**Name of Title:** Learning Nginx

**Video Name:** 00\_04 Comparing Nginx to Apache

**Estimated Length:**

**Author Name:** Michael Jenkins

**Chapter\_Section\_Video:**

**Video Objective:**

At the end of this video the learner will be able to...

**Introductory Statement:**

Type your introductory statement here.

**Speaking Points:**

1. NGINX was created in response to the [C10K challenge](http://www.kegel.com/c10k.html): the ability to handle at least 10,000 simultaneous client connections on a single server.
2. NGINX does not support something like the Apache’s .htaccess file.
   1. Using .htaccess files, one can override system-wide settings on a per directory basis
   2. however, for optimal performance, these .htaccess directives should be included in the main configuration file(s) whenever possible.
3. Point\_3
4. Point\_4
5. Point\_5

**Script:**

SLIDE: 00\_04 Comparing Nginx to Apache

EDITOR: Fly in <https://httpd.apache.org/>

We can’t discuss nginx without also taking a moment to reflect on The Apache HTTP Server. Apache, as its more commonly known as, is an open source web server that’s been leading the market for more than 20 years.

In fact, the term LAMP, is an acronym for servers that run Linux-Apache-MySQL-and PHP, a very popular application stack for delivering dynamic web content.

So you you might be wondering, “why use nginx when Apache is such an integral part of delivering web content? What’s the difference between Apache and nginx?”

First, let’s take a look at a few of the ways they’re similar

SLIDE: Apache and nginx Similarities

Both apache and nginx are free and open source software. This means that anyone can use the software without paying for it and the source code is available for download along with binary versions of the application.

NEXT SLIDE:

This also means that the community of users behind both applications can review the code for bugs and vulnerabilities and suggest changes to be included in future versions.

NEXT SLIDE:

Both applications are compiled binaries but they can be extended by dynamically including modules at run time. This allows users to add additional functionality as needed.

NEXT SLIDE:

Both apache and nginx can be configured as proxy servers, allowing them to pass requests on to other applications and then return the content back to the requester.

NEXT SLIDE:

And finally, nginx and newer versions of Apache use event based processing. This means they can both efficiently handle a large number of simultaneous connections.

In past versions of Apache, each connection was handled by a worker processes that required additional system resources for each new connection that was made. This could easily lead to a server consuming all system resources under heavy load. By adding event based processing, newer versions of Apache are able to handle a high number of connections on par with nginx.

Now let’s look at a few differences.

SLIDE: Apache and nginx Differences

One of the most obvious differences is the configuration format used by apache and nginx. Apache uses a format that’s based on XML and can be verbose with open and closing tags for each section. Nginx on the other hand, uses a syntax very similar to the C programing language with directives and blocks defined by opening and closing brackets. Programmers working with nginx configurations might find this a bit more like programming compared to Apache which is might feel more like writing a document.

NEXT SLIDE:

Apache also provides for a distributed configuration through the use of .htaccess files. Each directory under the web server's root directory can contain a configuration file specifically for that directory. This has its benefits by allowing default configurations to be overridden, but it also can slow down processing because the htaccess file must be process before each request. Nginx uses a more centralized configuration with all configuration files loaded at the same time. Instead of configuring by directory, nginx instead uses location blocks that can map requests to locations inside the web server’s root directory, somewhere else on the file system, or to some other application for processing.

NEXT SLIDE:

Speaking of other applications, when it comes to processing dynamic content, Apache can minimize or eliminate the need for other applications. By including modules for languages like PHP, Python, and Perl directly in its binary, Apache can serve these types of dynamic content on its own. Nginx would need to rely on an external application for dynamic processing.

NEXT SLIDE:

But when it comes to serving static content, Nginx takes the lead over apache. In various tests, nginx is found to be a little more than twice as fast as apache when serving static content.

NEXT SLIDE:

Nginx also has built in capabilities for caching content which can lead to faster responses from the server. Apache can be configured as a cache with a module but the configuration may be cumbersome.

So with these similarities and differences to consider, why should we choose nginx?

SLIDE: Why nginx?

While apache maintains a strong lead in web server deployments, nginx has been quickly gaining in popularity since its public release in 2004. As more developers and engineers switch to nginx, there’s a noticeable decline in Apache’s usage. Learning and using nginx now, will help you stay up to date with what may soon become the market leading web server.

We’ve also discussed how nginx is efficient and consistent under heavy loads. If developers are trying to deploy applications in environments that have limited resources like small CPUs and limited RAM, deploying a web server that operates consistently can provide for more predictable performance.

And, along with efficiency and multiple capabilities, many find nginx easier to configure than apache. This can make nginx easier to learn for people that are just getting into application development and system administration.

Now that we have a better idea of why we should learn nginx, let’s get started.

**Conclusion:**

Type your conclusion statement here.

**Script and Media:**

Break the script up into parts and align it with any media (slides, web, CLI, etc.)

| **Part** | **Script** | **Media** |
| --- | --- | --- |
|  |  |  |

**Exercise Files:**

**Basement:**

<https://httpd.apache.org/>

<https://www.digitalocean.com/community/tutorials/apache-vs-nginx-practical-considerations>

Apache proceessing methods: <https://httpd.apache.org/docs/2.4/mpm.html>