Using Varnish

eZ Publish 5 being built on top of Symfony 2, it uses standard HTTP cache headers. By default the Symfony 2 reverse proxy, written in PHP, is used to handle cache, but it can be easily replaced with any other reverse proxy like Varnish.

Configure Varnish

⚠ Provided VCL example is given for Varnish 3.x only

The first thing is to configure Varnish to advertise ESI usage and to activate cache invalidation:

ezpublish5.vcl

```
# Varnish 3.x - eZ Publish 5 - Complete VCL

# Our Backend - We assume that eZ Publish Web server listen on port 80
backend ezpublish {
    .host = "127.0.0.1";
    .port = "80";
}

# ACL for purgers IP.
# Provide here IP addresses that are allowed to send PURGE requests.
# PURGE requests will be sent by the backend.
acl purgers {
    "127.0.0.1";
    "192.168.0.0"/16;
}

# Called at the beginning of a request, after the complete request has been received
sub vcl_recv {

    # Set the backend
    set req.backend = ezpublish;

    # Advertise Symfony for ESI support
    set req.http.Surrogate-Capability = "abc=ESI/1.0";

    # Add a unique header containing the client address (only for master request)
    # Please note that /_fragment URI can change in Symfony configuration
    if (!req.url ~ "^/_fragment") {
        if (req.http.x-forwarded-for) {
            set req.http.X-Forwarded-For = req.http.X-Forwarded-For + ", " +
client.ip;
        } else {
            set req.http.X-Forwarded-For = client.ip;
        }
    }

    # Handle purge
    # Only works with "multiple_http" purge method
    if (req.request == "PURGE") {
        if (!client.ip ~ purgers) {
            error 405 "Method not allowed";
        }
```

```
        if ( req.http.X-Location-Id == "*" ) {
            # Purge all locations
            ban( "obj.http.X-Location-Id ~ ^[0-9]+$" );
            error 200 "Purge all locations done.";
        } elseif ( req.http.X-Location-Id ) {
            # Purge location by its locationId
            ban( "obj.http.X-Location-Id == " + req.http.X-Location-Id );
            error 200 "Purge of content connected to the location id(" +
req.http.X-Location-Id + ") done.";
        }
    }

    # Normalize the Accept-Encoding headers
    if (req.http.Accept-Encoding) {
        if (req.http.Accept-Encoding ~ "gzip") {
            set req.http.Accept-Encoding = "gzip";
        } elsif (req.http.Accept-Encoding ~ "deflate") {
            set req.http.Accept-Encoding = "deflate";
        } else {
            unset req.http.Accept-Encoding;
        }
    }

    # Don't cache Authenticate & Authorization
    if (req.http.Authenticate || req.http.Authorization) {
        return(pass);
    }

    # Don't cache requests other than GET and HEAD.
    if (req.request != "GET" && req.request != "HEAD") {
        return (pass);
    }

    # Don't cache authenticated sessions
    if (req.http.Cookie && req.http.Cookie ~ "is_logged_in=" ) {
        return(pass);
    }

    # If it passes all these tests, do a lookup anyway;
    return (lookup);
}

# Called when the requested object has been retrieved from the backend
sub vcl_fetch {

    # Optimize to only parse the Response contents from Symfony
    if (beresp.http.Surrogate-Control ~ "ESI/1.0") {
        unset beresp.http.Surrogate-Control;
        set beresp.do_esi = true;
    }

    # Don't cache response with Set-Cookie
    if ( beresp.http.Set-Cookie ) {
        set beresp.ttl = 0s;
        return (hit_for_pass);
    }

    # Respect the Cache-Control=private header from the backend
    if (beresp.http.Cache-Control ~ "private") {
```

```
    set beresp.ttl = 0s;
    return (hit_for_pass);
}

return (deliver);
```

```
}
```

You can of course add additional rules if you need.

Configure eZ Publish

Update your front controller

By default your front controller (`index.php`) uses the built-in reverse proxy, `EzPublishCache`. In order to use Varnish, you need to deactivate it by commenting the line where `EzPublishCache` is instantiated:

```
<?php
// ...
require_once __DIR__ . '/../ezpublish/EzPublishKernel.php';
require_once __DIR__ . '/../ezpublish/EzPublishCache.php';
$kernel = new EzPublishKernel( 'prod', false );
$kernel->loadClassCache();
// Comment the following line if you use an external reverse proxy (e.g. Varnish)
// $kernel = new EzPublishCache( $kernel );
$request = Request::createFromGlobals();
// Uncomment the following if your application is behind a reverse proxy you manage
and trust.
// (see
http://fabien.potencier.org/article/51/create-your-own-framework-on-top-of-the-symfony
2-components-part-2)
//Request::trustProxyData();
$response = $kernel->handle( $request );
$response->send();
$kernel->terminate( $request, $response );
```

Update YML configuration

ezpublish.yml

```
ezpublish:
    http_cache:
        # Use multiple_http method for purging content
        purge_type: multiple_http

    system:
        # Assuming that my_siteaccess_group your frontend AND backend siteaccesses
        my_siteaccess_group:
            http_cache:
                # Fill in your Varnish server(s) address(es).
                purge_servers: [http://my.varnish.server:6081]
```

Et voilà !