

Varnish deluxe – Geil, wir haben einen Cache! Und nun?

Einblicke und brauchbare
Tipps aus der Praxis.

Kevin Trieloff
Hamburg, 14. November 2017





Inhalt

1. Varnish: ein kurzer Überblick
2. SSL: Nein?!? Doch!!! Oh...
3. Das Geheimnis der Cookies
4. Einmal im Cache, immer im Cache?
5. Dynamischen Content gibt es nicht mehr? Doch!
6. Fazit



Varnish Ein kurzer Überblick.



In aller Kürze. **Varnish? Was ist das?**

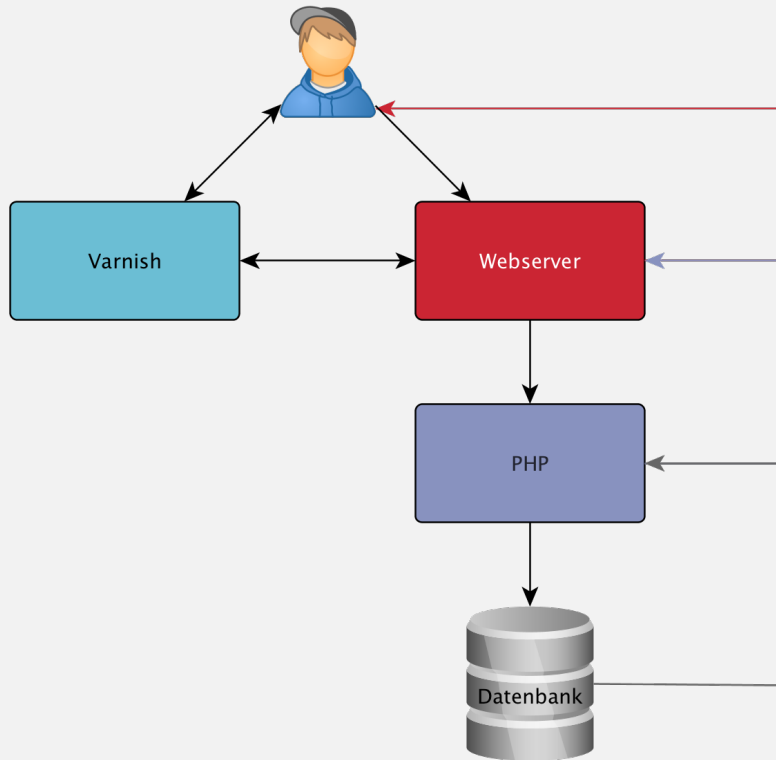
- Webbeschleuniger
- Von Grund auf als solcher konzipiert
- Im Einsatz u. a. bei Facebook, Twitter, Wikipedia, Eurosport*
- Ursprung: norwegische Boulevardzeitung Verdens Gang
- Hauptentwickler: Poul-Henning Kamp
- BSD-Lizenz
- Aktuelle Version: 5.2.0 vom 15. September 2017
- Offizielle Website: <https://varnish-cache.org>

* Quelle: <https://www.varnish-software.com/case-studies/>



In aller Kürze. **Technisch beleuchtet.**

- Daten im virtuellen Speicher (VMM)
- Thread-basiert
- Konfiguration mithilfe der Varnish Configuration Language (VCL)
- VCL wird nach C übersetzt
- Viele config-Parameter während der Laufzeit anpassbar
- Lastverteilung via Zufall oder Round Robin
- Index: URL (inkl. eventueller Parameter)



In aller Kürze. **Varnish aktivieren.**

- Datei im Varnish:
 - Direkte Auslieferung
 - Keine Webserver-Beteiligung
- Datei nicht oder nicht mehr im Varnish (neue URL oder abgelaufene Cache-Vorhaltezeit):
 - Anforderung an Webserver
 - Standard PHP-Rendering
 - DB-Abfragen
 - → Datei im Varnish ablegen

In aller Kürze. **Varnish aktivieren.**

- Globale Regel(n):
 - Caching aller Ressourcen
 - Durch VCL-Regeln unterschiedliche Zeiten (TTL) möglich
- Flexibilität durch HTTP-Header:
 - `cache-control` (Sekunden)
 - `Expires` (Timestamp)

http/1.1

Überschreibt http/1.0 Direktiven

```
Cache-Control: public, max-age=120, s-maxage=600
```

http/1.0

Heute noch notwendig?

```
Expires: Tue, 14 Nov 2017 20:00:00 GMT
```



In aller Kürze. **Varnish aktivieren.**

- Regeln denkbar für
 - Dateityp (Dateiendung)
 - MIME-Type
 - jede andere durch RegEx abbildbare Regel



SSL.
Nein?!? Doch!!! Oh...



