GNU/Linux Administration - Feature #265

Replicated Memcached Instances For High Availability

12/22/2013 05:02 AM - Daniel Curtis

Status:	Closed	**Start date:**	12/21/2013
Priority:	Normal	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:	Caching Server	**Estimated time:**	1.00 hour
Target version:		**Spent time:**	5.00 hours

Description

I have decided to add a memcache layer to the web services provided by the ALT VPS Infrastructure. To accomplish this, I have created an instance of memcached (more correctly repcached) on each node, possibly in a circular replication pattern if supported, then have the memcached instances load-balanced. This will have the benefit of increased performance and availability, with minimal restructuring of current web applications and the potential to utilize memcached for increased MySQL performance.

Installing repcache

First you need to visit http://repcached.lab.klab.org/ and download the lastest version (latest at time of writing: 2.2-1.2.8). After downloading the tar file, we will need to install some dependencies:

```
apt-get install memcached libevent-dev g++ make
```

From here we can continue the installation:

```
tar xvf memcached-1.2.8-repcached-2.2.tar
cd memcached-1.2.8-repcached-2.2/
./configure --enable-replication
make
make install
```

Configuring repcache

At this point you have two installations of memcached. Default memcached that came from apt packages, which is installed in /usr/bin/memcached and repcached, that installed itself in /usr/local/bin/memcached, leaving the original memcached intact.

Now that we have both versions installed, we can copy memcached's default settings and init script and modify them to use repcached. This way you can quickly switch between versions. I would even recommend using default ports (just remember to firewall them!) Arguments are saved in /etc/memcached.conf, so we will create /**etc/repcached.conf**

```
cp /etc/memcached.conf /etc/repcached.conf
vi /etc/repcached.conf
```

DAEMON_ARGS="-m 128 -p 11211 -u root -P /var/run/repcached.pid -d -x 172.16.100.2"

Note that the only differences with memcached.conf is the name (repcached) and two extra arguments, make sure to at least define the -x arugment:

1. **-x** for the server IP of the master memcached service to replicate
2. **-X** for replication port

Memcached has an enable/disable config in /etc/default so you can quickly switch between daemons or disable them.

Copy the system defaults

```
cp /etc/default/memcached /etc/default/repcached
vi /etc/default/repcached
```

Change the line to:

ENABLE_REPCACHED=yes

Then edit /etc/default/memcached"

```
vi /etc/default/memcached
```

and disable it, by changing the line to

ENABLE_MEMCACHED=no.

Create an init script

```
vi /etc/init.d/repcached
```

Then add:

```
#!/usr/bin/perl -w

# start-repcached
# 2011 - Jean Caffou <jean@briskula.si>
# This script handles the parsing of the /etc/repcached.conf file
# and was originally created for the Debian distribution.
# Anyone may use this little script under the same terms as
# memcached itself.

use strict;

if($> != 0 and $< != 0)
{
    print STDERR "Only root wants to run start-repcached.\n";
    exit;
}

my $params; my $etchandle; my $etcfile = "/etc/repcached.conf";

# This script assumes that repcached is located at /usr/local/bin/memcached, and
# that the pidfile is writable at /var/run/repcached.pid

my $memcached = "/usr/local/bin/memcached";
my $pidfile = "/var/run/repcached.pid";

# If we don't get a valid logfile parameter in the /etc/repcached.conf file,
# we'll just throw away all of our in-daemon output.
my $fd_reopened = "/dev/null";

sub handle_logfile
{
    my ($logfile) = @_;
    $fd_reopened = $logfile;
}

sub reopen_logfile
{
    my ($logfile) = @_;

    open *STDERR, ">>$logfile";
    open *STDOUT, ">>$logfile";
    open *STDIN, ">>/dev/null";
    $fd_reopened = $logfile;
}
```

```
# This is set up in place here to support other non -[a-z] directives

my $conf_directives = {
    "logfile" => \&handle_logfile,
};

if(open $etchandle, $etcfile)
{
    foreach my $line (<$etchandle>)
    {
        $line ||= "";
        $line =~ s/\#.*//g;
        $line =~ s/\s+$//g;
        $line =~ s/^\s+//g;
        next unless $line;
        next if $line =~ /^\-[dh]/;

        if($line =~ /^[^\-]/)
        {
            my ($directive, $arg) = $line =~ /^(.*?)\s+(.*)/;
            $conf_directives->{$directive}->($arg);
            next;
        }

        push @$params, $line;
    }

}else{
    $params = [];
}

push @$params, "-u root" unless(grep "-u", @$params);
$params = join " ", @$params;

if(-e $pidfile)
{
    open PIDHANDLE, "$pidfile";
    my $localpid = <PIDHANDLE>;
    close PIDHANDLE;

    chomp $localpid;
    if(-d "/proc/$localpid")
    {
        print STDERR "repcached is already running.\n";
        exit;
    }else{
        `rm -f $localpid`;
    }

}

my $pid = fork();

if($pid == 0)
{
    reopen_logfile($fd_reopened);
    exec "$memcached $params";
    exit(0);

}else{
    if(open PIDHANDLE,">$pidfile")
    {
        print PIDHANDLE $pid;
        close PIDHANDLE;
    }else{
```

```
        print STDERR "Can't write pidfile to $pidfile.\n";
    }
}
```

