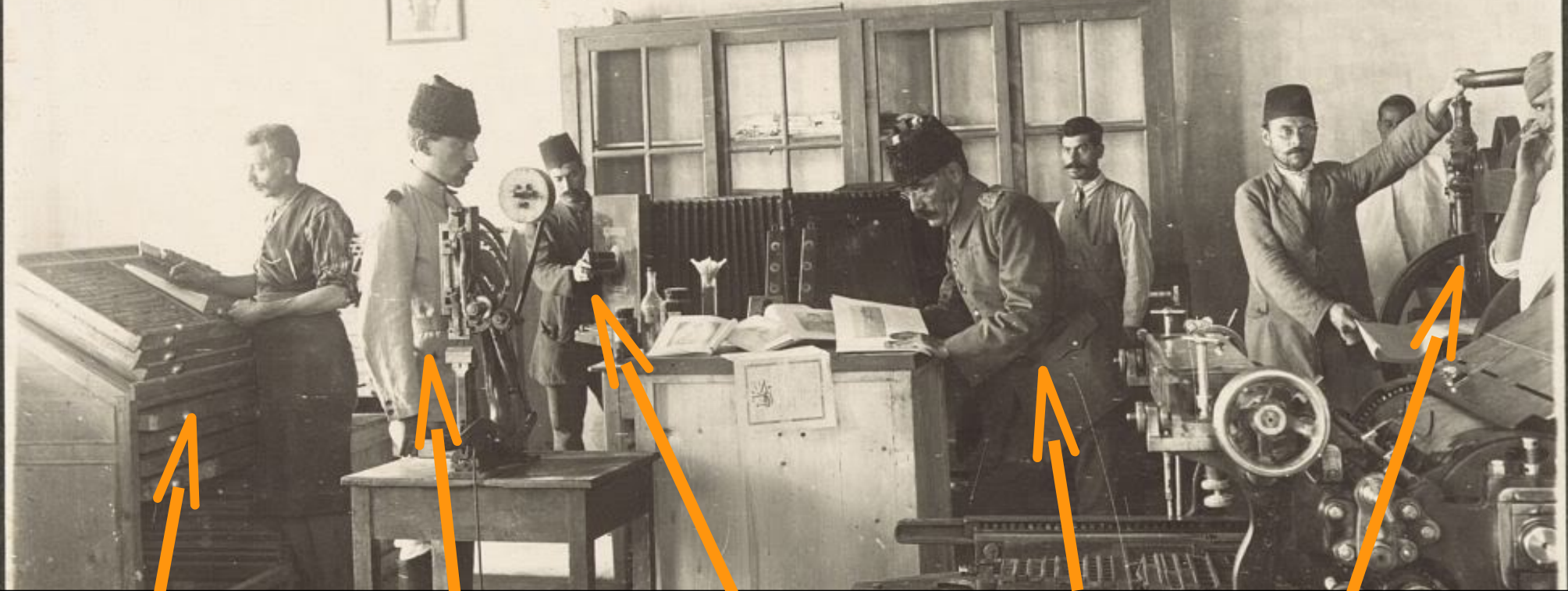
Heidelberg

to

Varnish

or:

Enough about
Johannes Gutenberg,
let's talk about
Karl Georg Ferdinand Gilke.



???

Typography:
Multiple Fonts,
Hyphenation,
Ligatures &c
(sepll checking
an optional extra.)

Multimedia:
Scaling, Cropping,
Gamma correction (?)

CMS

Production:
Replication
Delivery

Content Creation:

Diverse input methods:

Text Editors, Image scaling/cropping, File import filters, Feeds...

Flexible Layout/Typography tools

WYSIWYG, Semantic Markup, CSS

Complex cross referencing:

"Other articles about Paris Hilton"

"No airline ads, if «crash» present in headline"

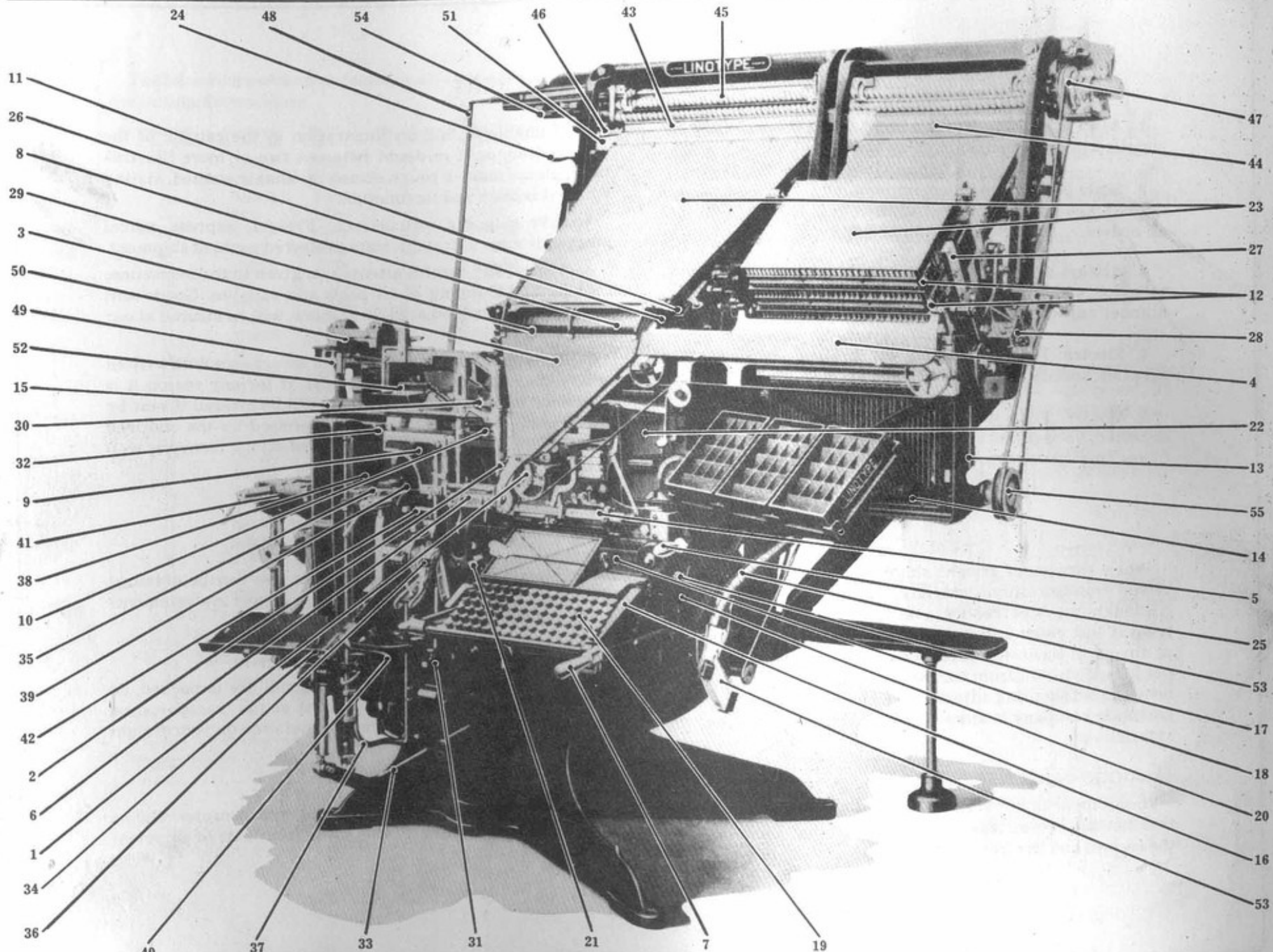
Expensive in database lookups

User Generated Content

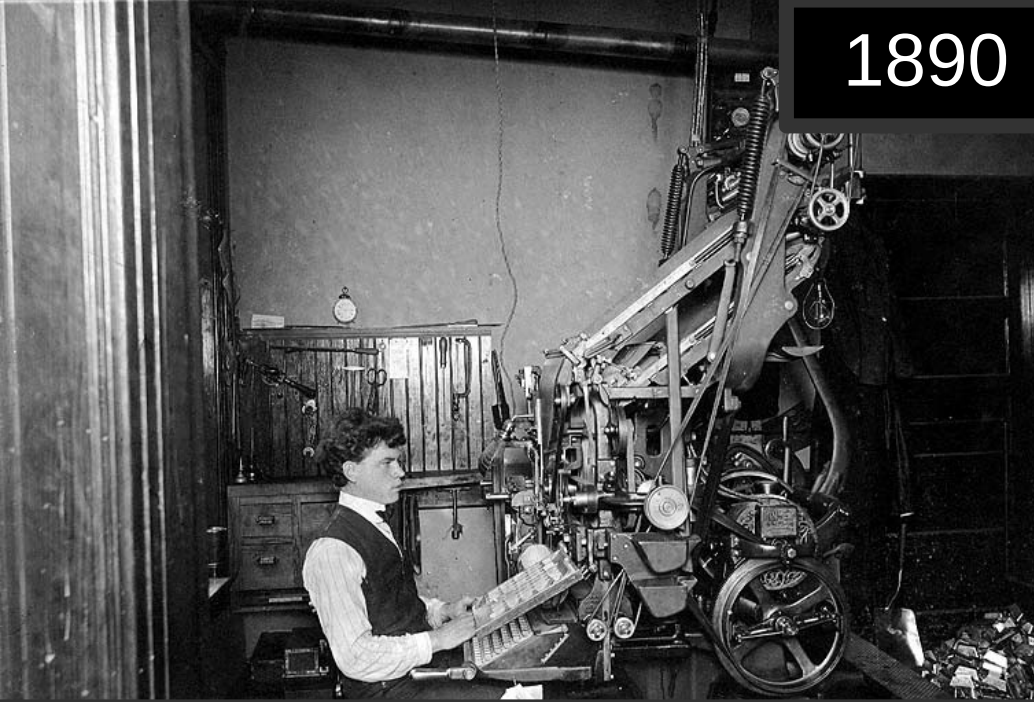
Discussions, galleries, personal views, chats etc.

Ohh, and: Rendering

KEY TO MECHANISMS



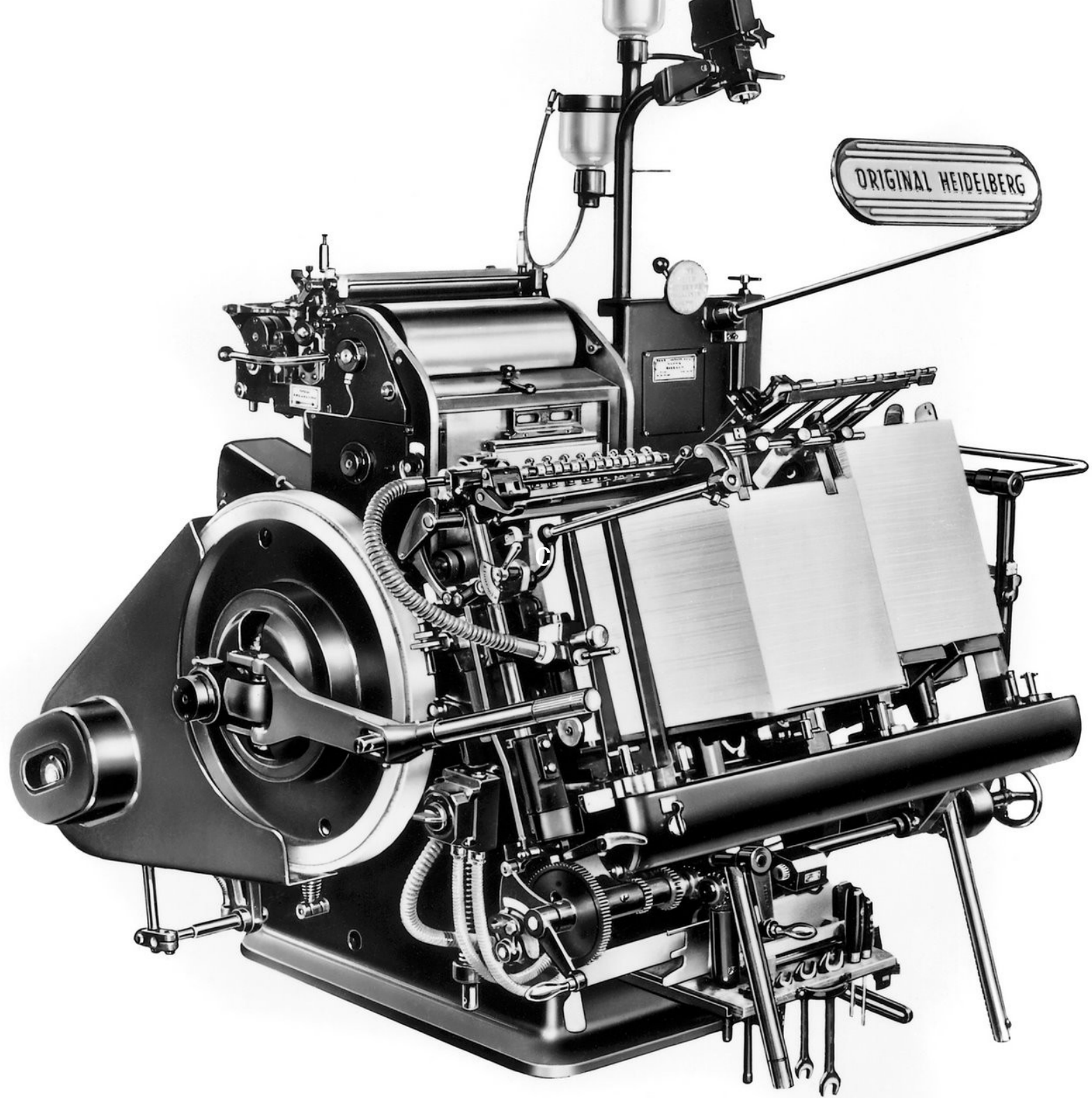
1890 ⇔ 1980



Content Production:

