



ORACLE[®]

Virtual Lab

High Impact Low Cost Virtual Lab

Vladimír Marek, Principal Software Engineer
Systems Revenue Product Engineering (RPE)

Program Agenda

- Who are we
- The problem
- Virtualization Technologies overview
- The solution
- Demo
- Technical details
- Q/A



Who We Are: Systems Revenue Product Engineering


Oracle Solaris Sustaining

- Responsible for fixing bugs in released (revenue) versions of Oracle Solaris
 - Solaris 11.1, Solaris 10, Solaris 9, Solaris 8
 - Dealing with bugs reported by customers and found internally
- Organized in technology teams
 - Kernel, drivers, security, networking, file systems, utilities, naming, install, desktop, free and open source, etc.
- Own technologies in maintenance mode
 - UFS, PCFS, ...
- Development of troubleshooting and debugging technologies
 - Kernel debugger, crash dump analysis, ...

Most Common Tasks

- Reproduce the reported problem
- Find a root cause
- Design and implement fix
- Test the fix
- Code review
- Integrate the fix
- Test resulting patch
- Eventually provide interim diagnostic relief (patch)
 - Design, implement and test

Most Common Tasks: We Need HW For Testing

- Reproduce the reported problem
 - Find a root cause
 - Design and implement fix
 - Test the fix
 - Code review
 - Integrate the fix
 - Test resulting patch
 - Eventually provide interim diagnostic relief (patch)
 - Design, implement and test
- 
- The diagram features a cluster of hardware components (server racks and modules) on the right side. A red oval labeled "The Problem" is positioned in the center-right. Arrows point from the hardware to the first two tasks: "Reproduce the reported problem" and "Find a root cause". Another arrow points from the hardware to "Test the fix". A fourth arrow points from the hardware to "Test resulting patch". A fifth arrow points from the hardware to the sub-item "Design, implement and test" under "Eventually provide interim diagnostic relief (patch)".

Classical Solution

Yes, we do have a lot of HW but ...

- Large labs with thousands of machines
 - Automated installation supported
- Allow very complex setups
- Definitely required for HW related issues

but ...

- Rather slow to get the set up prepared
 - Set up and installation takes time
 - Not very suitable for quick tests and experiments
- For a lot of tests any HW is sufficient

Engineers Started To Address The 'But...'

There must a better solution

- We develop and support Solaris Operating system
- We develop virtualization tools
- We develop and produce microprocessors (SPARC)
- We develop and produce servers

...

