Apache - mod_rewrite v příkladech (harvie.cz)

Přesunuto z http://blog.harvie.cz/2008/05/navody/par-uzitecnych-rewritu/

Při změně domény jsem trochu předělal již tak zajímavé rewrity (nastavení mod_rewrite (CZ)) v Apache 2. V současnosti jsou asi dost nehezky napsané, ale i tak vám to tu nechám jako ukázku, o které si myslím, že je celkem použitelná.

Mě osobně se toto nastavení rewritu vygooglit nepodařilo a tak mi nezbylo než se mod_rewrite naučit a s tím jsem se zase opět pocvičil ve mnou milovaných i zatracovaných regulárních výrazech (CZ)...

Tady jsou hlavní rewrity, které v současnosti pohání mého Apače, protože platí globálně pro celý server (s dalšíma párset řádkama konfiguráků - když nepočítám ty defaultní):

Nejnovnější verze 2.0

Zase jsem po dlouhé době šáhnul do nastavení rewritů za účelem zlepšení použitelnosti serveru a snadného přidávání nových domén do hostingu. Starou verzi nastavení najdete níže.

Dříve jsem měl všechny poddomény na harvie.cz namapované na adresáře a hostingy nových domén jsem přidával pomocí přidávání nových hostů do apache, což dělalo docela problémy a nikdy jsem s tímm nebyl spokojený, takže jsem hledal způsob, jak namapovat celou doménu na adresáře a poddomény na podadresáře.

Nejdříve jsem narazil na mod_vhost_alias, nějaké info najdete zde:
http://httpd.apache.org/docs/1.3/mod/mod_vhost_alias.html
http://httpd.apache.org/docs/1.3/vhosts/mass.html Vhost_alias je asi výkonný, ale dělal problémy ve spojení s dalšímy rewrity a ani mi nepostačoval z hlediska omezených možností (ve skutečnosti se mi zamlouval jen kvůli snadné konfiguraci). Rozhodl jsem se tedy vylepšit stávající rewrity.

pokud chceme používat jeden konfigurák pro více domén, je dobré tam šoupnout místo hostnamu hvězdičku a případně i port (to já nemohu, protože používám jeden konfigurák na víc portů - apache nadává, ale nakonec funguje, protože ve skutečnosti k žádné kolizi dojít nemůže):

```
NameVirtualHost *
<VirtualHost *>
...
</VirtualHost>
```

Další věcí, se kterou jsem si trochu vyhrál je reverzní proxy na jiné stroje (v mém případě virtuální server). Na to je potřeba následující konfigurace a zavedení modulů poznamenaných v komentářích:

```
        #Proxy
        #mods: proxy_connect proxy_http proxy rewrite
        #zabezpecime nas apache proti zneuziti jako anonymni proxy:
        ProxyRequests Off
        <Proxy *>
            AddDefaultCharset off
            Order deny,allow
            Allow from all #
        </Proxy>
```

Potom každou reverzní proxy nastavíme zvlášť, jako v tomto případě, kde všechny požadavky na vserver.harvie.cz přepošlu na http://192.168.2.137/.

```
        #vservers
         RewriteCond %{HTTP_HOST} ^vserver\.harvie\.cz
         RewriteRule ^/(.*) http://192.168.2.137/$1 [NC,P,L]
```

A když už víme, jak předat dotazy, které jdou mimo náš stroj, můžeme se věnovat těm, které musíme skutečně obsloužit sami.

V kódu níže můžete vidět, že se nám dvakrát opakují podobné tři řádky... První RewriteCond vždy zakáže parsování pokud je místo hostname poskytnuta pouze IPv4 adresa. Pokud máte více síťových rozhraní, možná se vám to naopak bude hodit. Pokud ale používáte IPv6 spojení na server, budete muset ještě přidat pravidlo na vyfiltrování požadavků na IPv6 adresy bez specifikace hostname.

Zbylé dva řádky už dělají tu podstatnou práci - tj. mapování domén a poddomén na adresářovou strukturu.

Sekce označená DOMAINS se využije pouze pro případy, kdy hostname obsahuje jen jednu tečku. Potom např. požadavek na doménu harvie.cz přesměruje do adresáře /srv/data1/domains/harvie.cz/.

Tady bych rád poznamenal, že se může hodit upravit rewrite, aby tam bylo něco jako %1.%2/www/%{REQUEST_URI} , potom totiž bude požadavek na harvie.cz přepsán do /srv/data1/domains/harvie.cz/www/, tedy požadavek na harvie.cz bude vyřízen stejně jako na www.harvie.cz, což je takový nepsaný zvyk, ale ne podmínka (mě osobně to nevyhovuje - řeším to jinak).

Další sekce označená SUBDOMAINS už vyřizuje dotazy s více jak jednou tečkou v hostname a to tím způsobem, že www.harvie.cz přepíše do domains/harvie.cz/www/ a www.old.harvie.cz přepíše do domains/harvie.cz/www.old/ , tudíž můžete použít třeba i pod-pod-pod-pod-doménu, ale už nebude umístěná hierarchicky v pod-pod-pod-pod-adresáři, ale přímo v kořenovém pro danou doménu. To lze brát jako výhodu i nevýhodu.

