Roller 5 – Multi-domain Case Study

A Ten Step Linux Installation

Author: Christopher Dodunski   Version: 5.0   Date: 16 November 2010

Congratulations on discovering Apache Roller! Perhaps, like me, you are a relative newcomer to Roller weblogging. Or maybe you are upgrading to Roller 5 from an earlier version. Either way, the new multi-domain functionality of version 5 is a significant advance, and further promotes Roller as the Weblog Server of choice for the enterprise environment. Naturally, hosting multiple domains from the one instance carries substantial benefits in terms of resource usage. Furthermore, as Roller is developed in Java – a professional object-oriented programming language – typically performance will be superior to comparable software developed in PHP or another scripting language.

Our own requirements were for a weblog server capable of handling a large number of internet 'blog' pages, generally each with its own domain. Optomus hosts websites for a number of clients. These clients expect to have their own blog sub-domain – and rightly so. Supporting URLs of the form http://blog.their-domain.com rather than http://blog.optomus.com/their-name was - and remains - a clear selling point. Another key requirement was that adding new domains should be a quick and simple affair; a fresh weblog installation for each domain simply wasn't an option. Happily, Apache Roller has met these demands, and more.

Roller 5 weblogs presently hosted by optomus.com

- http://blog.optomus.com
- http://blog.bread.co.nz
- http://blog.capereinga.co.nz
- http://blog.doubtlessbay.co
- http://blog.christopher.net.nz
- http://blog.isaac.net.nz

The purpose of this case study is to describe in detail a Roller 5 installation in a typical production environment, whereby Apache HTTP Server operates as a proxy to Apache Tomcat. A few challenges were encountered along the way, and I explain how these were resolved. Occasionally I've deviated from the standard installation instructions - reasons are given. Please note that the official installation guide contains important information not repeated in this case study. Consequently, this document should be viewed as a supplement only, and used in conjunction with the Roller 5 installation guide.

Before launching into the ten steps to Roller 5 multi-domaining, it is worth noting our particular webserver architecture. Despite the fact this document describes a Roller installation on a Debian Linux platform, the ten steps should broadly apply whatever your operating system. Importantly, though, Apache Tomcat, in our case, sits behind an Apache HTTP (proxy) Server. This is a very common scenario for a production environment, and possibly one with which you are familiar.

Overview of the optomus.com webserver architecture

- Debian Linux OS – version 2.6.30.5-xenU
- Apache HTTP Server – version 2.2.9
- Apache Tomcat Servlet Container – version 6.0.29

- MySQL Database – version 5.0.51
- Apache Roller - version 5.0.0-RC2

Ten Step Multi-domain Installation

(1) Create a user named 'roller'

Command...
```
adduser roller
```

Why create a user named roller? The reason is that the roller home directory will contain the roller application. Also contained here is its associated properties file for startup, and roller user files, such as image and audio files. In an enterprise environment it makes good sense to do it this way. For a start, it's good housekeeping. Many of us have worked with applications and associated resources scattered across several harddrive locations. It's messy, and maintenance becomes an issue. Also, there are security implications. Were we to assign a roller administrator, we could jail him within /home/roller, and similarly grant restricted access to the roller database.

(2) Make the Roller home directory structure

Command...
```
cd /home/roller
mkdir application
mkdir data
mkdir data/search-index
mkdir data/mediafiles
```

These commands create our home directory structure. Depending on the quantity of images and other media contained in hosted weblogs, the mediafiles directory can grow rather large.

(3) Download Apache Roller 5

Command...
```
wget http://people.apache.org/~snoopdave/apache-roller-5.0/roller-weblogger-5.0.0-RC2-binary.tar.gz
tar xvzf roller-weblogger-5.0.0-RC2-binary.tar.gz
```

At the time of writing, Apache Roller release candidate (RC) 2 is unofficially available for download. Release candidate 3 is in the wings, so please check the Apache Roller website - Dave's blog too - prior to downloading. Modify the above commands accordingly. The second command above unpackages the archive. Should you choose to download the zip file, use the 'unzip' command instead (tar.gz is similar in purpose to zip, although the latter is more commonly used on Windows).

(4) Package Roller into a WAR file

Command...

```
cd roller-weblogger-5.0.0-RC2/webapp/roller/
jar cvf /home/roller/application/ROOT.war *
```

Here is where the fun begins! Deploying by WAR keeps it nice and clean, and the above commands will generate ROOT.war, placing it exactly where we want it - in /home/roller/application. Why have we named it ROOT.war rather than, say, roller.war? The ROOT (upper case important) application has special meaning to Tomcat. It is the application invoked by Tomcat when the base URL is called, without any path name. Deploying Roller with ROOT context may not be required in the future. At the time of writing, however, the multi-domain feature of Roller 5 has only been tested fully in ROOT context. Therefore, for now at least, it is recommended that you likewise deploy Roller 5 with ROOT context. Perhaps you have read that Tomcat allows only one ROOT application, and that the existing default ROOT application found at tomcat/webapps/ROOT will need to be removed or renamed. Actually, this isn't necessary. Once we move onto configuring Tomcat, you will see how we have gotten around this general rule.

