Контрол на достъпа с htaccess

Конфигурационен файл .htaccess

.htaccess е файл с който могат да бъдат променяни настройките на сървъра и позволява да се изпълняват множество функции, най-популярни от които са:

- указване на собствена 404 error страница;
- показване на файловия листинг в защитени директории;
- отхвърляне на достъп от определени IP адреси;
- промяната на страницата по подразбиране (чрез задаване друг файл да бъде използван в качеството на index файл) ;
- други.

Всяка версия на Apache web сървър поддържа .htaccess

Предназначение

- Предоставя възможност да се извършват промени на конфигурацията по отношение на директориите.
- Реализира концепцията за “разпределени конфигурационни файлове”.
- Файлът (с една или повече конфигурационни директиви), се записва в определена директория и директивите се прилагат към тази директория и нейните поддиректории.

Възможности

- По подразбиране името на такъв файл е **.htaccess**.
- Ако е необходимо името да бъде друго, това се прави с директивата AccessFileName в http.conf.

Пример:

```
AccessFileName .myconfig
```

Конфигуриране на Apache

- За конфигуриране на Apache се използват текстови конфигурационни файлове, съдържащи множество директиви. Основния конфигурационен файл е */etc/httpd/httpd.conf*. Този файл се прочита от сървъра при стартиране или рестарт.
- Последователността на обработване на конф. файлове е
  1. httpd.conf
  2. .htaccess
  3. .htpasswd

Създаване на **.htaccess**

- **.htaccess** използва същия синтаксис като основния конфигурационен файл http.conf.
- Какво може да се постави в него се определя от директивата **AllowOverride**.
  - Тя описва в категории, кои директиви ще се вземат под внимание ако те са открити в .htaccess.
  - Ако директива е разрешена в .htaccess файла, документацията на тази директива ще съдържа Override секция, описваща каква стойност трябва да е в **AllowOverride**, за да може тази директива да е разрешена.

Пример:

Директивата **AddDefaultCharset** е разрешена в .htaccess файла. За нейната Override пише **FileInfo**.

Следователно в конфигурационния файл трябва да се запише **AllowOverride FileInfo**, за да може тази директива да бъде валидна в .htaccess.

Използване на **.htaccess**

- По принцип, не е необходимо използване на .htaccess ако имаме достъп до главния конфигурационен файл на сървъра.
- По принцип, използването на .htaccess трябва да се избягва когато е възможно.
- Всяка конфигурация която може да се включи в .htaccess, може лесно да бъде включена в <Directory> секцията в главния конфигурационен файл на сървъра.
- Използването на .htaccess може цялостно да бъде забранено с директивата **AllowOverride None**.

Използване на .htaccess

- Предоставящите за публичен достъп съдържание трябва да правят промени на конфигурацията по отношение на директории, но нямат административен достъп до системата.
- Ако администраторът не желае да прави чести промени, той може да разреши индивидуални потребители да правят тези промени в .htaccess файлове.

Пример:

Когато ISP поддържат много потребителски сайтове на една машина и потребителите трябва сами да си променят техните конфигурации.

Причини да не се използва

- Производителност
  - Когато директивата **AllowOverride** е включена за да осигури използването на .htaccess файлове, Apache ще преглежда всяка директория за тези файлове, дори и реално да не се ползват в някои директории.
  - При всяко заявяване на документ, файлът .htaccess от неговата директория винаги се прочита.
- Сигурност
  - Ако на потребителите им е разрешено да променят конфигурацията на сървъра, това може да рефлектира в промени, върху които администраторът няма контрол.

Пример за алтернативи

- Напълно еквивалентно е, което трябва да бъде в .htaccess да се включи като част в секцията Directory в главния конфигурационен файл.
- Пример: .htaccess е в директорията */www/htdocs/example*:

```
Съдържание на .htaccess file в /www/htdocs/example
AddType text/example .exm

Секция от httpd.conf файла
<Directory /www/htdocs/example>
        AddType text/example .exm
</Directory>
```

- Ако е записана в главния файл, това ще подобри производителността, защото конфигурацията ще се зареди само веднъж при стартиране на сървъра, вместо всеки път при заявяването на документа.

Прилагане на директивите

- Прилагат се към директорията, в която .htaccess е открит и към всички нейни поддиректории.
- Такива файлове може да има и в директориите от по-горните нива.
- Директивите се прилагат по реда на тяхното откриване - .htaccess файл в една директория може да препокрие директивите, открити в по-горните нива.

Пример:

В директорията */www/htdocs/example1* има .htaccess съдържащ

```
Options +ExecCGI
```

(Трябва да е била указана директива **AllowOverride Options** преди това)

В директорията */www/htdocs/example1/example2* има .htaccess, съдържащ

```
Options Includes
```

Поради втория .htaccess в дъщерната директория, там изпълнението на CGI не е разрешено, защото само **Options Includes** има ефект.

