Drupal Security Best Practices

1. Cross Site Scripting (XSS) is the number one vulnerability in Drupal code. Make sure that any user input(including those from the url) or data from database is passe through **filter_xss** or a more restrictive function. - **YES**
2. Use hook_permission for fine grained access controls. Instead of creating a single permission for controlling all the features of a functionality, create more permissions like read, write, delete with combinations of own and any. - **YES**
3. Make sure that input filters(full HTML and php) are given only to trust-worthy roles. - Only TLS should enable these fitlers
4. Make sure the while letting the end users upload files, only files with certain extensions that make sense in the given context are enabled. - Engineering Specs document. Limit memory(space) usage per user.
5. File permissions : We generally use chmod -R command for changing the permissions. This can compromise the security as different files need to have different permissions set. We should come out with a standard command that can change the permissions of only safe set of files.

We can also use the bash file mentioned at the end of http://drupal.org/node/244924**. Let us call the file file_permissions.sh** in home folder. **YES**

6. Validate all the view arguments. **YES**
7. Make sure that access is set appropriately in all the views. **YES**
8. When you are using links for important functions like update or delete some record in db use the confirmation forms. **YES**
9. If we cannot use the confirmation forms, then we must use one time token verification

==========

Link(token) creation side

```
l('Title', 'url', array('query' => array('token' => drupal_get_token('unique-content-id'))));
```

Token verification side

```
if (!drupal_valid_token($_GET['token'], 'unique-content-id')) {
    return drupal_access_denied();
  }
```

==========

10. Similar steps to be taken for all the ajax calls too. **YES**
11. http://drupal.org/project/security_review

General

1. Mention module dependencies in the info file.
2. If a module is not listed as a dependency in the file but you are using code from the module then use module_exists to see if the mentioned modules is enabled or if you are not sure as to which module is providing the function then use function_exists to check that the function exists or use.
3. Disable writing logs to screens on live sites.
4. Move all the non hook implementations to a common inc file
5. Any code that is written in views should be moved to code files and use the above mentioned functions.
6. For the current user user $user variables for any other use use $other_user or any other appropriate variable.
7. Make the /node inaccessible if it is not being used.
8. If the individual pages of a particular node type are not present in the site plan, change the access or redirect them accordingly.
9. Wherever possible use fields API instead of defining your own custom tables.
10. If a similar display is used in more than one locations, then use view_modes along with Display Suite to theme it accordingly.
11. Do not use functions like node_load ,user_load for retrieving one or two fields.As these are costly functions which takes more load time.

Basic module needed for common development.

1. Use http://drupal.org/project/clientside_validation to reduce the unnecessary load on the server for validations.
2.