HTTP HEADERS THAT WILL MAKE YOUR WEBSITE GO FASTER

BY THIJS FERYN
Slow websites SUCK
WEB PERFORMANCE IS AN ESSENTIAL PART OF THE USER EXPERIENCE
SLOW ~ DOWN
THROWING SERVERS AT THE PROBLEM
MO' MONEY
MO' SERVERS
MO' PROBLEMS
IDENTIFY SLOWEST PARTS
OPTIMIZE
AFTER A WHILE YOU HIT THE LIMITS
CACHE
HI, I'M THIJS
I'M THE TECH EVANGELIST
AT VARNISH SOFTWARE
About Varnish Software

We deliver enterprise solutions based on open source Varnish Cache

10M active websites worldwide powered by Varnish

22% of the world’s top 10,000 sites use Varnish

10M pulls on Docker hub of the Varnish image

2005 - Open source Varnish Cache project begins

2014 - Varnish Enterprise solution launched to support content-heavy, high-traffic businesses

2010 - Varnish Software spun out of Redpill Linpro

2020 - Varnish Edge Cloud launched to support 5G content delivery within telco networks

2017 - Varnish start delivering custom, private CDNs to enterprises
WE MAKE THE WEB FASTER
WE ENHANCE DIGITAL EXPERIENCES BY LOWERING NETWORK LATENCY
WE BUILD **SOFTWARE-DEFINED**
WEB ACCELERATION & CONTENT DELIVERY SOLUTIONS
ACHIEVE GROWTH, PERFORMANCE & SUSTAINABILITY GOALS

1.3 Tbps per server

1.17 Gbps per watt

WORLD’S FASTEST EDGE CONTENT DELIVERY SOFTWARE
VARNISH 6
BY EXAMPLE

A practical guide to web acceleration and content delivery with Varnish 6 technology
CACHING
WHY CACHE
HIGHER CONCURRENCY
HIGHER THROUGHPUT
IMPROVE QUALITY OF EXPERIENCE
WHY RECOMPUTE IF THE DATA HASN'T CHANGED?
DIFFERENT KINDS OF CACHING

✓ LOCAL KEY-VALUE STORE
✓ FILE CACHE
✓ DISTRIBUTE CACHE
✓ BROWSER CACHE
✓ REVERSE CACHING PROXY
✓ CONTENT DELIVERY NETWORK
BROWSER CACHE

USER

SERVER
UNDER PRESSURE

SERVER
THE EDGE

USER   VARNISH   SERVER
THE ORIGIN

USER  VARNISH  SERVER
EVERY IMPLEMENTATION HAS ITS OWN CACHE POLICY CONFIGURATION
HTTP HAS CONVENTIONAL BUILT-IN CACHING MECHANISMS
LIMITED OPTIONS
Cache-Control: private, no-cache, no-store
CACHING

HOLD THE RESPONSE AND SERVE IT AGAIN UPON SUBSEQUENT REQUESTS
PRIVATE CACHE

A CACHE THAT EXISTS IN THE CLIENT.
E.G. A LOCAL DEVICE OR BROWSER CACHE.
STORES DATA FOR A SINGLE USER.
SHARED CACHE

A CACHE THAT SERVES MULTIPLE USERS. USUALLY A CACHING PROXY OR CDN. YOU SHOULD AVOID STORING PERSONALIZED DATA.
TIME TO LIVE

THE AMOUNT OF SECONDS AN OBJECT IS CONSIDERED FRESH.
CACHED OBJECT HASN'T EXPIRED YET. RESPONSE CAN BE REUSED FOR SUBSEQUENT REQUESTS.
STALE CONTENT

EXPIRED CONTENT THAT SHOULD BE REVALIDATED BEFORE SERVING. IS NOT DIRECTLY REMOVED FROM THE CACHE.
REVALIDATE CONTENT

ASK THE ORIGIN SERVER IF THE REQUESTED OBJECT IS STILL FRESH.
CACHE-CONTROL RESPONSE DIRECTIVES

✓ PRIVATE
✓ PUBLIC
✓ IMMUTABLE
✓ MAX-AGE
✓ S-MAXAGE
✓ NO-CACHE
✓ NO-STORE
✓ NO-TRANSFORM
✓ MUST-REVALIDATE
✓ PROXY-REVALIDATE
✓ MUST-UNDERSTAND
✓ STALE-WHILE-REVALIDATE
✓ STALE-IF-ERROR
Cache-Control: public
Cache-Control: public