Автентициране

- Автентицирането на потребител е необходимо първо да бъде това разрешено с директивата AllowOverride AuthConfig.
- За осъществяване на достъп е необходимо създаване на специален файл с пароли.
- Този файл може да бъде записан на всяко място, до което няма достъп от Web.

Пример:

Ако документите се поддържат в */usr/local/apache/htdocs*, то файлове с паролите могат да бъдат в */usr/local/apache/passwd*.

Създаване на пароли

- За създаването на файл се изпозва програмата *htpasswd*.

```
htpasswd -c /usr/local/apache/passwd/passwords testuser
```

- Пита се за парола и нейното повторно въвеждане.
- Опцията -c създава нов файл за потребител *testuser*.
- Необходимо е и задаване на минимални права за четене от web-сървъра.

Пример:

Ако сървърът е конфигуриран да се изпълнява като потребител *nobody* и група *nogroup*, то само web сървърът да може да чете и само *root* да пише:

```
chown root.nogroup /usr/local/apache/passwd/passwords
chmod 640 /usr/local/apache/passwd/passwords
```

Конфигуриране на автентикацията

- Конфигурира се сървъра да пита за парола и да се укаже за кои потребители да има достъп.
- Например, ако е необходимо контрол на достъпа до */usr/local/apache/htdocs/secret*, могат да се използват следните директиви, описани или в *httpd.conf* в секцията **<Directory /usr/local/apache/apache/htdocs/secret>** или във файла */usr/local/apache/htdocs/secret/.htaccess*.

```
AuthType Basic
AuthName “Access to restricted files”
AuthUserFile /usr/local/apache/passwd/passwords
Require user testuser
```

Директиви

- **AuthType** - определя метода, използван за автентикация.
  - По-често използваният метод е Basic. Той изпраща паролата от клиента в чист текст.
  - Другия по сигурен метод е Digest. Паролата се изпраща MD5 криптирана.
  - В този случай се използва друга програма за създаване на паролите:

```
htdigest -c /usr/local/apache/passwd/htdigest realm
                                                username
```

- **AuthName** - указва име (Realm), което да бъде използвано в автентикацията. Това ще се изведе в диалога за въвеждане на име/парола за потребителя.

Диалог между клиента и сървъра

- Когато се изисква ресурс, Apache изпраща хедър 401 (Authentication Required), за да уведоми клиента, че трябва да се дадат кредениалите на потребителя, за да му се предостави ресурса.
- Ако клиентския браузър поддържа Basic автентикация, ще пита потребителя за име и парола, които се пращат на сървъра.
- Понеже HTTP е stateles, всяка заявка се обработва отделно, дори те да са от един и същи клиент. Т.е. за всеки ресурс трябва отново да се пращат кредениалите.
- За оптимизирането на това, браузърът поддържа информация, така, че е достатъчно само еднократно въвеждане на кредениалите от страна на потребителя.
- Заедно с 401 отговора ще се изпрати и друга информация обратно към клиента. В частност се изпраща име, асоциирано със защитена област на сайта. Това се нарича Realm или автентикационно име.
- Браузърът кешира потребителското име и парола и ги съхранява с този Realm. Ако се изисква ресурс за същия Realm, ще се изпратят същите потребителско име и парола.

Директиви

- **AuthUserFile -** указва пътя до файла с паролите.
  - Ако има множество потребители не е изгодно търсене в текстов файл.
  - Тогава може да използва база данни с директивата **AuthDBMUserFile**.
  - Тези файлове могат да се създадат с програмата *dbmanage*.
- **Require -** указва потребителя, който ще има достъп до тази област на сървъра.
  - Ако е необходимо да не се указва конкретен потребител за достъп, то може да се зададе всеки валиден потребител, който е създаден с *htpasswd/htdigest*

Require valid-user

Всеки описан потребител във файла с паролите и въвел коректна парола ще бъде допуснат до ресурса.

Ако има включени и директиви **Allow** може да се укаже как да се прилагат **Allow** и **Require** директивите.

- **Satisfy all** - Клиентът трябва да удовлетвори И име/парола И разрешенията по адрес.
- **Satisfy any** - Клиентът трябва да удовлетвори поне едно от име/парола ИЛИ разрешаване по адрес.