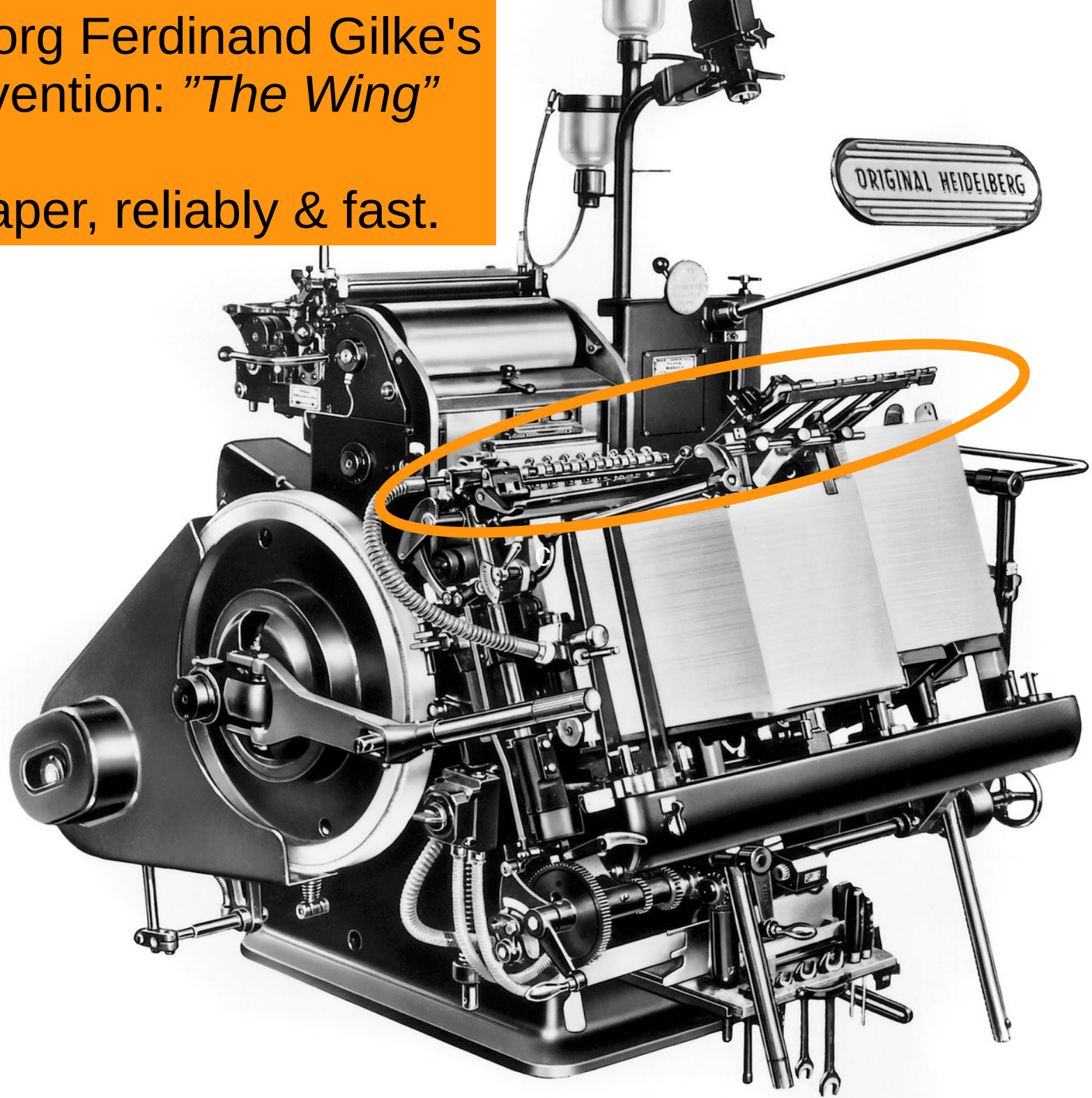
Repeated reproduction of rendered master.



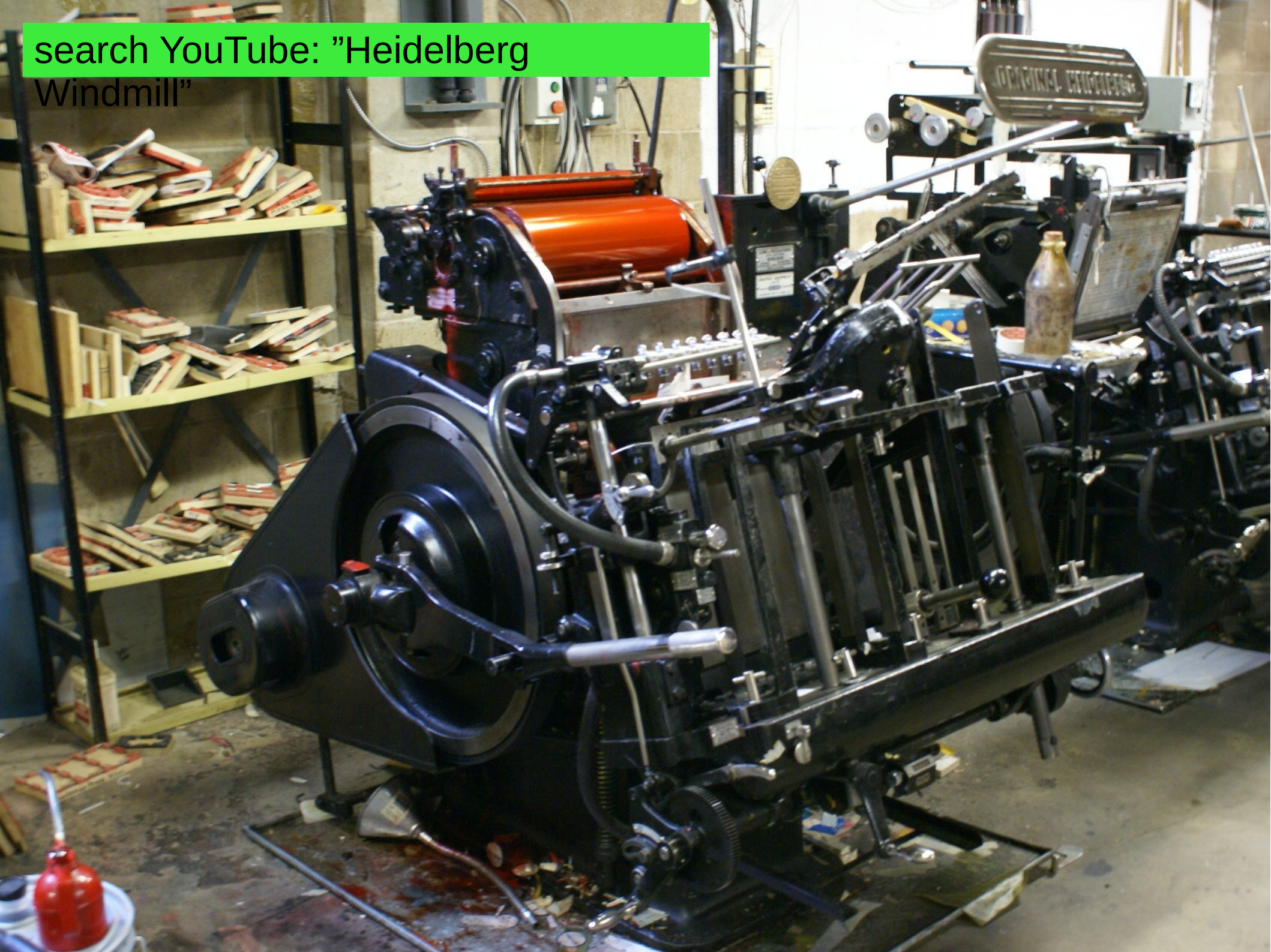


Karl Georg Ferdinand Gilke's
1912 invention: "*The Wing*"

Move paper, reliably & fast.



search YouTube: "Heidelberg
Windmill"



The Varnish Elevator Pitch:

Varnish is a HTTP delivery engine

... general purpose HTTP processor

... HTTP Cache

... content composition

... cheap hardware does 100+ kreq/s

... is Free & Open Source Software

... has commercial support



The website VG.no is one of Norway's largest in terms of traffic.

Classical news site: rapidly changing contents in a slow CMS system.

12 Squid caches used as accelerators.

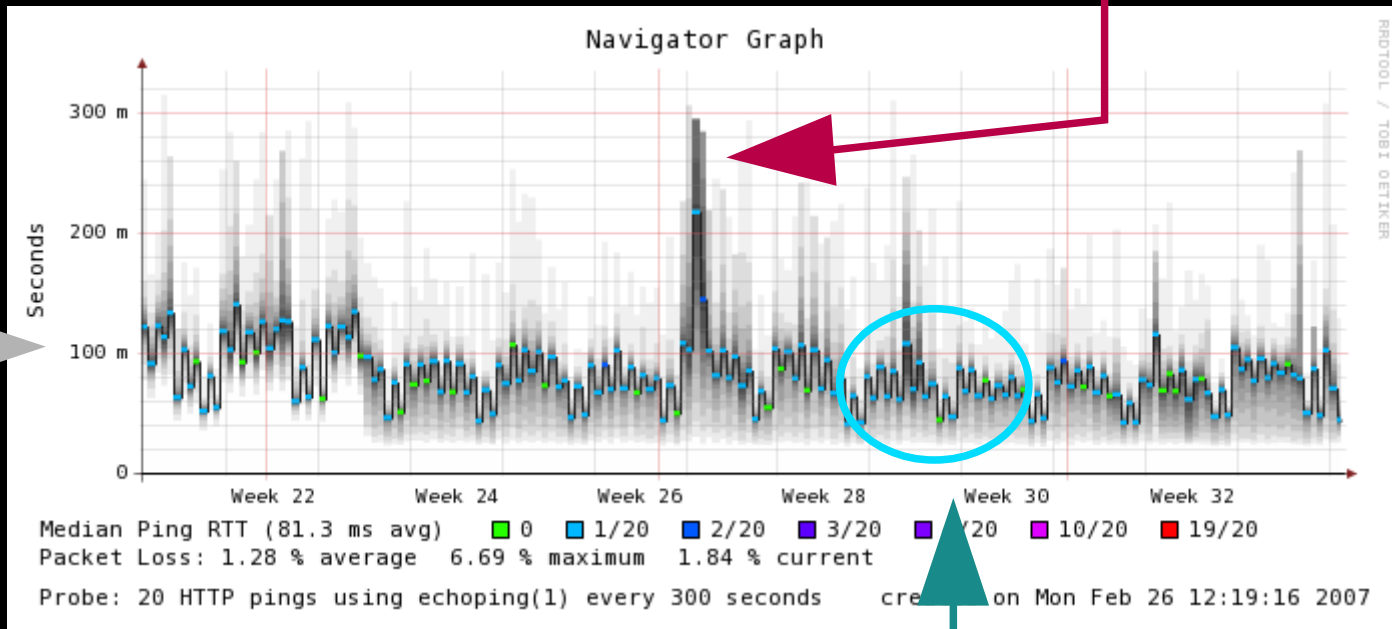
Unhappy with performance and stability.



