

mod_rewrite

(URL Rewriting Engine)

Tobias Walter (tobias@unwichtig.org)

www.unwichtig.org

04. Mai 2006

- **Grundlagen**
- Suchmaschinenoptimierung
- Reverse Proxy
- Rewrite Maps
- Sonstige Beispiele
- Links

Was ist mod_rewrite?

- URL Rewriting Engine des Apache Webservers
- Kann URLs auswechseln oder umschreiben
 - `unwichtig.org/path1/ -> unwichtig.org/path2/`
 - `unwichtig.org/?page=news -> unwichtig.org/news/`
 - `unwichtig.org/foo/ -> www.google.de/search?q=bar`
- Seit Apache 1.3 offizielles Apache Modul
- Arbeitet Hilfe von Regular Expressions

"mod_rewrite, the Swiss Army Knife of URL manipulation!"

- Ralf S. Engelschall, Autor von mod_rewrite -

"Despite the tons of examples and docs, mod_rewrite is voodoo. Damned cool voodoo, but still voodoo."

- Brian Moore -

"The great thing about mod_rewrite is it gives you all the configurability and flexibility of Sendmail. The downside to mod_rewrite is that it gives you all the configurability and flexibility of Sendmail."

- Brian Behlendorf, Apache Group -

- `mod_rewrite` wird eigentlich von allen Distributionen im Apache mitgeliefert
- Ist jedoch meist nicht geladen
- Modul laden (`httpd.conf`):
 - `LoadModule rewrite_module modules/mod_rewrite.so`
- Starten der Engine:
 - `RewriteEngine On`
 - direkt in der `httpd.conf`, in einer `VirtualHost`-Direktive, in einer `Directory`-Direktive oder in einer `.htaccess`-Datei

Reguläre Ausdrücke in mod_rewrite

| | |
|--|---|
| <code>.</code> (<code>a</code>) | ein beliebiges Zeichen (<code>a</code>) |
| <code>+</code> (<code>a+</code>) | ein oder mehrere beliebige Zeichen (<code>a</code>) |
| <code>*</code> (<code>a*</code>) | kein oder mehrere beliebige Zeichen (<code>a</code>) |
| <code>?</code> (<code>a?</code>) | kein oder ein beliebiges Zeichen (<code>a</code>) |
| <code>\.</code> <code>\+</code> <code>\\</code> <code>...</code> | Steuerzeichen können mit <code>\</code> escaped werden |
| <code>^a</code> | <code>a</code> am Anfang des Ausdrucks |
| <code>a\$</code> | <code>a</code> am Ende des Ausdrucks |
| <code>a b</code> | <code>a</code> oder <code>b</code> |
| <code>[a-z]</code> | ein kleiner Buchstabe |
| <code>[^a]</code> | kein <code>a</code> |
| <code>[a-z0-9]*</code> | beliebig viele kleine Buchstaben und Zahlen |
| <code>(.*)</code> | beliebig viele Zeichen werden in <code>\$1</code> gespeichert |
| <code>([a-z]*)</code> | beliebig viele kein Buchstaben werden in <code>\$1</code> gespeichert |

Einfaches Beispiel:

```
<Directory /var/www/>  
    RewriteEngine On  
    RewriteBase /  
    RewriteRule ^index\.html$ hallo.html  
    ...  
</Directory>
```

- `123.de/index.html -> 123.de/hallo.html`
- **Außerhalb der Directory-Direktive (keine RewriteBase):**

```
RewriteRule ^/index\.html$ /hallo.html
```

Ein paar Beispiele:

```
RewriteRule ^index\.html$ hallo.html
```

- liefert `hallo.html` an Stelle von `index.html`

```
RewriteRule ^index\.html$ hallo.html [R]
```

- redirected den Browser nach `hallo.html`

```
RewriteRule ^index\.html$ - [F]
```

- verbietet den Zugriff auf `index.html`

Und noch mehr:

```
RewriteRule ^index\.html$ http://google.de/
```

– Redirect zu Google

```
RewriteRule ^(.*)$ http://google.de/$1
```

– Redirect zu Google mit Pfad und Parametern

Bedingungen:

```
RewriteCond %{HTTP_USER_AGENT} ^Lynx.*
```

```
RewriteRule ^index\.html$ index.ascii.html
```

