The doRedis Linux Service

The doRedis package includes a script that configures doRedis R workers to run as a Linux system service. This vignette outlines installation and configuration of the service.

The service installation script should work on at least RHEL/CentOS and Ubuntu/Debian Linux systems, and probably most Linux systems (with or without systemd). We outline installation instructions for Ubuntu/Debian Linux flavors. Installation and configuration proceeds similarly for RHEL, replacing the apt-get lines with comparable yum ones.

Install R and Redis

The bash shell is required by the doRedis service.

```
sudo apt-get build-dep r-base libxml2-dev libssl-dev \
     libcurl4-openssl-dev unzip  redis-server
# (substitute with current version of R as required)
wget --quiet -O - \
    https://cran.r-project.org/src/base/R-3/R-3.6.2.tar.gz | \
tar -zxvf -

cd R-3.6.2
./configure --enable-R-shlib --enable-memory-profiling
make -j 2     # use an appropriate number of CPUs here
sudo make install
```

Configure Redis to listen on all interfaces

It is *important* to combine this step with reasonable security firewall rules. You need access on the Redis port 6379 between nodes in your compute cluster, *but not to nodes on the internet*. The following command will configure your Redis server to listen on all interfaces:

```
sudo sed -i.bak  "s/^bind 127.0.0.1/bind 0.0.0.0/"  \
     /etc/redis/redis.conf && sudo /etc/init.d/redis-server restart
```

Install R packages

Finally, we need to install the foreach, redux, and doRedis packages, plus whatever other R packages you might want to use. The following script installs redux and doRedis from their source repository on GitHub using the devtools package.

```
sudo R --slave -e "install.packages(c('devtools', 'foreach', 'redux'),
                  repos='https://cran.r-project.org')"
sudo R --slave -e "devtools::install_github('bwlewis/doRedis')"
```

Install the doRedis service

The doRedis package includes an example script for setting up R workers as a service. The script works on generic Linux systems including RHEL/CentOS and Ubuntu.

```
script=$(R --slave -e "cat(
         system.file('scripts/redis-worker-installer.sh',
                     package='doRedis'))")
sudo $script
```

The script creates a sample configuration file in `/etc/doRedis.conf` and starts the service. You may wish to edit the config file and restart the service with your own configuration.

The script also compiles and installs a sentinel program into `/usr/local/bin/doRedis` that monitors and restarts if necessary R worker processes used by doRedis. The script configures the doRedis service to start automatically in default runlevels.

Uninstall the doRedis service

To fully un-install the service, run:

```
sudo R --slave -e "system(system.file(
     'scripts/redis-worker-uninstaller.sh', package='doRedis'))"
```

That will terminate any running R workers and remove the service.

Configure the service

Service configuration is controlled by the `/etc/doRedis.conf` file. The `#` character means comment and everything after that character is ignored per line. Settings appear as `key: value` and may appear in any line order. Here is an example:

```
n: 2              # number of R workers to start
queue: RJOBS      # foreach job queue name
R: R              # path to R (default assumes 'R' is in the PATH)
timeout: 5        # wait in seconds after job queue is
                  # deleted before exiting
iter: 50          # max tasks to run before worker exit and restart
host: localhost   # redis host
port: 6379        # redis port
user: nobody      # user that runs the R workers
```

The example config file above will start and maintain two R worker processes listening on the doRedis work queue named "RJOBS". If an R worker crashes

or otherwise terminates the service will replace it with a new one to maintain a total of `n` (2 in the above example) workers.

The `n:` and `queue:` entries may list more than one set of worker numbers and queue names delimited by exactly one space. If more than one queue is specified, then the n: and queue: entries *must* be of the same length. For example,

```
n: 2 1
queue: RJOBS SYSTEM
```

starts a set of two R workers listening on the queue named "RJOBS" and, separately, a single R worker listening on the queue named "SYSTEM".

Why multiple queues? We find it sometimes convenient, especially on larger and/or EC2 deployments, to keep a back-channel open to the workers for system and other lighter-weight tasks. That way, even if the workers are busy with a computation and a deep work queue, we can run quick ad hoc jobs on the system queue.

Managing the service

Stop and start the service with the usual system service commands in your operating system, examples are shown in the listing below.

```
sudo /etc/init.d/doRedis stop
sudo /etc/init.d/doRedis start

# or

sudo service doRedis stop
sudo service doRedis start
```

Logging

The service version of doRedis logs its output to the system logger when the `loglevel=1` option is set. When `loglevel=1`, all output from the worker R processes are logged. A good way to add printf-like debugging statements to your parallel doRedis programs is to simply insert R's `message()` function in your code with informative messages. Those messages can later be viewed in the system logs on the computers where the doRedis service runs.