Automated Deployment of hundreds of OpenSolaris machines

Philip Torchinsky
Sun Microsystems

My thanks to:
Peter Karlsson
Scott Dickson

Full paper: http://voyager-eng.livejournal.com/1155.html

When we need a lot of similar machines?

- a classroom
- an office or a call-center
- lab environment, test farm
- datacenter, application (i.e. web) hosting facility

Issues with identical software config

- hostname must be unique
- (a bit) different hardware
- post-install updates

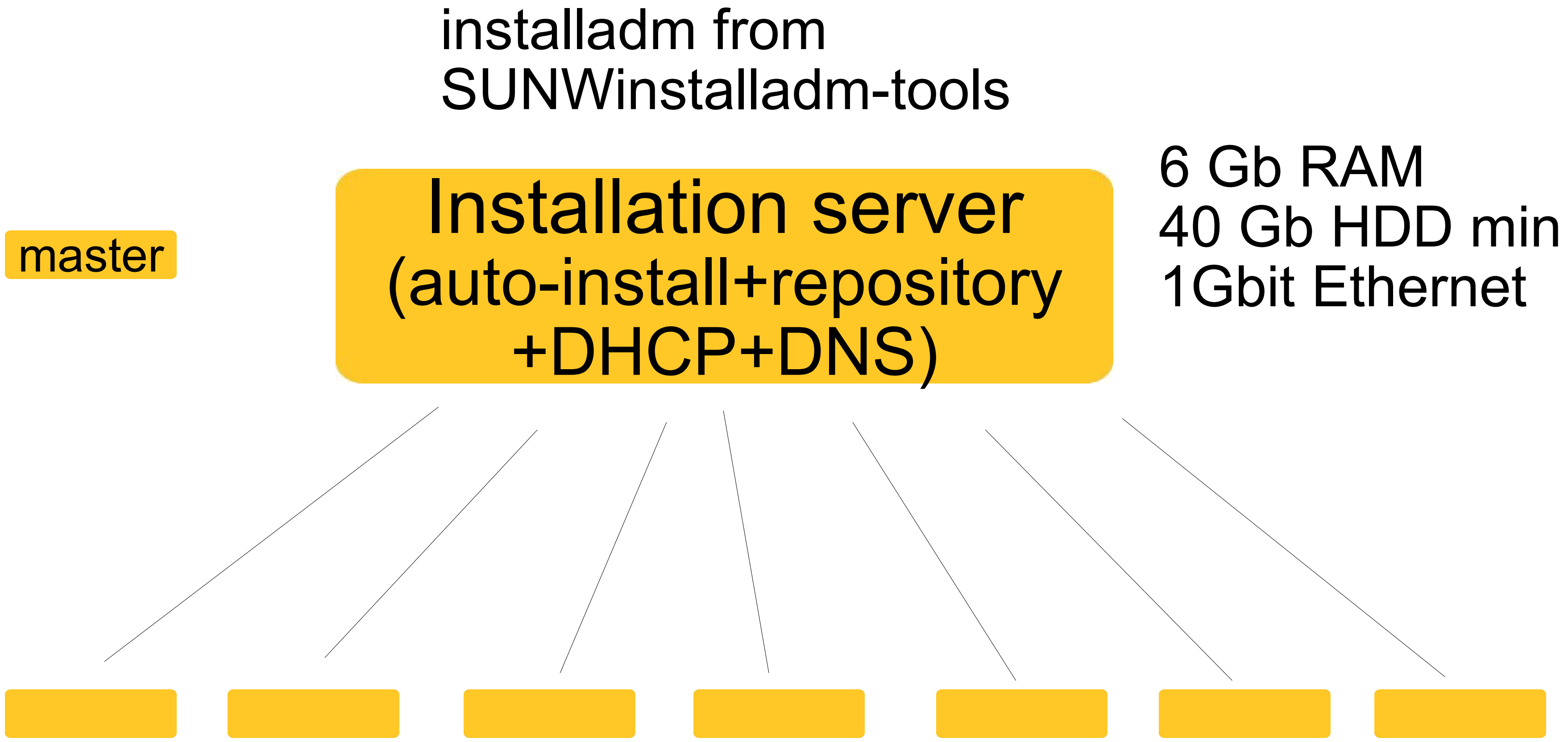
Tasks

- Setup **installation server** (where everything will be installed from)
- Setup local **mirror** of OpenSolaris **repository**
- Make master machine
- Make a snapshot from master machine and put it on the server as a file
- Configure remote machines for network boot
- Make **a service to pull ZFS snapshot** of master machine filesystem after installation and first boot
- Compose a package with the pulling service
- Make **local repository**, put the “puller” package there
- Install everything
- Perform post-install actions with scripts

What's new in this concept?

