FabLab Server

SLQ Wiki Fabrication Lab 2026/04/24 08:40

This is an archived page , no further development is anticipated

FabLab Server

The fablab server is an in-house edge programming server, running the internal wiki. The need for a hosted wiki has come about through the mortal need for documentation, inspired in part by the CCC project. Researching the available wiki software (including confluence available on SLQ networks) we settled Dokuwiki. Dokuwiki is simple to edit, maintain and back-up. It doesn't require a database for installation and can be edited from any modern web browser. The wiki uses a custom template and has a few extensions installed.

> The Fablab Server is used for development over the internal networks only!! It is not secured enough for external deployment, the instructions here are considered solely for an internal, private server.

Hardware

The server runs on a recycled HP8100 mid-tower. Its physically located in The Edge's Fabrication Lab.

Software

~~The server is running Ubuntu 15.10, with a LAMP stack.~~Ugraded to 16.04.

rdiff backup with cron to external USB drive

The back-up routine is daily to an external drive. This will be augmented to a second drive and hopefully some space on an ICTS asset somewhere…

The server runs a cron job (a regular scheduled task) to back up the entire web directory (www) to an external drive.

Finding and Mounting drives and directories

First lets check the www directory - normally it resides in /var/www/html

```
ls -al /www/www
```

UIn our case, these are symlinked to another location[1)]. For example the dokuwiki install is:

```
lrwxrwxrwx 1 www-data www-data 17 Mar 17  2016 /var/www/html/dokuwiki ->
/www/www/dokuwiki
```

The /www folder is actually another drive mounted on boot - check it with `nano /etc/fstab`

```
# /www was on /dev/sdb1 during installation
UUID=d497c901-515c-456f-bddd-aaa5684c4bd3 /www            ext4    defaults
0       2
```

Now lets mount our backup USB on boot - First chuck it in then find the UUID using `sudo blkid`

```
/dev/sdc1: LABEL="edgeserverbkup"
UUID="620df2f1-4787-4584-93d3-2cb4ad24d983" TYPE="ext4"
PARTUUID="ad59a428-01"
```

Then edit fstab to mount on boot.

```
UUID=620df2f1-4787-4584-93d3-2cb4ad24d983 /mnt/externalusb        ext4
defaults        0       1
```

Now its time to reboot.

Using rdiff-backup

From rdiff-backup;

rdiff-backup backs up one directory to another, possibly over a network. The target directory ends up a copy of the source directory, but extra reverse diffs are stored in a special subdirectory of that target directory, so you can still recover files lost some time ago. The idea is to combine the best features of a mirror and an incremental backup. rdiff-backup also preserves subdirectories, hard links, dev files, permissions, uid/gid ownership, modification times, extended attributes, acls, and resource forks. Also, rdiff-backup can operate in a bandwidth efficient manner over a pipe, like rsync. Thus you can use rdiff-backup and ssh to securely back a hard drive up to a remote location, and only the differences will be transmitted. Finally, rdiff-backup is easy to use and settings have sensical defaults.

What we want is a incremental back-up of the entire /www/www/ directory to /mnt/externalusb/www. With rdiff-backup this simple. First lets make our directory on our external drive.

```
sudo mkdir /mnt/externalusb/www
```

This is the command we want to run:

```
rdiff-backup --preserve-numerical-ids /www/www /mnt/externalusb/www
```

run it once (as sudo) to check its working.

Next step is to make this happen daily..

Set up a Crontab

Access the system crontab.

```
sudo crontab -e
```

We want daily backups so add this line

```
@daily rdiff-backup --preserve-numerical-ids                    /www/www
/mnt/externalusb/www
```

Save and exit.

Automount USB

https://askubuntu.com/a/772387

Mobile Uploads

Why it was done

Motivated by the finicky nature of uploading I've hacked a together a solution that copies over uploaded photos to this namespace. The old method was:

1. take photo
2. upload over usb to computer
3. log into wiki
4. upload to wiki
5. go to media manager and insert photo.

New method is;

1. take photo on mobile (with this camera app - upupu on iOS)
2. or DAVsync on Android.
3. uploaded via webdav (this is built into upupu and occurs automagically with DAVsync)

4. go to media manager and insert photo.

Setup For Staff

iOS

IF you want to use your own IOS device - device install upupu app from store.

