1.5 Server Standards

> This section is just starting.

Many server standards are covered in build documentation, for example, Linux and Unix Server Administration.

This area is for common standards across operating systems and has more in depth details about the why we use them.

Passwords

Vault

Ideally our organization has central Vault Password Management capability. This technology also often includes increased security (ie strong generated passwords, multi-factor authentication) and reporting. Also, when integrated, can include one-time.

Personal Password Manager

There are many free personal password manager tools that can also be leveraged.

Algorithm

Where the above options are not viable...

For small scale systems, we select a standard password and then append an algorithm to all our passwords based upon what is being protected. This seems to be a good compromise which prevents password overload.

Here is an example. We pick something that is at least 7 characters, mixed case, special characters but is easy to remember and fast to type. For example,

bonsai trees. = Btr33s.

Next we look at the item we are protecting. I use the following flow picking elements that are variable,

- Domain Name www.krypton.com
- Server Name
- User Name

For example, I want to setup a master password for my database. In this case, I would use the server name which is krypton. We apply an algorithm to it. Look at the first letter which is k and the last letter n. We then define our algorithm,

k Letter x + 1 + uppercase = L
n letter n + 1 = o

The password is then,

Btr33s.Lo

Use this to technique to also protect the database on server Earth. The password for Earth is,

Btr33s.Fi

Between Environments IST, QAT and Production, use a distinguisher. That distinguisher also helps to prevent accidents such as deleting files in the wrong environment. In this case, we put add to as a first letter,

IST = iBtr33s.Fi
QAT = qBtr33s.Fi
Production = Btr33s.Fi

I tend to **not** have a distinguisher in production to keep things simpler there.

Folder Backup

In addition to version control it is often useful to backup folders. Especially since we tend to build self-contained applications.

First we use the tar command to ensure permissions are kept. Second we use relative paths to prevent accidents where you can easily untar and overwrite your work.

Last, we use a very specific naming structure, 2011-01-20.version.folder.user.comment.bck.extension and generally keep our backups in one single backup folder, /opt/backup/

For example, if we make a tar file to backup our /opt/tomcat-a/ folder,

```
cd /opt
tar -cvf ./backup/2011-01-20.v1.opt.tomcat-a.bhitch.before_upgrade.bck.tar
```

Here are the details of how we use the naming structure,

- 2011-01-20 - the date, yyyy-mm-dd ensures the **ls** listing is sort-able
- version - references a version for the release
- folder - specifies what folder the backup should be restored to
- bck - the key we have chosen to make searching for backup easier

File Backup

With file backups the recommended approach is to keep any changes in source control and this should be the primary approach.

However, it is understood that sometimes you are just trying out something and not sure it will work. The only environment possible to try this out is an actual server environment.

File backup uses a similar convention to folder backups. However, with file backups we find it more useful to sort by the file name first then the date,

```
file.2011-01-20.version.user.comment.bck.extension
```

For example the backup file for welcome.jsp would be named,

```
welcome.2011-01-20.v0.0.bhitch.before_custom_java_set.bck.jsp

# If the change is simple then it can be,
welcome.2011-01-20.v0.0.bhitch

# If it is one-time change to the original file by the serveradmin account
I have started to use,
welcome.ori
```

> Sometimes we want to retain the original file extension to prevent security breaches. For example, renaming a .jsp to .bck would allow attackers to view the source code of the backup file. However, in some cases you want to ensure that the file is never used. In that case, use *.jsp.bck. The key is to have the bck so searches and filters can easily find backup files.