Server Side Performance Tricks

Over time I've learned that performance is a two way street that requires good server configurations and good content management.

You should avoid using .htaccess files completely if you have access to httpd main server config file. Using .htaccess files slows down your Apache http server. Any directive that you can include in a .htaccess file is better set in a Directory block, as it will have the same effect with better performance.

Cross Origin Requests

These rulles allow servers to work around the same-origin policy and accept resources origin

This is not without risks as it opens your server to potentially maliciously crafted resources originating from outside your domain and which you have no control over.

Generic CORS requests

Sends the CORS headers for all resources.

```
<IfModule mod_headers.c>
  Header set Access-Control-Allow-Origin "*"
</IfModule>
```

Cross-origin images

Send the CORS header for images when browsers request it.

- https://developer.mozilla.org/en-US/docs/Web/HTML/CORS_enabled_image

- https://blog.chromium.org/2011/07/using-cross-domain-images-in-webgl-and.html

```
<IfModule mod_setenvif.c>
  <IfModule mod_headers.c>
    <FilesMatch "\.(bmp|cur|gif|ico|jpe?g|png|svgz?|webp)$">
      SetEnvIf Origin ":" IS_CORS
      Header set Access-Control-Allow-Origin "*" env=IS_CORS
    </FilesMatch>
  </IfModule>
</IfModule>
```

Cross-origin web fonts

Send the CORS header for web fonts when browsers request them.

```
<IfModule mod_headers.c>
  <FilesMatch "\.(eot|otf|tt[cf]|woff2?)$">
    Header set Access-Control-Allow-Origin "*"
  </FilesMatch>
</IfModule>
```

Cross-origin resource timing

Allow cross-origin access to the timing information for all resources.

If a resource isn't served with a `Timing-Allow-Origin` header that would allow its timing information to be shared with the document, some of the attributes of the `PerformanceResourceTiming` object will be set to zero.

- https://www.w3.org/TR/resource-timing/
- http://www.stevesouders.com/blog/2014/08/21/resource-timing-practical-tips/

```
<IfModule mod_headers.c>
  Header set Timing-Allow-Origin: "*"
</IfModule>
```

Media Types

Serve resources with the proper media types (MIME types).

- https://www.iana.org/assignments/media-types/media-types.xhtml
- https://httpd.apache.org/docs/current/mod/mod_mime.html#addtype

Normalize Javascript to standard type per https://tools.ietf.org/html/rfc4329#section-7.2

```
<IfModule mod_mime.c>
  # Data interchange
    AddType application/atom+xml                        atom
    AddType application/json                            json map topojson
    AddType application/ld+json                         jsonld
    AddType application/rss+xml                         rss
    AddType application/vnd.geo+json                    geojson
    AddType application/xml                             rdf xml

  # JavaScript
    AddType application/javascript                      js mjs

  # Manifest files
    AddType application/manifest+json                   webmanifest
    AddType application/x-web-app-manifest+json         webapp
    AddType text/cache-manifest                         appcache

  # Media files
    AddType audio/mp4                                   f4a f4b m4a
    AddType audio/ogg                                   oga ogg opus
    AddType image/bmp                                   bmp
    AddType image/svg+xml                               svg svgz
    AddType image/webp                                  webp
    AddType video/mp4                                   f4v f4p m4v mp4
    AddType video/ogg                                   ogv
    AddType video/webm                                  webm
    AddType video/x-flv                                 flv
  # Web fonts
    AddType font/woff                                   woff
    AddType font/woff2                                  woff2
    AddType application/vnd.ms-fontobject              eot
    AddType font/ttf                                    ttf
    AddType font/collection                             ttc
    AddType font/otf                                    otf

  # Other
    AddType application/octet-stream                    safariextz
    AddType application/x-bb-appworld                   bbaw
    AddType application/x-chrome-extension              crx
    AddType application/x-opera-extension               oex
    AddType application/x-xpinstall                     xpi
    AddType text/calendar                               ics
    AddType text/markdown                               markdown md
    AddType text/vcard                                  vcard vcf
    AddType text/vnd.rim.location.xloc                  xloc
    AddType text/vtt                                    vtt
    AddType text/x-component                            htc

</IfModule>
```

Character encodings

Serve all resources labeled as `text/html` or `text/plain` with the media type `charset` parameter set to `UTF-8`.

