Name-based Virtual Host Support

This document describes when and how to use name-based virtual hosts.

Topics

See also

- IP-based Virtual Host Support[1]
- An In-Depth Discussion of Virtual Host Matching[2]
- Dynamically configured mass virtual hosting[3]
- Virtual Host examples for common setups[4]
- ServerPath configuration example[5]

Name-based vs. IP-based Virtual Hosts

IP-based virtual hosts use the IP address of the connection to determine the correct virtual host to serve. Therefore you need to have a separate IP address for each host. With name-based virtual hosting, the server relies on the client to report the hostname as part of the HTTP headers. Using this technique, many different hosts can share the same IP address.

Name-based virtual hosting is usually simpler, since you need only configure your DNS server to map each hostname to the correct IP address and then configure the Apache HTTP Server to recognize the different hostnames. Name-based virtual hosting also eases the demand for scarce IP addresses. Therefore you should use name-based virtual hosting unless there is a specific reason to choose IP-based virtual hosting. Some reasons why you might consider using IP-based virtual hosting:

- Some ancient clients are not compatible with name-based virtual hosting. For name-based virtual hosting to work, the client must send the HTTP Host header. This is required by HTTP/1.1, and is implemented by all modern HTTP/1.0 browsers as an extension. If you need to support obsolete clients and still use name-based virtual hosting, a possible technique is discussed at the end of this document.
- Name-based virtual hosting cannot be used with SSL secure servers because of the nature of the SSL protocol.
- Some operating systems and network equipment implement bandwidth management techniques that cannot differentiate between hosts unless they are on separate IP addresses.

Using Name-based Virtual Hosts

Related Modules	Related Directives
core	DocumentRoot NameVirtualHost ServerAlias ServerName ServerPath VirtualHost

To use name-based virtual hosting, you must designate the IP address (and possibly port) on the server that will be accepting requests for the hosts. This is configured using the NameVirtualHost

directive. In the normal case where any and all IP addresses on the server should be used, you can use * as the argument to NameVirtualHost. Note that mentioning an IP address in a NameVirtualHost directive does not automatically make the server listen to that IP address. See Setting which addresses and ports Apache uses[6] for more details. In addition, any IP address specified here must be associated with a network interface on the server.

The next step is to create a <VirtualHost> block for each different host that you would like to serve. The argument to the <VirtualHost> directive should be the same as the argument to the NameVirtualHost directive (ie, an IP address, or * for all addresses). Inside each <VirtualHost> block, you will need at minimum a ServerName directive to designate which host is served and a DocumentRoot directive to show where in the filesystem the content for that host lives.

Main host goes away

If you are adding virtual hosts to an existing web server, you must also create a <VirtualHost> block for the existing host. The ServerName and DocumentRoot included in this virtual host should be the same as the global ServerName and DocumentRoot. List this virtual host first in the configuration file so that it will act as the default host.

For example, suppose that you are serving the domain www.domain.tld and you wish to add the virtual host www.otherdomain.tld, which points at the same IP address. Then you simply add the following to httpd.conf:

```
NameVirtualHost *

<VirtualHost *>
  ServerName www.domain.tld
  ServerAlias domain.tld *.domain.tld
  DocumentRoot /www/domain
</VirtualHost>

<VirtualHost *>
  ServerName www.otherdomain.tld
  DocumentRoot /www/otherdomain
</VirtualHost>
```

You can alternatively specify an explicit IP address in place of the * in both the NameVirtualHost and <VirtualHost> directives. For example, you might want to do this in order to run some name-based virtual hosts on one IP address, and either IP-based, or another set of name-based virtual hosts on another address.

Many servers want to be accessible by more than one name. This is possible with the ServerAlias directive, placed inside the <VirtualHost> section. For example in the first <VirtualHost> block above, the ServerAlias directive indicates that the listed names are other names which people can use to see that same web site:

```
ServerAlias domain.tld *.domain.tld
```

then requests for all hosts in the domain.tld domain will be served by the www.domain.tld virtual host. The wildcard characters * and ? can be used to match names. Of course, you can't just make up names and place them in ServerName or ServerAlias. You must first have your DNS server properly configured to map those names to an IP address associated with your server.

Finally, you can fine-tune the configuration of the virtual hosts by placing other directives inside the <VirtualHost> containers. Most directives can be placed in these containers and will then change

the configuration only of the relevant virtual host. To find out if a particular directive is allowed, check the Context[7] of the directive. Configuration directives set in the *main server context* (outside any `<VirtualHost>` container) will be used only if they are not overriden by the virtual host settings.

Now when a request arrives, the server will first check if it is using an IP address that matches the `NameVirtualHost`. If it is, then it will look at each `<VirtualHost>` section with a matching IP address and try to find one where the `ServerName` or `ServerAlias` matches the requested hostname. If it finds one, then it uses the configuration for that server. If no matching virtual host is found, then **the first listed virtual host** that matches the IP address will be used.

As a consequence, the first listed virtual host is the *default* virtual host. The `DocumentRoot` from the *main server* will **never** be used when an IP address matches the `NameVirtualHost` directive. If you would like to have a special configuration for requests that do not match any particular virtual host, simply put that configuration in a `<VirtualHost>` container and list it first in the configuration file.

Compatibility with Older Browsers

As mentioned earlier, there are some clients who do not send the required data for the name-based virtual hosts to work properly. These clients will always be sent the pages from the first virtual host listed for that IP address (the *primary* name-based virtual host).

How much older?

Please note that when we say older, we really do mean older. You are very unlikely to encounter one of these browsers in use today. All current versions of any browser send the `Host` header as required for name-based virtual hosts.

There is a possible workaround with the `ServerPath` directive, albeit a slightly cumbersome one:

Example configuration:

```
NameVirtualHost 111.22.33.44

<VirtualHost 111.22.33.44>
  ServerName www.domain.tld
  ServerPath /domain
  DocumentRoot /web/domain
</VirtualHost>
```

What does this mean? It means that a request for any URI beginning with "`/domain`" will be served from the virtual host `www.domain.tld` This means that the pages can be accessed as `http://www.domain.tld/domain/` for all clients, although clients sending a `Host:` header can also access it as `http://www.domain.tld/`.

In order to make this work, put a link on your primary virtual host's page to `http://www.domain.tld/domain/` Then, in the virtual host's pages, be sure to use either purely relative links (*e.g.*, "`file.html`" or "`../icons/image.gif`" or links containing the prefacing `/domain/` (*e.g.*, "`http://www.domain.tld/domain/misc/file.html`" or "`/domain/misc/file.html`").

This requires a bit of discipline, but adherence to these guidelines will, for the most part, ensure that your pages will work with all browsers, new and old.

URI References

[1] http://httpd.apache.org/docs-2.1/vhosts/ip-based.html
[2] http://httpd.apache.org/docs-2.1/vhosts/details.html
[3] http://httpd.apache.org/docs-2.1/vhosts/mass.html
[4] http://httpd.apache.org/docs-2.1/vhosts/examples.html
[5] http://httpd.apache.org/docs-2.1/vhosts/examples.html#serverpath
[6] http://httpd.apache.org/docs-2.1/bind.html
[7] http://httpd.apache.org/docs-2.1/mod/directive-dict.html#Context