CACHING ALLOWED, BOTH BY PRIVATE & SHARED CACHES
CACHING ALLOWED, BOTH BY PRIVATE & SHARED CACHES
Cache-Control: private
Cache-Control: private

CACHING ALLOWED, BUT ONLY BY PRIVATE CACHES
Cache-Control: private=Set-Cookie
Cache-Control: private=Set-Cookie

CACHING ALLOWED BY ALL CACHES, UNLESS A SET-COOKIE HEADER IS SET. THEN THE RESPONSE IS ONLY HANDLED BY PRIVATE CACHES
Cache-Control: public, max-age=100
Cache-Control: public, max-age=100

ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 100 SECONDS.
Cache-Control: private, max-age=100
Cache-Control: private, max-age=100

ONLY PRIVATE CACHES ALLOWED.
CONTENT IS FRESH FOR 100 SECONDS.
Cache-Control: public, s-maxage=100
Cache-Control: public, s-maxage=100

ALL CACHES ALLOWED.
CONTENT IN SHARED CACHES IS FRESH FOR 100 SECONDS.
Cache-Control: public, max-age=60, s-maxage=100
Cache-Control: public, max-age=60, s-maxage=100

ALL CACHES ALLOWED.
CONTENT IN PRIVATE CACHES IS FRESH FOR 60 SECONDS.
CONTENT IN SHARED CACHES IS FRESH FOR 100 SECONDS.
Cache-Control: public, max-age=60
Age: 40

AGE HEADER DESCRIBES THE TIME IN SECONDS THE OBJECT WAS IN A PROXY CACHE.
Remaining TTL = TTL - Age
REVALIDATION
Is the content still fresh?
CONDITIONAL REQUESTS
HTTP/1.1 304 Not Modified
GET / HTTP/1.1
Host: localhost

HTTP/1.1 200 OK
Host: localhost
Etag: 7c9d70604c6061da9bb9377d3f00eb27
Content-type: text/html; charset=UTF-8

Hello world output
GET / HTTP/1.1
Host: localhost
If-None-Match: 7c9d70604c6061da9bb9377d3f00eb27

HTTP/1.1 304 Not Modified
Host: localhost
Etag: 7c9d70604c6061da9bb9377d3f00eb27
GET / HTTP/1.1
Host: localhost

HTTP/1.1 200 OK
Host: localhost
Last-Modified: Fri, 22 Jul 2016 10:11:16 GMT
Content-type: text/html; charset=UTF-8

Hello world output
GET / HTTP/1.1
Host: localhost
If-Last-Modified: Fri, 22 Jul 2016 10:11:16 GMT

HTTP/1.1 304 Not Modified
Host: localhost
Last-Modified: Fri, 22 Jul 2016 10:11:16 GMT
VALIDATE QUICKLY
EARLY
ASYNCHRONOUS REVALIDATION
SEND STALE RESPONSE WHILE FETCHING

ASYNC FETCH

USER          PROXY          ORIGIN

ASYNC FETCH
Cache-Control: public, max-age=3600, stale-while-revalidate=100
Cache-Control: public, max-age=3600, stale-while-revalidate=100

ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 1 HOUR.
STALE CONTENT CAN BE SERVED UP TO 100 SECONDS PAST THE TTL WHILE DOING AN ASYNCHRONOUS REVALIDATION.
SEND STALE RESPONSE WHILE FETCHING FAILS

FETCH FAILED

USER  PROXY  ORIGIN
Cache-Control: public, max-age=3600, stale-if-error=86400
Cache-Control: public, max-age=3600, stale-if-error=86400

ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 1 HOUR.
STALE CONTENT CAN BE SERVED UP TO 1 DAY PAST THE TTL
WHILE THE ORIGIN IS UNREACHABLE.
Fresh = TTL > 0
Async revalidation = TTL + stale > 0
Synchronous revalidation = TTL + stale <= 0
Fresh = TTL > 0
Async revalidation = TTL + stale > 0
Synchronous revalidation = TTL + stale <= 0

Revalidation can be done conditionally.
Cache-Control: public, max-age=3600, must-revalidate
Cache-Control: public, max-age=3600, must-revalidate

ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 1 HOUR.
SERVING STALE CONTENT NOT ALLOWED.
Cache-Control: public, max-age=3600, proxy-revalidate
Cache-Control: public, max-age=3600, proxy-revalidate

SAME AS MUST-REVALIDATE BUT FOR PROXY SERVERS
Cache-Control: public, max-age=86400, immutable
Cache-Control: public, max-age=86400, immutable

ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 1 DAY.
CONTENT WILL NOT BE UPDATED WHILE FRESH
ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 1 HOUR.
CONTENT WILL NOT BE UPDATED WHILE FRESH

Cache-Control: public, max-age=86400, immutable

USEFUL FOR BROWSER CACHING
Cache-Control: public, **immutable**

**DATA IS IMMUTABLE, ASSUME HIGH TTL IN PROXY CONTEXT**
Cache-Control: public, max-age=3600, no-transform
Cache-Control: public, max-age=3600, no-transform

ALL CACHES ALLOWED.
CONTENT IS FRESH FOR 1 HOUR.
CONTENT CANNOT BE TRANSFORMED BY INTERMEDIARY CACHES
Cache-Control: public, max-age=3600, no-transform

EDGE COMPUTE
Cache-Control: no-cache
Cache-Control: no-cache

STORE OBJECT IN CACHE
BUT REVALIDATE BEFORE EVERY REUSE
Cache-Control: no-cache=Set-Cookie

STORE OBJECT IN CACHE
BUT REVALIDATE BEFORE EVERY REUSE
IF THE SET-COOKIE HEADER IS SET
Cache-Control: no-store
Cache-Control: no-store

DON'T STORE OBJECT IN THE CACHE
Cache-Control: private, no-cache, no-store
CACHE VARIATIONS
Vary: Accept-Language
GET / HTTP/1.1
Host: test.com
Accept-Language: fr

GET / HTTP/1.1
Host: test.com
Accept-Language: en

http://test.com/
- Accept-Language: fr
- Accept-Language: nl
- Accept-Language: en
Vary: Accept-Encoding, Accept-Language, X-Forwarded-Proto
SURROGATES
THE EDGE
THE EDGE IS NO LONGER IN THE ORIGIN DATA CENTER
THE EDGE MOVES CLOSER TO THE END USER
Surrogate-Control: max-age=300
Surrogate-Control: max-age=300+100
Surrogate-Control: max-age=300+100
Surrogate-Control: no-store
Surrogate-Control: no-store-remote,
max-age=3600
SURROGATE CAPABILITY
Surrogate-Capability: key="ESI/1.0"
Surrogate-Control: content="ESI/1.0"
Surrogate-Capability: varnish="ESI/1.0"
Surrogate-Control: max-age=60, max-age=86400;varnish, max-age=3600;cdn, content="ESI/1.0";varnish
SEPARATE HTTP REQUEST
AJAX
EDGE-SIDE INCLUDES ESI
<esi:include src="/header" />
ESI

✓ PLACEHOLDER
✓ PARSED BY EDGE CACHE [VARNISH]
✓ OUTPUT IS A COMPOSITION OF BLOCKS
✓ STATE PER BLOCK
✓ TTL PER BLOCK
Surrogate-Capability: key="ESI/1.0"

Surrogate-Control: content="ESI/1.0"

<esi:include src="/header" />

Parse ESI placeholders
Welcome
The current time is 21:07:53.
Welcome
THIS IS JUST THE TIP OF THE ICEBERG
EVERY IMPLEMENTATION HAS ITS OWN CACHE POLICY CONFIGURATION

REMEMBER THIS ONE?
VARNISH CONFIGURATION LANGUAGE
Welcome to the Varnish Developer Portal by Varnish Software. This is the place to learn about Varnish and VCL. This portal provides tutorials, install guides, documentation references, information about the official book and ways to engage with us and get help.

What are you looking for?

Quick start
Want to get going right away? Here you can find a quick start guide to learn the basics of Varnish and be up-and-running in minutes.
Get started →

Downloads
Download Varnish Cache 6.0 LTS and other utilities. We provide official Linux packages, images for Docker and various cloud platforms.
Go to downloads →

Varnish 6 by Example Book

Documentation
HTTPS://VARNISH-SOFTWARE.COM/DEVELOPERS
THE END