- https://httpd.apache.org/docs/current/mod/core.html#adddefaultcharset

```
AddDefaultCharset utf-8
```

Serve the following file types with the media type `charset` parameter set to `UTF-8`.

- https://httpd.apache.org/docs/current/mod/mod_mime.html#addcharset

```
<IfModule mod_mime.c>
AddCharset utf-8 .atom \
  .bbaw \
  .css \
  .geojson \
  .ics \
  .js \
  .json \
  .jsonld \
  .manifest \
  .markdown \
  .md \
  .mjs \
  .rdf \
  .rss \
  .topojson \
  .vtt \
  .webapp \
  .webmanifest \
  .xloc \
  .xml
</IfModule>
```

Rewrite Engine

1. Turn on the rewrite engine (this is necessary in order for the `RewriteRule` directives to work).

    https://httpd.apache.org/docs/current/mod/mod_rewrite.htmlRewriteEngine

2. Enable the `FollowSymLinks` option if it isn't already.

    https://httpd.apache.org/docs/current/mod/core.htmloptions

3. If your web host doesn't allow the `FollowSymlinks` option, you need to comment it out or remove it, and then uncomment the `Options +SymLinksIfOwnerMatch` line (4), but be aware of the performance

impact.

https://httpd.apache.org/docs/current/misc/perf-tuning.htmlsymlinks

4. Some cloud hosting services will require you set `RewriteBase`.
    1. https://www.rackspace.com/knowledge_center/frequently-asked-question/why-is-modrewrite-not-working-on-my-site
    2. https://httpd.apache.org/docs/current/mod/mod_rewrite.htmlrewritebase
5. Depending on how your server is set up, you may also need to use the `RewriteOptions` directive to enable some options for the rewrite engine.

    https://httpd.apache.org/docs/current/mod/mod_rewrite.htmlrewriteoptions

6. Set `%{ENV:PROTO}` variable, to allow rewrites to redirect with the appropriate schema automatically (http or https).

```
<IfModule mod_rewrite.c>
    RewriteEngine On # (1)
    Options +FollowSymlinks # (2)
    # Options +SymLinksIfOwnerMatch # (3) (4)
    # RewriteBase /
    # RewriteOptions <options> # (5)
    RewriteCond %{HTTPS} =on  # (6)

    RewriteRule ^ - [env=proto:https]
    RewriteCond %{HTTPS} !=on
    RewriteRule ^ - [env=proto:http]

</IfModule>
```

Forcing https

Redirect from the `http://` to the `https://` version of the URL.

- https://wiki.apache.org/httpd/RewriteHTTPToHTTPS

```
<IfModule mod_rewrite.c>
   RewriteEngine On
   RewriteCond %{HTTPS} !=on
   RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [R=301,L]
</IfModule>
```

Content Security Policy (CSP)

Mitigate the risk of cross-site scripting and other content-injection attacks.

This can be done by setting a `Content Security Policy` which whitelists trusted sources of content for your website.

The example header below allows ONLY scripts that are loaded from the current website's origin (no inline scripts, no CDN, etc). That almost certainly won't work as-is for your website!

To make things easier, you can use an online CSP header generator such as: http://cspisawesome.com/.

- https://content-security-policy.com/
- https://www.html5rocks.com/en/tutorials/security/content-security-policy/
- https://w3c.github.io/webappsec-csp/

`mod_headers` cannot match based on the content-type, however, the `Content-Security-Policy` response header should be send only for HTML documents and not for the other resources.

```
<IfModule mod_headers.c>
  Header set Content-Security-Policy "script-src 'self'; object-src 'self

  <FilesMatch "\.(appcache|atom|bbaw|bmp|crx|css|cur|eot|f4[abpv]|flv|geo
      Header unset Content-Security-Policy
  </FilesMatch>
```

```
</IfModule>
```

File access

Block access to directories without a default document.

You should leave the following uncommented, as you shouldn't allow anyone to surf through every directory on your server (which may includes rather private places such as the CMS's directories).

```
<IfModule mod_autoindex.c>
    Options -Indexes
</IfModule>
```

Block access to all hidden files and directories with the exception of the visible content from within the `/.well-known/` hidden directory.

These types of files usually contain user preferences or the preserved state of an utility, and can include rather private places like, for example, the `.git` or `.svn` directories.

The `/.well-known/` directory represents the standard (RFC 5785) path prefix for "well-known locations" (e.g.: `/.well-known/manifest.json`, `/.well-known/keybase.txt`), and therefore, access to its visible content should not be blocked.

