bids.manikaranpowerltd.in

Installation instructions

Info: http://bids.manikaranpowerltd.in
Author: Abhijit Menon-Sen <abhijit@menon-sen.com>
Date: 2011-04-05, revised 2013-06-05

Introduction

This document explains how MPL's online bid management system is set up. The instructions are aimed at any competent Linux system administrator.

Server

The application is hosted at E2E Networks in their NOIDA data centre. It runs on a virtual server (plan VPS-HDD-2B). The server configuration is summarised below:

CPU	Intel Xeon E5-2620 @ 2.00GHz (x4)
RAM	6GB
Disk	100GB
OS	Debian 7.0 (minimal)
IP	180.179.212.227
Traffic	1000GB/month

The application was originally hosted on a VPS-HDD-1B virtual server in E2E's Mumbai data centre, but was moved to its present location in June 2013. These instructions have been updated to refer to the new setup.

Backups

E2E maintains incremental six-hourly snapshot-based backups of the entire disk using Idera Server Backup Enterprise. One of the four daily backups is retained for 14 days, and 2 monthly backups are retained for 2 months. Restoring from the backups directly to the server requires a support request, but E2E can provide a web interface to download files from backups.

We also store three-hourly Postgres dump files in `/var/lib/postgresql/backups`, retaining only the latest backup in prior years, previous months in the current year, and each day of the current month. See `utils/pg_retain_backups`.

Domain

The manikaranpowerltd.in domain is registered through Net4India. The A record for the bids subdomain is created through Net4India's domain management web interface.

SSL Certificate

The application uses a RapidSSL SSL certificate for the bids.manikaranpowerltd.in domain. This must be renewed periodically.

The private key and CSR (Certificate Signing Request) are generated using the following OpenSSL command:

```
openssl req -newkey rsa:2048 -nodes -keyout bids.manikaranpowerltd.in.key \
    -out bids.manikaranpowerltd.in.csr
```

The RapidSSL support web site explains how to "Generate a Certificate Signing Request (CSR) for Apache + Mod SSL + OpenSSL": `https://knowledge.rapidssl.com/support/ssl-certificate-support/index?page=content&id=SO13985`

The private key is installed as `/home/mpl/bids.manikaranpowerltd.in.key` and the signed certificate as `/home/mpl/bids.manikaranpowerltd.in.crt` (this file should also contain the intermediate CA certificates from RapidSSL).

Installation

1. Package installation

    1. Add the wheezy-backports repository: `http://backports.debian.org`
    2. Add the PGDG repository: `https://wiki.postgresql.org/wiki/Apt`
    3. Install package updates:

        ```
        apt-get update && apt-get upgrade
        ```

    4. Install packages:

        ```
        apt-get install build-essential nginx-light postfix pure-ftpd \
            bind9 git memcached postgresql-9.2 postgresql-contrib-9.2 \
            postgresql-server-dev-9.2 poppler-utils mailutils
        ```

2. Set up a user for the application

    1. Create the user:

        ```
        useradd -m mpl
        ```

    2. **Do not** set a password. Authentication should be through SSH keys only.
    3. Install SSH public key(s) into `~mpl/.ssh/authorized_keys`

3. Set up the application repository

    1. Create the repository and working directory:

        ```
        su - mpl
        git init --bare /home/mpl/git
        mkdir /home/mpl/web
        ```

    2. Add a post-receive hook to update the working directory:

        ```
        touch git/hooks/post-receive
        chmod +x git/hooks/post-receive
        cat > git/hooks/post-receive <<EOF
        #!/bin/sh
        GIT_WORK_TREE=/home/mpl/web git checkout -f
        EOF
        ```

    3. Push the application to the new repository (from the source repository):

        ```
        git remote add web \
            ssh://mpl@bids.manikaranpowerltd.in/home/mpl/git
        git push web master
        ```

Application dependencies

1. Set up a private Perl installation

    1. Install perlbrew:

        ```
        wget -O - http://install.perlbrew.pl | bash
        ```

    2. Install cpanm:

        ```
        perlbrew install-cpanm
        ```

    3. Set up `~/.bash_profile`:

        ```
        cat >> ~/.bash_profile << EOF
        source ~/perl5/perlbrew/etc/bashrc
        export PERL_CPANM_OPT="-q --skip-installed"
        EOF
        ```