- A virtual lab came in mind

Virtualization On Solaris

What to consider



Virtualization On Solaris

What to consider

- HW Platform – SPARC and x86

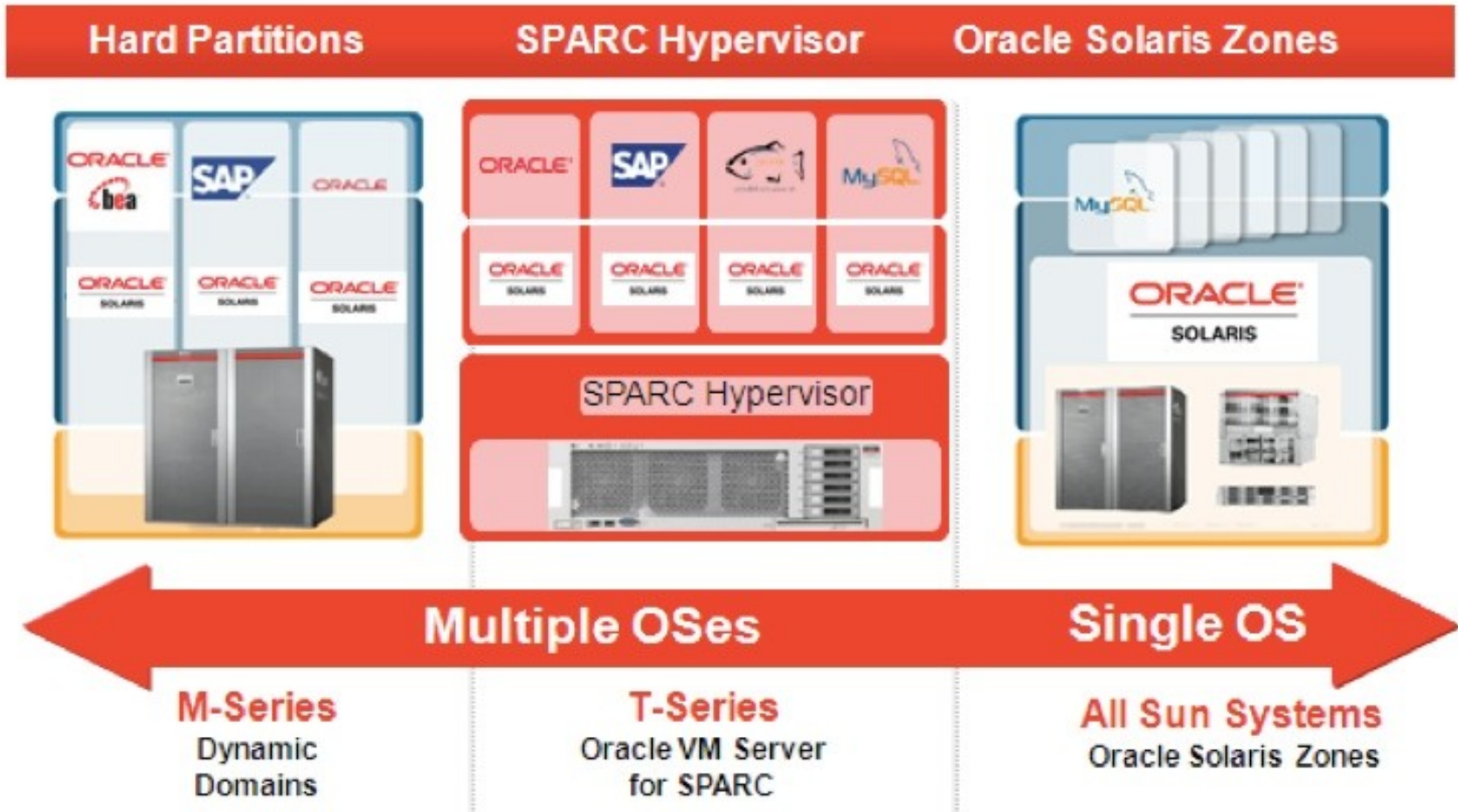


- Type of virtualization

Virtualization Available On Solaris

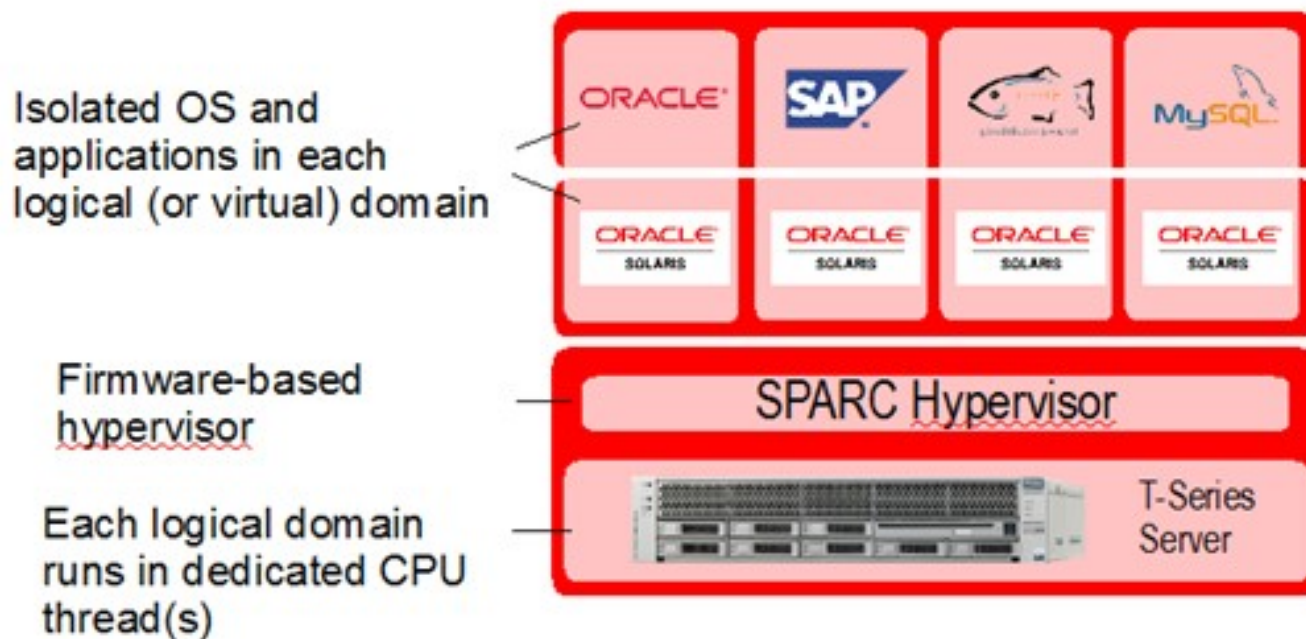
- SPARC
 - Dynamic Domains (hard partitioning)
 - M-Series Servers
 - Para-Virtualization
 - **Oracle VM Server for SPARC**
 - Formerly known as Logical Domains (LDOM)
 - Operating system level Virtualization
 - Solaris containers (zones), SPARC, x86, any HW
- x86
 - Full virtualization (both hardware and software)
 - **Oracle VM VirtualBox**, any x86 HW
 - Operating system level Virtualization (zones)