(5) Define the Roller configuration file

Command...

```
cd /home/roller
jed roller-custom.properties
ln -s /home/roller/roller-custom.properties /usr/local/tomcat/lib/roller-custom.properties
```

This is the custom properties file Roller looks to at startup, principally for establishing database and mail server connectivity. The example below contains some additional directives that are of special interest to us. The two highlighted relate to file storage, specifying where Roller is to store search index data and uploaded media files. By specifying these locations in our properties file we are, in fact, overriding Roller's default behaviour which is to store these files elsewhere. Recall our original intention of placing all Roller resources in the one place? Indeed, even the properties file will be located here. The above "ln" linux command creates a symbolic link within Tomcat's lib directory. For all intents and purposes, the linked to properties file is now within the Tomcat classpath.

In case this is the first time you have encountered 'jed', it just happens to be my preferred console text editor. If you use an alternative text editor, certainly stick with the one you are comfortable with. Otherwise, jed may be installed with the command "apt-get install jed" - if you don't already have it. Tip: to access the jed menu bar hit F10 on your keyboard.

The next of our directive types is 'weblog.absoluteurl'. This is the means by which multi-domain functionality is achieved in Roller 5. These directives merely associate URLs with Roller handle names. If you are a Roller 4 user, you are likely familiar already with the concept of weblog handles. It is the handle that uniquely identifies a Roller weblog (blog page). Essentially, we are directing Roller to serve up a specific weblog based on the URL contained in the incoming HTTP request header. An unfortunate artifact of earlier (mono-domain) versions, however, is that Roller still parses the URL with the intention of extracting the handle name. If no handle is present, Roller will simply serve up the default weblog. Generally this would dictate that URLs be of the form 'http://example.com/handle'. But, as you shall soon see, we sidestep this minor inconvenience with the help of a little URL rewriting.

File: /home/roller/roller-custom.properties

```
installation.type=manual
database.configurationType=jdbc
database.jdbc.driverClass=com.mysql.jdbc.Driver
database.jdbc.connectionURL=jdbc:mysql://localhost:3306/roller
database.jdbc.username=roller
database.jdbc.password=[password]
mail.configurationType=properties
mail.hostname=[hostname]
mail.username=[username]
mail.password=[password]
newuser.categories=General
weblog.export.enabled=true
search.index.dir=/home/roller/data/search-index
mediafiles.storage.dir=/home/roller/data/mediafiles
weblog.absoluteurl.optomus=http://blog.optomus.com
weblog.absoluteurl.bread=http://blog.bread.co.nz
weblog.absoluteurl.capereinga=http://blog.capereinga.co.nz
weblog.absoluteurl.doubtlessbay=http://blog.doubtlessbay.co
weblog.absoluteurl.christopher=http://blog.christopher.net.nz
weblog.absoluteurl.isaac=http://blog.isaac.net.nz
```

(6) Grant Tomcat write access to the roller home directory

Command...

```
chown -R tomcat.nogroup /home/roller
```

This step is all about giving Tomcat write access to the Roller home directory. Without it, Tomcat would not be able to unpack ROOT.war or save uploaded media files. Of course, there are a number of ways to grant other users write access. You could simply run "chmod -R a+w /home/roller". But this could hardly be considered a secure solution in a production environment. Another safe option would be to make tomcat and roller members of the same group, and then run the command "chmod -R g+w /home/roller".

(7) Add Apache HTTP Server virtual host(s)

Command...

```
httpd -M    or...
apache2ctl -M
```

The above command queries Apache HTTP Server for the names of loaded modules. Look for mod_proxy and mod_rewrite amongst the modules listed (names may differ slightly). If they aren't present, you will need to install them. The mod_proxy module allows us to run a reverse proxy arrangement, whereby Apache HTTP Server may forward requests on to Apache Tomcat. The mod_rewrite module allows us to tweak, if necessary, URLs contained in the inbound HTTP request header, thereby eliminating the need to include handle names in URLs (refer step 5, paragraph 3).

For an insight into how these two modules work, and virtual hosting in general, let's look at a few virtual host definition files. The first directive we'll consider is 'ServerName'. Apache HTTP Server, upon receiving a HTTP request on port 80, will look to the virtual host descriptor with a ServerName value matching the URL contained in the request header. Apache HTTP Server then steps through the various directives contained within that descriptor.

Referring to the examples below, the next group of directives instruct the mod_rewrite module. Should a HTTP request arrive without a handle name in the URL, then we would want to remedy this. The 'RewriteRule' directive accomplishes this with a minimum of fuss. Were we not to append the handle name, Roller would simply serve up the default weblog - probably not what we want to occur. If you would like to observe this URL rewriting in action, click on any of the weblog domains listed above, and watch what happens in the address bar of your browser.

The final group of directives instruct the mod_proxy module, by means of 'ProxyPass' and 'ProxyPassReverse'. These merely forward the incoming request on to local port 8009, which Apache Tomcat happens to be listening on. Tomcat invokes the Roller web application, which in turn serves up our weblog. All too beautiful for words!