A tady už je kód:

```
         #DOMAINS
             RewriteCond %{HTTP_HOST} !(^[0-9]*\.[0-9]*\.[0-9]*\.[0-9]*)
             RewriteCond %{HTTP_HOST} ^([^\.]*)\.([^\.]*)$
             RewriteRule (.*) /srv/data1/domains/%1.%2/%{REQUEST_URI}
    #SUBDOMAINS
             RewriteCond %{HTTP_HOST} !(^[0-9]*\.[0-9]*\.[0-9]*\.[0-9]*)
```

```
            RewriteCond %{HTTP_HOST} (.+)\.(.*)\.(.*)$
            RewriteRule (.*) /srv/data1/domains/%2.%3/%1/%{REQUEST_URI}
```

Ještě bych poznamenal, že v případě nutnosti je možné rewrite snadno upravit tak, aby se harvie.cz nepřepsalo do domains/harvie.cz/, ale do domains/cz/harvie/, to se hodí například pokud vám dochází místo na disku při hostování velkého počtu domén. Potom je možné rozhodit weby podle TLD na různé disky. Další možností (používanou hlavně velkými clustery) je rozhození na více disků (nebo i na více strojů) podle prvního písmenka (nebo více písmenek) hostnamu - www.harvie.cz by se potom přepsalo třeba na /srv/cz/h/a/r/harvie.cz/www. To by ale vyžadovalo trochu složitější úpravu celého rewritu.

Toť vše a jedinná věc, která mi skutečně zůstala z minulé verze nastavení je přesměrování některých souborů/adresářů, které mají být dostupné na všech (pod) doménách:

```
#společná favicona:
RewriteRule favicon\.ico /mnt/data1/favicon.ico
#css, který používám v apache autoindexu (listingu):
RewriteRule harvie-admin\.css /mnt/data1/domains/harvie.cz/info/admin.css
#a ikonky vyžadované opět autoindexem:
RewriteCond %{REQUEST_URI} ^/icons/(.*)
RewriteRule icons/(.*) /usr/share/apache2/icons/$1
```

Verze 1.2

1.) Povolíme rewritování:

```
RewriteEngine On
```

2.) ZINA JuKeBox si nerozumí s poddoménou, proto nerewritujeme, ale rovnou přesměrujeme:

```
RewriteCond %{HTTP_HOST} ^music.harvie\.cz
RewriteRule (.*) http://harvie\.cz/wan/music%{REQUEST_URI} [R=301,L]
```

3.) Novinka! potřebuju přesměrovat URL ze staré domény harvie.ath.cx na novou harvie.cz - včetně poddomén a jak pro HTTP tak pro HTTPS: - URL bez subdomén:

```
RewriteCond %{HTTP_HOST} ^harvie\.ath\.cx$
RewriteRule (.*) http://harvie\.cz%{REQUEST_URI} [L,R=301]


#Pro SSL:
RewriteCond %{HTTPS} on
RewriteCond %{HTTP_HOST} ^harvie\.ath\.cx$
RewriteRule (.*) https://harvie\.cz%{REQUEST_URI} [L,R=301]
```

- URL se subdoménou

```
RewriteCond %{HTTP_HOST} ^(.*).harvie\.ath\.cx$
RewriteRule (.*) http://%1.harvie\.cz%{REQUEST_URI} [L,R=301]
```

```
#Pro SSL:
RewriteCond %{HTTPS} on
RewriteCond %{HTTP_HOST} ^(.*).harvie\.ath\.cx$
RewriteRule (.*) https://%1.harvie\.cz%{REQUEST_URI} [L,R=301]
```

4.) Potřebuju nejaka-subdomena.harvie.cz transparentne namapovat (rewritnout) na /foo/bar/baz/nejaka-subdomena/:

```
RewriteCond %{HTTP_HOST} ^(.*).harvie\.cz
RewriteRule (.*) /foo/bar/baz/%1%{REQUEST_URI}
```

5.) Potřebuji, aby soubor favicon.ico a admin.css byl stejný pro komplet všechny domény/subdomény a adresáře, protože ho často používám...

```
RewriteRule favicon\.ico /foo/bar/favicon.ico
RewriteRule harvie-admin\.css /foo/bar/baz/info/admin.css
```

6.) Protože nechci, aby mě google penalizoval za dvojitý výskyt stejného obsahu, přidal jsem ještě do adresáře s blogem následující soubor .htaccess:

```
RewriteEngine On
RewriteCond %{REQUEST_URI} /wan/blog
RewriteRule (.*) http://blog\.harvie\.cz/ [R=301]
```

Ten způsobí, že pokud by někdo na můj blog "lezl" přes url harvie.cz/wan/blog, bude automaticky "hozen" na adresu http://blog.harvie.cz/ a jako bonus od cesty dostane jeho prohlížeč pěknou "třistajedničku" ;)

Na závěr poznamenám snad už jen to, že "R=301" znamená redirect "moved permanently" - tj. přesměrování, které si prohlížeč zapamatuje a příště jde rovnou k cíli bez requestování. Pokud není jasný rozdíl mezi redirectem (přesměrováním) a rewritem (přepsáním) URL, tak je v tom, že redirect je na straně prohlížeče (prohlížeč jde na jinou adresu) a přepsání je na straně serveru (server vrátí jiný soubor). Přepisovat se dá i na jiný server (tzv. PROXY rewrite).

From:
https://wiki.spoje.net/ - **SPOJE.NET**

Permanent link:
https://wiki.spoje.net/doku.php/howto/hosting/apache-mod_rewrite

Last update: **2016/10/29 23:05**