Going crazy with Varnish

Caching pages of logged in users

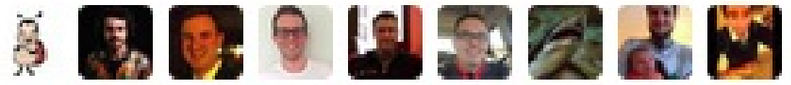
Confoo, Montreal, Canada
February 21st, 2024

Greg Brockman
@gdb

Follow

Web programming is the science of coming up with increasingly complicated ways of concatenating strings.

RETWEETS	LIKES
1,370	380

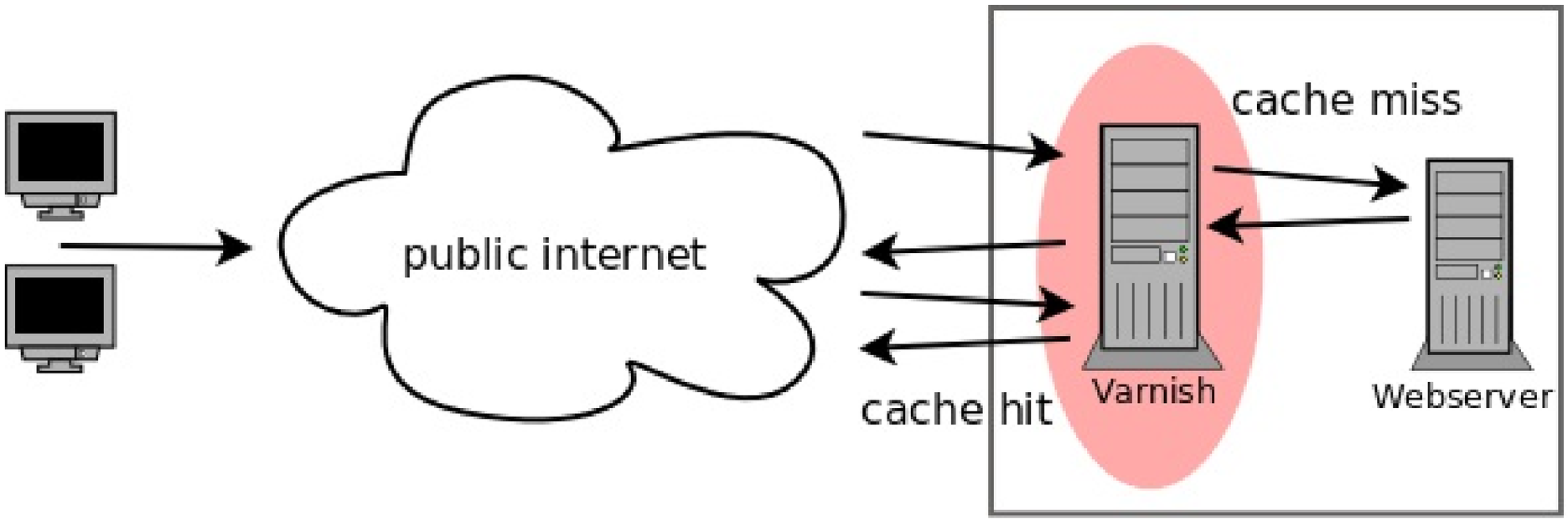


10:49 PM - 17 Dec 2011

HTTP Caching

- Why re-render content that does not change?
- Scaling and response time
- Less servers, more users

What is a reverse proxy again?



What about the real live?

Abonnieren ANMELDEN

Personal Information

Location based

Zürich -9°C / -4°C

Neue Zürcher Zeitung

E-Paper / Webpaper

MENÜ | Startseite | Meinung | International | Wirtschaft | Finanzen | Schweiz | Feuilleton | Zürich | Sport | Wissenschaft | Panorama

Kanadas Premierminister

Der Anti-Trump

von Martin Beglinger / 20.1.2017

Kein Regierungschef in der westlichen Welt verkörpert ein schärferes Gegenprogramm zu Donald Trump als Justin Trudeau. Wer ist dieser vermutlich politisch korrekteste Premierminister der Welt?