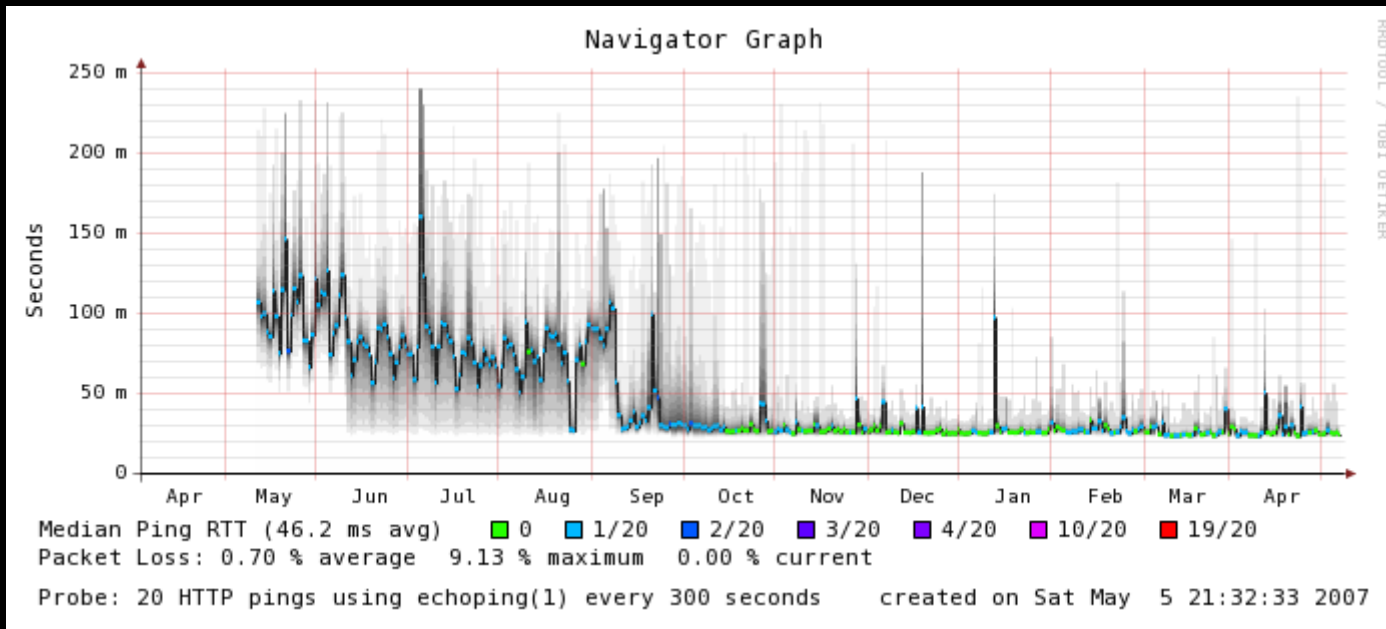
Multimedia arm of Norwegian newspaper "Verdens Gang"

Slow response

Terrible peak-handling



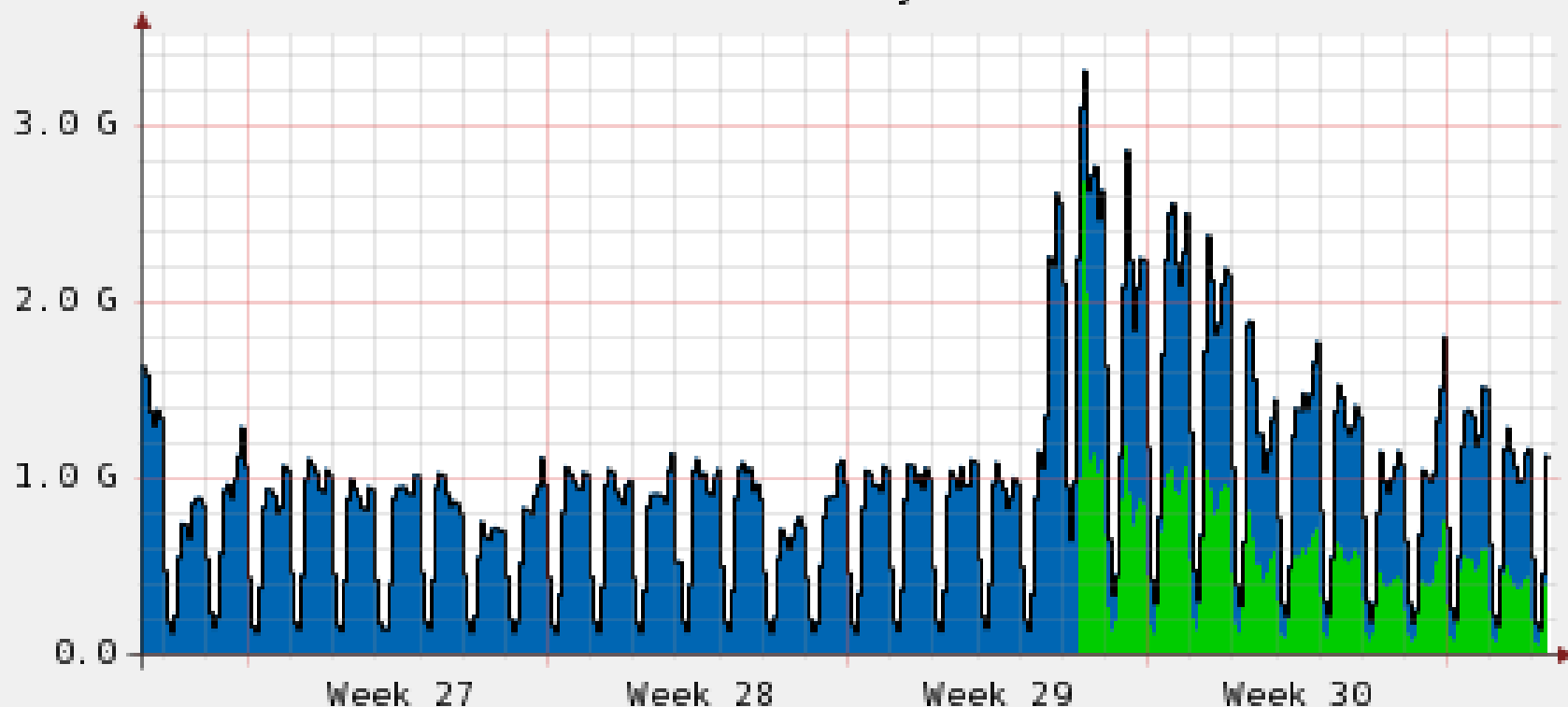
Significant loss



Squid
12 servers

Varnish
3 servers

Bandwidth - by month

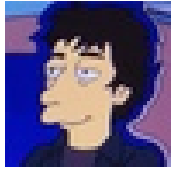


	Cur:	Min:	Avg:	Max:
lbhmg9	394.66M	0.00	175.62M	4.12G
lbm323	729.50M	0.00	751.52M	3.08G
total	1.12G	112.46M	927.13M	6.17G

Last update: Wed Aug 3 11:40:23 2011

Wanted: Website terrorist





neilhimsel Neil Gaiman

I just [#neilwebfailed](#) The Beano's site. I think they can put you in prison for that. I'm going off to write now. You didn't see anything.

11 hours ago

#neilwebfailed:

Getting all the attention
you ever dreamt about,
and your webserver dies.



Neil Gaiman ✓

@neilhimsel mostly near minneapolis
will eventually grow up and get a real job. Until
then, will keep making things up and writing them
down.

<http://www.neilgaiman.com>

+ Follow

Text follow neilhimsel to your carrier's shortcode



About @neilhimsel

27,562

Tweets

668

Following

1,628,492

Followers

25,309

Listed

2 days later:



= 45 new followers/h



About @neilhimsel

27,699

Tweets

668

Following

1,630,642

Followers

25,395

Listed



Neil Gaiman I may have just #neilwebfailed the @BigIssueScots website. Wait a little while then click on those links again. Sorry..

ReplyRetweetPermalink

4:40 AM Aug 5th



Neil Gaiman Seem to have #neilwebfailed the allhallowsread.com site. Here's the original blog post from last year: <http://t.co/TMkPRZO>

ReplyRetweetPermalink

9:49 PM Jul 19th



Neil Gaiman I think that <http://www.marklynas.org/> link was #neilwebfailed. Sorry. I should've mirrored it.

ReplyRetweetPermalink

1:52 PM Jul 9th



Neil Gaiman For the curious, The Den of Iquity Turntable Stats & Playlist from tonight so far (mirrored so it won't #neilwebfail) <http://goo.gl/MI9jV>

ReplyRetweetPermalink

9:07 PM Jul 7th



Neil Gaiman Damn. #Neilwebfail on the Dirda essay. I'll take down the tweet until the website's working again. Then I'll mirror it. Sorry.

ReplyRetweetPermalink

5:47 PM Jul 5th

Starting from scratch, setting goals:

Varnish is a HTTP accelerator.

Better configuration.

Better management.

Much faster.

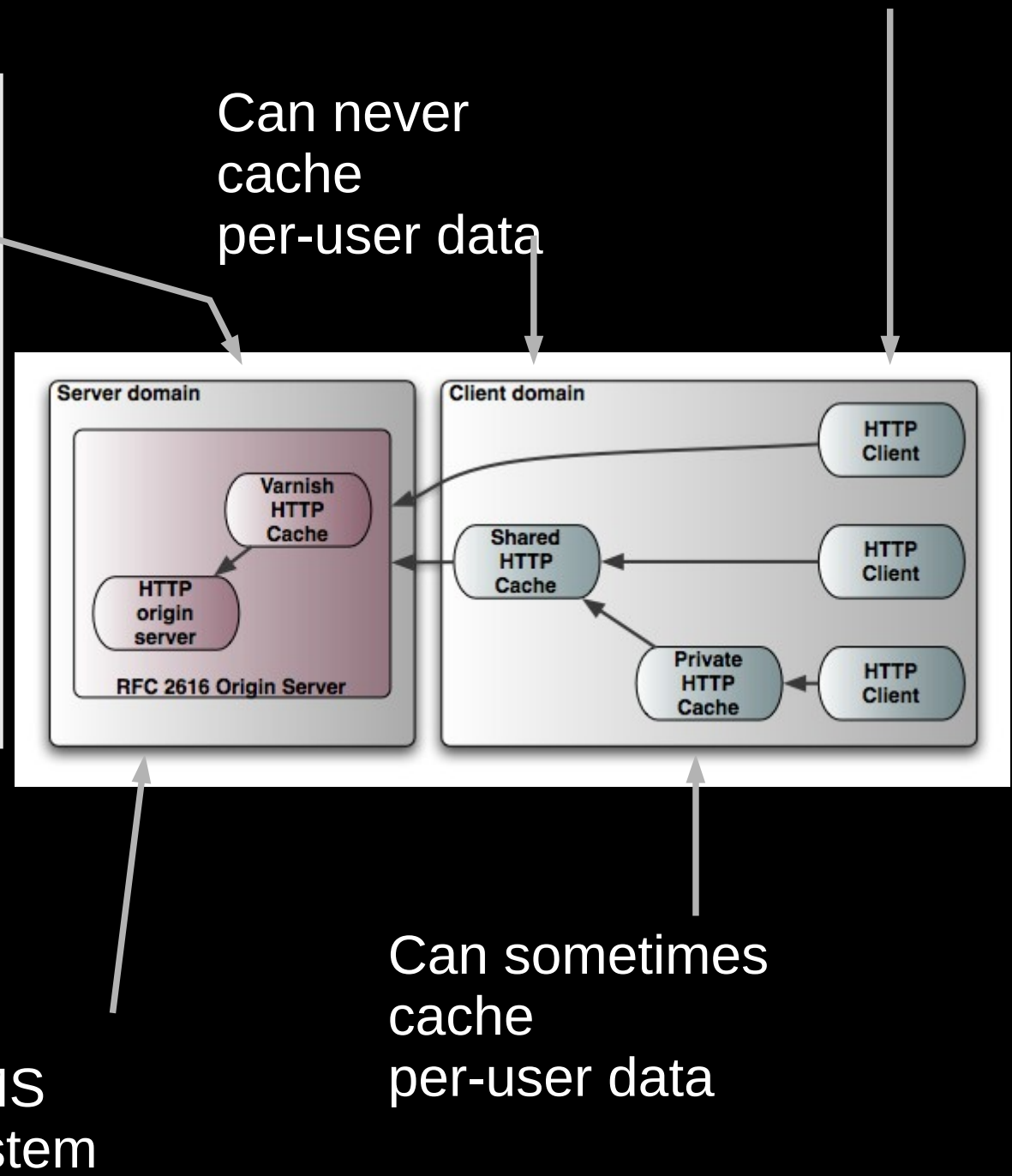
Content management feature set

RFC2616 cast list

A HTTP Accelerator is not a HTTP cache.