    4. Install Perl 5.14.4:

        ```
        perlbrew install perl-5.14.4
        ```

    5. Install Mojolicious 3.97 (**not** 4.x):

        ```
        git clone git::://github.com/kraih/mojo.git
        cd mojo
        git checkout -b v3.97 v3.97
        perl Makefile.PL; make; make test
        make install
        ```

    6. Install other module dependencies:

        ```
        cd web
        cpanm --installdeps .
        ```

2. Start the MPL server at boot

    1. Append the following to `/etc/rc.local`:

        ```
        su mpl -l -c 'cd ~/web; hypnotoad ./mpl'
        su mpl -l -c 'cd ~/web; utils/mail-daemon &'
        ```

    (Application configuration via `mpl.conf` is described in `docs/app-overview`. See also the "Routine operations" section later in this document.)

Other services

1. Configure BIND as a caching resolver

    1. Add the following to `/etc/bind/named.conf.options`:

        ```
        forwarders { 8.8.8.8; 8.8.4.4; };
        listen-on { 127.0.0.1; };
        ```

    2. Tell the system to use this as a resolver:

        ```
        echo "nameserver 127.0.0.1" > /etc/resolv.conf
        ```

2. Configure Postfix to send mail and keep copies of everything

    1. Install configuration files from the source repository:

        ```
        cp misc/main.cf misc/virtual misc/sender_bcc \
            /etc/postfix
        ```

    2. Run `postmap /etc/postfix/virtual /etc/postfix/sender_bcc`

3. Configure nginx

    1. Install nginx server definitions from the source repository:

        ```
        cp misc/nginx.site /etc/nginx/sites-enabled/default
        ```

        (The above sets up two servers, with the one on port 80 redirecting everything to the one on port 443, and the latter proxies everything to the application.)

4. Configure pure-ftpd

    1. Enable PureDB authentication:

        ```
        cd /etc/pure-ftpd/auth
        ln -sf ../conf/PureDB 50pure
        echo no > ../conf/PAMAuthentication
        ```

    2. Create a user and set the password:

        ```
        useradd -s /sbin/nologin -d /dev/null ftpuser
        pure-pw useradd mplexports -d /home/mpl/web/uploads/exports \
            -u $(id -u ftpuser) -g $(id -g ftpuser)
        pure-pw mkdb
        ```

Periodic tasks

1. Install crontabs for `postgres` and `mpl`:

```
cp ~mpl/web/misc/crontab /etc/cron.d/mpl
```

2. Install `logrotate` configuration for application log files:

```
cp ~mpl/web/misc/logrotate /etc/logrotate.d/mpl
```

Note that the `misc/crontab` file is meant only to be installed in `/etc/cron.d` and not as a user crontab. It runs jobs as user `postgres` as well as `mpl`. Output is sent to `root`.

The file contains comments explaining the various jobs.

Database backups

One job runs `pg_dump` every three hours to create a compressed dump file in `/var/lib/postgresql/backups`. We also run a backup retention script every night to delete older backups.

The script retains only the latest backup from previous years, previous months in the current year, and previous days in the current month. It leaves everything for the current day alone.

Mail retention

Postfix is configured to deliver a copy of *all* outgoing mail to user `mpl`, i.e. to `/var/mail/mpl`. It's useful to have this mail for a few days after it's sent, but it also takes up a lot of space, since we send out all archived files as attachments.

These cron jobs copy each day's mail into dated files in `~mpl/mail`, and keep the last seven days' worth of such files. These may be useful for debugging, or if mail needs to be resent for some reason. (But note that `/var/log/mail.log` is a better place to track down any "I didn't get mail" complaints.)

Utility scripts

The system depends on various utility scripts to be run periodically, for example to process daily reports. These are documented in `docs/app-overview`.

A copy of VeriSign's International Server CA Class 3 CA certificate must be installed as `/etc/ssl/certs/Verisign_Class_3_International_Server_-_G3.crt`. This is used to validate the certificate presented by the IEX FTP/SSL server. A copy of this file is included in `misc/`, but this may need to be updated from time to time. Remember to run `c_rehash` after changing the certificate (or list the certificate in `/etc/ca-certificates.conf` and use the `update-ca-certificates` mechanism).