SPARC Server Virtualization



Virtual Lab: Oracle VM Server For SPARC (Logical Domains)

- Hypervisor is part of the system itself
 - Hidden in Open Boot PROM (roughly like BIOS on x86 systems)



Virtual Lab: x86 – Oracle VM VirtualBox

- Full virtualization
 - make possible to run unmodified guest operating system
 - Uses hardware virtualization extensions (Intel VT-x and AMD-V) where available
 - Can use paravirtualized drivers for better performance
- Multi-platform support
 - Host – Solaris, Linux, Mac OS X, Windows
 - Guest – Solaris, FreeBSD, OpenBSD, OS/2, Mac OS X, Linux, Windows, DOS



SPARC T-series processors

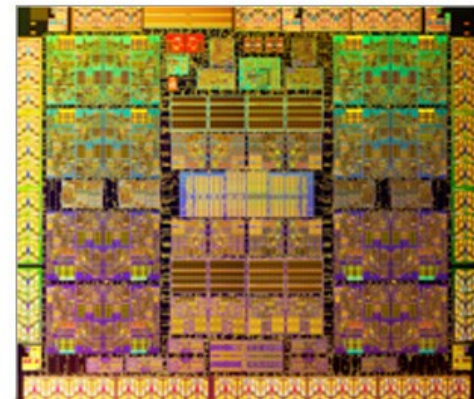
HW To Host Virtual Machines

- T3, T4 SPARC processors
 - Chip Multi Threading (CMT)
 - 8 or 16 cores (T3), 8 cores (T4)
 - Each core has 8 hardware threads – act as virtual CPU's
 - Up to 4 sockets (=> up to 512 threads)
- CPU cores executes instructions concurrently
- Each core switches between threads
- Virtualization
 - Hypervisor support (Hyper-Privileged execution mode)
 - Single CPU thread is enough for a virtual machine (aka domain)
 - Multiple threads can be assigned to one domain

SPARC T5 Processor

What is coming

- Better, faster, smaller ...
 - 28 nanometers
 - possible to have 1024 CPU's from operating system point of view
 - Enhanced single core performance (3,6GHz)



- <http://www.oracle.com/us/corporate/innovation/sparc-t5-deep-dive/index.html>

Oracle Solaris: ZFS

Zettabyte FileSystem

- Many features
 - 128-bit filesystem, includes volume manager functionality, easy to administer, deduplication, focus on data integrity (copy-on-write), snapshots and clones, ...
 - <http://en.wikipedia.org/wiki/ZFS>
- ZFS snapshots and clones
 - Killer feature for virtual lab
 - Creating clone is very fast and efficient
 - Initially clone consumes almost no additional disk space

Solution: Virtual Lab

A virtual lab created by engineers for engineers

- Controlled by simple command line interface (vm script) common to both x86 and SPARC platforms
- Fast and easy to use
- Use existing technologies
 - x86 – VirtualBox
 - SPARC – LDOMs
 - ZFS is used for fast cloning of virtual machine templates
 - DHCP
- Virtual machines accessible through
 - Console
 - Secure shell (ssh)
 - Graphical console (via rdesktop on x86 only)