- https://www.mnot.net/blog/2010/04/07/well-known
- https://tools.ietf.org/html/rfc5785

```
<IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteCond %{REQUEST_URI} "!(^|/)\.well-known/([^./]+./?)+$" [NC]
    RewriteCond %{SCRIPT_FILENAME} -d [OR]
    RewriteCond %{SCRIPT_FILENAME} -f
    RewriteRule "(^|/)\." - [F]
</IfModule>
```

Block access to files that can expose sensitive information.

By default, block access to backup and source files that may bleft by some text editors and can pose a security risk when anyonhas access to them.

= https://feross.org/cmsploit/

(!) Update the `<FilesMatch>` regular expression from below tinclude any files that might end up on your production server and can expose sensitive information about your website. These files mainclude: configuration files, files that contain metadata about the project (e.g.: project dependencies), build scripts, etc.

```
<FilesMatch "(^#.*#|\.(bak|conf|dist|fla|in[ci]|log|orig|psd|sh|sql|sw[op]

    # Apache < 2.3
    <IfModule !mod_authz_core.c>
        Order allow,deny
        Deny from all
        Satisfy All
    </IfModule>

    # Apache ≥ 2.3
    <IfModule mod_authz_core.c>
        Require all denied
    </IfModule>

</FilesMatch>
```

HTTP Strict Transport Security (HSTS)

Force client-side SSL redirection. If a user types `example.com` in their browser, even if the server redirects them to the secure version of the website, that still leaves a window of opportunity (the initial HTTP connection) for an attacker to downgrade or redirect the request. The following header ensures that browser will ONLY connect to your # server via HTTPS, regardless of what the users type in the browser’s address bar.

(!) Remove the `includeSubDomains` optional directive if the website's subdomains are not using HTTPS.

- https://www.html5rocks.com/en/tutorials/security/transport-layer-security/
- https://tools.ietf.org/html/draft-ietf-websec-strict-transport-sec-14#section-6.1
- [https://blogs.msdn.microsoft.com/ieinternals/2014/08/18/strict-transport-security/]

```
<IfModule mod_headers.c>
  Header always set Strict-Transport-Security "max-age=16070400; includeSu
</IfModule>
```

Reducing MIME type security risks

Prevent some browsers from MIME-sniffing the response.

This reduces exposure to drive-by download attacks and cross-origin data leaks, and should be left uncommented, especially if the server is serving user-uploaded content or content that could potentially be treated as executable by the browser.

- https://www.slideshare.net/hasegawayosuke/owasp-hasegawa
- https://blogs.msdn.microsoft.com/ie/2008/07/02/ie8-security-part-v-comprehensive-protection/
- https://msdn.microsoft.com/en-us/library/ie/gg622941.aspx
- https://mimesniff.spec.whatwg.org/

```
<IfModule mod_headers.c>
    Header set X-Content-Type-Options "nosniff"
</IfModule>
```

Reflected Cross-Site Scripting (XSS) attacks

1. Try to re-enable the cross-site scripting (XSS) filter built into most web browsers.

    1. The filter is usually enabled by default, but in some cases it may be disabled by the user. However, in Internet Explorer for example, it can be re-enabled just by sending the `X-XSS-Protection` header with the value of 1.

2. Prevent web browsers from rendering the web page if a potential reflected (a.k.a non-persistent) XSS attack is detected by the filter.

    1. By default, if the filter is enabled and browsers detect a reflected XSS attack,they will attempt to block the attack by making the smallest possible modifications to the returned web page.
    2. Unfortunately, in some browsers (e.g.: Internet Explorer), this default behavior may allow the XSS filter to be exploited, thereby, it’s better to inform browsers to prevent the rendering of the page altogether, instead of attempting to modify it [IE’s XSS Filter Creates XSS Vulnerabilities] (https://hackademix.net/2009/11/21/ies-xss-filter-creates-xss-vulnerabilities)

(!) Do not rely on the XSS filter to prevent XSS attacks! Ensure that you are taking all possible measures to prevent XSS attacks, the most obvious being: validating and sanitizing your website’s inputs.