Set up the webdav settings in upupu with:

> Webdav ON
>
> server http://192.168.36.78/webdav/mobileupload/yourname
>
> user : yourname
>
> pass : yourpass

Source can be found on github

Android

Install DavSync from Google play or compile it yourself. Then, chose “DavSync” in your application drawer. Set up the following fields:

> server
> http://192.168.36.78/webdav/mobileupload/yourname/
>
> user : yourname
>
> pass : yourpass

Source can be found on github.

Server Set-Up

The server has folder, accessable over webdav (a simple file sharing system that is accessible over http), that is monitored by lsyncd. Lsyncd has a configuration file (written in lua) that tells its to copy the contents of the webdav folder into the media directory in the wiki.

Set-up webdav

Originally webdav was set-up with digest authorisation however a couple of the syncing apps would only use password, so we've switched to basic auth.

> This is another reason the server is not production ready - basic auth should be over SSL only!!!

Apache Setup

First up webdav needs to be turned on in our apache2 server using

```
sudo a2enmod dav
sudo a2enmod dav_fs
```

Next we make a webdav directory on our data drive:

```
sudo mkdir /www/www/webdav
```

And link to to our web directory

```
sudo ln -s /www/www/webdav/ /var/www/html/webdav
```

Lets make a directory for mobile uploads.

```
sudo mkdir /var/www/html/webdav/mobileupload
```

Now make directories in the webdav directory for each staff member

```
 sudo mkdir /var/www/html/webdav/mobileupload/phil
```

Change ownership so the apache2 server can read the link and directories;

```
sudo chown -R www-data:www-data /var/www/html/webdav
```

Now edit the apache2 server sites-available with

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

And enter the following:

```
  Alias /webdav /var/www/html/webdav
    <Directory /var/www/html/webdav>
        DAV On
```

```
      AuthType Basic
      AuthName "webdav"
      AuthUserFile /var/www/passwd.dav
      Require valid-user
      </Directory>
```

Webdav Passwords

Now we need to add webdav users and create the passwords. For the first user use:

```
sudo htpasswd -c /var/www/passwd.dav phil
```

You will be prompted to create a password for this user.

Then for following users try this:

```
sudo htpasswd  /var/www/passwd.dav mick
```

Repeat for remaining users.

Next make sure your passwd.dav file is owned by the apache server.

```
sudo chown www-data:www-data /var/www/passwd.dav
```

Lsyncd

Next install lsyncd with

```
sudo apt install lsyncd
```

Create a log directory

```
sudo mkdir /var/log/lsyncd
touch /var/log/lsyncd/lsyncd.{log,status}
```

Then create a cofiguration file directory.

```
sudo mkdir /etc/lsyncd
```

```
sudo nano /etc/lsyncd/lsyncd.conf.lua
```

And enter this lua code

```

```

```
settings ={
        logfile = "/var/log/lsyncd/lsyncd.log",
        statusFile = "/var/log/lsyncd/lsyncd.status"
}

--
-- This syncs our webdav to dokuwiki for photo uploads.

sync  {
        default.rsync,
--      wait = 15,
        delete = false,
        source = "/www/www/webdav/mobileupload/",
        target = "/www/www/dokuwiki/data/media/mobileupload/",
        rsync={
        owner = true,
        group = true,
        perms = true
        }
}
```

Now lets start lsyncd as a service

```
sudo service lsyncd start
```

This performs a basic rsync command whenever the “/www/www/webdav/mobileupload/” changes. We use the command `delete = false` to force lsyncd to leave new files in our destination directory untouched -why? See below...

Fixing Capitalisation issues

For some reason dokuwiki does not like CAPITALS in media filenames, neither in the name nor exstention. So while `2971057.jpg` will appear in dokuwiki `IMG_2971057.jpg` or `2971057.JPG` will not.

This is fine when using upupu, but both davsync (Android) and whatever Phil is using use capitalisation in the file names. As an added bonus Phil's app creates directories, which means we need to look through them recursivley.

