FileSystemSecurity

The Apache HTTP Server needs read access to its configuration files and the files it serves. In and of itself, the server does not need write access anywhere on the system: even its log files are opened for write when the server is still root, and the open file descriptors passed to the child processes which change their user id to the lesser privileged user.

Read access only. The web server user should not own, or be able to write to, its configuration files or content.

Content, other than CGI scripts, generally does not need Execute permissions. Even PHP files that are interpreted by the server do not need to be Executable.

Certain applications, especially publishing platforms and Content Management Systems that you manage and populate through the web server itself using a browser, require that certain directories on the system be made writable by the web server user. You can do this by changing the owner of the directory to that user (usually www but ymmv), or by making the directory group-writable and changing the group to the group as which Apache runs.

Making directories writable by the web server should be done only with care and consideration. The typical attack model is: someone manages to upload (for instance) a PHP script of their own making into the document root, and simply executes that by accessing it through a browser. Now your machine is executing their code under their control.

If a web app needs writable directories, it's often better to have those outside the Document Root (http://httpd.apache.org/docs/2.2/mod/core.html#documentroot): that way the uploads can't be accessed from the outside through a direct URL. Some applications, such as WordPress http://wordpress.org/ support this; others do not.

In many cases, writable directories are not strictly necessary even though the web app might like them: rather than upload plugins (which contain code that gets executed or interpreted, yech!) through the web browser, upload them through ssh and manually unpack them on the server. The Joomla! CMS, for instance, attempts to write its configuration file to the Document Root during installation – this is therefore a popular target – but if it can't write to the Document Root, it will output the config to the browser to the user can manually upload it.

(Credit to Sander Temme for elaborating on this subject in a much more concise fashion than I could have achieved)