Caching policy can be tailored to CMS system and site policies.

RFC2616 compliance as "origin server".



```
$ cat /etc/foobar.conf
# copied from example.conf
#      /svend 19870104
# updated to new version
#      /knud 19941231
# new version, some changes
#      /valdemar 20060523
# DON'T MESS WITH THIS!!!
```

See manual for details.

HDXHSVVaCS=0y

Dimensionality of FTL-space used

$V_{\text{neutrino}} = 3.1418 \oplus \aleph + \int \int \int f(21.4^{\mu} \cdot \beta e^{-ij})$

overflow method (0-7) [default=3]

overflow_method=8.03

Man



Willis Looy 1998-2001

Woman



Vigerslev Nær
F 15 35 55
F+ 25 45 05

„Machtwort/Kongspil/Trollen“

Hilling → Lansen → Flakkeby: Nordhavn
Flakkeby → Lansen: Nordhavn
Flakkeby → Lansen: Nordhavn
Lansen (by Jægers) → Hilling: Fx Nord

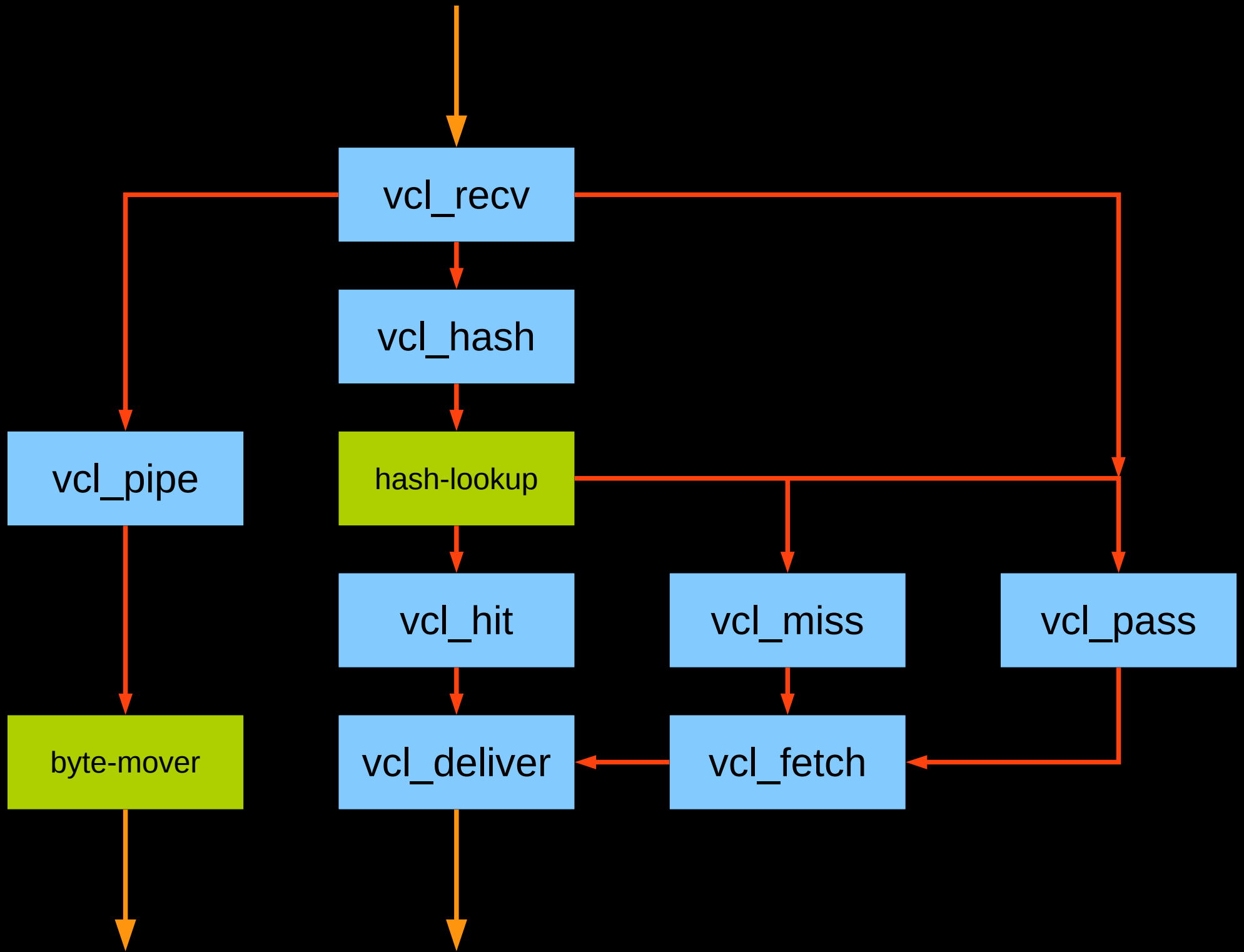
H 03 23 43
H+ 10 30 50
A+ 18 38 58

A+ 02 22 42
H+ 09 29 49
H 16 36 56

B+ 09 29 49
F 10 30 50
A 12 32 52
B 19 39 59
F+ 17 37 57
F+ 20 40 00
E 22 42 02
C 24 44 04

F 14 24 54
C 16 36 56
E 18 38 58
B 20 40 00
F+ 24 44 04
A 28 48 08
B+ 30 50 10

HI



VCL - Varnish Configuration Language

```
sub vcl_recv {
    if (req.url ~ "(\\.\\.|\\.exe)") {
        error(999, "Bugger off.");
    }
    if (client.ip ~ editor_ips) {
        set req.http.x-cms = "no-stats";
        return(pass);
    }
    if (req.url ~ "\\.(jpg|png|gif|css)$") {
        unset req.http.cookie;
        unset req.http.authenticate;
        set req.backend = static_backend;
    }
    if (req.url == "hotstory.html") {
        set req.url = "hotstory.html";
    }
}
```

Why VCL rocks:

Compiled to binary/shlib via C-code

⇒ Runs full speed

You can have multiple VCL's loaded at the same time

⇒ Switch between them without restart

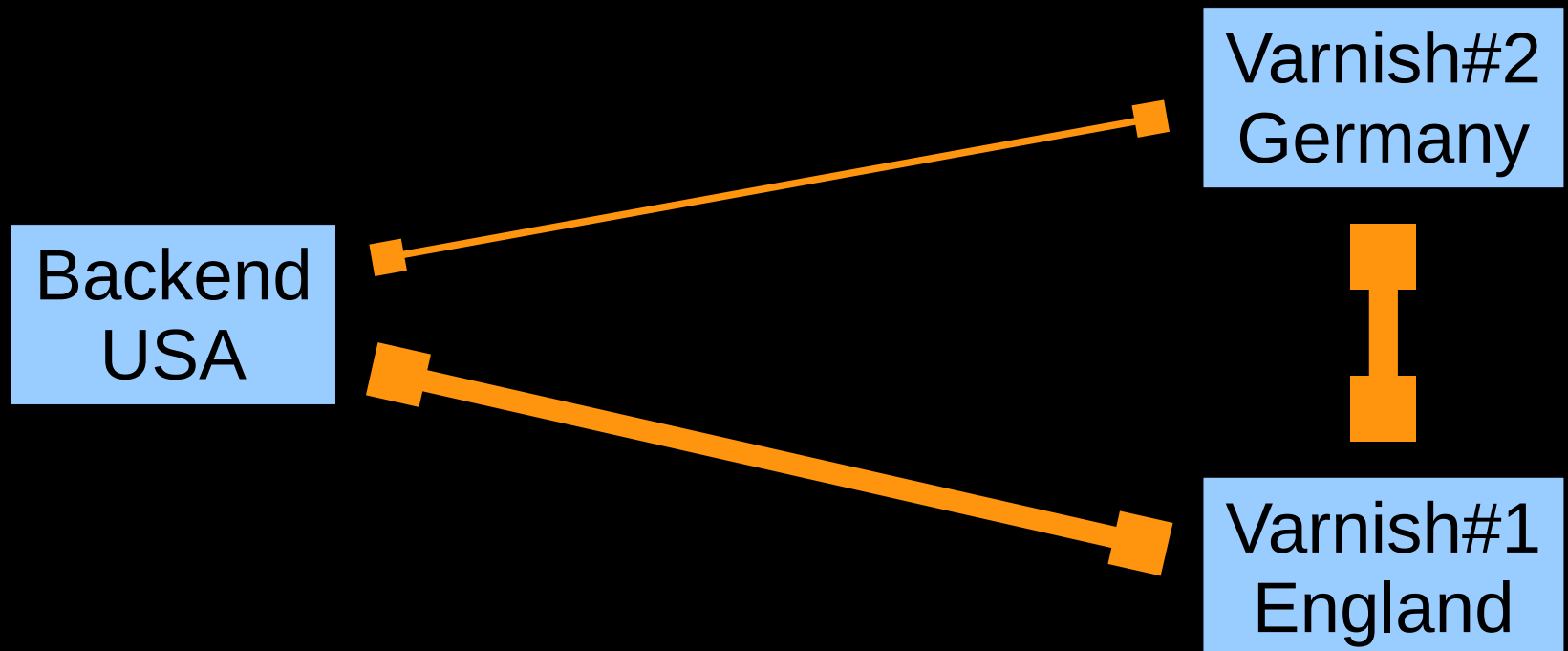
⇒ Instantaneous

Allows you to do anything you might fancy

⇒ VMOD libraries (Written in C)

⇒ Inline C-code (if you're hard-core)

```
sub vcl_recv {  
    if (client.ip == "varnish1") {  
        set req.backend = usa;  
    } else {  
        set req.backend = england;  
    }  
}
```