Setting up repcached to start at boot

We need to be sure that /etc/init.d/repcached is executable. If you copied it from memcached, everything should be OK, but if init's not recognising the repcached service, you need to

```
chmod +x /etc/init.d/repcached
```

After you've run update-rc.d command in the terminal it will create shortcuts in rc?.d files which are read at boot:

```
update-rc.d repcached defaults
```

You have successfully configured repcached as a service and to start on boot.

To start/stop repcached use:

```
service repcached start
service repcached stop
```

Try to run repcached by hand at first with the configuration you provided in /etc/repcached.conf.
In my example it's this:

```
/usr/local/bin/memcached -m 64 -p 11211 -u memcache -X 11212 -x 10.11.22.33
```

After installing repcached on another machine I've found out that the default user for memcached is **nobody**, not **memcache**, so please always check the differences from the default memcache config with the repcached config you've modified or copied from here.

Testing

Before we move on we will test that the 2 nodes are replicating.

- **server1**:

  ```
  telnet 127.0.0.1 11211
  ```

Then type in:

```
set foo 0 0 3
bar
```

You should see the word STORED appear under it. Good:

```
quit
```

and it will return to the console. Go to your other server and type in the following.

- **server2**:

  ```
  telnet 127.0.0.1 11211
  ```

Then type in

```
get foo
```

and it should return the value you entered on the first server. If it has then the replication is working.

Troubleshooting

I encountered an error while compiling repcache:

> memcached.c: In function ‘add_iov’:
> memcached.c:697:30: error: ‘IOV_MAX’ undeclared (first use in this function)
> memcached.c:697:30: note: each undeclared identifier is reported only once for each function it appears in

Luckily enough, someone else had this problem and posted a fix on the developers forum, here

```
In memcached.c code, we can see that IOV_MAX is defined if the OS is FreeBSD or iOS, so I decided
to just define it anyway.
```

Here is my diff of the code that made it compile, in case you need it.

```
57,59c57
< #if defined(__FreeBSD__) || defined(__APPLE__)
< # define IOV_MAX 1024
< #endif
---
> #define IOV_MAX 1024
```

So I just needed to remove the IF condition and define the IOV_MAX.

~~I also had a problem after installing the modified memcached server where the init script would display an error, preventing the server from starting:~~

```
/usr/local/bin/memcached: invalid option -- '-'
Illegal argument "?"
```

I had forgotten to add the --enable-replication argument while configuring the software.

Resources

http://dev.kafol.net/2011/03/configuring-repcached-service-on.html
http://dev.kafol.net/2012/03/repcached-does-not-compile.html
http://www.howtoforge.com/how-to-install-repcached-memcached-replication-for-high-availability-over-2-nodes-on-ubuntu-11.04

History

#1 - 12/26/2013 05:59 PM - Daniel Curtis

- Description updated

After many hours of troublshooting, I found that the init script provided causes a crash when starting the repcached service, I had thought that the error was due to an missing compilation argument:

```
/usr/local/bin/memcached: invalid option -- '-'
Illegal argument "?"
```

