Cross-Site Request Forgeries (CSRF)

Taken from PHP Under Attack:
Cross Site Request Forgery (CSRF) exploits the trust a web site has for a particular user. It involves tricking a user into unknowingly sending a HTTP request of the attacker's choosing to the vulnerable web site. The following example demonstrates CSRF:

1. A trusted user logs into http://top-secret-site.com/vulnerable.cgi
2. The user is tricked into visiting a malicious site.
3. The malicious web site sends the user the followig HTML:

< img src="http://top-secret-site.com/vulnerable.cgi?action=trusteduseraction&user=gullibledude">

The following is an article from ApacheCon - very good reading.
Secure PHP Programming: Foiling Cross-Site Attacks

Here are some pieces of background for your personnal knowledge :
(excerpt from many mails from Jun Moriya)

Jun 13, 2001 bugtraq at securityfocus.com
Cross Site Request Forgeries (CSRF, pronounced "sea surf")
http://www.tux.org/~peterw/csrf.txt

2003/07 php conference at oscon 2003
http://conferences.oreillynet.com/os2003/php/

PHP Under Attack
http://conferences.oreillynet.com/cs/os2003/view/e_sess/4114

PHP Under Attack OSCON 2003 (slide)
http://talks.php.net/show/php-under-attack
http://talks.php.net/show/php-under-attack/11
http://talks.php.net/show/php-under-attack/15

PHP Quebec 2005, Security and common vulnerabilities
Slides
Full audio track on CD ROM

Read documentation in comments of file lib/tikiticketlib.php :
http://de.tiki.org/xref-head/lib/tikiticketlib.php.source.html

security bugtraq

List of Security issues Past

---

security bugtraq

Easy DOS?

- Hitting F5 repeatedly to refresh a page can ratchet up server load easily... try it at home see what happens
- the ab command if you've got apache running... try this one at home
    - ab -n 15 http://YOURTIKISITE.com:80/tiki-index_raw.php
        - I watched my server load climb quickly as this benchmark was run... This seems like an easy DOS tactic to me.

Security issues when developing Tiki

Writing secure code

Go here for this topic: SecurityRules

How to Secure Tiki

Content in filesystem

While most of the data in Tiki is stored in a database, some can be stored in filesystem, like file or image galleries, or backups. Those directories could be outside the web tree (but accessable from apache/php), or have some restriction preventing the webserver from accessing them directly, i.e. for apache servers a file named .htaccess could be put in those directories with something like

))AuthType((Basic
))AuthUserFile((/dev/null
))AuthGroupFile((/dev/null
Require valid-user

This should work in any directory which content should not be accesible directly, but what happens with i.e. img/wiki_up? For those dirs in apache configuration could be made, i.e. adding a directory directive for the wiki_up directory, and inside that directive, a files ".php" directive to block the loading from that dir of php files (and could be done more for other kind of executable files that could be put there, i.e. shtml) much like the disabling of loading .htaccess files is already done in apache configuration.
(this description should be heavily rewritten for clarity 😀)

Protecting the Apache))WebServer((

modsecurity
Take a look at http://www.modsecurity.org/. ModSecurity is an open source intrusion detection and prevention engine for web applications (or a web application firewall), operating as an Apache Web server module or standalone webapp. See TikiModSecurity for tips on howto use.

modrewrite
If you are using mod_rewrite, you can restrict the files served by Apache. This example rule prevents Apache from serving files that are not typical for web pages. The file gallery and attachment files are delivered to the user through tiki-download_file.php, so their file type will not activate this rule.

```
RewriteEngine on RewriteBase /tiki RewriteRule !\.(?i:php|gif|jpe?g|png|tiff?|css|js)$ - [F]
```

Protecting from PHP running in unsolicited locations

Consider setting php_openbasedir value in .htaccess or your vhost setup. As an example, take a look at the code below:

```
php_admin_value open_basedir /var/www/tiki19 php_admin_value upload_tmp_dir /tmp
php_admin_value session_save_path /tmp
```

Disable execution of PHP by default

Disable PHP execution by default and then enable it for directories that should be accessible by Apache. This will also prevent PHP execution in directories used to store uploaded files; although these should be outside of the Apache tree (DocumentRoot) anyway. Remember to enable the PHP engine for other non-TikiWiki directories with PHP pages.

```
php_admin_flag engine off php_admin_flag engine on
```

Using Admin Security

See here: AdminSecurity

PHP Settings

Some suggested php.ini settings. The last one is needed though for Blog Trackback Pings to work.

```
register_globals = Off
safe_mode = On
allow_url_fopen = Off
```

Apply a "web-application-firewall"

HowToApacheModSecurity

Use Request and Bandwith Limiting

HowToApacheModCbandRateLimiting

---

in tikiwiki internal admin/general page set "disallow access to site if load is above threshold" , set threshold to reasonable load, say 7