- Using OpenSolaris auto-install
- Benefiting from ZFS and boot environment feature
- Post-install actions approach

How it does work?



Make full repository
1)http://blogs.sun.com/migi/entry/create_your_own_opensolaris_ips2
2)http://opensolaris.org/jive/thread.jspa?threadID=91113
3)http://ekschi.com/technology/2009/09/10/build-your-own-opensolaris-200906-ips-repository-on-your-laptop/

Tips and tricks

- Do not try to install less packages than by default
- Use installadm and dhcpmgr to configure DHCP server
- Do not expect less than 30 min for initial installation
- When making a repository please use separate zfs filesystem
  - zfs set atime=off /export/repository/mirror
  - zfs set compression=on /export/repository/mirror

Prepare zfs snapshot from master machine

- zfs snapshot /rpool/ROOT/opensolaris-2@master
- zfs send /rpool/ROOT/opensolaris-2@master > /export/image/zflar.zfs; gzip /export/image/zflar.zfs
- scp /export/image/zflar.zfs.gz server:/export/image/zflar.zfs.gz

ssh from server to clients without password prompt

- Make server certificate (on server):
  ssh-keygen -t rsa
  (this will create an id_rsa.pub in /root/.ssh)

- Transfer id_rsa.pub to master machine with scp or other method to
  /root/.ssh/id_rsa.pub_<server_host_name>
  In our case it was /root/.ssh/id_rsa.pub_j1holsrv
- On master machine do as folows:
  - cd ~/.ssh
  - cat id_rsa.pub_j1holsrv >> authorized_keys
- in /etc/ssh/sshd_config:
  - PermitRootLogin no -> PermitRootLogin yes

http://blogs.sun.com/jkini/entry/how_to_scp_scp_and

Steps to perform

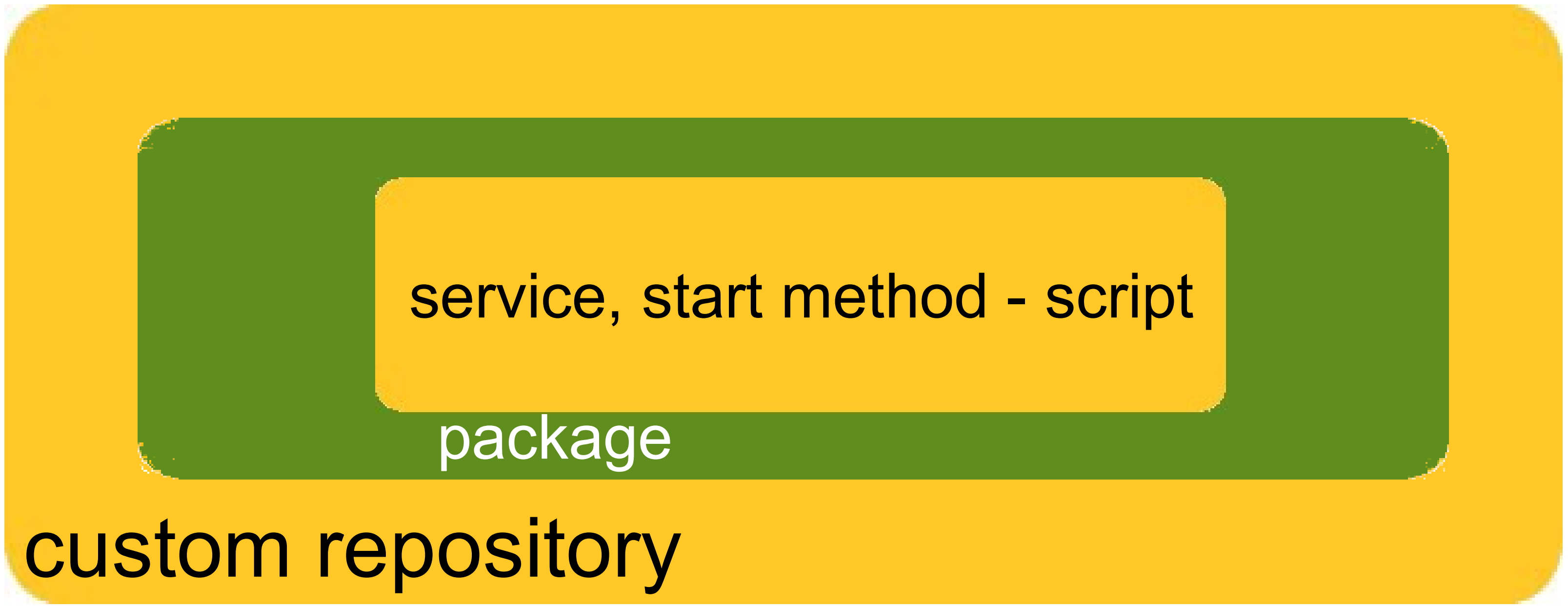
- Install OpenSolaris 2009.06 on the clients
- Retrieve master machine filesystem snapshot to a client and make default (active) boot environment on its base

  How to do it?