Routine operations

The application is started from `/etc/rc.local` as described above. At any time, there should be a handful of `mpl` processes running:

```
mpl       1494     1  0 Jul04 ?        00:00:15 /home/mpl/web/mpl
mpl       1498  1494  0 Jul04 ?        00:00:56 /home/mpl/web/mpl
mpl       1499  1494  0 Jul04 ?        00:00:59 /home/mpl/web/mpl
mpl       1500  1494  0 Jul04 ?        00:00:57 /home/mpl/web/mpl
mpl       1501  1494  0 Jul04 ?        00:00:56 /home/mpl/web/mpl
```

In this case, the parent process is 1494 (`~mpl/web/hypnotoad.pid`), and the application can be restarted by rerunning the command from `/etc/rc.local` thus:

```
su mpl -l -c 'cd ~/web; hypnotoad ./mpl'
```

Note that the application *must* be restarted if Postgres is restarted for any reason (e.g. a package upgrade with automatic server restart). It will fail mysteriously otherwise ("Can't load user").

To shut down the application completely, run `hypnotoad --stop ./mpl` as above (or stop `nginx` to prevent access to the site).

Apart from the `mpl` processes, `mail-daemon` must also be running for (some, but not all) outgoing mail to be generated. Other important processes are `nginx`, `postgres`, `postfix`, and `named`. All of these are installed from Debian packages, so their configuration and log files are in the standard places, and they can be started, stopped, and restarted using the system init scripts.

Exported bids

Exported bids are stored in `~mpl/web/uploads/exports/YYYYMMDD` as numbered `.zip` files. These are downloaded by MPL staff via FTP. The server runs `pure-ftpd` in standalone mode. An `mplexports` user is defined in `/etc/pure-ftpd/pureftpd.passwd`, and has access to the exports directory. Its password can be changed with `pure-pw passwd mplexports` (remember to run `pure-pw mkdb` afterwards).

Report processing

The `utils/process-reports` script is started by cron every afternoon. Sometimes it fails and needs to be restarted by hand (usually because it couldn't connect to the FTP server despite repeated attempts). It can be started as follows:

```
su mpl -l -c 'cd ~/web; utils/process-reports'
```

The process will stay in the foreground and is expected to run silently for several hours (usually from early afternoon to about 20:00). It will produce messages only in case of errors. While it runs, it logs messages to `~/web/uploads/files/YYYY/MM/DD/results.log`. The FTP server and username are in `mpl.conf`, and the password is in `secrets.conf`.

Postgres configuration

This is a summary of the changes made to the Postgres configuration. It is not exhaustive because, in general, the steps involved depend on the target server's hardware configuration.

The following table shows the changes made to `postgresql.conf`.

`shared_buffers`	512MB
`temp_buffers`	16MB
`work_mem`	16MB
`maintenance_work_mem`	256MB
`checkpoint_segments`	32
`checkpoint_timeout`	15min
`effective_cache_size`	3GB

These values are suitable for the existing server. For general advice about performance tuning, consult a Postgres tuning guide.

Note that the setting of `shared_buffers` depends on the value of `kernel.shmmax` being set to a larger value in `/etc/sysctl.conf`.

In addition to the above, the logging parameters have also been changed to log connections, disconnections, checkpoints, and all queries, lock waits, and large temporary files. These settings are useful for tuning as well as forensics.

In `pg_hba.conf`, local connections should be set to `trust` authentication. Of course, the server should not need to be configured to accept non-local connections at all.

System administration checklist

Here is a non-exhaustive list of routine maintenance tasks:

1. Follow debian-security announcements and install important security updates.
2. Follow the latest PostgreSQL 9.2.x packages from the PGDG repository.
3. Install updates to nginx, Postfix, BIND, and other services.
4. Monitor PostgreSQL backups in /var/lib/postgresql/backups.
5. Monitor filesystem backups and available disk space.
6. Monitor process status and logs for: nginx, postgres, memcached, postfix, pure-ftpd.
7. Monitor overall system health and parameters (CPU usage, memory, process list, etc.).
8. Monitor /var/log/syslog for any unusual activity (warnings or errors from system daemons, security threats, etc.).
9. Monitor output from routine cronjobs.
10. Monitor /home/mpl/web/uploads

Package upgrades

The `unattended-upgrades` package has been installed and configured to install security updates without intervention. Note that the postgresql packages have been specifically blacklisted, so as to avoid downtime on the production database. Also, this does not eliminate the requirement to follow security updates, since there may be special considerations for certain security problems (for example, it will not reboot after a kernel upgrade, for obvious reasons).

Disclaimer

E2E has administrative access to the server, and have configured (at least) the backup agent. Any changes made by E2E to the system are outside the scope of this document.