BUILDING A SOFTWARE UPDATE SERVER REPLICA WITH FREEBSD 05 Nov

A SHORT INTRO

One of the interesting facets of starting a business is that you don't have any money. This forces you to find creative, and most importantly, cost-effective (i.e. free) solutions to problems. For example, our service center is divided into two networks - the "office" network (employee's machines, printers, servers) and the "service" network (customer machines).
The only server license we could afford is being used by the Mac mini server in the office network, providing mail, wiki, calendar and chat services. But we also needed a robust system to host netboot, AFP, NFS and local Software Update services as well as to act as a secure gateway between the service and office networks. So I built a PC, set it up with 3 drives and RAID-Z and two gigabit NICs and FreeBSD - all for roughly half the price of a Mac mini server.
Now the question was how do I get this PC to act as a Software Update server to all the service machines. There's actually a million ways, here's just one:

THE PIECES

First, let's find out where all the software update data is:

```
$ defaults read /etc/swupd/swupd updatesDocRoot
/Volumes/Drobo/SoftwareUpdate/
```

I wanted to avoid using any messy authentication (since software updates aren't exactly trade secrets), so I set up a read-only rsync repository on the Mac mini to share the software update document root with a very simple rsyncd.conf file:

```
[sus]
comment = software update server
path = /Volumes/Drobo/SoftwareUpdate/html
readonly = true
```

That's all the changes we have to do to the Mac server. On the FreeBSD side, we need to install lighttpd and dnsmasq:

```
$ cd /usr/ports/www/lighttpd
$ sudo make install clean
$ cd /usr/ports/dns/dnsmasq
$ sudo make install clean
```

The FreeBSD server also functions as a DHCP server for the service network and sets itself as the primary DNS server for all the DHCP clients. The idea was that we wouldn't have to make any changes to the customer's machines to have them use our local SUS, ie to use a transparent software update solution. So I added these lines to /usr/local/etc/dnsmasq.conf (172.17.65.1 is the IP of the FreeBSD server):

```
address=/swscan.apple.com/172.17.65.1
address=/swcdn.apple.com/172.17.65.1
address=/swquery.apple.com/172.17.65.1
```

I then create a directory for the SUS data on my RAID-Z, ZFS pool:

```
sudo mkdir /data/sus
```

and edit /usr/local/etc/lighttpd/lighttpd.conf to use the SUS docroot as the default docroot:

```
server.document-root = "/data/sus/"
```

Then just fire up dnsmasq and lighttpd.

MAKING IT RUN

The final step is to put it all together. At first I thought it would be a trivial matter of just dumping the data, but there are some additional steps that need to be taken, so I threw together this little script:

```
#! /usr/bin/env bash

if [[ $# -lt 4 ]]
then
  echo "usage: $(basename $0) server repo port destination" 2>&1
  exit 1
fi

MAC_SERVER=$1
REMOTE_REPO=$2
REMOTE_PORT=$3
LOCAL_DOCROOT=$4
HOSTNAME=$(hostname)

# download latest update packages, removing deprecated
rsync -av --delete rsync://${MAC_SERVER}:${REMOTE_PORT}/${REMOTE_REPO} ${LOCAL_DOCROOT}

if [[ ! $? ]]
then
  echo "Failed to fetch software update content" 2>&1
  exit 1
fi

# update hostnames in catalog files
find ${LOCAL_DOCROOT} -name *.sucatalog -type f -exec sed -i '' "s/${MAC_SERVER}:8088/${HOSTNAME}/g" {} \;

# rebuild symlinks
cd ${LOCAL_DOCROOT}
rm ./index*
ln -s ./content/catalogs/others/index-leopard-snowleopard.merged-1.sucatalog ./index-leopard-snowleopard.merged-1.sucata$
ln -s ./content/catalogs/others/index-leopard.merged-1.sucatalog ./index-leopard.merged-1.sucatalog
ln -s ./content/catalogs/index.sucatalog ./index.sucatalog
# 10.4 seems to need this
ln -s ./content/catalogs/index.sucatalog ./content/catalogs/index-1.sucatalog
```

Save that into a file and run it:

```
$ ./supdate.sh macserver.example.com sus 874 /data/sus/
```

If all worked as expected, save it so that it's run periodically (say, once a week). Notice that the script is assuming the Mac server was hosting the updates on port 8088! I had a version of the script that actually read the repo path and port from the Mac server using SSH, but I like this

solution much more - no authentication required.

Posted by filipp in server | freebsd | deployment

PREV POSTS NEXT POSTS