Was heute wichtig ist

Updated frequently

Zuletzt aktualisiert um 13:23 Uhr

Donald Trump vor Amtsübernahme / «El Chapo» an die USA ausgeliefert / Acht Überlebende in verschüttetem Hotel in Abruzzen gefunden / Autofahrer rast in Melbourne in Menschenmenge

NEWS-TICKER

User profile specific

Ukraine-Krise

Ukraine stoppt Mauerbau an der Grenze zu Russland

vor 17 Minuten

Schneechaos in Spanien

Hunderte verbringen die Nacht auf eingeschneiter Autobahn

vor 36 Minuten

Krieg in Syrien

Dutzende von Jihadisten bei Luftangriff getötet

vor 3 Stunden

Krieg in Syrien

IS zerstört weiteres Monument in Palmyra

vor 3 Stunden

earch

Pull requests Issues Gist

Personal informatio

ymfony / **FOSHttpCache**

Unwatch 15 Star

Personal, repo specific

Varnishadm CLI client #231

vants to merge 1 commit into `master` from `varnish-admin`

Commits 1 Files changed 11

Role specific +37

ommented on 5 Jul

Owner

dea is that using this client involves less custom VCL. However, for the ban lurker to work, some m VCL is still necessary to copy e.g. the URL header to `X-Url`.

as I know, purge cannot be done through varnishadm, so we should either not implement `eInterface` or fake the purge by actually doing a ban when `purge()` is called.

goes for refresh. We could fake it by doing a ban and then manually getting a fresh response

Labels

None yet

Milestone

2.0

Assignee

No one—a

Caching content that is not the same for all users

- Avoid Sessions, remove when no longer needed
- Build information in the frontend:
  "Cheating" with Javascript
- Cache despite cookies / cache per user
- User Context: Cache by relevant group

Avoid Sessions

Avoid Session

- Delete cookie as soon as no longer needed
- Watch out for autostarted PHP sessions

```twig
{% if app.request.hasPreviousSession %}
    {% for message in app.flashes('notice') %}
        <div class="flash-notice">
            {{ message }}
        </div>
    {% endfor %}
{% endif %}
```

Cleanup Cookies: Remove all but session cookie

```
sub vcl_recv {
    # using a capturing sub pattern, extract the continuous string of
    # alphanumerics that immediately follows "PHPSESSID="
    set req.http.X-Varnish-PHP_SID = regsuball(req.http.Cookie,
                                ";? ?PHPSESSID=([a-zA-Z0-9]+)( |;| ;).*","\1");
    if (req.X-Varnish-PHP_SID != "") {
        set req.http.Cookie = req.X-Varnish-PHP_SID;
    } else {
        unset req.http.Cookie;
    }
    unset req.X-Varnish-PHP_SID;
}
```

https://www.varnish-cache.org/docs/4.1/users-guide/increasing-your-hitrate.html

Logic in the Frontend

Render varying parts in Javascript

- User state is communicated to Javascript in cookie
- Alter page when loaded:

```
1 $(document).ready(function () {
2     if (is_editor()) {
3         $("#labels-edit").show();
4         $("#milestone-edit").show();
5         $("#assignee-edit").show();
6     }
7 });
```

Logic in the Frontend

- Pro
  - Fine grained control
  - No network overhead, easy on backend
  - Can handle complex user interfaces
- Con
  - Templating in frontend and backend?
  - Development effort and hard to maintain
  - Additional data needs to be fetched separately

Ajax for User Specific Parts

Ajax

- Separate endpoint in application for each fragment
- Do request when page loaded:

```
1 $(document).ready(function () {
2     $.ajax({
3         url: "/sidebar.html"
4     }).done(function( html ) {
5         $( "#sidebar" ).append( html );
6     });
7 });
```

