

Контрол на достъпа с htaccess

проф. д-р инж. Венета Алексиева

Основни моменти

- Понятия
- Предназначение
- Използване
- Примери

Понятия

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

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

Всяка версия на Apache уеб 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**.
- Например:
AccessFileName .myconfig

Създаване на файла `.htaccess`

- `.htaccess` използва същия синтаксис като основния конфигурационен файл `httpd.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.
- Например:

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

- Не е необходимо използване на .htaccess ако има достъп до главния конфигурационен файл на сървъра.
- 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.

Но използването му е оправдано когато:

- Трябва да се правят промени на конфигурацията по отношение на директории, но без root достъп до системата.
- Ако администраторът не желае да прави чести промени, той може да разреши индивидуални потребители да правят тези промени в .htaccess файлове.
- Например:

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

Причини за използването му:

- **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 алтернативите

- Напълно еквивалентно е, което трябва да бъде в `.htaccess` да се включи като част в секцията `Directory` в главния конфигурационен файл.
- Пример:
 - `.htaccess` file in `/www/htdocs/example`:
 - Contents of `.htaccess` file in `/www/htdocs/example`**
 - `AddType text/example .exm`
 - Section from your `httpd.conf` file**
 - `<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 директории 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.

Автентизиране

- Автентизирането на потребител е необходимо първо да бъде 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.
- Например:

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

Автентизиране- 2

- За създаването на файл се използва програмата `htpasswd`.
`htpasswd -c /usr/local/apache/passwd/passwords testuser`
- Ще се пита 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
```

Следващата стъпка

- 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`):

```
Auth Basic
```

```
AuthName "Restricted files"
```

```
AuthUserFile /usr/local/apache/passwd/passwords
```

```
Require user testuser
```

Пояснения

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

```
htdigest -c  
/usr/local/apache/passwd/digest 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.

Как работи автентикацията?

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

И още

- 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 - Клиента трябва да удовлетвори И име/парола И разрешенията по адрес.
- Satisfy any - Клиента 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 стандартните.
- 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

- Например:

ErrorDocument 404 errorfolder/notfound.html

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

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, а не само на index.html

DirectoryIndex index.php alabala.my

Пренасочване

- Една от повече 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> - 301

преместена е “постоянно”. Това запазва текущия search engine ранг.

Redirect 301 / <http://www.newsite.com> - пренасочване на цял сайт

Въпроси ?

Благодаря за вниманието !