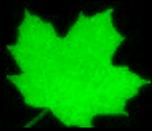


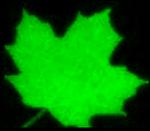
FreeBSD Unified Configuration

Andrew Pantyukhin
infofarmer@FreeBSD.org

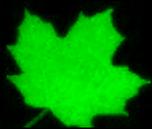


once upon a time

a private cloud



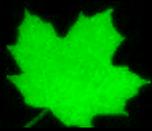
petabytes of data
dozens of gigabits of transfers
teraflops of processing



4 countries

10 cities

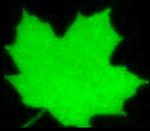
13 data centers



11 service providers

15 support contracts

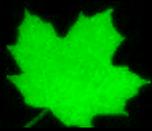
5 SLA types



~100 machines

~20 hardware configurations

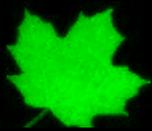
~1000 hard drives



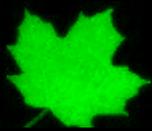
30 local networks

5 network types

7 out-of-band console types



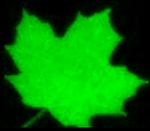
1 operating system
(potentially more)
5 boot types



1 systems engineer

1 network engineer

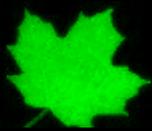
1 field engineer



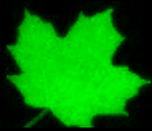
initial tactics

owned -> cluster

leased -> setup & forget



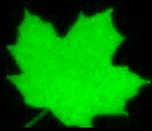
briefly considered
puppet, chef, cfengine
scripted per-node management



priorities

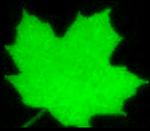
**extremely low ops load and
complexity**

**extremely high performance and
flexibility**



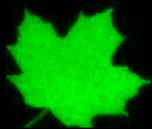
solution

unified configuration management
unified deployment



unified?

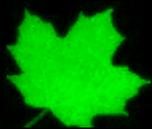
exactly same root fs everywhere
exactly same configs everywhere



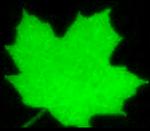
/.git

/usr/local/project/.git

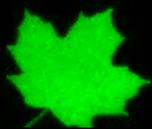
/usr/home/*/.git



fully distributed
flexible semi-auto master-master
sync
no symlinking, copying (almost)



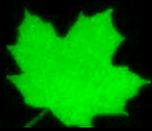
**concentrated
complexity
smarter specialization
role-aware configs**



roles

passwd, group

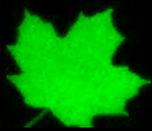
aware.map



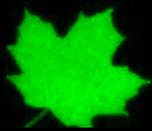
role-aware boot

who am I? what are my MACs?

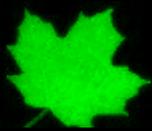
MAC -> aware.map -> host -> roles



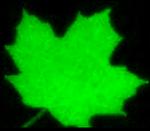
rc.conf - role-aware shell script intricate evaluation



```
ntp_enable="YES"  
role.www() { nginx_enable="YES"  
          }  
role.host1() { hack_enable="YES"  
            }
```

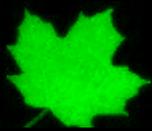


```
for i in $myroles  
do  
  role.$i
```



nginx.conf role- compatible

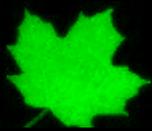
```
{ server_name www1; }  
{ server_name www2; }
```



syslog.conf role- unaware

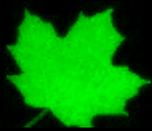
syslog.conf - most nodes

syslog.conf.collect - log collector

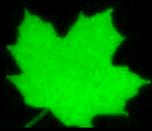


rc.conf-based work-around

```
role.logcol() {  
  syslog_flags="-c  
  syslog.conf.collect" }
```



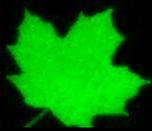
```
fstab role-unaware  
#empty  
loader.conf, scripts
```



boot drive

`/dev/ufs/root1 - 10G`

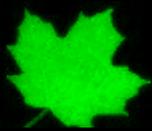
`/dev/ufs/root2 - 10G`



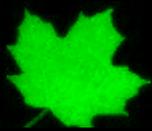
boot drive

/dev/gpt/swapserial - 4G

/dev/ufs/serial - leftover



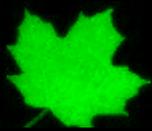
loader.conf
vfs.mountroot
falls back to NFS root



deployment

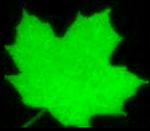
aware.map, configs adjustment

dhcp, etc



deployment

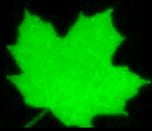
**find & partition a suitable drive
untar recent image into root1**



full upgrade

untar new image into root2

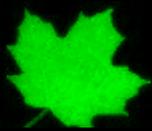
pivot root1<->root2 (kernel!!)



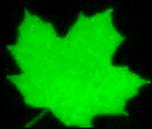
full upgrade

rsync? pkgng?

freebsd-update?

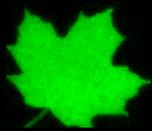


pkg upgrade
pkgng

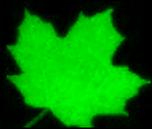


continuous upgrade

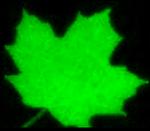
git pull



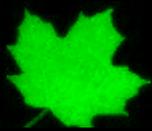
edit on any box
commit, push
powerful conflict resolution



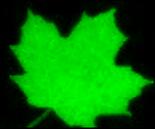
pretty scalable



git is awful
rsync is lacking
need more smart configs



pretty simple
fool-proof
single-view cloud-wide config



Q&A