- **Dem Browser Lynx wird `index.ascii.html` statt `index.html` geliefert**
- **Es können beliebige Bedingungen durch UND und ODER kombiniert werden**

ODER:

```
RewriteCond %{HTTP_USER_AGENT} ^Lynx.* [OR]  
RewriteCond %{HTTP_USER_AGENT} ^Links.*  
RewriteRule ^index\.html$ index.ascii.html
```

UND:

```
RewriteCond %{HTTP_USER_AGENT} ^Lynx.*  
RewriteCond %{REQUEST_METHOD} ^GET$  
RewriteRule ^index\.html$ index.ascii.html
```

- Bedingungen können unter anderem auf folgende Variablen angewandt werden:
 - `HTTP_USER_AGENT` Browser des Clients
 - `HTTP_REFERER` Referer (Woher kommt der Client)
 - `REMOTE_ADDR` IP des Clients
 - `REMOTE_HOST` Hostname des Clients
 - `REMOTE_USER` Name des Users (bei Authentifizierung)
 - `AUTH_TYPE` Authentifizierungstyp
 - `THE_REQUEST` Request (z.B. `GET /index.html HTTP/1.1`)
 - `HTTPS` "On" wenn per SSL zugegriffen wird
 - `ENV:PATH` Zugriff auf beliebige Environment variablen

Logging:

```
RewriteLog "/var/log/apache2/rw.log"
```

```
RewriteLogLevel 9
```

- LogLevel von 0 bis 9
- Alles über 2 nur zum Debugging
- Macht den Server lahm, deshalb am besten:

```
RewriteLog "/dev/null"
```

```
RewriteLogLevel 0
```

- `LogLevel 9`, ein Zugriff, Rewrite von `index.html -> hallo.html`:

```
[per-dir /var/www/] strip per-dir prefix: /var/www/index.html -> index.html
```

```
[per-dir /var/www/] applying pattern '^index\.html$' to uri 'index.html'
```

```
[per-dir /var/www/] rewrite index.html -> hallo.html
```

```
[per-dir /var/www/] add per-dir prefix: hallo.html -> /var/www/hallo.html
```

```
[per-dir /var/www/] trying to replace prefix /var/www/ with /
```

```
strip matching prefix: /var/www/hallo.html -> hallo.html
```

```
add subst prefix: hallo.html -> /hallo.html
```

```
[per-dir /var/www/] internal redirect with /hallo.html [INTERNAL REDIRECT]
```

```
[per-dir /var/www/] strip per-dir prefix: /var/www/hallo.html -> hallo.html
```

```
[per-dir /var/www/] applying pattern '^index\.html$' to uri 'hallo.html'
```

```
[per-dir /var/www/] pass through /var/www/hallo.html
```

- Grundlagen
- **Suchmaschinenoptimierung**
- Reverse Proxy
- Rewrite Maps
- Sonstige Beispiele
- Links

- URLs wie
`news.php?year=2006&month=may&day=04`
- sind hässlich
- schlecht zu merken
- Suchmaschinen mögen keine dynamischen Seiten
 - Mehr Parameter -> wird nicht indiziert oder schlechter geranked
 - Links werden möglicherweise nicht weiter verfolgt
- Unsere Wunsch URL ist:
`/news/2006/may/05/`

- `news/2006/may/05/`
- `news.php?year=2006&month=may&day=04`

RewriteRule

```
^( [^/ ]+ ) / ( [^/ ]+ ) / ( [^/ ]+ ) / ( [^/ ]+ ) / $  
$1 .php?year=$2&month=$3&day=$4
```