Създаване на потребителски страници за грешки

- Едно от приложенията на .htaccess е създаване на собствени страници за грешки, вместо стандратните.
- За всеки код на грешка може да се зададе

ErrorDocument <errornumber> <file_html>

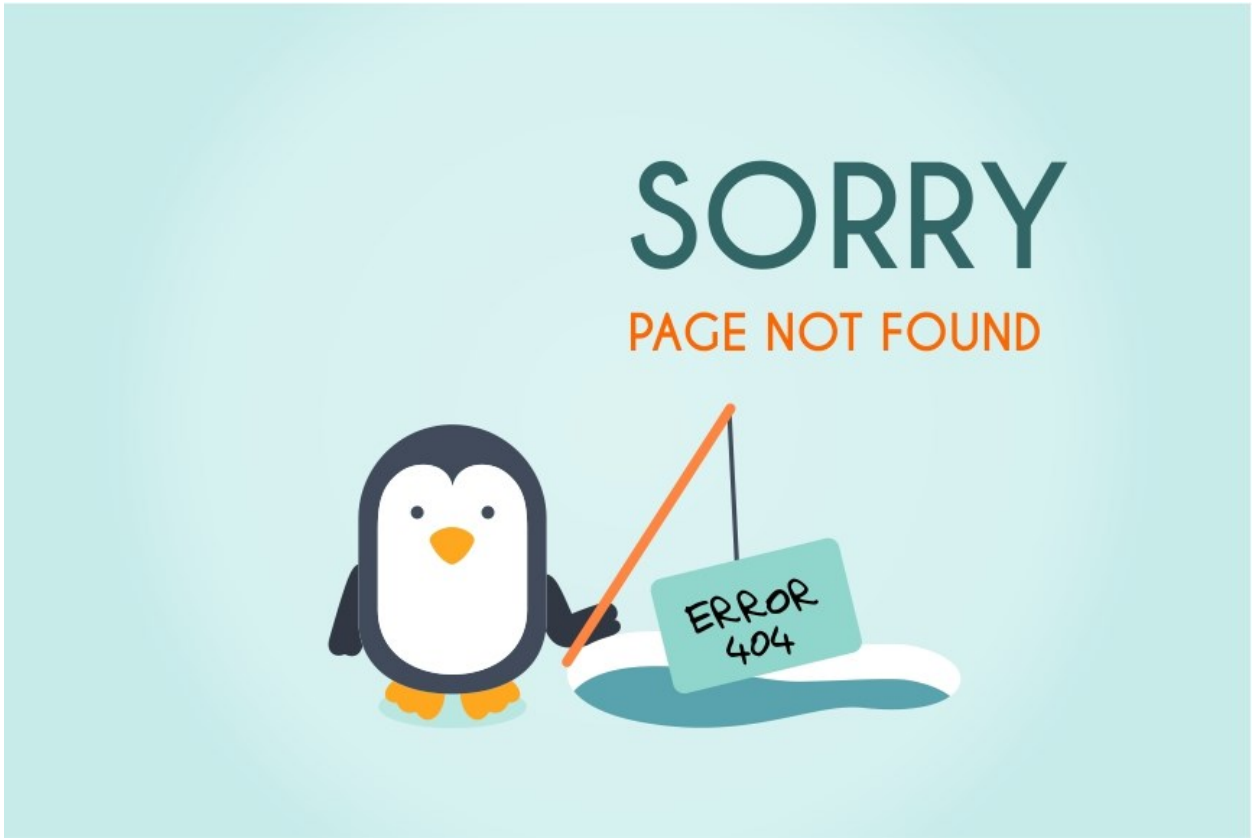
Някои общи кодове за грешки
401 - Authorization required
400 - Bad request
403 - Forbidden
500 - Internal Server Error
404 - Wrong Page

Пример:

ErrorDocument 404 errorfolder/notfound.html

(файлът не е в основната директория, а е в *errorfolder*)

Потребителски страници



Кодове за грешки

Client Request Errors	
400	Bad Request
401	Authorization Required
402	Payment Required (not used yet)
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable (encoding)
407	Proxy Authentication Required
408	Request Timed Out
409	Conflicting Request
410	Gone
411	Content Length Required
412	Precondition Failed
413	Request Entity Too Long
414	Request URI Too Long
415	Unsupported Media Type

Server Errors	
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported

Други опции

- Забрана за показване на файлове и папки

  Options -Indexes

- Разрешаване за показване на файлове и папки

  Options +Indexes

- Ако е необходимо използване на други индексни файлове не само на *index.html*

  DirectoryIndex index.php index.my

Пренасочване

- Една от често използваните функции на .htaccess е възможността за пренасочване на заявките към различни файлове, или на същия сървър или на изцяло друга машина.
- Използва се ако са променени имената на файлове, а потребителите не знаят това.
- Друг случай е пренасочване към дълги URL (реалните), докато потребителите използват по-кратки.

Пренасочване на файлове:

```
Redirect /location/from/file.ext http://www.othersite.com/new/location.xyz
```

Пренасочване на цяла директория:

```
Redirect /olddirectory http://www.newsite.com/newdirectory
```

```
Redirect 301 /old/old.html http://www.newsite.com/new.html
```

Код 301 указва, че е преместена за “постоянно”. Това запазва текущия Search Engine ранг.

Пренасочване на цял сайт

```
Redirect 301 / http://www.newsite.com
```

Въпроси ?