

Network Engineering Assignment 1

Wiki & Dynamic Web Server Selection, Deployment &
Documentation



1 EXECUTIVE SUMMARY

This Assignment will explain how to setup a wiki and web server on ubuntu, looking into all phases from researching into what specific we server/wiki to use, deciding on what based on a set or criteria to installing and configuring them so that there is a running wiki behind a server.

I give permission to have this assignment shared for peer evaluation.

2 CONTENTS

1	Executive Summary.....	1
3	Web Server List:	2
3.1	Apache HTTP	2
3.2	Nginx.....	2
3.3	Caddy.....	3
4	Wiki List:.....	3
4.1	Wiki.js	3
4.2	Media Wiki	3
4.3	XWiki.....	4
5	Scripting Languages List:.....	4
5.1	Node.js (JavaScript)	4
5.2	PHP	4
5.3	Python	5
6	Selection.....	5
7	Installation	6
7.1	Install Software & Enable the services.....	6
	6
7.2	Configure Nginx.....	6
7.3	Installing Wiki.js.....	7
7.4	Configure Wiki.js	7
8	Modules & Expansion	8
9	Bibliography	8
10	Appendix	9

3 WEB SERVER LIST:

3.1 APACHE HTTP

Apache HTTP is the most dominate web server, powering over 50% of the worlds websites, and is often touted as the default best open source server to use, though it lacks a lot of features to start with, there are tons of open sourced modules for Apache HTTP that can make it do just about anything. [1]–[3]

3.1.1 Features:

- The dominate web server
- Offers more functionality through modules
- Provides logs of visitor information in plain text
- Virtual Hosting
- Password authentication
- IPv6 & HTTP/2 compatibility through modules, there is a large amount of modules that can be easily implemented

3.1.2 Dependencies:

- Apr-Util-1.6.1
- ANSI-C Compiler
- PCRE-8.43

3.2 NGINX

Nginx is another top contender for most popular web servers in use, the main attraction to Nginx is that it's asynchronous and that it can handle a very large amount of concurrent requests. It's also very fast & lightweight to run, often being used as a proxy server.[1]–[3]

3.2.1 Features:

- Second most popular web server
- Capable of handling large numbers of parallel requests without slowing down (very fast & lightweight)
- Asynchronous & event-driven
- IPv6, TLS/SSL encryption, SMTP/POP3, IMAP in-built support
- Often used as proxy servers (HTTP, email) & as a load balancer
- FLV and MKV streaming

3.2.2 Dependencies:

- PCRE
- Zlip
- OpenSSL

3.3 CADDY

Caddy is the newest server on this list, launching in 2015. It's designed to be an easy to use and configure open source web server for enterprises and personal use alike. It's major difference that is a big con, is that caddy is only free for personal use, enterprises need to pay a subscription fee to use Caddy, this does however come with consulting and support.[1]–[3]

3.3.1 Features:

- No Dependencies – static single binary executable
- Uses HTTPS by default (other security features that other servers have)
- Is open-source, however is only free for personal use, commercial use requires a payment plan
- http/2, IPv6, Websockets, FASTCGI protocols compatibility
- Dedicated support team
- Very easy to install and run

4 WIKI LIST:

4.1 WIKI.JS

Wiki.js is a very new wiki engine which is build from Node.js (JavaScript)

License: AGPLv3

4.1.1 Features:

- Works on any platform
- Markdown editing support backed by Git
- Runs on Node.js (JavaScript)
- Intelligently makes use of available resources
- Can use both 3rd party authentication i.e. Google, Github or Local authentication i.e. LDAP, Azure AD
- Has a Cloud module coming in 2020 for both self-hosting and managed infrastructure

The above information has been gathered from the Wiki.js website available at: <https://wiki.js.org/> and more information from source [4]

4.2 MEDIA WIKI

The poster child and most known wiki software, Wikipedia itself uses Media wiki, as it's suited/built for large and high-traffic websites.

License: GPL

4.2.1 Features:

- Written in PHP

- Easily scalable
- Uses mySQL database as storage
- Uses it's own wiki-text format instead of HTML/CSS to edit pages
- Optimized for large websites
- Has the largest selection of interface languages

4.2.2 Cons:

- Has less in-built functionality then the rest, though most extra functionality are easily installed through modules

The above information has been from source [4]

4.3 XWiki

A wiki engine written in Java, that is proclaimed to be a 'Second Generation' wiki, which by their websites means that it's structed and can be used to create collaborative web applications.