Ist das euer Ernst? **Keine SSL-Unterstützung.**

Poul-Henning Kamp*:

- OpenSSL-Code ist ein Alptraum
- SSL und TLS ist wahnsinnig komplex
- Integration würde mindestens sechs Monate dauern
- Politische Gründe

* Quelle: https://varnish-cache.org/docs/trunk/phk/ssl_again.html

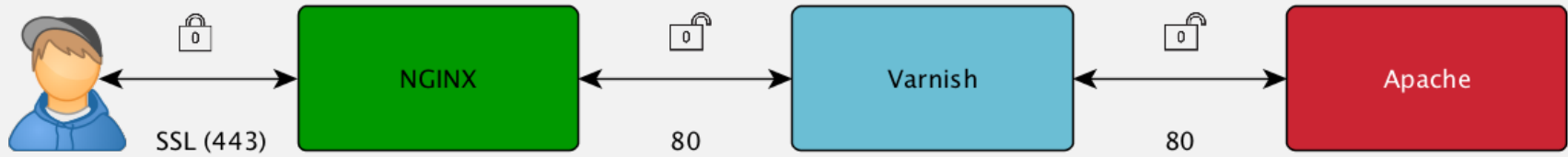


Die Lösung: **Proxy, z. B. NGINX.**

- Sehr schnell*
- Erprobt
- Stabil
- In der Plus-Version auch mit eigenem Cache, allerdings
 - nicht so flexibel
 - kein Komponenten-Caching möglich
 - nicht kostenlos (2 Stages mind. \$6.000 pro Jahr)**

* Quelle: z. B. <https://timmehosting.de/benchmarks>

** Quelle: <https://www.nginx.com/products/buy-nginx-plus/>



NGINX als Proxy.

SSL wird über NGINX terminiert (443) und intern an Varnish und Apache auf Port 80 geroutet



Das Geheimnis der Cookies.
Cookie dabei, am Cache vorbei.



Cookie dabei – am **Cache vorbei.**

- Bypass: Varnish cacht keine Seiten mit Cookie
- Grund: individuelle Daten, bspw. Sessions
- Weder im Request-Header
- noch im Response-Header

Was tun?



Cookie dabei – am Cache vorbei. **Das muss nicht sein!**

- Durch VCL-Konfiguration sind Ausnahmen möglich
- Trifft Ausnahme zu: Cookie-Objekt löschen
- Regel für Cookies ohne Auswirkungen auf den Cache bei uns:
 - beginnen mit Unterstrich ()

```
if (req.http.Cookie) {  
  
    set req.http.Cookie =  
        regsuball(req.http.Cookie,  
            "(^|;\s*)(_[^=]+|has_js)=[^;]*", "");  
  
}
```



Einmal im Cache, immer im Cache?
Purging & Banning.



Purging. **Cache gezielt leeren.**

- Standard-Mechanismus
- Gezielt einzelne Dateien aus dem Cache entfernen
- http PURGE Request
- via Kommandozeile bspw. mithilfe von cURL

```
curl -X PURGE [URL]
```

```
curl -X PURGE  
http://www.domain.de/meine/datei.txt
```



Purging. **Cache gezielt leeren.**

Empfehlung: IP-Sperre

VCL:

```
acl purge {  
    "localhost";  
    "203.0.113.0";  
}
```

```
if (req.method == "PURGE") {  
    if (!client.ip ~ purge) {  
        return (synth(405,"Not allowed.));  
    }  
}
```



Banning. **Die Wildcard-Methode.**

- Standard-Mechanismus
- ebenfalls http PURGE Request
- per RegEx beliebige und beliebig viele Dateien löschen
- Weitere(r) Parameter benötigt
- Empfehlung: Secret-Key (zusätzlich zur IP-Sperre)



Banning. **Zusätzliche Parameter.**

- X-Ban-Authorize
 - Secret-Key
 - beliebige Zeichenfolge
- X-Ban-Regexp
 - Regulärer Ausdruck
 - z. B. "\\..png\$"