- https://blogs.msdn.microsoft.com/ie/2008/07/02/ie8-security-part-iv-the-xss-filter/
- https://blogs.msdn.microsoft.com/ieinternals/2011/01/31/controlling-the-xss-filter/
- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

`mod_headers` cannot match based on the content-type, however, the `X-XSS-Protection` response header should be send only for HTML documents and not for the other resources.

```
<IfModule mod_headers.c>
```

```
  #                          (1)    (2)
  Header set X-XSS-Protection "1; mode=block"

  <FilesMatch "\.(appcache|atom|bbaw|bmp|crx|css|cur|eot|f4[abpv]|flv|geo
      Header unset X-XSS-Protection
  </FilesMatch>

</IfModule>
```

Server-side technology information

Remove the `X-Powered-By` response header that:

- is set by some frameworks and server-side languages (e.g.: ASP.NET, PHP), and its value contains information about them (e.g.: their name, version number)
- doesn't provide any value to users, contributes to header bloat, and in some cases, the information it provides can expose vulnerabilities

(!) If you can, you should disable the `X-Powered-By` header from the language / framework level (e.g.: for PHP, you can do that by setting`expose_php = off` in `php.ini`)

- https://php.net/manual/en/ini.core.php#ini.expose-php

```
<IfModule mod_headers.c>
    Header unset X-Powered-By
</IfModule>
```

Server software information

Prevent Apache from adding a trailing footer line containing information about the server to the server-generated documents (e.g.: error messages, directory listings, etc.)

- https://httpd.apache.org/docs/current/mod/core.html#serversignature

```
ServerSignature Off
```

Prevent Apache from sending in the `Server` response header its exact version number, the description of the generic OS-type or information about its compiled-in modules.

(!) The `ServerTokens` directive will only work in the main server configuration file, so don't try to enable it in the `.htaccess` file!

- https://httpd.apache.org/docs/current/mod/core.html#servertokens

```
#ServerTokens Prod
```

Compression

Force compression for mangled `Accept-Encoding` request headers.

- Pushing Beyond Gzipping

```
<IfModule mod_deflate.c>
  <IfModule mod_setenvif.c>
      <IfModule mod_headers.c>
          SetEnvIfNoCase ^(Accept-EncodXng|X-cept-Encoding|X{15}|~{15}|-{1
          RequestHeader append Accept-Encoding "gzip,deflate" env=HAVE_Acc
      </IfModule>
  </IfModule>
</IfModule>
```

Compress all output labeled with one of the following media types.

(!) For Apache versions below version 2.3.7 you don't need toenable `mod_filter` and can remove the `<IfModule mod_filter.c>`and `</IfModule>` lines as `AddOutputFilterByType` is still inthe core directives.

- https://httpd.apache.org/docs/current/mod/mod_filter.html#addoutputfilterbytype

```
<IfModule mod_filter.c>
AddOutputFilterByType DEFLATE "application/atom+xml" \
    "application/javascript" \
    "application/json" \
    "application/ld+json" \
    "application/manifest+json" \
    "application/rdf+xml" \
    "application/rss+xml" \
    "application/schema+json" \
    "application/vnd.geo+json" \
    "application/vnd.ms-fontobject" \
    "application/x-font-ttf" \
    "application/x-javascript" \
    "application/x-web-app-manifest+json" \
    "application/xhtml+xml" \
    "application/xml" \
    "font/collection" \
    "font/eot" \
    "font/opentype" \
    "font/otf" \
    "font/ttf" \
    "image/bmp" \
    "image/svg+xml" \
    "image/vnd.microsoft.icon" \
    "image/x-icon" \
    "text/cache-manifest" \
    "text/calendar" \
    "text/css" \
    "text/html" \
    "text/javascript" \
    "text/plain" \
    "text/markdown" \
    "text/vcard" \
    "text/vnd.rim.location.xloc" \
    "text/vtt" \
    "text/x-component" \
    "text/x-cross-domain-policy" \
    "text/xml"
```

```
</IfModule>
```

Map the following filename extensions to the specified encoding type in order to make Apache serve the file types with the appropriate `Content-Encoding` response header(do note that this will NOT make Apache compress them!).

If these files types would be served without an appropriate`Content-Enable` response header, client applications (e.g.:browsers) wouldn't know that they first need to uncompressthe response, and thus, wouldn't be able to understand thecontent.

- https://httpd.apache.org/docs/current/mod/mod_mime.html#addencoding

```
<IfModule>
  <IfModule mod_mime.c>
      AddEncoding gzip              svgz
  </IfModule>
</IfModule>
```

ETags

Remove `ETags` **as resources are sent with far-future expires headers.**

- https://developer.yahoo.com/performance/rules.html#etags
- https://tools.ietf.org/html/rfc7232#section-2.3

```
# `FileETag None` doesn't work in all cases.
<IfModule mod_headers.c>
  Header unset ETag
</IfModule>

FileETag None
```

Expires headers

Serve resources with far-future expires headers.