License: LGPL

4.3.1 Features:

- Version Control for the wiki
- Can Export Wiki pages to PDF, XML or HTML
- RESTful remote API to integrate into existing applications
- Easy to use as a File management system / Attachments to pages
- Has fairly powerful plugins/modules, i.e. in-site Image or SVG editing

The above information has been gathered from the XWiki Website:

<https://www.xwiki.org/xwiki/bin/view/Documentation/UserGuide/Features/SecondGenerationWiki/> and more information from source [4]

5 SCRIPTING LANGUAGES LIST:

5.1 NODE.JS (JAVASCRIPT)

Node.js is a framework that enables JavaScript to be used as a scripting language on the server side, as JavaScript is inherently a front-end language. Most sources online state that JavaScript is incredibly fast when it comes to handling client-server communication, such as chats/messaging.

Notable websites that use Node.js are Uber, PayPal and Netflix [5], [6]

5.2 PHP

PHP is the most common scripting language for servers world wide, many major websites such as Facebook and Wikipedia use it, and is best suited for sites that's main focus is

database manipulation. PHP is only a web language however so you may have to learn a new language, though PHP is a very simple scripting language to learn. [5], [6]

5.3 PYTHON

Python can be used as a scripting language by itself, or used as part of a specific server-side framework such as Django or Flask. Python is an all-round great and easy programming language to learn that can be used for a wide variety of tasks outside of sever-side scripting.

On the web however, python is used by a wide variety of popular sites such as Google & Youtube

While it's frameworks of Flask & Django run sites such as Reddit or Airbnb [5], [6]

6 SELECTION

Through reading up about them, I concluded that many of the open-source are very similar to each other and will mostly all suit the needs of this assignment. They're all fairly easy to install & configure, all major engines have a vast amount of open-sourced extensible modules.

From the list earlier I would choose each wiki respectively if I were making an application that they advertised being suited for, examples being;

I would choose media wiki, if the goal would be to create a fairly large open-contribution wiki-style website, in a vain much similar to Wikipedia, or one of the many other fan-made wiki's for specific TV shows.

I would choose XWiki if the goal would be to create a fully functioning website application.

In the end I have decided to go with setting up a wiki.js as it suits my selection criteria quite nicely, as I wanted to setup a specific type of wiki/server environment which was;

- Lightweight & capable of running the wiki service as a self-hosted web service

This means that the wiki can run inherently without a server in front of it, though this leads into why I chose to use Nginx as the web server, as Nginx can be easily setup to be a light weight reverse proxy server in front of the wiki.js service.

The reasoning behind this choice is the main purpose of the wiki, I want the wiki to be a more personal location for keeping specific code/programs, notes or any other work. All while running the services off of my personal computer, without causing much memory to be used up by either the wiki or server. While using a wiki/server ensures that I can easily search back and also invite others to view/contribute to the work.

I'm also a fan of the node.js environment as well as the markdown/git features of wiki.js.

Selected Web Server: Nginx

Selected Wiki: Wiki.js

6.1.1 Dependencies:

The main dependencies for Wiki.js are:

- Node.js & npm
- Git
- MongoDB (or another Database service)

Though during the installation phase, I used curl to download, curl was not installed by default on my machine.

7 INSTALLATION

7.1 INSTALL SOFTWARE & ENABLE THE SERVICES

The software/dependencies that will need to be installed right now are; curl, Nginx, git nodejs, npm and mongodb, below is the apt-install command to install all of them, terminal commands throughout this tutorial will be in similar text boxes.

```
sudo apt install nginx nodejs npm mongodb curl git
```

Start the mongodb service

```
sudo systemctl start mongodb.service  
sudo systemctl enable mongodb.service
```

Start the Nginx service

```
sudo systemctl enable nginx.service  
sudo systemctl start nginx.service
```

7.2 CONFIGURE NGINX

Use any text editor to edit the contents of Nginx wiki.js config file in the location below, I used nano as the text editor

```
sudo nano /etc/nginx/conf.d/wiki.js.conf
```

Below is my configuration file, setting Nginx up as a reverse proxy to listen on both port 80 (HTTP) & 443 (HTTPS), sever_name should be the real domain name, however I'll just be using local host

```
GNU nano 2.9.3 /etc/nginx/conf.d/wiki.js.conf
server {
    listen [::]:443 ssl http2;
    listen 443 ssl http2;
    listen [::]:80;
    listen 80;

    server_name localhost;

    charset utf-8;
    client_max_body_size 50M;

    location / {
        proxy_set_header Host $http_host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_pass http://127.0.0.1:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_next_upstream error timeout http_502 http_503 http_504;
    }
}
```