Sample Session (Get a Machine In 2 Minutes)

```
$ vm list
$ vm create S11 S11mytest
$ vm start S11mytest
$ vm ssh S11mytest
Password:
Last login: Tue Jan 10 12:42:00 2012 from 10.0.2.2
Oracle Corporation      SunOS      5.11      11.0      November 2011

-bash-4.1$
...
-bash-4.1$ exit

$ vm destroy S11mytest
```

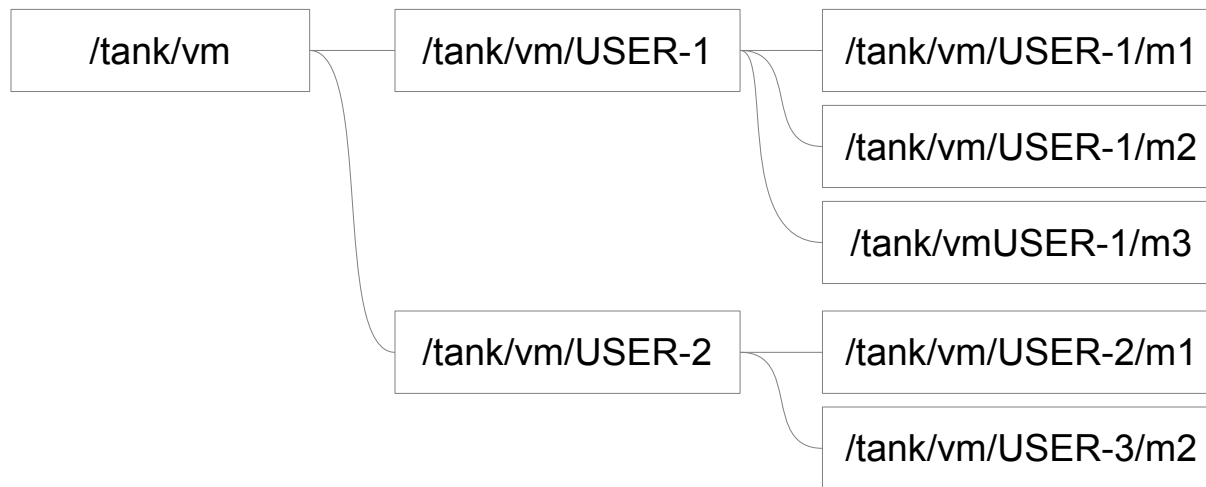
Implementation Details

Everything necessary is available in Solaris

- Written in KSH
 - x86 has 2000 lines of code + 1000 for indentation and comments
 - SPARC is smaller (1000+500 lines)