To fix this there are a couple of scripts running in start up, triggered by a cronjob. Lets set them up now. First go super with

```
   sudo su
```

Next lets make a directory for our scripts

```
mkdir -p /usr/scripts/photo_rename
```

The first scripts will set up some inotifywait watches on the mobileupload directories. I've used one per user so for Phil lets do:

```
nano inotify_phil.sh
```

Then we set up a while loop that uses inotifywait to detect a file action, then triggers another script called `exiftool_rename_phil.sh`

```
   #!/bin/bash
   DIRTARGET=/var/www/html/dokuwiki/data/media/mobileupload/phil
   while true #run indefinitely
   do''
   inotifywait -r -e close_write  $DIRTARGET  &&  /bin/bash
/usr/scripts/photo_rename/exiftool_rename_phil.sh
   done
```

Now lets make our script to rename the photo with the date and time using `exiftool`, then change the extension with `mv`

```
  #!/bin/bash
  DIRTARGET=/var/www/html/dokuwiki/data/media/mobileupload/phil
  exiftool '-FileName<CreateDate' -d %Y%m%d_%H%M%S%%-c.%%e $DIRTARGET/*.JPG
&&
  find $DIRTARGET -name *JPG -exec sh -c 'mv "$0" "${0%%.JPG}.jpg"; echo
"Moved $0 to ${0%%.JPG}.jpg"' {} \;
```

Finally set-up a crontab to run the script on reboot. So `'crontab -e` then add this line.

```
    @reboot /usr/scripts/photo_rename/inotify_phil.sh
```

Exporting Sections of the Wiki

When we need to take sections of the wiki offline to another location, follow this procedure;

Install New Wiki

Follow the steps here to make a new wiki.

Export Wiki Data

Now we

```
 sudo tar -pvczf dokuwiki.tar.gz  /www/www/dokuwiki  --
exclude=/www/www/dokuwiki/data/page
```

Next lets grab only the one last apocalypse data.

```
sudo find /www/www/dokuwiki/ -type d  -name one_* | tar -pvczf
dokuwiki_data.tar.gz  --include-from -

tar -tf dokuwiki_data.tar.gz
```

Recovery Procedure

This document covers recovery of content from your wiki, fixing broken plugins or templates, and total back-up.

Content Recovery

The easiest way to recover content is using the built in `old revisions` editor accessible from the right menu.

Then your can view all the changes made to that page.

And choose two changes to view the differences. Either side by side;

Or inline.

If you want to revert changes, select the version of the page you want (click the blue link) and it will open the page with a note at the top..

Edit the page, once again you will see note at the top. IF this is correct, then just save the edit and the changes will be made, creating a new version of the page.

Fixing broken Plugins or Templates

If the formating or function of the wiki goes haywire because of template or plugin updates or changes, you can use the server back-up to restore previous versions. This is useful because most plugin and template makers don't keep old versions around, relying on the wiki user to keep backups of their own wikis.

Re-installing Plugins and Templates

Restoring old versions of Plugins or Templates

We use rdiff-backup to daily backup the wiki (and the entire www directory). To understand how the back-up works and set it up see here

Restoring a folder from rdiff-backup

Lets say the we need to restore a working version of the revealjs plugin, after foolishly upgrading it and breaking all our existing slideshows[2)]. We know that we want to go back to how the wiki was about 20 days ago, when the slideshow was still working.

First we need to know where the plugin is installed. Our wiki is located in

```
/www/www/dokuwiki
```

All dokuwiki plugs live in:

```
 dokuwiki/lib/plugins
```

They all have there own folder, so the full path to revealjs would be

```
/www/www/dokuwiki/lib/plugins/revealjs/
```

Now we know where we want our restored files to go to, lets see where rdiff-backup keeps our backups. The file structure is mostly preserved by rdiff-backup so our revealjs folder would be in on the back drive.

```
/mnt/externalusb/www/
```

inside the rdiff-backup-data/ directory

```
/mnt/externalusb/www/rdiff-backup-data/
```

Then we are after the incremental backups...(this will let us go back a certain time)

```
/mnt/externalusb/www/rdiff-backup-data/increments/
```

Finally we find the revealjs directory.

```
/mnt/externalusb/www/rdiff-backup-
data/increments/dokuwiki/lib/plugins/revealjs/
```

The command we want is

```
rdiff-backup -r 20D
```

This will restore (-r) the file from 20 days ago.