Ajax

- Pro
  - Can be executed after page already displayed
  - Good for slow, non-critical information
- Con
  - Network overhead (mobile…)
  - Lots of requests to the backend
  - Additional logic in the backend
  - Complicated to maintain

Caching despite cookies

Caching despite cookies

- Very easy to shoot yourself in the foot!
- The backend must send correct cache headers
  - Not cacheable:
    `Cache-Control: no-cache`
  - Depends on session:
    `Vary: Cookie`
  - Static:
    `Cache-Control: public, s-maxage: ...`

Make Symfony keep the cache headers

```
// Symfony overwrites caching headers when session was used.
// Need to explicitly tell it we intend to cache the response
use Symfony\Component\HttpKernel\EventListener
\AbstractSessionListener;
...
$response->headers->set(
    AbstractSessionListener::NO_AUTO_CACHE_CONTROL_HEADER, 'true');
```

builtin.vcl

```
sub vcl_recv {
    // ...
    if (req.method != "GET" && req.method != "HEAD") {
        /* We only deal with GET and HEAD by default */
        return (pass);
    }
    if (req.http.Authorization || req.http.Cookie) {
        /* Not cacheable by default */
        return (pass);
    }

    return (hash);
}
```

Cache lookup despite cookies

```
// default.vcl
sub vcl_recv {
    // ...
    if (req.method != "GET" && req.method != "HEAD") {
        /* We only deal with GET and HEAD by default */
        return (pass);
    }

    // Cache lookup even with Cookie or Authorization header
    // Not adjusting vcl_cache to use Cookie in cache key
    return (hash);
}
```

Edge Side Includes

Use Edge Side Includes

Like server side include, but on Varnish:

- Content embeds URLs to sub-parts of the content
- Varnish fetches and caches elements separatly
- Individual caching rules per fragment. E.g. some elements vary on cookie, different TTL, ...

ESI HTML

```
<html>
    <body>
        Main body.
        <esi:include src="fragment.php" />
    </body>
</html>
```

And activate ESI in Varnish

ESI

- Pro
  - Separate caching for fragments
  - Server side, makes search engines happy
  - Simple to use in the application
- Con
  - Each include must be resolved before response can be sent

Symfony has built-in ESI support

```
# app/config/config.yml
framework:
  esi: { enabled: true }
  fragments: { path: /_fragment }
```

```
{# index.html.twig #}
{% render_esi(controller(
    'AppBundle:Comments:comments',
    {'param': 42 })
) %}
```

