RewriteSecurity

Substitution paths

The second argument of the RewriteRule directive specifies the substituted URI reference or file path. A very common misconception is that the substituted string is always taken to mean a URI reference relative to the DocumentRoot.

For example, one might try the following rule to append a query parameter to an incoming request.

```
# DO NOT DO THIS!
RewriteEngine On
RewriteRule (.*) $1?foo=bar [QSA]
```

But given that the substitution string is first tried as an absolute filesystem path, and if that doesn't work, as relative to the DocumentRoot, a request such as http://example.com/etc/passwd would serve the system password table.

One solution is to prepend the document root in cases where the resulting path could be ambiguous. Such as:

```
RewriteEngine On
RewriteRule (.*) %{DOCUMENT_ROOT}/$1?foo=bar [QSA]
```

Mitigating Information Disclosure

The best way to avoid situations like this is to only allow access to file paths relevant to the context at hand. For example:

```
<Directory />
 Deny from all
</Directory>

NameVirtualHost *:80

<VirtualHost *:80>
 ServerName www1.example.com
 DocumentRoot /var/www/www1.example.com
 <Directory /var/www/www1.example.com>
  Allow from all
 </Directory>
</VirtualHost>

<VirtualHost *:80>
 ServerName www2.example.com
 DocumentRoot /var/www/www2.example.com
 <Directory /var/www/www2.example.com>
  Allow from all
 </Directory>
</VirtualHost>
```

In this way, vhosts are restricted to their relevant areas in cases of overly promiscuous rewrite rules. Most, if not all, installations of Apache do this already (denying from all in the / directory).