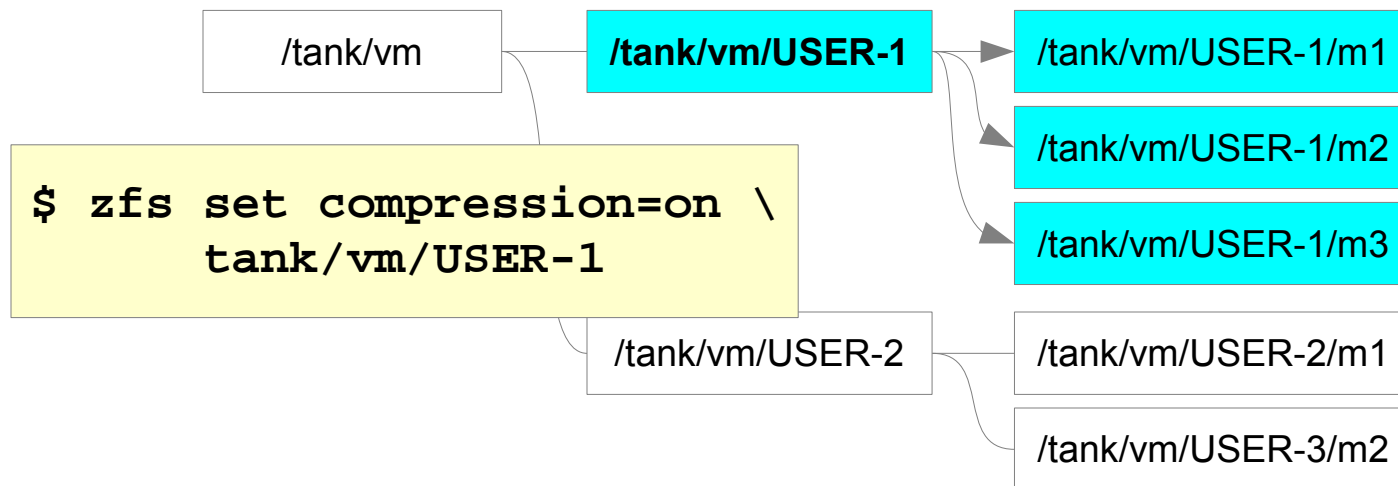
ZFS configuration

- Every template has to be separate ZFS filesystem
- User has to be able to create/clone ZFS filesystems into his directory, but must not be able to modify other's
 - ZFS permission inheritance to solves this problem



ZFS configuration

- Every template has to be separate ZFS filesystem
- User has to be able to create/clone ZFS filesystems into his directory, but must not be able to modify other's
 - ZFS supports permission inheritance to solve this issue



ZFS configuration

- `zfs create tank/vm` # Create base filesystem
- `zfs allow -s @virtual \` # Create permission set
`clone,create,destroy,mount,hold,rollback,share,sn`
`apshot,userprop,send,receive rpool/vm`
- `zfs allow -c @virtual tank/vm` # set during
"create" moment
- `zfs allow -l -g staff create,mount tank/vm`
Limited to group "staff"
 - `/usr/bin/chmod \` # Group staff can create dirs
`A+group:staff:add_subdirectory:fd:allow /tank/vm`

LDOMs example

Create SPARC virtual machine from scratch

- `zfs create -V 80G tank/vm/USER/TEST` # Create disk device
- `ldm add-domain TEST` # Create new LDOM
- `ldm add-vcpu 4 TEST` # Add virtual cpus
- `ldm add-memory 2g TEST` # Add memory
- `ldm add-vnet vnet0 vm-switch TEST` # NIC
- `ldm set-vconsole port=$PORT TEST` # "Serial" console
- `ldm add-vdsdev /dev/zvol/rdisk/tank/vm/USER/TEST TEST@vm-vds` # Register new virtual disk
- `ldm add-vdisk disk TEST@vm-vds TEST` # Assign the disk to the LDOM
- `ldm bind TEST; ldm start TEST`

Security/permissions

- x86
 - Mostly no special privileges required, every user runs it's own VirtualBox
 - Exception is reading and writing dhcp server configuration
- SPARC
 - Any operation on LDOMs requires authorization, but the authorization **solaris.ldoms.write** grants access to all virtual machines
 - Instead **vm** itself makes sure that user is working with his LDOM and uses **sudo** for obtaining root privileges
 - dhcp server too

What Is It Good For

- Extremely fast creation and maintenance of virtual machines
 - Controlled environment, which allows to track state of the running machines
 - Each user has his/her own environment (almost sand-box like)
- Suitable solution for any testing, which does not require specific HW
 - Unit testing, feature testing, patch verification, regression testing
- High number of OS templates available – Solaris 8, 9, 10, 11, ZFSSA, OEL, Windows, Ubuntu
 - Suitable also for fast checking of differences between platforms

Most Common Use Cases

- Test feature XY on specific Solaris build / release
- Get a machine for root causing a bug
- Find when a regression happened – what is the last build when feature XY worked properly
- Compare a feature XY between Solaris and OEL
- Get a machine for testing my fix both on x86 and SPARC
- iSCSI testing (COMSTAR) – multiple targets and initiators over virtual network

