One of the major tasks that we need to finish before our initial release is the modification of the file management system in Drupal. This is extremely important for AMBuzz.com due to the large number of files that we are going to be dealing with. This will increase incrementally soon after the initial release when we start implementing some of the other functionality that the site was originally designed for. These files consist of images, movie, audio, flash, pdf, and documents mainly, but the file system should handle most anything that we throw at it.

Here are the major issues that I see with the file system as it exists currently.

I. Files are stored at directory level with its original file name and extension.

    1. Allows users to type in http://www.ambuzz.com/files/teaser_trailer1.mov and download the file. This i way out of line with our security needs.

    2. What happens when two movies share teaser_trailer1.mov as a file name?

    3. Provides no logical directory structure.

II. There is no option for storing files as binary data in database form.

    1. For data warehousing applications, the lack of ability to store files in databases is detrimental.

    2. Drupal offers an excellent base for asset management applications but the lack of database storage makes it very difficult to obtain.

III. Files should be a node by default. This allows even the simplest image within a page to have all the benefits that any other node on the page is allowed.

    1. Files should should have permission options as fine grained as any other object in the Drupal universe.

    2. Extra data fields should accessible when a file content type is selected.

    3. Creating files as nodes allows their inclusion into views very simply.

Here are the AMBuzz specific needs that file system should cover

I. Security. I was going to address this last, but this seems to be the brick wall that I run up against every time while dealing with the existing file system.

    1. Files cannot be downloaded at random from AMBuzz. A large part of our core functionality is providing sales information about releases to our field reps. Our major customer cannot be allowed to have access to this information.

    2. Files are prevalent throughout AMBuzz. Movie trailers, music samples (and sometimes even full albums), images, sales notes, sales information documents, book cover images and many other types of files can be related to a single release. Sales reps, divisional managers, vendors, customers, and other types of users all have access to releases and I can see this quickly becoming a nightmare to

maintain with the current system.

3. Workflow makes files available to only certain roles before publishing. This is also a concern of mine as the file system, as it sits, allows absolutely no security. Since AMBuzz will have pre-release information listed on it before it is general public knowledge, this become a priority item on our list.

II. Galleries. Our galleries are not just image galleries, but media galleries. While this is possible using existing functionality, storing files as nodes allows the ability to attach a thumbnail image to the node as well as creating fine-grained security on those files.

III. Robust searching. The ability to quickly locate and view any file in the system. Storing files as nodes allows Drupals native search functionality to work as intended for files as well.

IV. For the future the ability to provide secure web services for these files would be extremely useful to our company. Search, update, create, categorization, deletion, addition of files to our system via web service methods would be an especially useful item

V. Asset management. The ability to archive, categorize, apply fine-grained security to, version tracking, and all of the myriad of functions available to major commercial asset management packages should be a goal for us to reach. This would be best suited for a module, but the ability to store binary versions of files in a database would be a huge boon for the creator of the module.

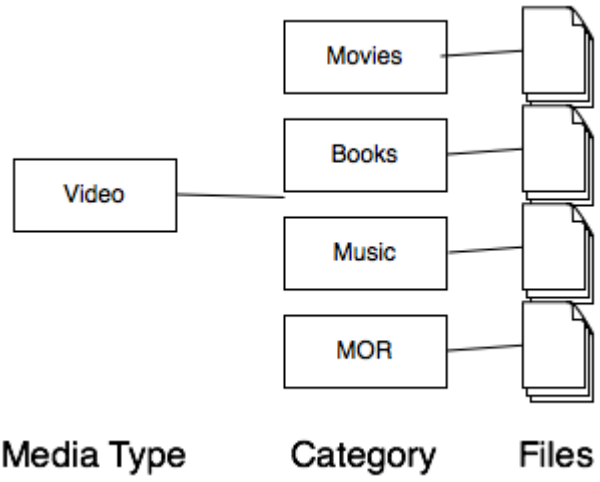
VI. The ability to attach several file types and information on those files.

VII. Tagging, tagging, tagging all within the boundaries of our security schema.

Okay, so now that we know what the requirements are, here are some of my suggestions to meet these goals.

I. Files are stores with serialized filename and stripped of their extenstion. Data about file type, mime type, default “player”, file size, and other attributes are stored in a table within the database. While this only provides a very false sense of security in the beginning, it is at least a step in the right direction for file storage. This would be best suited for a “Phase I” release.

II. A directory structure based off of categorization of files. For example:



III. The ability to choose database storage of files. Definitely a Phase II release item.

IV. Make files a node object. This solves a ton of problems right off the bat for us.