Restart the Nginx service afterwards

```
sudo systemctl reload nginx.service
```

7.3 INSTALLING WIKI.JS

Make a directory for wiki.js in the /var/www folder, and navigate into it

```
sudo mkdir -p /var/www/wiki.js
cd /var/www/wiki.js
```

Download and install wiki.js using curl

```
sudo curl https://wiki.js.org/install.sh | sudo bash
```

7.4 CONFIGURE WIKI.JS

To configure the wiki use the wiki.js wizard, start the wizard with

```
sudo node wiki configure
```

This start the wiki configuration wizard, the wizard is open on port 3000, so use any web browser and go to: localhost:3000

The configuration wizard is pretty straight forward to follow, however I've included my screenshots in the appendix.

8 MODULES & EXPANSION

I made the mistake of installing Wiki.js version 1.0.117, which is apart of the stable branch. Wiki.js currently has a near completed version 2.0 which is currently in beta right now. The only difference of installing the newer version 2.0 is instead of using curl to download with the command.

```
sudo curl https://wiki.js.org/install.sh | sudo bash
```

You would get the latest beta release from github at:

<https://github.com/Requarks/wiki/releases/download/2.0.0-beta.208/wiki-js.tar.gz>


I bring this up as version 1 does not have module compatibility while Wiki.js version 2 does have full module support, as well as better scalability support. So the first step would be to reinstall Wiki.js as version 2.

Modules are placed in the directory `/var/www/wiki.js/server/modules` – but currently due to how recent the addition of modules/module creation for wiki.js, the vast majority of modules are used to support easy integration of Wiki.js to existing infrastructure. With time many more modules will be created as it's open sourced.

9 BIBLIOGRAPHY

- [1] R. Muilwijk, "Top 5 open source web servers," *opensource.com*, 18-Aug-2016. .
- [2] Nucuta, "Best Open Source Web Servers for Linux," *linuxhint*, 20-Jan-2019. .
- [3] A. Thakur, "7 Open Source Web Servers for Small to Large Sites," *Geekflare*, 13-Mar-2019. .
- [4] D. Morelo, "Best Self-Hosted Wiki Software Products," *linuxhint*, 10-Jul-2018. .
- [5] W.-M. Thor, "5 top programming languages to learn server-side web development," *twm*, 11-Jan-2018. .
- [6] C. Wodehouse, "Server-Side Scripting: Back-End Web Development Technology," *upwork*. .

10 APPENDIX

 WIKI.JS

General

Site Title

NetEng Wiki

The site title will appear in the top left corner on every page and within the window title bar.

Host

https://localhost

The full URL to your wiki, without the trailing slash. E.g.: http://wiki.domain.com. Note that sub-folders are not supported.


Port

3000

The port on which Wiki.js will listen to. Usually port 80 if connecting directly, or a random port (e.g. 3000) if using a web server in front of it. Set \$(PORT) to use PORT environment variable.

Site UI Language

English

 WIKI.JS

Database

Wiki.js stores administrative data such as users, permissions and assets metadata in a MongoDB database. Article contents and uploads are not stored in the DB. Instead, they are stored on-disk and synced automatically with a remote git repository of your choice.

MongoDB Connection String

mongodb://localhost:27017/wiki

The connection string to your MongoDB server. Leave the default localhost value if MongoDB is installed on the same server. You can also specify an environment variable as the connection string, e.g. \$(MONGO_URI).

BACK

CONNECT

Paths

It is recommended to leave the default values.

Local Data Path

The path where cache (processed content, thumbnails, search index, etc.) will be stored on disk.

Local Repository Path

The path where the local git repository will be created, used to store content in markdown files and uploads.

[BACK](#)[CONTINUE](#)

Git Repository

Wiki.js stores article content and uploads locally on disk. All content is then regularly kept in sync with a remote git repository. This acts a backup protection and provides history / revert features. While optional, it is **HIGHLY** recommended to setup the remote git repository connection.

Repository URL

The full git repository URL to connect to.

Branch

The git branch to use when synchronizing changes.

Authentication

The authentication method used to connect to your remote Git repository.

Private Key location

The full path to the private key on disk.

☒ Verify SSL

Administrator Account

An administrator account will be created for local authentication. From this account, you can create or authorize more users.

Administrator Email

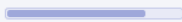
The email address of the administrator account

Password

At least 8 characters long.

Confirm Password

Verify your password again.

[BACK](#)[CONTINUE](#)