Future Plans

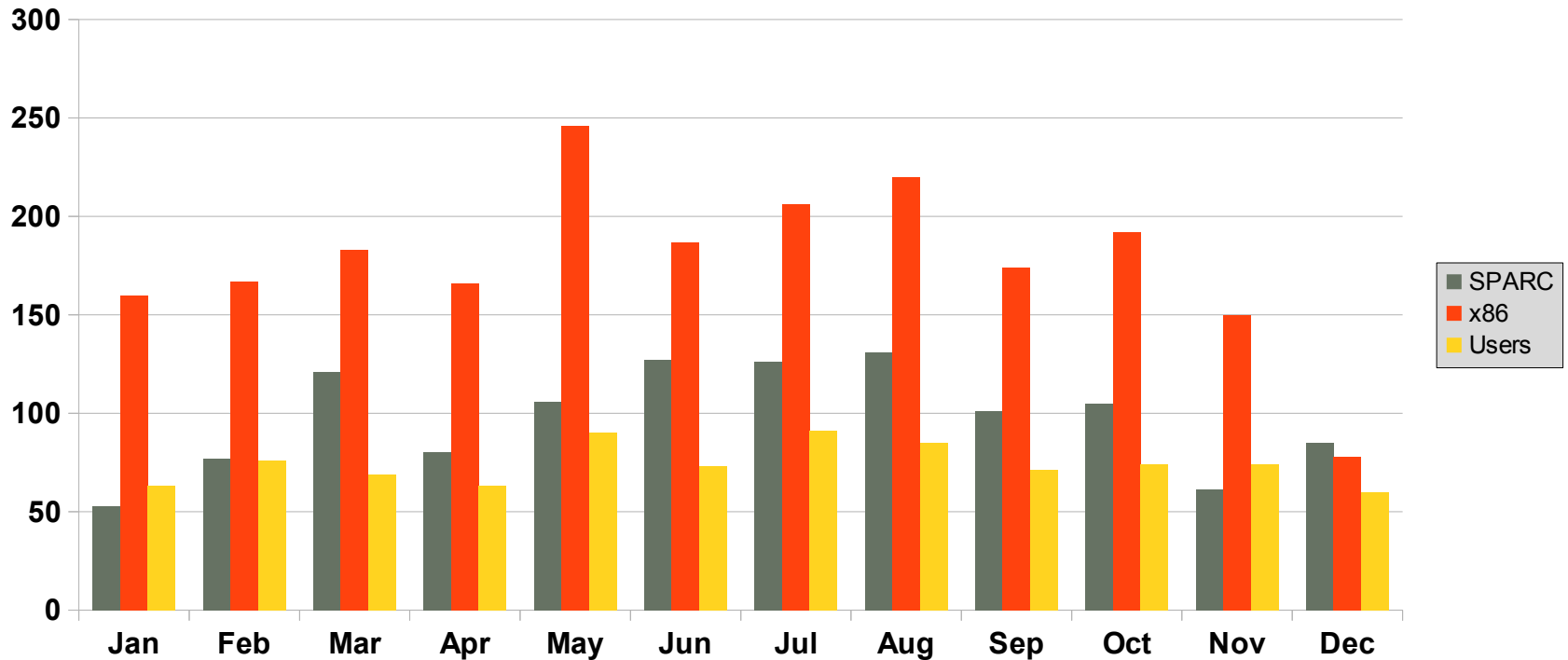
- Load balancing across multiple physical machines
- Improve network configurability
 - Deliver the same features available on x86 also on SPARC
- Improve scriptability
- Integration with other tools and frameworks
 - E.g., automate creation of a test machine from a successful automatic build

Appendix – Stats and Quotes

```
$  
$ vm stat  
$
```

Virtual Machines Created And Users

January 2012 – December 2012



3202 machines in total in 2012
889 (non-unique) users in 2012

Quote Attribution

Jan Hnátek, Solaris G11n

July 14th, 2010

“Allow me to add my *big thanks* to the virtual.czech team. Having used the system on our own server since ~ January 2010 I must say it's a truly amazing time-saver and our whole team (Solaris G11n Prague) got used to work with it.”

Quote Attribution

Darren Moffat, Solaris Security

September, 2011

“(16:48:24) darrenm: vlad just been pointed to the virtualbox and LDOMs based vm system you guys setup - that is amazing very cool! THANK-YOU THANK-YOU

(16:50:10) darrenm: mrj vm based "lab" "vm create snv_172 mybugfixvm && vm start mybugfixvm && vm ssh mybugfixvm" in less than 1 minute I have an installed machine to use

Questions and Answers