Managing Varnish

Command Line Interface for real-time control

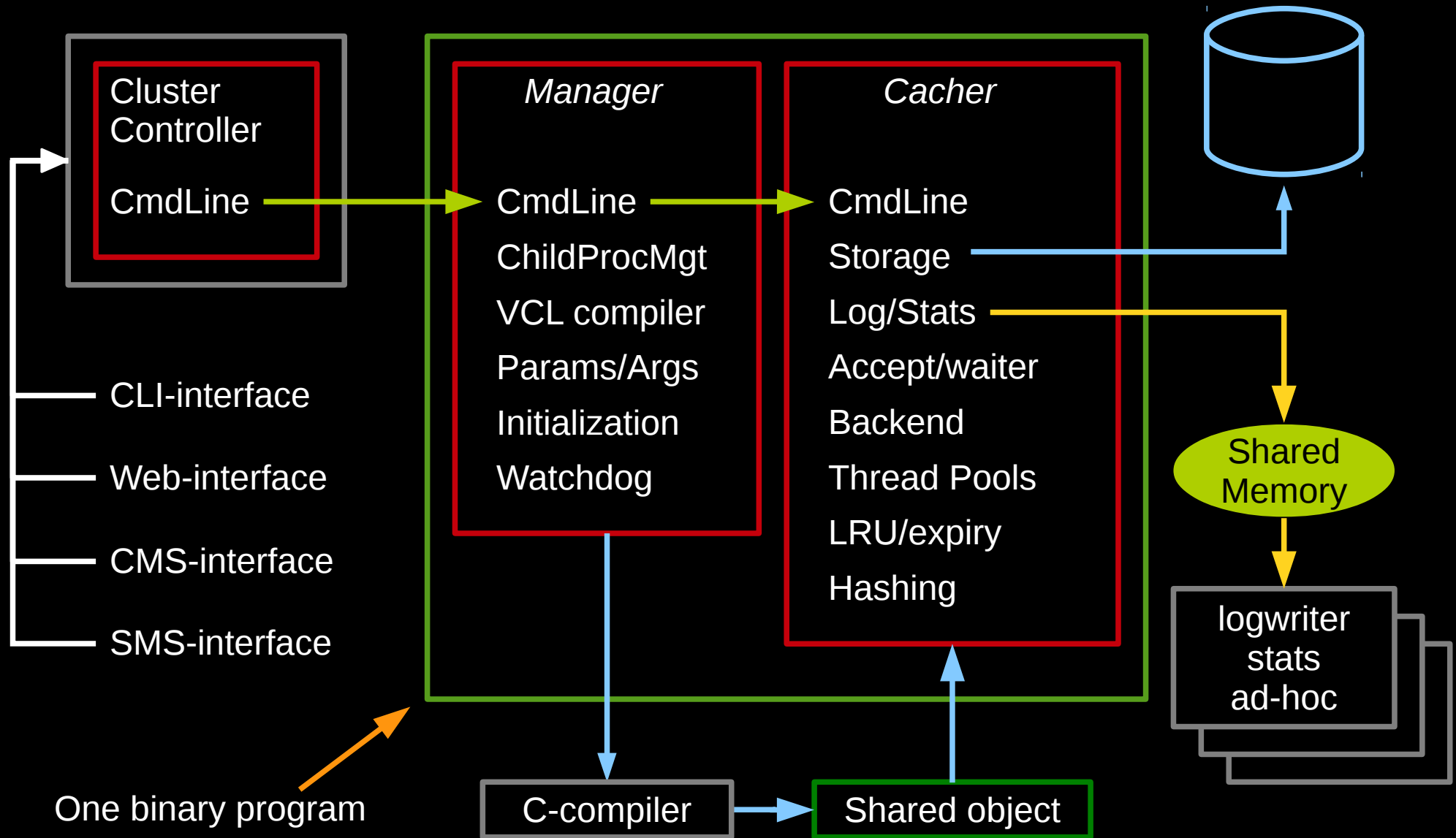
Management/Worker process split:

- Manager (re)starts worker

- Allows privilege separation

- Contains multithreading to worker process

Varnish architecture



CLI management

```
$ telnet localhost 81
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
param.show
200 675
default_ttl          120 [seconds]
thread_pools         5 [pools]
thread_pool_max      1500 [threads]
thread_pool_min       1 [threads]
thread_pool_timeout  120 [seconds]
overflow_max         100 [%]
http_workspace       8192 [bytes]
sess_timeout         5 [seconds]
pipe_timeout         60 [seconds]
send_timeout         600 [seconds]
auto_restart         on [bool]
[...]
```

CLI management

```
param.show overflow_max
```

```
200 330
```

```
overflow_max      100 [%]
```

```
Default is 100
```

```
Limit on overflow queue length in  
percent of thread_pool_max parameter.
```

```
NB: We don't know yet if it is a good  
idea to change this parameter.
```

```
Caution advised.
```

Performance and speed

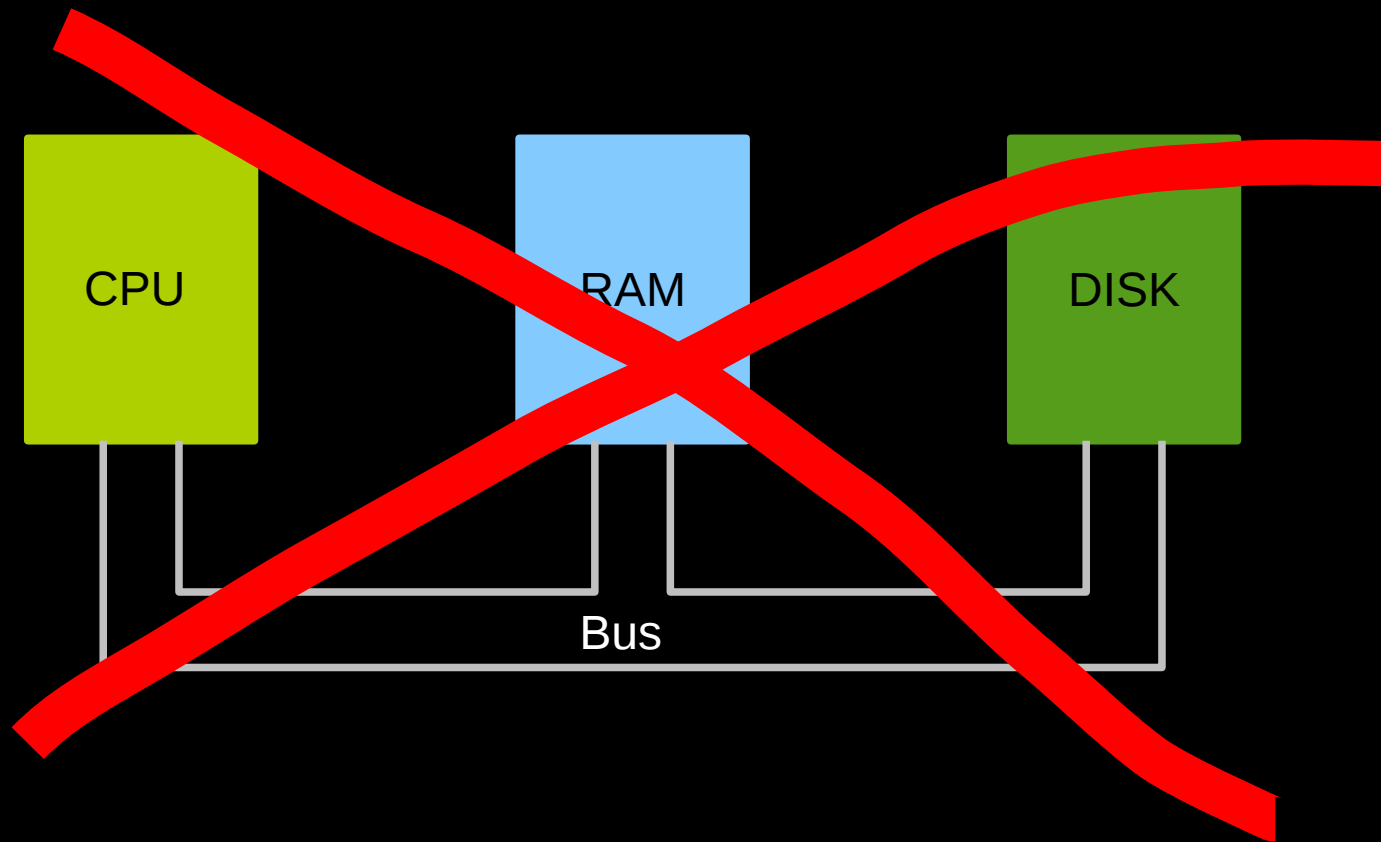
Program for performance from day one

Use modern features:

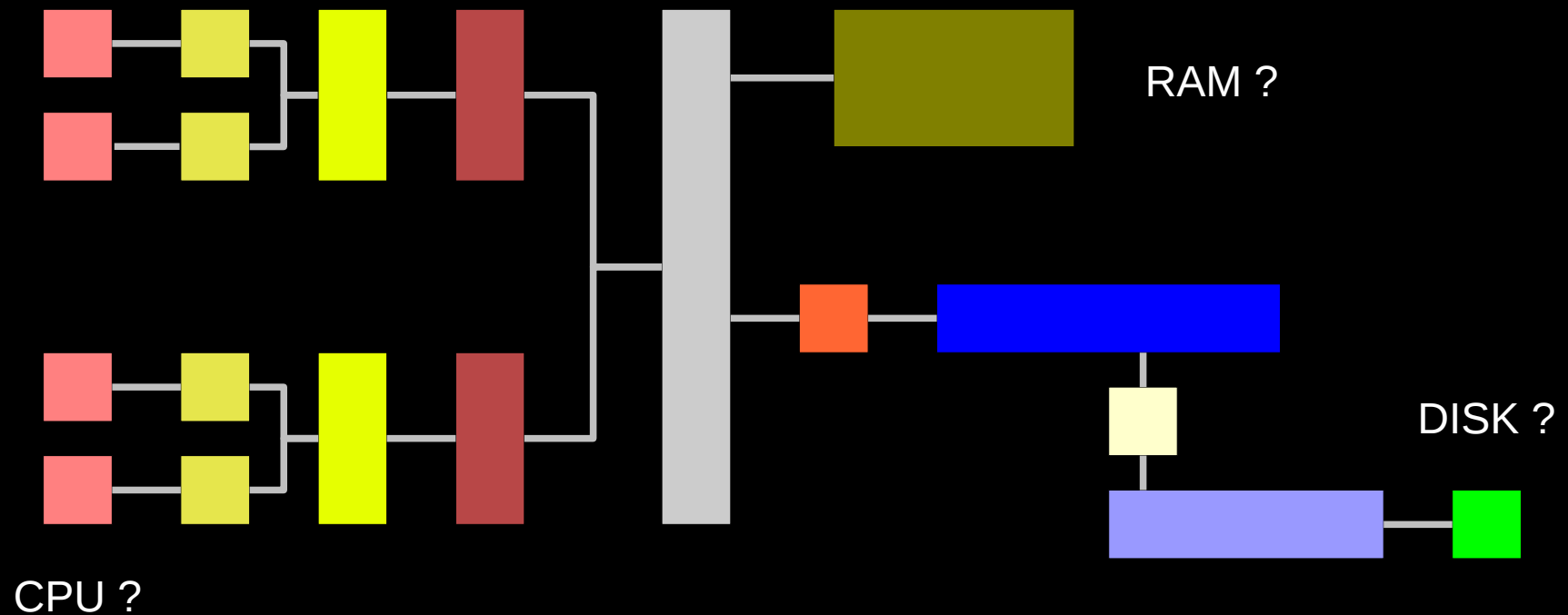
Virtual Memory

`sendfile(2)`, `accept_filters(2)`, `kqueue(2)`

(and every other trick in the book)



Not your dads computer anymore:

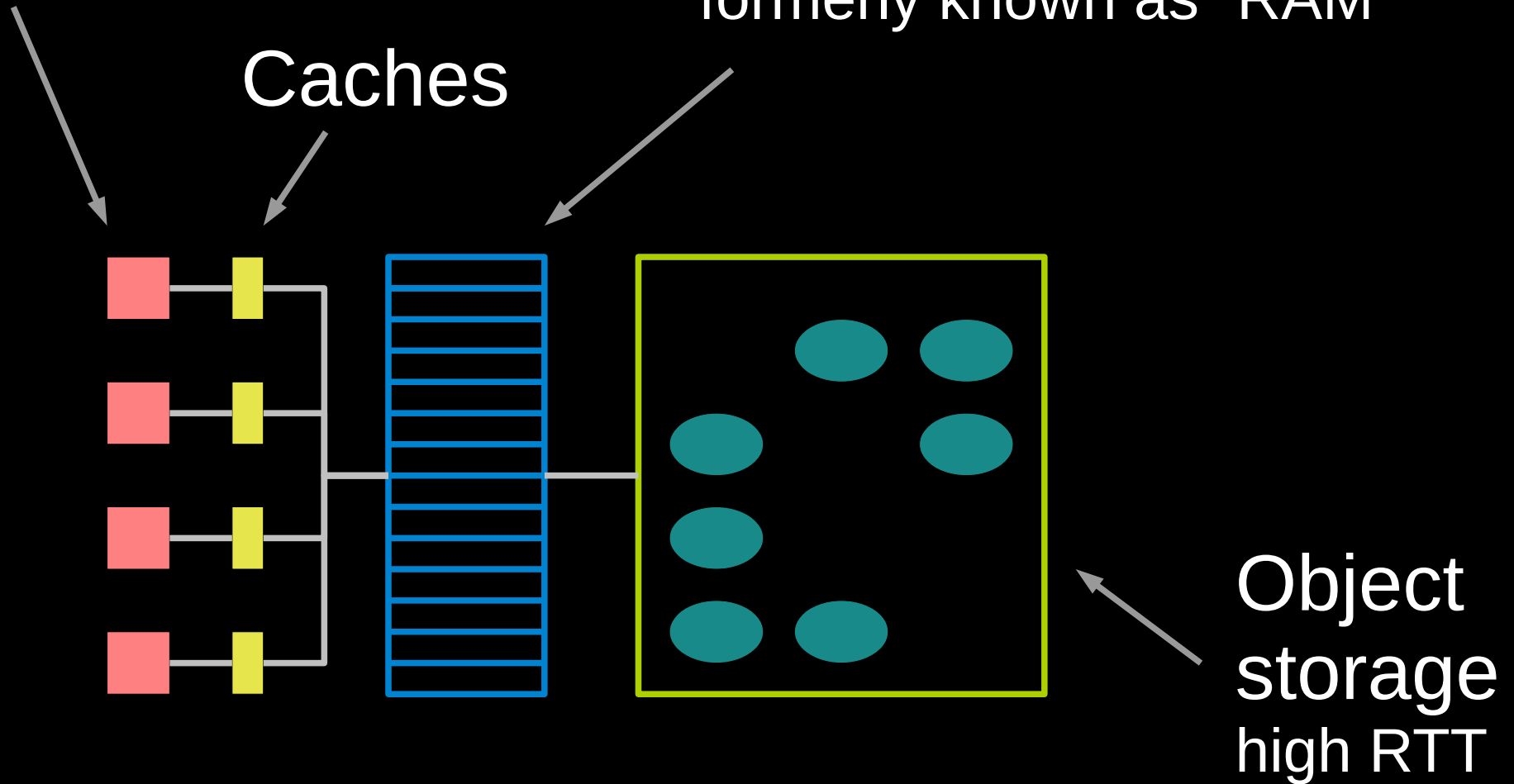


...and besides, the operating system
virtualizes all of this

CPUs/Cores

The Virtual Page Cache
formerly known as "RAM"