Cache still either global or individual

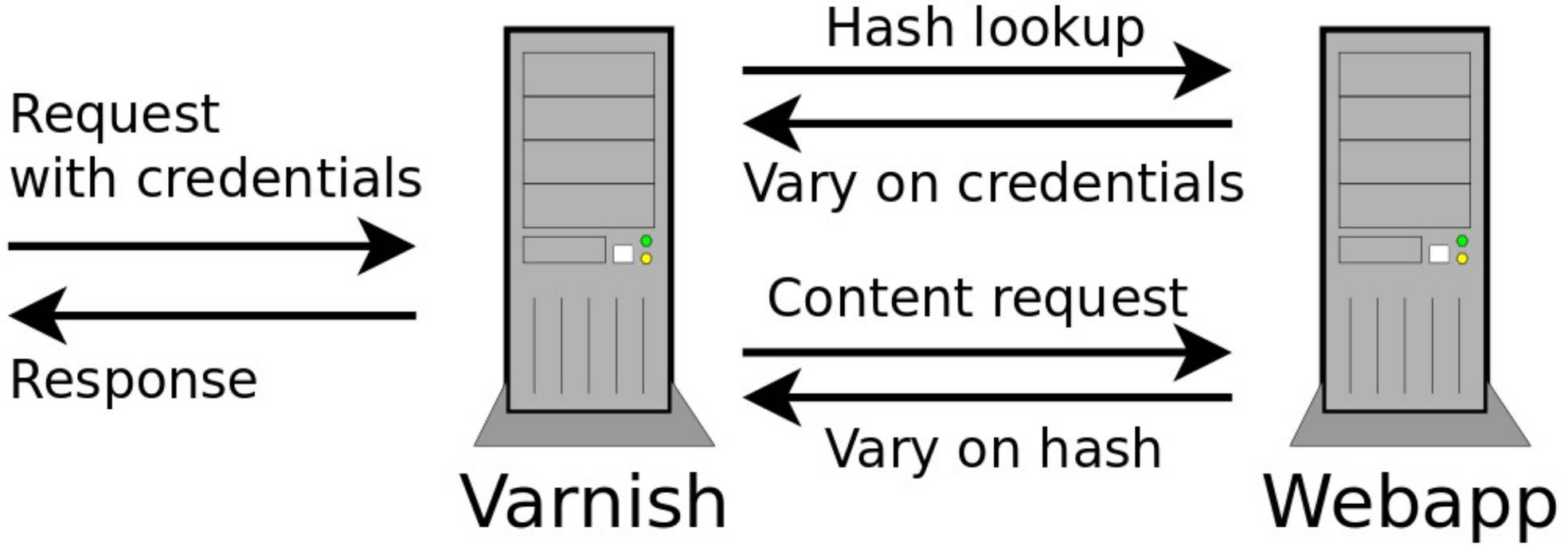
User Context


UK/EU passports
EU ID cards
Including Switzerland, Norway and the European Economic Area
UK Border
UK/EU Passports
EU ID cards