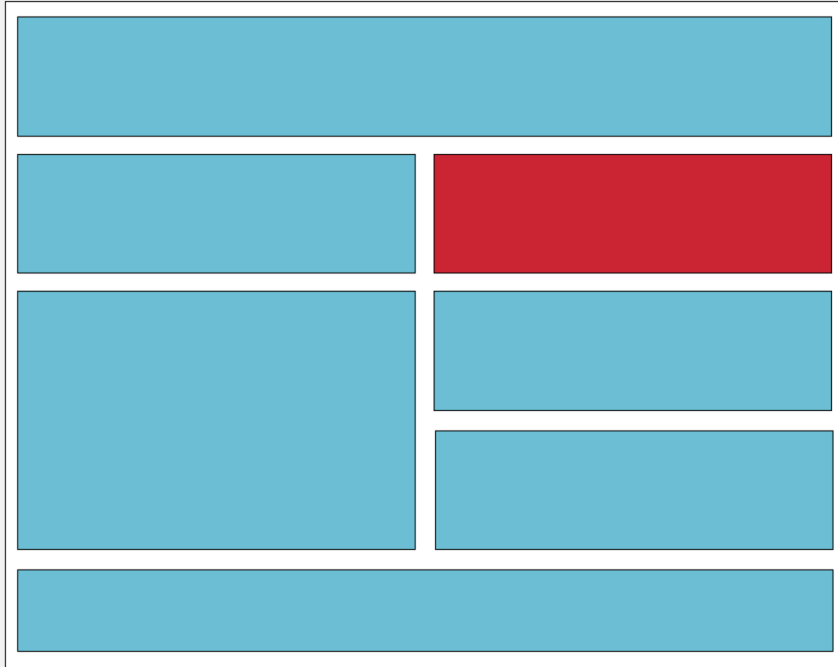
```
if (req.method == "PURGE") {  
  
    if (!client.ip ~ purge) {  
        return (synth(405,"Not allowed.));  
    }  
  
    if ( req.http.X-Ban-Regex ~ ".+" ) {  
  
        if ( ! req.http.X-Ban-Authorize == "mein$trengGehelmerString" ) {  
            return (synth(405,"Not allowed.));  
        }  
  
        ban("req.http.host ~ " + req.http.host + " && req.url ~ " + req.http.X-Ban-Regex );  
        return (synth(200, "Banned " + req.http.X-Ban-Regex));  
  
    }  
  
}
```

Banning



inklusive IP-Sperre und Secret Key

<esi/>

Dynamischen Content gibt es nicht mehr? Doch!
Edge Side Includes (ESI).



Edge Side Includes (ESI). Cache und dynamischer Content.

- Fragmente direkt ausliefern 
- Alle andere Teile der Website weiterhin aus dem Cache 
- Nomen est omen: Snippets (Includes)
- Sinnvoll oder gar notwendig für volatile Elemente wie Preis oder Verfügbarkeit
- ⚡ ESI muss im Varnish aktiviert werden.
- ⚡ ESI-Parsing nur innerhalb von HTML-Seiten
 - mindestens benötigt:
 - `<html>`
 - `<body>`

Edge Side Includes (ESI).

ESI-Tags: esi:include.

- Includes werden mithilfe von ESI-Tags eingebunden
- Optisch ähnlich HTML- oder XML-Tags
- esi:include bindet Snippets ein:

```
<esi:include src="inc/myInclude.php"/>
```

- Einbettung wie im PHP-Umfeld gewohnt
- PHP-Interpreter gibt Snippet an Varnish weiter (Bypass)

Edge Side Includes (ESI).

ESI-Tags: esi:remove.

- esi:remove greift, wenn ESI nicht funktionieren:
 - Fehlkonfiguration
 - gewünscht, bspw. auf Staging-Umgebung

```
<esi:remove>
  <?php include_once("inc/myinclude.php"); ?>
  <strong>Achtung, kein ESI!</strong>
</esi:remove>
```

- Wenn ESI-Prozessor aktiv, keine Ausgabe
- andernfalls in diesem Beispiel PHP-Interpreter + Hinweis

Fazit.

Varnish rockt!

- Schnelle Installation = schnell erste Ergebnisse
- Reverse Proxy: einfach vor nahezu jede Anwendung einrichtbar
- Flexibler als viele andere Caches
- Hot Reload (dynamische Konfiguration)
- Businesslogik via VCL möglich
- Viele Module* erhältlich, z. B.
 - A/B-Test
 - Drupal-Funktionen
 - LDAP-Authentifizierung

* Quelle: <https://varnish-cache.org/vmods/>

Fragen?

Fragen!



Vielen Dank.

Kevin Trieloff

Teamleiter Entwicklung

trieloff@for-sale-digital.de

<http://www.for-sale-digital.de>

Bildnachweis

- Folie 1: <https://varnish-cache.org>
Folie 3: <https://worldvectorlogo.com/logo/varnish>
Folie 9: <https://i.ytimg.com/vi/OL8Eh2XLp80/hqdefault.jpg>
Folie 10: <http://www.freeiconspng.com/img/4979>
Folie 11: https://commons.wikimedia.org/wiki/File:Nginx_logo.svg
Folie 13: https://upload.wikimedia.org/wikipedia/ru/b/be/Cookie_Monster_Pointing_Right_Backat.png
Folien 14 & 15: <http://pluspng.com/png-22583.html>
Folie 16: https://commons.wikimedia.org/wiki/File:Broom_icon.svg
- Diagramme: © by Kevin Trieloff, Tool: [yED](#)