oder

RewriteRule

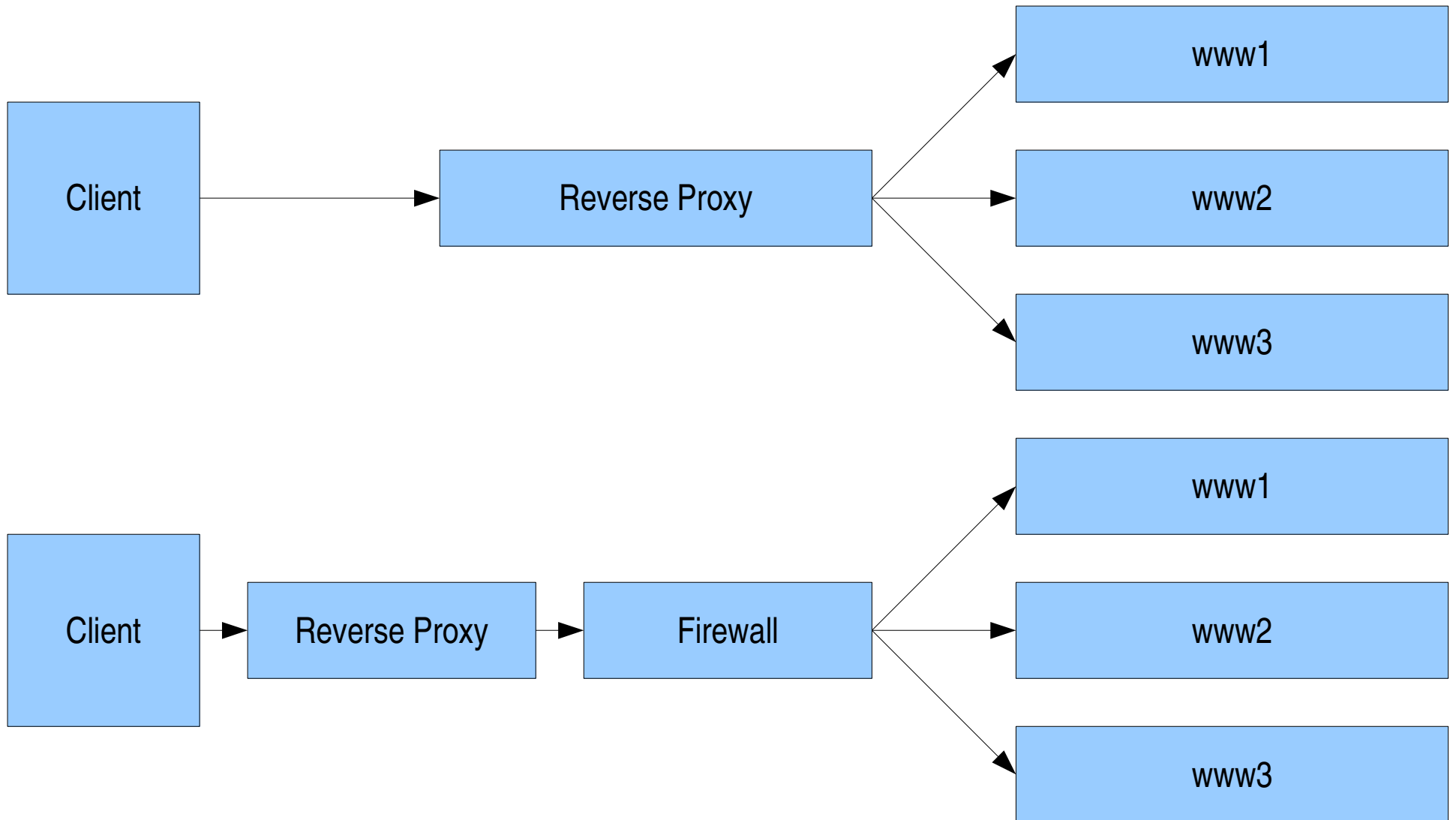
```
^news / ( [^/ ]+ ) / ( [^/ ]+ ) / ( [^/ ]+ ) / $  
news.php?year=$2&month=$3&day=$4
```

- Homepage ist über mehrere URLs erreichbar, z.B.:
 - www.unwichtig.org
 - www.twalter.de
 - unwichtig.org
 - twalter.de
 - ...
- Für Suchmaschinen vier verschiedene Seiten
- Links gehen natürlich auch auf unterschiedliche URLs
- Das Ranking wird also für jede URL einzeln berechnet

```
RewriteCond %{HTTP_HOST}
    !^unwichtig\.org$ [NC]
RewriteCond %{HTTP_HOST} !^$
RewriteRule ^(.*)$
    http://unwichtig.org/$1 [R=301]
```

- [NC] = No Case
- [R=301] = Redirect mit Code 301 (Moved Permanently)
- 2. Zeile ist nötig da ältere Browser keinen HTTP_HOST mitsenden

- Grundlagen
- Suchmaschinenoptimierung
- **Reverse Proxy**
- Rewrite Maps
- Sonstige Beispiele
- Links



Gründe für Reverse Proxys:

- Sicherheit, nur Reverse Proxy hat Zugriff auf die Webserver
- SSL Proxy, Verschlüsselung macht der Proxy
- Caching, Reverse Proxy cached dynamische Seiten
- Verschiedene Server über einen Namen zugänglich machen

- Klassische Methode mit `mod_proxy` ist wenig flexibel
- `mod_proxy` wird jedoch dennoch benötigt
- Konfiguration:

```
LoadModule proxy_module modules/mod_proxy.so
```

```
ProxyRequests Off
```

```
ProxyVia On
```

- `ProxyRequests On` würde den Apache zu einem offenen Proxyserver machen
- `ProxyVia On` fügt eine VIA Header ein

```
RewriteRule ^/(.*)$  
  http://www.unfug.org/$1 [P]
```

- **Und schon sind wir unfug.org**

```
RewriteRule ^/unfug/(.*)$  
  http://www.unfug.org/$1 [P]
```

```
RewriteRule ^/unwichtig/(.*)$  
  http://www.unwichtig.org/$1 [P]
```

- **Unter /unfug/ liegt unfug.org**
- **Unter /unwichtig/ liegt unwichtig.org**

```
RewriteRule ^([\^/]+)/(.*)$  
  http://$1.unfug.de/$2 [P]
```

```
www.unfug.org/news/ -> news.unfug.org/
```

```
www.unfug.org/plan/ -> plan.unfug.org/
```

```
RewriteRule ^news/(.*)$  
  http://newsserver/news/$1 [P]
```

```
RewriteRule ^plan/(.*)$  
  http://zeus/extern/plan/$1 [P]
```

- Grundlagen
- Suchmaschinenoptimierung
- Reverse Proxy
- **Rewrite Maps**
- Sonstige Beispiele
- Links

```
RewriteRule ^news/(.*)$  
  http://newserver/news/$1 [P]
```

```
RewriteRule ^plan/(.*)$  
  http://zeus/extern/plan/$1 [P]
```

...

- Bei vielen Servern schwillt der Apache Konfigurationsfile unnötig an
- Wird unübersichtlich

- Lösung:

```
RewriteMap proxyurls txt:proxy.txt
```

```
RewriteRule ^([^/]+)/(.*)$
```

```
    http://${proxyurls:$1}/${2} [P]
```

- proxy.txt:

```
news news.unfug.org
```

```
plan zeus.unfug.org/extern/plan
```

```
search www.google.de
```

- Grundlagen
- Suchmaschinenoptimierung
- Reverse Proxy
- Rewrite Maps
- **Sonstige Beispiele**
- Links

Dynamische Seiten verstecken:

```
RewriteRule    ^ (.*) \.html$    $1.cgi  
    [T=application/x-httpd-cgi]
```

- Alle Zugriffe auf `irgendwas.html` werden in `irgendwas.cgi` umgeschrieben
- `[T=application/x-httpd-cgi]` ist nötig damit das CGI ausgeführt wird

Spambots ausschließen:

```
RewriteCond %{HTTP_USER_AGENT}  
^NameOfBadRobot.*
```

```
RewriteCond %{REMOTE_ADDR}  
^123\.123\.123\.[1-254]$
```

```
RewriteRule ^(.*)$ - [F]
```

- **Alle Zugriffe von BadRobot aus dem Netz
123.123.123.0/24 werden verboten**

Bilderklau verhindern:

```
RewriteCond %{HTTP_REFERER} !^$
```

```
RewriteCond %{HTTP_REFERER}  
    !^http://unwichtig.org/.*$ [NC]
```

```
RewriteRule ^.*\.png$ - [F]
```

- Zugriffe auf PNGs die mit einem Referer ungleich `unwichtig.org` aufgerufen werden, werden **verboten**

- Grundlagen
- Suchmaschinenoptimierung
- Reverse Proxy
- Rewrite Maps
- Sonstige Beispiele
- **Links**

Nützliches:

- http://httpd.apache.org/docs/2.0/mod/mod_rewrite.html
- <http://httpd.apache.org/docs/2.0/misc/rewriteguide.html>
- <http://www.modrewrite.de/>
- [http://www.google.de/search?q=mod_rewrite ;-\)](http://www.google.de/search?q=mod_rewrite;-))

Historisches (aber auch nützlich):

- <http://www.heise.de/ix/artikel/1996/12/149/>