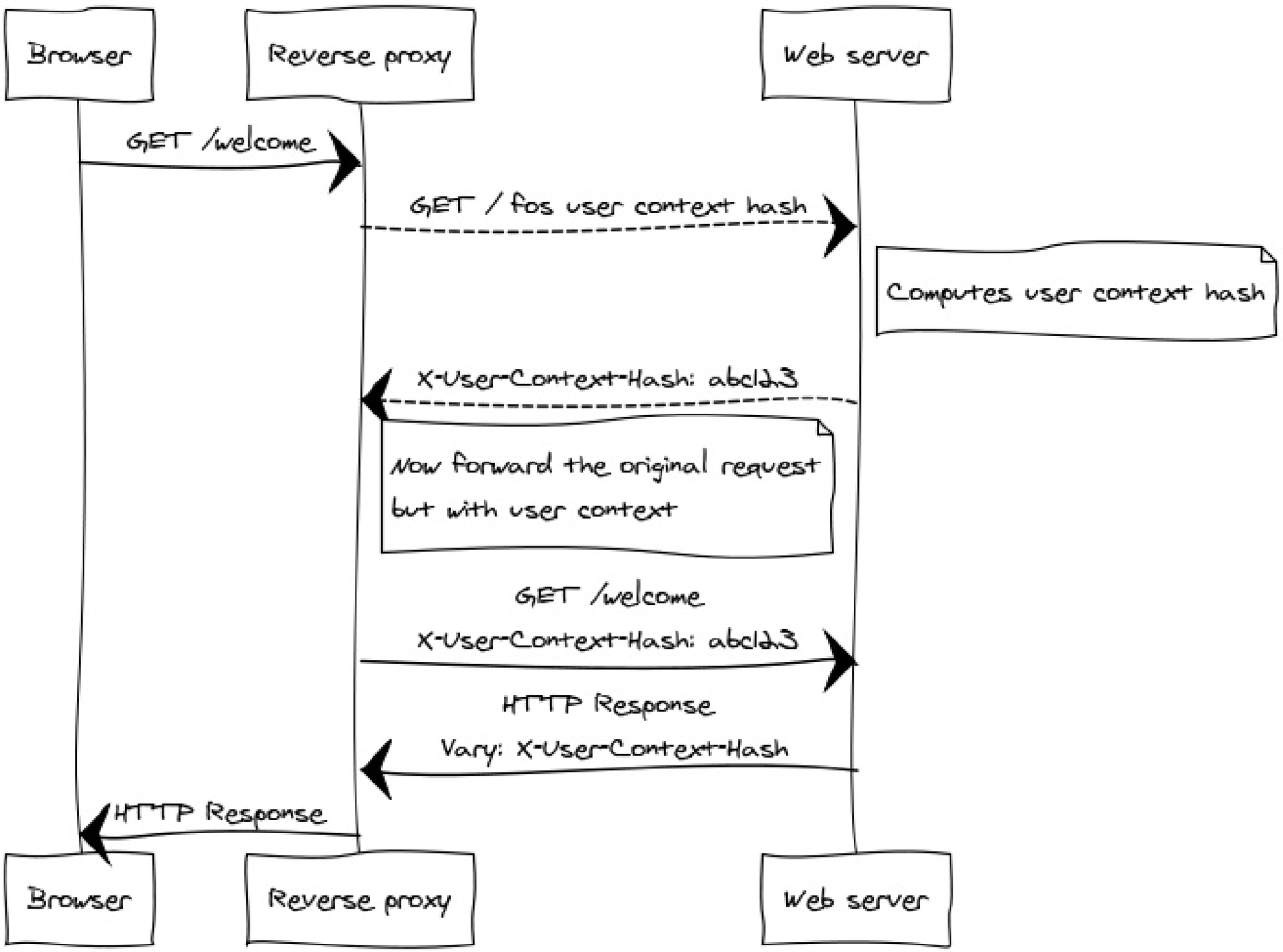
Introducing the User Context Hash

- Groups with the same permissions
- Transparent (reverse proxy does most of the job)
- Computed for every user of application
- Hash generation can be customized
- Cached by session ID, using HTTP cache