Caches



Performance Pricelist

- `char *p += 5;`
- `strlen(p);`
- `memcpy(p, q, l);`
- Locking
- System Call
- Context Switch
- I/O Disk Access
- File operation
- Directory operation



CPU

1,000,000,000/s

Memory

Protection

Mechanical

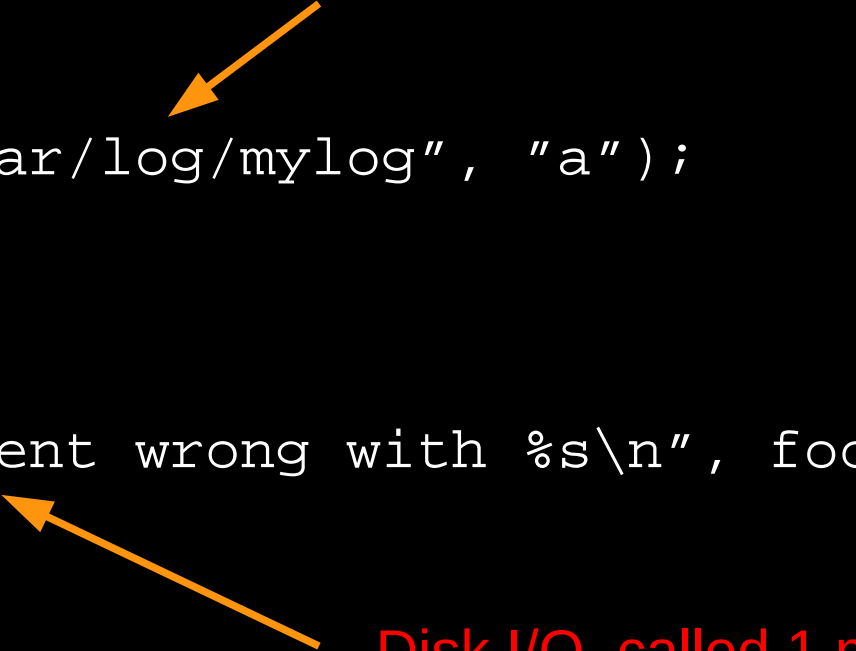


100/s

Classical logging is horribly expensive:

Filesystem operation, called once.

```
FILE *flog;  
flog = fopen("/var/log/mylog", "a");  
[...]  
fprintf(flog,  
        "Something went wrong with %s\n", foo2str(object));  
fflush(flog);
```



Disk I/O, called 1 mio times

$$1 * 0.010s + 1,000,000 * 0.001s = 16 \text{ minutes}$$

Logging to shared memory is almost free:

```
char *logp, *loge;
```

```
fd = open(...);
```

```
logp = mmap(..., size);
```

```
loge = logp + size;
```

Filesystem ops, called once.



```
[...]
```

```
logp[1] = LOG_ERROR;
```

```
logp[2] = sprintf(logp + 3,
```

```
    "Something went bad with %s", foo2str(obj));
```

```
logp[3 + logp[2]] = LOG_END;
```

```
logp[0] = LOG_ENTRY;
```

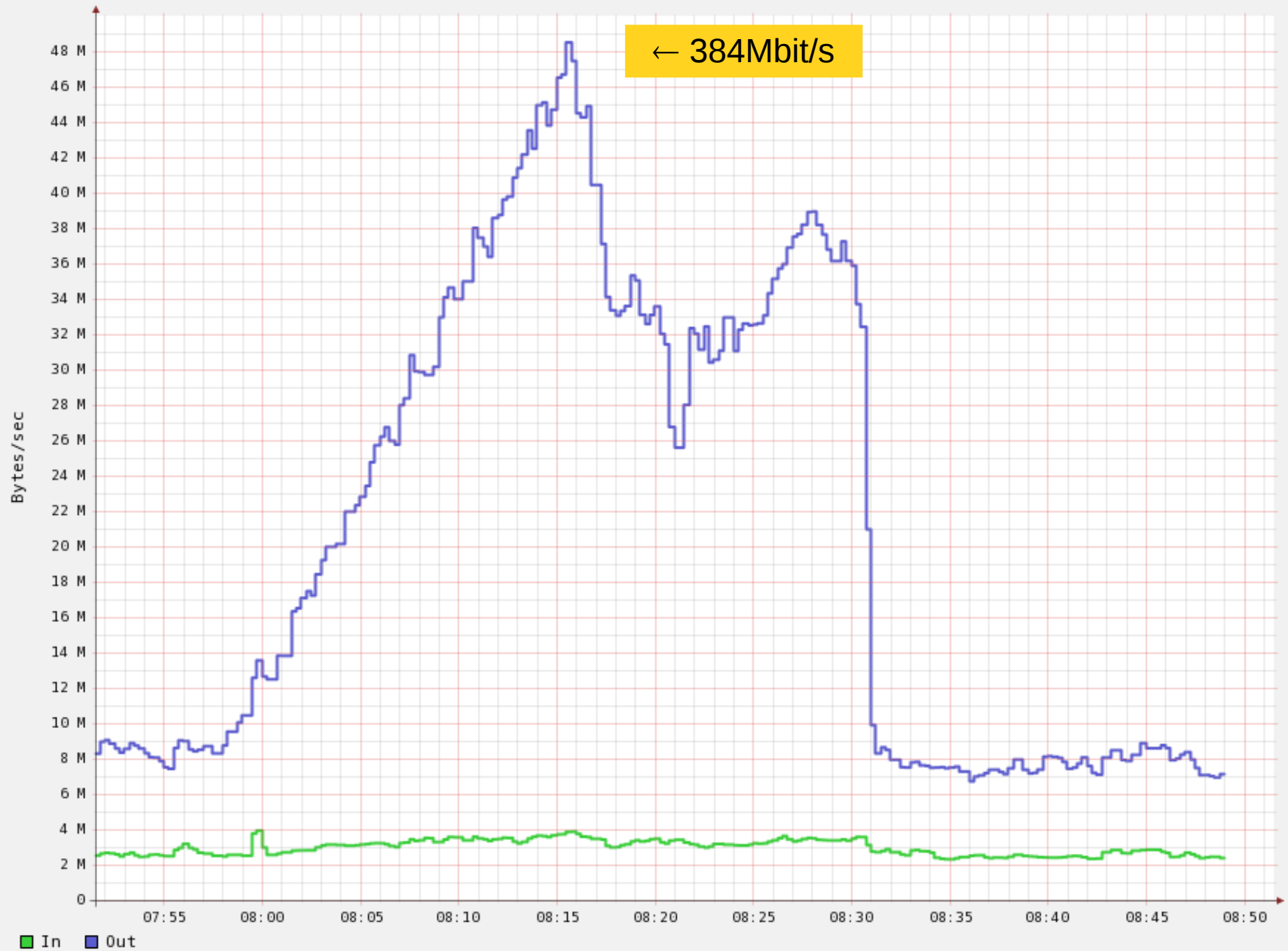
```
logp += 3 + logp[2];
```

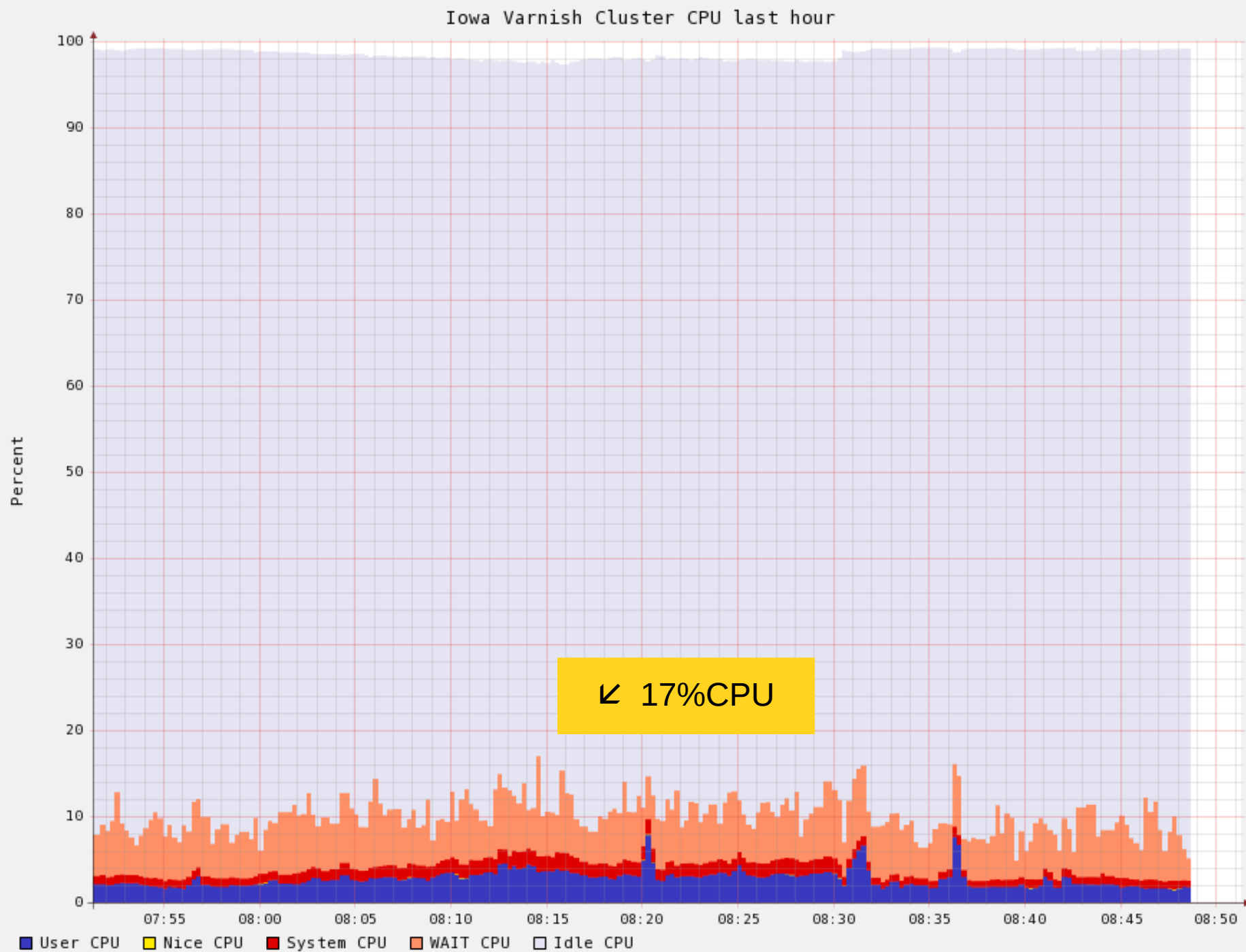
Memory and arithmetic, 1 mio calls



$$2 * 0.010s + 1,000,000 * .000001s = 10 \text{ seconds}$$

Iowa Varnish Cluster Network last hour





Where does my traffic come from ?

```
$ varnishtop -i rxheader -I Referer
```

```
33913.74 Referer: http://www.vg.no/  
4730.72 Referer: http://vg.no/  
925.62 Referer: http://www.vg.no  
510.10 Referer: http://www.vg.no/pub/vgart.hbs?artid=169075  
434.37 Referer: http://www.vg.no/export/Transact/menu.html  
349.55 Referer: http://www.vg.no/pub/vgart.hbs?artid=169063  
344.66 Referer: http://www.vg.no/pub/vgart.hbs?artid=155682  
324.06 Referer: http://www.vg.no/export/Transact/top.hbs  
297.25 Referer: http://www.nettby.no/user/  
263.82 Referer: http://www.vg.no/sport/fotball/  
242.55 Referer: http://www.vg.no/pub/vgart.hbs?artid=169081
```