So the full command to restore the revealjs directory contents from 20 days ago is..

```
rdiff-backup -r 20D  /mnt/externalusb/www/rdiff-backup-
data/increments/dokuwiki/lib/plugins/revealjs/
/www/www/dokuwiki/lib/plugins/revealjs
```

Full Restore of the Wiki From USB drive

If disaster strikes and the wiki goes down but the server is fine, you can restore it all from the USB external backup. Because rdiff-backup makes a *mirror* (an exact copy) we can just use the built in usual linux copy command with the '-a' for archive flag.

```
sudo cp -a /mnt/externalusb/www/dokuwiki /www/www/dokuwiki
```

Log in an check the functionality.

Full Restore of the Wiki and Server

If disaster double strikes, and the only thing left working is the USB external backup - then to recover the wiki will require a re-install of the server and dokuwiki. These instructions are adapted from

ccclabs.edgqld.gov.au. They will require a computer on the The Edge basement SLQ network, with an ICT request to set a permanent DHCP lease for a MAC address.

Sort The Hardware

You will need an i5, with 4gig RAM and 100Gig HD. Just about any recent donation from ICTS will suite, but try to use one that is (a) not part of an existing workshop line (b) not radically overspec.

Install Server OS

I've adapted this from the official dokuwiki instructions. You will need internet access. We will install the desktop version of (X)ubuntu for ease of use.

First up, grab workshop USB stick and make a fresh install of Xubuntu 14.04 (follow the steps in CCC) then install using these instructions.

Get (or request) a fixed DHCP lease

If you are in an organisation that has an ICT department - get this underway early, as it can take a while. The purpose is to make the server available over a network with a fixed address.

You will need to give your ICT a port number where you are plugging the server in, and the MAC address of the network card in the new computer. The port number is where you want to physically plug the server in. The MAC address can be found with

```
ifconfig
```

Once you know these details, let your ICT department know.

Install Server Software

Now make sure you are up to date.

```
sudo apt update && sudo apt upgrade
```

Dokuwiki need a web server and PHP. We use apache2 and PHP7.0.

```
sudo apt install apache2 libapache2-mod-php7.0
```

Enable Apache Rewrite module. This lets us use pretty URLs

```
  sudo a2enmod rewrite
```

Apache Set-up

Next we need to set-up Apache,lets edit our default configuration.

```
  sudo nano /etc/apache2/sites-avialable/000*.conf
```

and change

```
  DocumentRoot /var/www/html
```

to

```
 DocumentRoot /var/www/dokuwiki
```

Next we need to deny access to directories we want to keep secure in the main apache.conf file.

```
sudo nano /etc/apache2/apache2.conf
```

For directory /var/www/ replace

```
      AllowOverride None
```

with

```
      AllowOverride All
```

Restart Apache2 service.

```
  sudo service apache2 restart
```

PHP Set-up

First lets install the GD library, which we need for using the PDF export.

```
sudo apt install php7.0-gd
```

Then lets add sqlite support for bits and pieces.

```
sudo apt install php7.0-sqlite3
```

We'll need to make a few modifications to our PHP settings to allow larger uploads and posts to our

wiki.

```
 sudo nano /etc/php7.0/apache2/php.ini
```

Look for these two lines and change them to at least 15M (15 megabytes)

```
upload_max_filesize = 15M
post_max_size = 15M
```

Use CTRL+w to search in nano

We then need to make sure php has enough memory allocated to run the upload scripts.

```
memory_limit = 256M
```

Copy Wiki Content

Before we copy the wiki data back, lets install usbmount

```
sudo apt install usbmount
```

Mount the external USB backup

Once you've got the server up and running lets grab the USB backup - just plug it in and it should mount automagically - both on the desktop and under `/home/yourusername/media`

Finally - this is the same as above - just use the `cp` command to copy from the backup to the server.

```
 sudo cp -a /home/yourusername/media/externalusb/www/dokuwiki
/www/www/dokuwiki
```

Change Template Paths

The final jiggery pokery is to set the template path for your new IP address. We need this as the theme we use is has some customisation. Go to the admin panel;

Then open *Configuration*

and scroll down to the Bootstrap3 Template config

Now change this IP

```
http://wiki.edgeqld.org.au/lib/tpl/bootstrap3/assets/bootstrap/edge/bootstra
p.min.css
```

To your new IP discovered earlier.

1)
This is so we can do a systemback of the server, excluding the www data
2)
like I just did