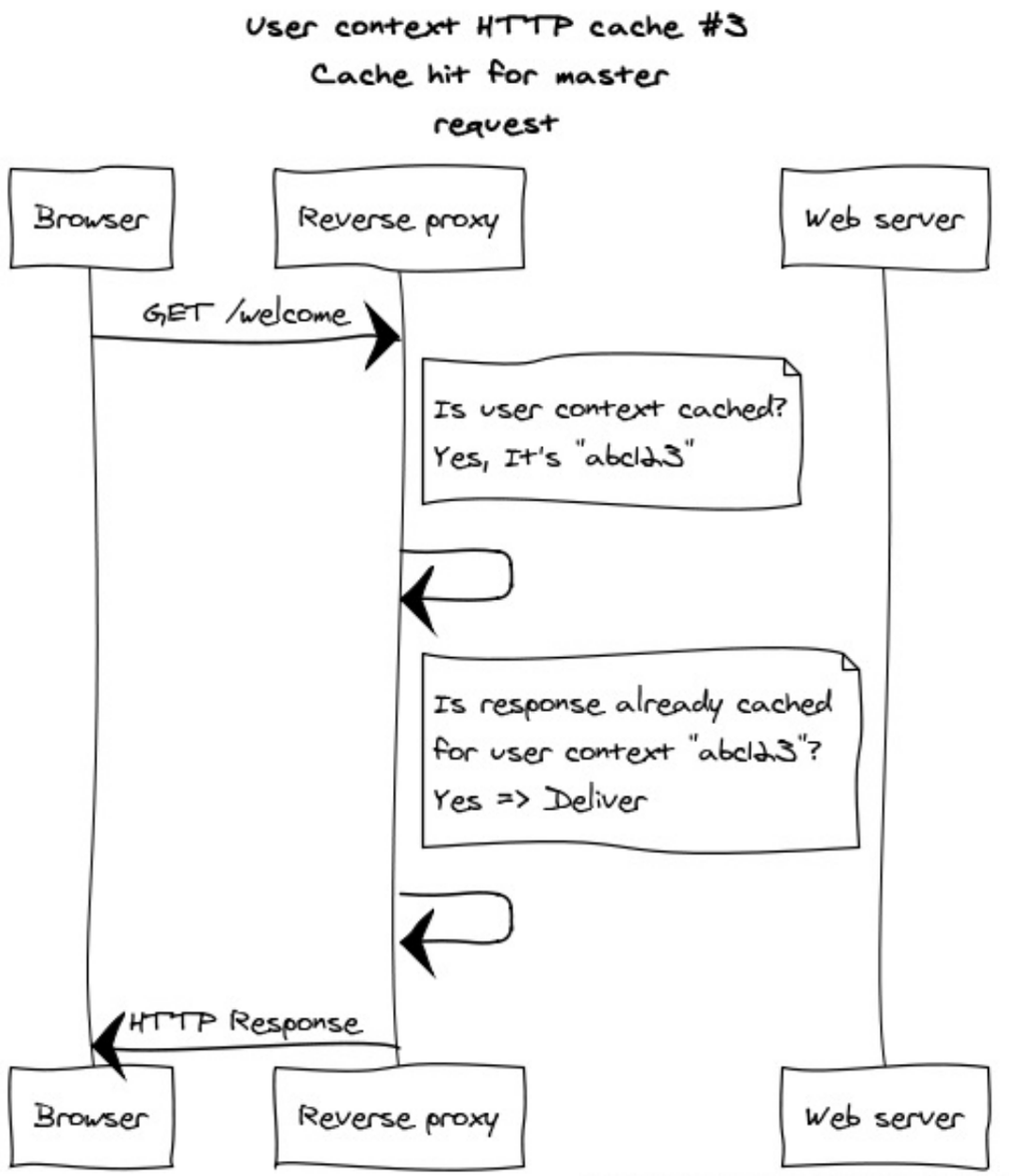
User Context



User context HTTP cache #1



www.websequencediagrams.com


User context HTTP cache #3
Cache hit for master
request
Browser
Reverse proxy
Web server
GET /welcome
Is user context cached?
Yes, It's "abc123"
Is response already cached
for user context "abc123"?
Yes => Deliver
HTTP Response
Browser
Reverse proxy
Web server
www.websequencediagrams.com

```
sub vcl_recv { ...
  // Copy client request headers to auth request
  curl.header_add_all();
  // We go through varnish itself to cache
  curl.header_add("Host: auth");
  curl.get("http://localhost/");
```

```
if (200 != curl.status()) {
    return (synth(curl.status()));
}
set req.http.X-User-Context =
      curl.header("X-User-Context");
curl.free();
```

```
sub vcl_recv {
    // Cache auth lookup
    if ("auth" == req.http.Host)
    {
        if (!client.ip ~ self) {
            return (synth(405, "Not allowed"));
        }

        set req.backend_hint = auth;
        // force caching despite auth headers
        return (hash);
    }
    ...
```

```php
class RoleProvider implements ContextProvider
{
  public function updateUserContext(
    UserContext $context
  ) {
    ...
    $roles = array_map(function (Role $role) {
        return $role->getRole();
    }, $token->getRoles());

    // Order should not change hash
    sort($roles);
    $context->addParameter('roles', $roles);
  }
}
```

Wrap Up

Mix solutions

- Edge side includes
- Ajax
- Move logic to frontend
- User context

Write your own context hash provider

- Other sources for user groups
- Geographical region
- User agent / client capabilities
- A/B testing
- …

Symfony FOSHttpCacheBundle

- Cache tagging
- Invalidation service for active invalidation
- Annotations for caching headers and invalidation
- Request matcher for caching headers
- Listener for user context


Questions / Input / Feedback ?
https://joind.in/talk/9fb1a
Twitter: @dbu