Varnish
The HTTP Accelerator

www.varnish-cache.org

What is Varnish ?

Varnish is a HTTP accelerator for server-side caching of web-pages.

Many Content Management systems (CMS) suffer from performance problems on the HTTP-server side. This is a natural result of the complex content generation process which requies many database lookups and a CPU intensive page composition.

Varnish reduces the load on the CMS by caching frequently used pages, images and other web-objects and serving these cached copies to the clients.

Varnish is written bottom up to be a server side cache, designed to exploit computer hardware and operating system facilities to the very limit of their performance.

Varnish is *also* written as a content provider tool, to give full and precise control over the content and offers unlimited flexibility to improve on the service of the CMS system backend.

Varnish is designed for usability

The designers of Varnish have many years experience from the trenches and have tried to reflect the best and avoid the worst of these experiences in Varnish.

For instance Varnish is a single binary program, easy to install, easy to upgrade, easy to put on a machine at 2:05AM.

When running, varnish has two processes, a management process and a cacher process.

The manager process offers the administrator a Command Line Interface

(CLI) for operation and management etc.

The cacher process which does the real work is a child of the manager, and if for some reason it should crash, the manager will restart it, with the currently in force parameters and policy configuration.

Varnish server parameter configuration

The server parameters, TCP port numbers, file storage location and other such operational paramters. can be specified on the command like and many of them can be changed at runtime.

Varnish content policy configuration

Varnish does not enforce a particular policy with respect to handling of HTTP requests and content objects, but it does come with a default policy which will Do The Right Thing for any reasonably normal application.

When the default policy is not sufficient, Varnish allows the content manager to expres the desired policy with a domain specific programming language, VCL, which gives full control over how the content is handled.

VCL programs

When loaded, VCL programs are transparantly compiled to executable machine code and therefore run at full CPU speed when executed. Unlike traditional WEB languges like PERL, PHP, and Java, VCL runs fast and does not soak up resources.

A VCL program to add 5 minutes extra cache time for certain objects:

```
sub vcl_fetch {
        if (req.url ~ ”worldcup”) {
                obj.ttl += 5m;
        }
}
```

Multiple VCL programs can be loaded at the same time but only one of them is active at any one point. Switching to a different VCL program is an instant, cost-free operation, but all requests already started on the old VCL program will also complete under that VCL program.

Content Invalidation

Many CMS can issue "purge" requests to frontend caches like Varnish, and thanks to the VCL language and CLI interface, Varnish can be interfaced to them all.

Varnish offers instant invalidation based on exact URL matching, but also based on regular expressions.

Instantly invalidating all articles in the Varnish cache is as simple as sending the CLI command:

```
url.purge .*html$
```

Logging and statistics

Varnish writes all log information and statistics to shared memory, at very high speed and practically no cost.

Separate programs can "`tail -f`" the shared memory file and produce NCSA log files, real-time displays or trawl for specific events almost like `tcpdump` does for a network.

Varnish provides a C language library for interfacing to the shared memory information making it easy to write custom log data extractors.

Because shared memory logging is practically free, Varnish logs full detail on each request, for instance all headers received *and* sent to both client *and* backend are recorded, along with IP numbers, portnumbers etc.

Performance

Varnish is written to exploit modern hardware and operating systems features like SMP, multi-core, virtual memory, sendfile, 64bit, epoll, kqueue etc.

We do not know yet how fast Varnish is, here are some of the numbers we do know:

- From a complete HTTP request is received to the response rs ready for transmission, a cache hit will typically take 25 µseconds (one 40.000th of a second).

- A cache-hit typically takes 12 system calls to service.
- One Varnish machine can replace 5 to 10 Squid machines.
- One 3 Ghz Xeon can service 18.000 hits/s (1 kilobyte each)
- A dual 2.4 Ghz Opteron can do 10.000 req/s, 700 Mbit/s and still have approx. 40% idle CPU.

Varnish license and portability.

Varnish is open source software, distributed under the BSD license.

Varnish should run any any reasonably modern UNIX. Linux 2.6 and FreeBSD 6.x are our reference platforms.

The Varnish History

The major Norvegian newspaper"Verdens Gang' (www.vg.no) sponsored Varnish version 1.0. Poul-Henning Kamp of FreeBSD fame designed and coded. Linpro.no assisted and provides the project infrastructure.

We hope Varnish users will pool together money to support the future development of Varnish.

Why is it called "Varnish" ?

> **Varnish** tr. v. varn·nished, varn·nish·ing, varn·nish·es. 1. To cover with varnish. 2. To give a smooth and glossy finish to. 3. **To give a deceptively attractive appearance to.**

Varnish makes your webserver look good.

How to start Varnish (at 2:05AM)

```
Varnishd -b 10.0.0.1
```

Varnish will use up to 50% of the diskspace in `/tmp`, listen on port 80 and use `10.0.0.1` as backend server. Redirect your traffic to the Varnish machine and go back to sleep.