(!) If you don’t control versioning with filename-based cache busting, you should consider lowering the cache times to something like one week.

- https://httpd.apache.org/docs/current/mod/mod_expires.html

```
<IfModule mod_expires.c>
  ExpiresActive                                 on
  ExpiresDefault                                "access plus 1 month"

# CSS
  ExpiresByType text/css                        "access plus 1 year"

# Data interchange
  ExpiresByType application/atom+xml            "access plus 1 hour"
  ExpiresByType application/rdf+xml             "access plus 1 hour"
  ExpiresByType application/rss+xml             "access plus 1 hour"
  ExpiresByType application/json                "access plus 0 seconds"
  ExpiresByType application/ld+json             "access plus 0 seconds"
  ExpiresByType application/schema+json         "access plus 0 seconds"
  ExpiresByType application/vnd.geo+json        "access plus 0 seconds"
  ExpiresByType application/xml                 "access plus 0 seconds"
  ExpiresByType text/calendar                   "access plus 0 seconds"
  ExpiresByType text/xml                        "access plus 0 seconds"

# Favicon (cannot be renamed!) and cursor images
  ExpiresByType image/vnd.microsoft.icon        "access plus 1 week"
  ExpiresByType image/x-icon                    "access plus 1 week"

# HTML
  ExpiresByType text/html                       "access plus 0 seconds"

# JavaScript
  ExpiresByType application/javascript          "access plus 1 year"
  ExpiresByType application/x-javascript        "access plus 1 year"
  ExpiresByType text/javascript             "access plus 1 year"

# Manifest files
  ExpiresByType application/manifest+json   "access plus 1 week"
  ExpiresByType application/x-web-app-manifest+json   "access plus 0 secon
  ExpiresByType text/cache-manifest         "access plus 0 seconds"

# Markdown
  ExpiresByType text/markdown               "access plus 0 seconds"

# Media files
  ExpiresByType audio/ogg                   "access plus 1 month"
  ExpiresByType image/bmp                   "access plus 1 month"
  ExpiresByType image/gif                   "access plus 1 month"
  ExpiresByType image/jpeg                  "access plus 1 month"
  ExpiresByType image/png                   "access plus 1 month"
  ExpiresByType image/svg+xml               "access plus 1 month"
  ExpiresByType image/webp                  "access plus 1 month"
  ExpiresByType video/mp4                   "access plus 1 month"
  ExpiresByType video/ogg                   "access plus 1 month"
  ExpiresByType video/webm                  "access plus 1 month"

# Web fonts
  # Collection
  ExpiresByType font/collection             "access plus 1 month"
  # Embedded OpenType (EOT)
  ExpiresByType application/vnd.ms-fontobject  "access plus 1 month"
  ExpiresByType font/eot                    "access plus 1 month"
  # OpenType
  ExpiresByType font/opentype               "access plus 1 month"
  ExpiresByType font/otf                    "access plus 1 month"
  # TrueType
  ExpiresByType application/x-font-ttf      "access plus 1 month"
  ExpiresByType font/ttf                    "access plus 1 month"
  # Web Open Font Format (WOFF) 1.0
  ExpiresByType application/font-woff       "access plus 1 month"
  ExpiresByType application/x-font-woff     "access plus 1 month"
  ExpiresByType font/woff                   "access plus 1 month"
  # Web Open Font Format (WOFF) 2.0
  ExpiresByType application/font-woff2      "access plus 1 month"
  ExpiresByType font/woff2                  "access plus 1 month"

# Other
  ExpiresByType text/x-cross-domain-policy  "access plus 1 week"
</IfModule>
```

Filename-based cache busting

If you're not using a build process to manage your filename version revving, you might want to consider enabling the following directives to route all requests such as `/style.12345.css` to `/style.css`.

To understand why this is important and even a better solution than using something like `*.css?v231`, please see: [Revving Filenames: don't use querystring]

```
<IfModule mod_rewrite.c>
  RewriteEngine On
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^(.+)\.(\d+)\.(bmp|css|cur|gif|ico|jpe?g|m?js|png|svgz?|webp
</IfModule>
```