File: /etc/apache2/sites-enabled/blog.bread.co.nz.conf

```
<VirtualHost *:80>
DocumentRoot "/home/bread/htdocs"
ServerName blog.bread.co.nz
<Directory "/home/bread/htdocs">
allow from all
Options +Indexes
</Directory>

RewriteEngine on
RewriteRule ^/$ /bread/ [R]

ProxyPass / ajp://127.0.0.1:8009/
ProxyPassReverse / ajp://127.0.0.1:8009/
</VirtualHost>
```

File: /etc/apache2/sites-enabled/blog.capereinga.co.nz.conf

```
<VirtualHost *:80>
DocumentRoot "/home/capereinga/htdocs"
ServerName blog.capereinga.co.nz
<Directory "/home/capereinga/htdocs">
allow from all
Options +Indexes
```

```
</Directory>

RewriteEngine on
RewriteRule ^/$ /capereinga/ [R]

ProxyPass / ajp://127.0.0.1:8009/
ProxyPassReverse / ajp://127.0.0.1:8009/
</VirtualHost>
```

File: /etc/apache2/sites-enabled/blog.doubtlessbay.co.conf

```
<VirtualHost *:80>
DocumentRoot "/home/doubtlessbay/htdocs"
ServerName blog.doubtlessbay.co
<Directory "/home/doubtlessbay/htdocs">
allow from all
Options +Indexes
</Directory>

RewriteEngine on
RewriteRule ^/$ /doubtlessbay/ [R]

ProxyPass / ajp://127.0.0.1:8009/
ProxyPassReverse / ajp://127.0.0.1:8009/
</VirtualHost>
```

Command...

```
/etc/init.d/apache2 reload    or...
/etc/init.d/apache2 restart
```

Perhaps you are wondering why the above files contain 'DocumentRoot' directives, although no content from these directories are actually served up. These directories contain the base website (e.g. www.bread.co.nz). If we ever wished to disable Roller for some reason, we could comment out the ProxyPass directives and have Apache HTTP Server, for a period, serve up the base website instead. It's nice to have this option available should we need it.

(8) Add a dedicated Roller Apache Tomcat host

Command...

```
jed /usr/local/tomcat/conf/server.xml
/etc/init.d/tomcat restart
```

Recall what was stated earlier regarding Tomcat ROOT applications, namely that there can be only one? Well, more accurately, there may be **one per host**. And with Apache Tomcat, extending the number of hosts is an easy task. The below excerpt from server.xml demonstrates how this is done. Note, too, that by giving our new host an application base of home/roller/application, we have realised our goal of keeping all Roller resources in the one place. The 'Alias' tags ensure that Tomcat correctly invokes the Roller application whenever it receives a request for any of the 'blog' sub-domains listed. Having added a dedicated Roller host to server.xml, remember to restart Tomcat for this change to take effect.

File: /usr/local/tomcat/conf/server.xml (excerpt only)

```
<Host name="localhost"  appBase="webapps"
        unpackWARs="true" autoDeploy="true"
        xmlValidation="false" xmlNamespaceAware="false">
</Host>

<Host name="blog.optomus.com" appBase="/home/roller/application"
        unpackWARs="true" autoDeploy="true"
        xmlValidation="false" xmlNamespaceAware="false">

        <Alias>blog.bread.co.nz</Alias>
        <Alias>blog.capereinga.co.nz</Alias>
        <Alias>blog.doubtlessbay.co</Alias>
        <Alias>blog.christopher.net.nz</Alias>
        <Alias>blog.isaac.net.nz</Alias>
</Host>
```

(9) Create the Roller database and user

Command...

```
mysql -u root -p
mysql> CREATE DATABASE roller DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_general_ci;
mysql> CREATE USER 'roller'@'localhost' IDENTIFIED BY 'password';
mysql> GRANT ALL ON roller.* TO 'roller'@'localhost';
mysql> quit
```

Naturally, the above commands are only to be executed if you are starting with a fresh install of Roller. If you are upgrading from an earlier version you may skip this step. In this case, Roller will recognise that your database already exists, and merely make the required changes for version 5.

(10) Finalise installation via Roller interface, as per installation guide

The time has come to test out your installation. Direct your browser to the URL you recorded as the host name in server.xml. All going to plan, you should receive a response screen from Roller 5 confirming that the database connection attempt was successful, and offering to create the database tables - should they not exist. Finally, you are invited to setup an admin user account prior to establishing your first weblog. Welcome to Roller weblogging!

Struck a problem? A chain of processes are responsible for serving up your Roller weblog, but fortunately it is usually evident where the break is, when there is one. Both Apache HTTP Server and Apache Tomcat provide detailed error reports whenever either encounters a problem. In most cases, revisiting the relevant step(s) above will reveal the cause - perhaps an omission or oversight. If you do get stuck, however, feel free to drop a line to the Roller users mailing list for assistance. Please give a clear account of the nature of your problem, and include any error messages reported. I or someone else would be pleased to help.