Varnishtop(1) – logfile "top" program

What is my most popular URL ?

```
$ varnishtop -i rxurl
```

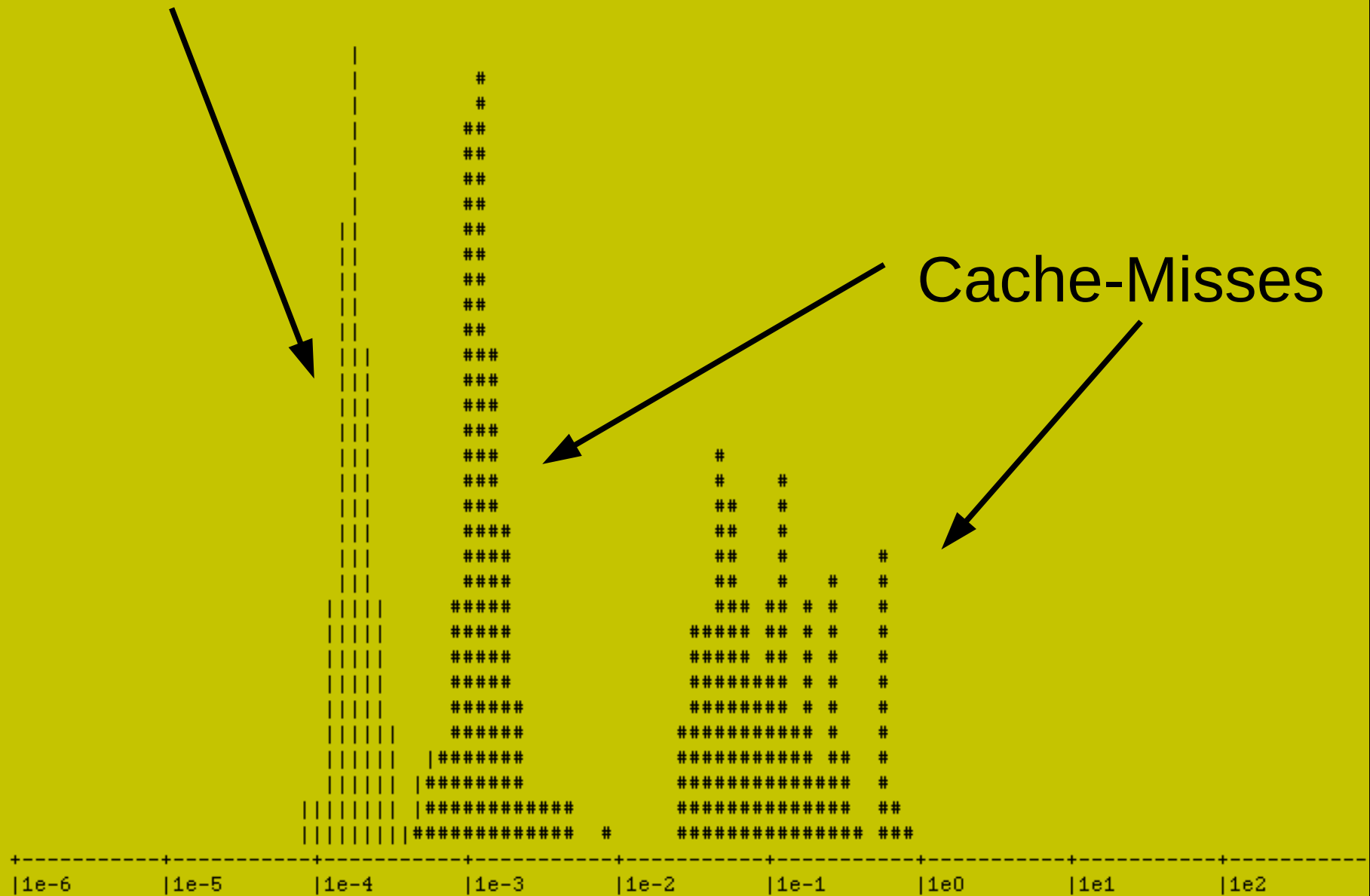
```
1304.86 /tmv11.js
 989.08 /sistenytt.html
 495.05 /include/global/art.js
 491.01 /css/hoved.css
 490.05 /gfk/ann/n.gif
 480.08 /gfk/ann/ng.gif
 468.12 /gfk/front/tipsvg.png
 352.66 /css/ufront.css
 317.75 /t.gif
 306.79 /gfk/plu2.gif
 298.84 /css/front.css
 292.84 /gfk/min2.gif
 280.94 /css/blog.css
 279.84 /
```

Varnishhist(1) - Response-time histogram

1:5, n = 2000

Cache-Hits

Cache-Misses



Real-time statistics via shared memory

16:23:13

Hitrate ratio:

9

9

9

Hitrate avg:

0.9986

0.9986

0.9986

17772105	435.55	301.26	Client connections accepted
130213161	3623.22	2207.26	Client requests received
129898315	3617.23	2201.93	Cache hits
85043	0.00	1.44	Cache hits for pass
227180	4.99	3.85	Cache misses
313630	4.99	5.32	Backend connections initiated
439	0.00	0.01	Backend connections recycles
54	0.00	0.00	Backend connections unused
6196	1.00	0.11	N struct srcaddr
1656	-24.97	0.03	N active struct srcaddr
3222	0.00	0.05	N struct sess_mem
2258	-51.95	0.04	N struct sess
65685	5.99	1.11	N struct object
65686	5.99	1.11	N struct objecthead

Content Management Features:

Instant action purges/bans (regexp or exact match)

TTL/Caching policy control in VCL

Load/Situation mitigation in VCL

Header Washing

Vary Washing ("Vary: User-Agent")

Edge-Side-Includes (ESI)

Purges:

Cache eviction based on exact criteria

Only through http transaction (= cache hit)

Can take all "Vary:" versions of object.

Bans:

Cache-hit prevention based on loose criteria

```
ban req.url ~ ".*royal.*naked.*"
```

CLI or http transaction

All the other stuff you can do in VCL

- TTL control
- cache/pass/pipe decision
- URL rewrites
- header washing
- IP based access control
- DoS prevention
- Spider-dieting
- mod_security-like screening
- &c &c.

And if VCL can not do it ?

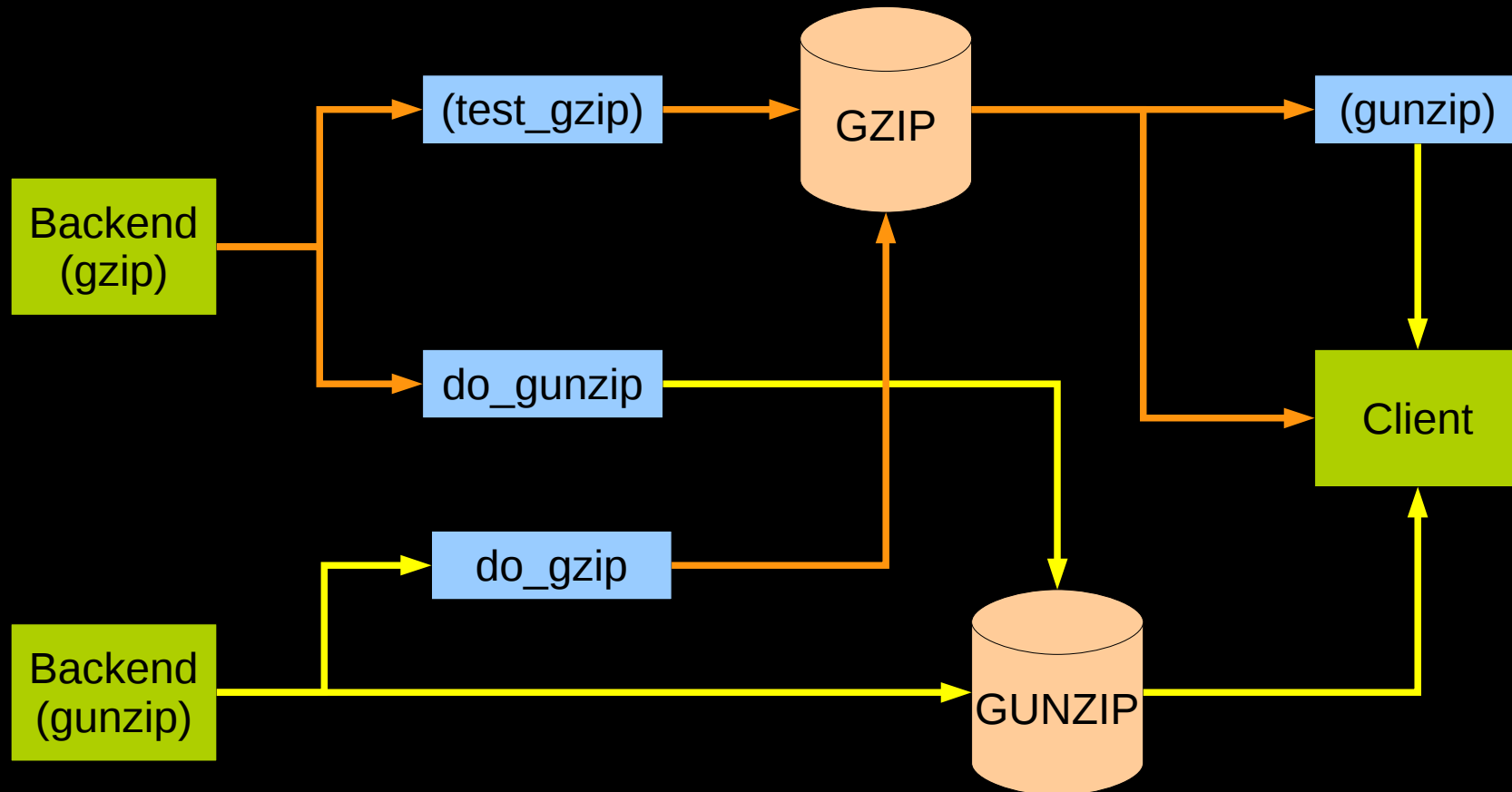
“Modules” in the form of shared libraries:

```
import std;  
  
req.url = std.tolower(req.url);
```

Inline C code:

```
sub vcl_recv {  
    C{  
        syslog(LOG_INFO, "Trouble: %u", foo);  
    }C  
}
```

GZIP support



Edge-Side-Includes ("ESI")

Per-user Page Scaffold, uncached (pass)