I finally settled on a hacked init script with the server arguments hard coded into the script itself:

```
#! /bin/sh
### BEGIN INIT INFO
# Provides:             repcached
# Required-Start:       $syslog
# Required-Stop:        $syslog
# Should-Start:         $local_fs
# Should-Stop:          $local_fs
```

```
# Default-Start:        2 3 4 5
# Default-Stop:         0 1 6
# Short-Description:    repcached - Memory caching daemon replicated
# Description:          repcached - Memory caching daemon replicated
### END INIT INFO
# Author: Marcus Spiegel <marcus.spiegel@gmail.com>
#
# Please remove the "Author" lines above and replace them
# with your own name if you copy and modify this script.
# Do NOT "set -e"
# PATH should only include /usr/* if it runs after the mountnfs.sh script
PATH=/sbin:/usr/sbin:/bin:/usr/bin
DESC="repcached"
NAME=memcached
DAEMON=/usr/local/bin/$NAME
DAEMON_ARGS="-m 128 -p 11211 -u root -P /var/run/repcached.pid -d -x 172.16.100.2"
PIDFILE=/var/run/repcached.pid
SCRIPTNAME=/etc/init.d/$DESC
VERBOSE="yes"
# Exit if the package is not installed
[ -x "$DAEMON" ] || exit 0
# Read configuration variable file if it is present
[ -r /etc/default/$DESC ] && . /etc/default/$DESC
# Load the VERBOSE setting and other rcS variables
. /lib/init/vars.sh
# Define LSB log_* functions.
# Depend on lsb-base (>= 3.0-6) to ensure that this file is present.
. /lib/lsb/init-functions
#
# Function that starts the daemon/service
#
do_start()
{
    # Return
    #   0 if daemon has been started
    #   1 if daemon was already running
    #   2 if daemon could not be started
    start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $DAEMON --test > /dev/null \
        || return 1
    start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $DAEMON -- \
        $DAEMON_ARGS \
        || return 2
    # Add code here, if necessary, that waits for the process to be ready
    # to handle requests from services started subsequently which depend
    # on this one.  As a last resort, sleep for some time.
}
#
# Function that stops the daemon/service
#
do_stop()
{
    # Return
    #   0 if daemon has been stopped
    #   1 if daemon was already stopped
    #   2 if daemon could not be stopped
    #   other if a failure occurred
    start-stop-daemon --stop --quiet --retry=TERM/30/KILL/5 --pidfile $PIDFILE --name $NAME
    RETVAL="$?"
    [ "$RETVAL" = 2 ] && return 2
    # Wait for children to finish too if this is a daemon that forks
    # and if the daemon is only ever run from this initscript.
    # If the above conditions are not satisfied then add some other code
    # that waits for the process to drop all resources that could be
    # needed by services started subsequently.  A last resort is to
    # sleep for some time.
    start-stop-daemon --stop --quiet --oknodo --retry=0/30/KILL/5 --exec $DAEMON
    [ "$?" = 2 ] && return 2
    # Many daemons don't delete their pidfiles when they exit.
    rm -f $PIDFILE
    return "$RETVAL"
}
#
# Function that sends a SIGHUP to the daemon/service
#
do_reload() {
```

```
    #
    # If the daemon can reload its configuration without
    # restarting (for example, when it is sent a SIGHUP),
    # then implement that here.
    #
    start-stop-daemon --stop --signal 1 --quiet --pidfile $PIDFILE --name $NAME
    return 0
}
case "$1" in
  start)
    [ "$VERBOSE" != no ] && log_daemon_msg "Starting $DESC" "$NAME"
    do_start
    case "$?" in
        0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
        2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
    esac
    ;;
  stop)
    [ "$VERBOSE" != no ] && log_daemon_msg "Stopping $DESC" "$NAME"
    do_stop
    case "$?" in
        0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
        2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
    esac
    ;;
  #reload|force-reload)
    #
    # If do_reload() is not implemented then leave this commented out
    # and leave 'force-reload' as an alias for 'restart'.
    #
    #log_daemon_msg "Reloading $DESC" "$NAME"
    #do_reload
    #log_end_msg $?
    #;;
  restart|force-reload)
    #
    # If the "reload" option is implemented then remove the
    # 'force-reload' alias
    #
    log_daemon_msg "Restarting $DESC" "$NAME"
    do_stop
    case "$?" in
      0|1)
        do_start
        case "$?" in
            0) log_end_msg 0 ;;
            1) log_end_msg 1 ;; # Old process is still running
            *) log_end_msg 1 ;; # Failed to start
        esac
        ;;
      *)
          # Failed to stop
        log_end_msg 1
        ;;
    esac
    ;;
  *)
    #echo "Usage: $SCRIPTNAME {start|stop|restart|reload|force-reload}" >&2
    echo "Usage: $SCRIPTNAME {start|stop|restart|force-reload}" >&2
    exit 3
    ;;
esac
:
```

#2 - 12/27/2013 01:02 AM - Daniel Curtis

- Description updated

#3 - 12/27/2013 08:28 AM - Daniel Curtis

- Project changed from 21 to 59

#4 - 12/27/2013 03:57 PM - Daniel Curtis

- File memcached-1.2.8-repcached-2.2.1.tar.gz added

#5 - 12/27/2013 04:27 PM - Daniel Curtis

- Description updated

#6 - 02/15/2015 09:24 PM - Daniel Curtis

- Project changed from 59 to GNU/Linux Administration

- Category set to Caching Server

Files

memcached-1.2.8-repcached-2.2.1.tar.gz	223 KB	12/27/2013	Daniel Curtis