- Make a service containing a script, which retrievs snapshot and make new boot environment
- Make a package to wrap a service
- Make custom repository containing this package only
- Include the package into auto-install manifest

OSDevCon

Design of post-install zfs snapshot pull



Script to pull zfs snapshot (key commands)

```
mkdir ${MNT}
mkdir ${ZMNT}
mount -o ro -F nfs ${NFS} ${MNT}

beadm create {$NBEADM}
zfs destroy -r rpool/ROOT/{$NBEADM}
gzcat ${IMG}/${FLAR} | zfs receive -vF rpool/ROOT/{$NBEADM}

beadm mount {$NBEADM} {$ZMNT}
rm ${ZMNT}/etc/nodename

beadm umount {$NBEADM}
zfs set mountpoint=/ ${NBOOTFS}
zpool set bootfs=${NBOOTFS} rpool
beadm activate {$NBEADM}
reboot
```

Service manifest (a fragment)

```
<instance name='default' enabled='true'>

        <exec_method
            type='method'
            name='start'
            exec='/lib/svc/method/zfs-flar.sh'
            timeout_seconds='0' />

        <exec_method
                        type='method'
                        name='stop'
                        exec=':true'
                        timeout_seconds='0' />
    </instance>
```

OSDevCon

install new repository

```
# svccfg -s pkg/server
> add local
> select local
> addpg pkg application
> addpg start method
> setprop pkg/mirror = boolean: faulse
> setprop pkg/port = 8001
> setprop pkg/inst_root = astring:"/export/pkgservers/local"
> setprop pkg/threads = count: 50
> exit

# svcadm refresh pkg/server:local
# svcadm enable pkg/server:local
```

Post install actions

Setting nodename (hostname)

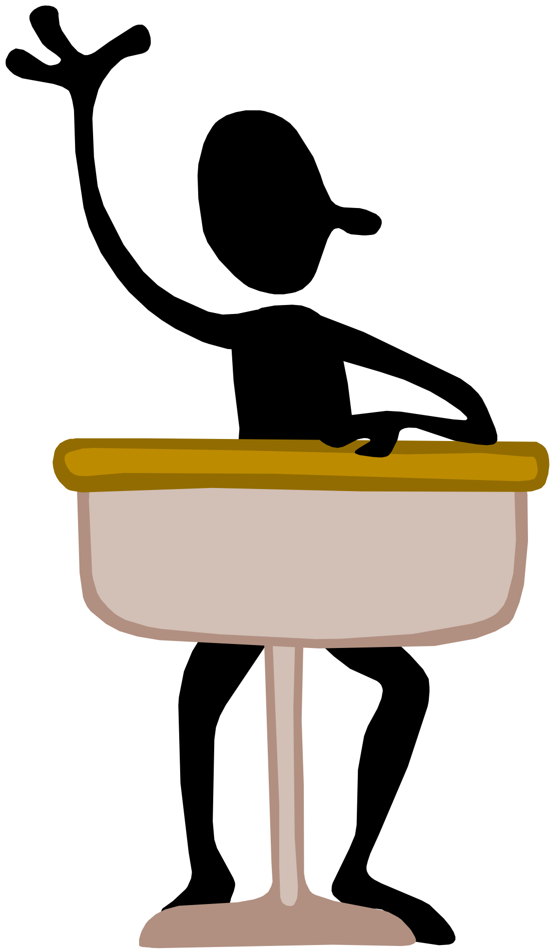
```
#!/bin/sh

echo "Setting hostname on $1"
ssh $1 "echo $1 > /etc/nodename; hostname $1"
```

Reading?

- developers.sun.com/developer/technicalArticles/opensolaris/boot-environments.html
- dlc.sun.com/osol/docs/content/2009.06/AIinstall/
- opensolaris.org/os/project/pkg/Mirroring/
- blogs.sun.com/jkini/entry/how_to_scp_scp_and

Questions?



thank you

Automated Deployment of hundreds of OpenSolaris machines

Philip Torchinsky
philip.torchinsky@sun.com