Index
TTL =
1 hour

Article

TTL = 1 month

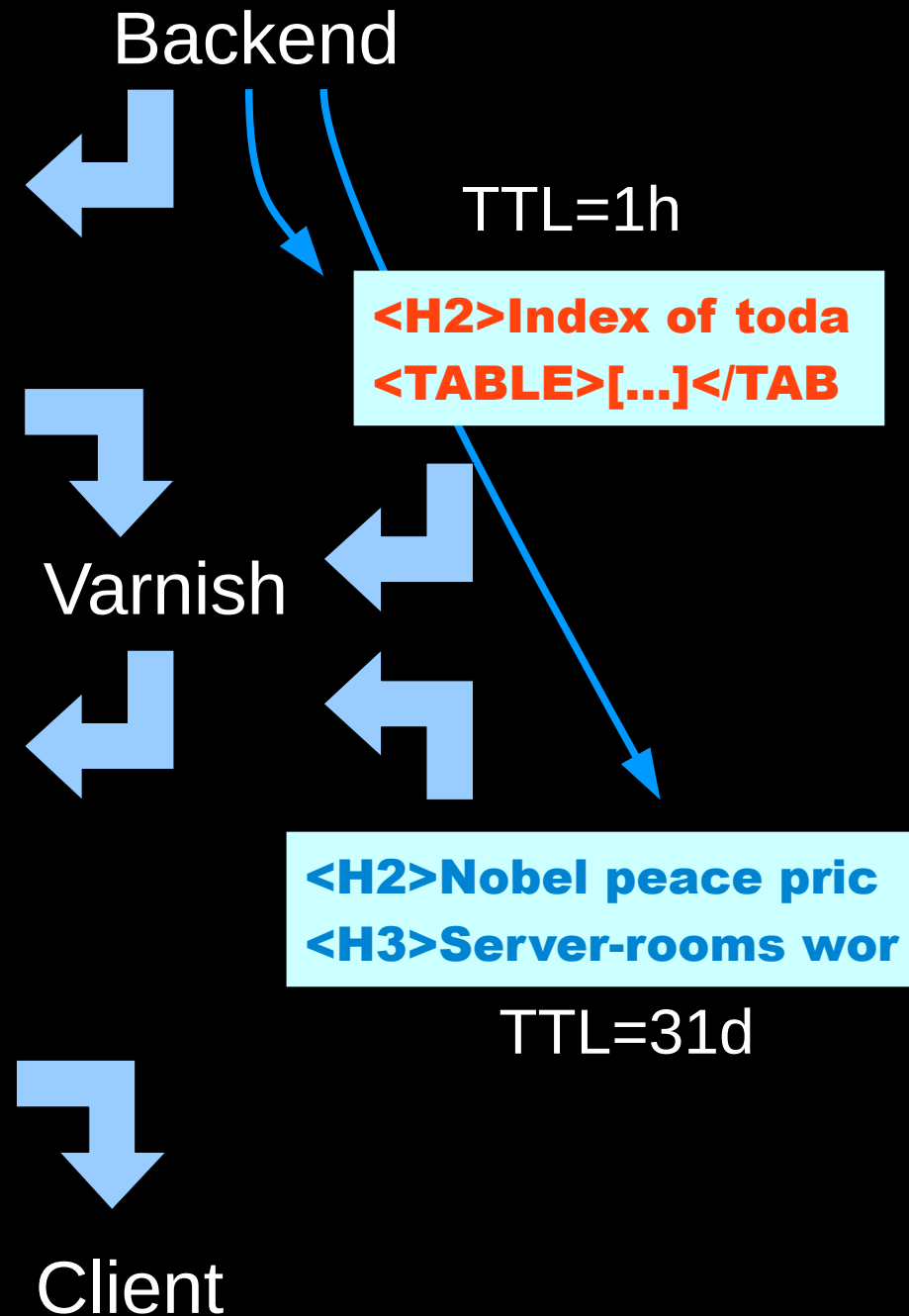
Top 5
Stories
TTL=
1min

Ad
(pass)

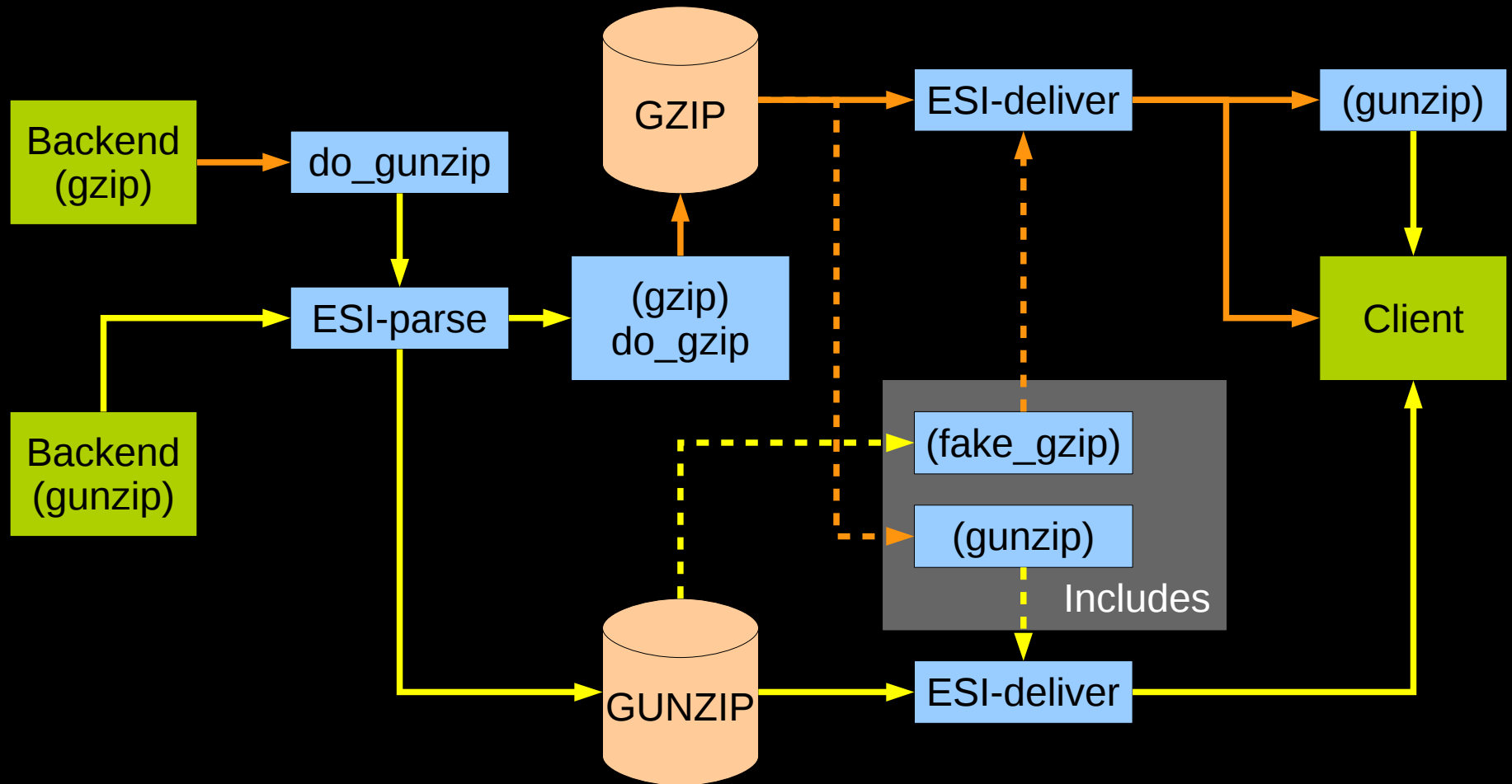
Feature
TTL = 1d

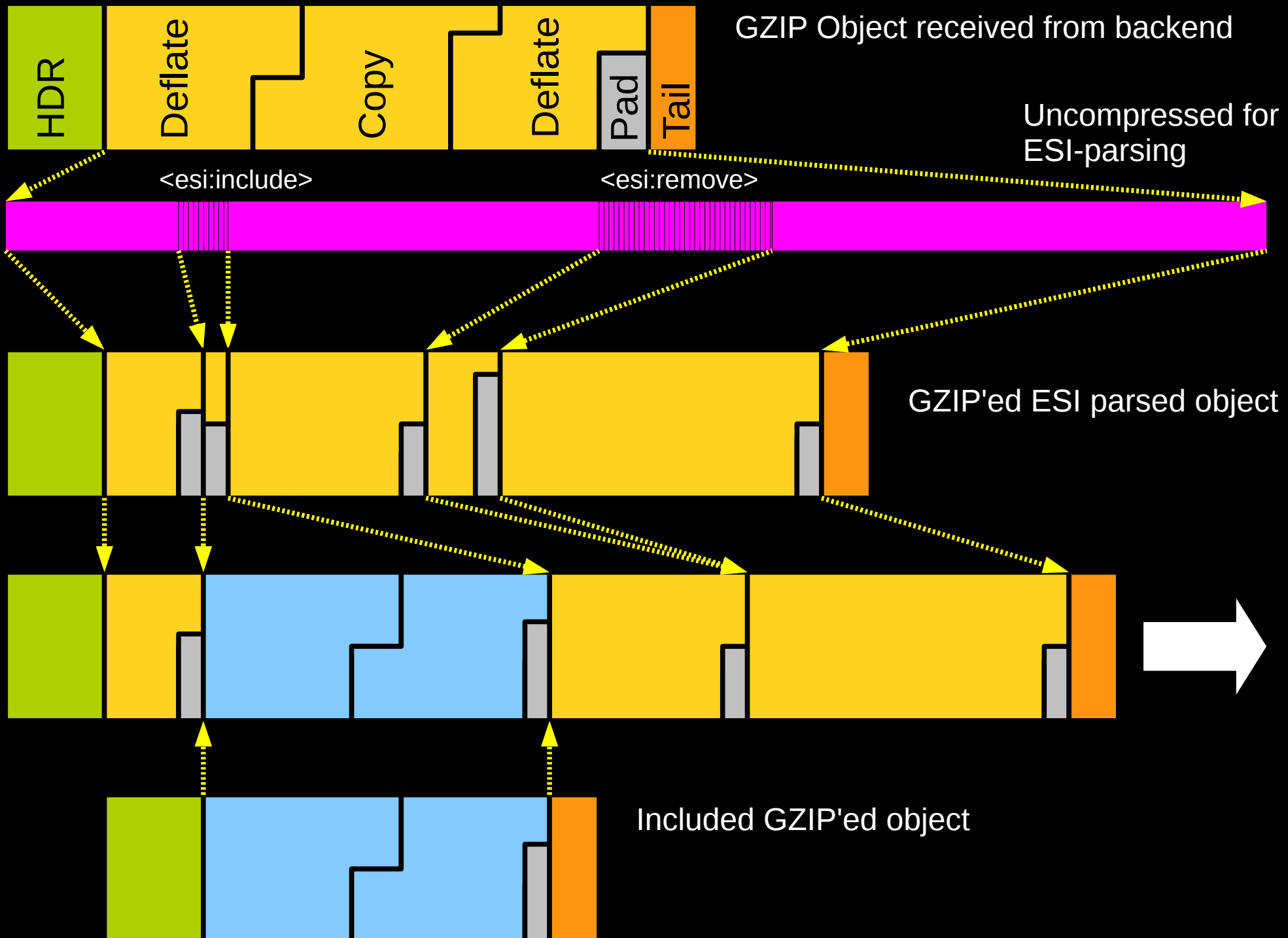
```
<html>
<H1>Hello Samuel B. Nobody</H1>
[...]
<esi include src="right_index.html">
[...]
<esi include src="article_1723.html">
[...]
TTL=30s
```

```
<html>
<H1>Hello Samuel B. Nobody</H1>
[...]
<H2>Index of todays top stories</H2>
<TABLE>[...]</TABLE>
[...]
<H2>Nobel peace price to software gen
<H3>Server-rooms world-wide silent af
[...]
TTL=30s
```



GZIP support with ESI





var·nish (vär'nĭsh) n.

1. a. A paint containing [...]

tr.v. **var·nished**, **var·nish·ing**, **var·nish·es**

1. To cover with varnish.

2. To give a smooth and glossy finish to.

3. To give a deceptively attractive appearance to.

And if I had more time I would also mention:

Round-Robin, Random, Client & Hash backend directors
Backend Health-polling, grace mode, saint mode
Modular storage-, hash-, waiter-code APIs
Libvarnishapi.a library for stats/log/cli access
Ipv6 support on all network connections
ACL's are compiled to C-code too = very fast
PSK security on CLI connection
Privilege separation/drop (manager/worker process)
Written entirely in C, only 64kloc (incl. 12kloc JEmalloc)
Cache hit = 7 system calls (typ: 15-30 μ s)
Almost 10% of source lines are asserts
85.2% of approx 50kloc covered by 276 test-cases
Builtin useful panic/backtrace formatter
Why marines.com sent email saying "You saved our ass"

Varnish Moral License:

<http://phk.freebsd.dk/VML>

From the FAQ:

Isn't this more a sort of sponsorship than a license ?

Somebody started the rumour that the name was chosen to sneak this expense into your IT departments budget, along with other software licenses, rather than put it under your marketing department, where all sponsorships belong and where it would compete with the local soccer team.

That rumour is very possibly true. I might even have started it myself. Possibly right here.

Varnish-cache.org:

← What you get

What you pay for

