COCKPIT
the Linux Server UI
I'm Stef Walter, work for Red Hat
I'm passionate about open source, but more specifically making it usable, coherent, and polished
Introducing you to cockpit today
Why we built it
What it is
How it works
Today we're going to talk about servers
Obligatory car analogy
But first we've gotta have an analogy
Actually this is not about cars it's about trucks
Building your own truck server
Deploying linux servers today is like building your own truck
You can build powerful purpose built trucks
Can built truck factories
Can virtualize your servers
Build servers that nobody else thinks are a good idea
Build workhorse servers that can be overloaded
Build fast low-latency servers
Or you can build it completely wrong
But but, I just want to drive it
But not everyone knows how to build a truck, as we've all seen
This is what it looks like to "drive" Windows server
[root@server ~]# uname
Linux
[root@server ~]#
This is what it looks like to "drive" Linux servers
Which looks like this to people who are not yet intimate with linux
Linux should be discoverable and configurable by non-experts
The learning curve is too steep
Cockpit is a discoverable face for RHEL, Fedora and Linux Servers
Cockpit is the server UI done right.
Cockpit is a prototype stage. It's alpha software.
This is what we're talking about
Look Ma!
(proof of concept)
<table>
<thead>
<tr>
<th>Virtual Disks</th>
<th>Storage Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>VirtIO Disk</td>
<td>8.0 GiB Hard Disk</td>
</tr>
<tr>
<td>VirtIO Disk</td>
<td>12.0 GiB Hard Disk</td>
</tr>
</tbody>
</table>

**Volume Groups**

<table>
<thead>
<tr>
<th>Volume Group</th>
<th>Storage Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>fedora</td>
<td>12.5 GiB Volume Group</td>
</tr>
</tbody>
</table>

**Storage Jobs**

(NO current jobs)

**Storage Log**

**November 5, 2013**

- udisksd: Acquired the name org.freedesktop.UDisks2 on the system message bus 10:40
- udisksd: udisks daemon version 2.1.0 starting 10:40
We built a proof of concept, that should give you a basic idea. The design and look-feel here is a bit dated, from what we're currently working on, but I hope it gives you an idea.
In Fedora now
# yum install --enablerepo=updates-testing cockpit
# setenforce 0
# systemctl enable cockpit-ws.socket
# xdg-open http://localhost:21064
Don't run this on a machine you care about (yet)
Architecture
Cockpit
Multi-Server Architecture

Management Services: NetworkManager, udisks, systemd, sssd, realmd...

kernel

CIMOM/OpenLMI

cockpitd

sshd

cockpit-agent

cockpit-web

Transports and Ports
- **SSH**: port 22
  a. gssapi-mic or
  b. password auth
- **WBEM**: port 5989
  a. GSSAPI + SSL or
  b. Basic Auth + SSL
- **WebSocket**: port 21064
  a. GSSAPI + SSL or
  b. Basic Auth + SSL

Messages and Models
- **DBus messages**
- **CIM messages**
- **syscall/sysfs**
- **API (Interface)**

Processes
- **kernel**
- **system service** running as root
- **user process** subject to authorization and audit

Note: (nearly) all processes socket-activated or run-on-demand

Initial: Local CIM
(access over unix socket, cockpitd uses polkit to authorize/audit)

Future: CIM over WBEM
(undecided how user identity authorization/audit will happen)
Discuss the architecture here
Goal: Discoverable
So one of our main goals is to make Linux and its various aspects discoverable.
lvcreate - create a logical volume in an existing volume group

NAME

LVCREATE(8) System Manager's Manual LVCREATE(8)

SYNOPSIS

lvcreate [-addtag Tag] [-alloc AllocationPolicy] [-a|--activate [a|e|l]{y|n}] [-k|--setactivationskip {y|n}] [-K|--ignoreactivation- skip] [-A|--autobackup {y|n}] [-C|--contiguous {y|n}] [-d|--debug] [-h|--help] [-noudesync] [-ignoremonitoring] [-monitor {y|n}] [-[raid]maxrecoveryrate Rate] [-[raid]minrecoveryrate Rate] [-i|--stripes Stripes [-I|--stripesize StripeSize]] [-l|--extents LogicalExtentsNumber[%(VG|PVS|FREE)] | -L|--size LogicalVolumeSize[bBSkKmMgGtTpPeE]] | -V|--virtualsize VirtualSize[bBSkKmMg- GtTpPeE]] [-M|--persistent {y|n}] [-m|--minor minor] [-m|--mirrors Mirrors] [-nosync] [-m|--mirrorlog {disk|core|mirrored}] [-c|--corelog] [-R|--region- size MirrorLogRegionSize]] [-n|--name LogicalVolume{Name|Path}] [-p|--permission {r|w|rw}] [-r|--readahead {ReadAheadSectors|auto|none}] [-t|--test] [-T|--thin [-c|--chunksize ChunkSize[bBSkKmMg]]] [-dis- cards {ignore|nopassdown|passdown}] [-s|--poolmetadatasize MetadataVolumeSize[bBSkKmMg]]] [-[poolmetadataspare {y|n}]] [-i|--thinpool Thinpool LogicalVolume{Name|Path}] [-s|--snapshot [VolumeGroup{Name|Path}] | ExternalOriginLogicalVolumeName]] [-t|--type SegmentType] [-v|--verbose] [-Z|--zero {y|n}]] VolumeGroup{Name|Path} [/ThinPoolLogicalVolumeName] [PhysicalVolumePath[:PE|PE]]...

lvcreate [-l|--extents LogicalExtentsNumber[%(VG|FREE|ORIGIN)] | -L|--size LogicalVolumeSize[bBSkKmMgGtTpPeE]] [-c|--chunksize ChunkSize[bBSkK]] [-noudesync] [-ignoremonitoring] [-monitor {y|n}] [-n|--name SnapshotLogicalVolume{Name|Path}] [-s|--snapshot [VolumeGroup{Name|Path}] | OriginalLogicalVolumeName -V|--virtualsize VirtualSize[bBSkKmMgGtTpPeE]]

Manual page lvcreate(8) line 1 (press h for help or q to quit)
This is not discoverable
Goal: Plays well with others
Allows management via other tools and reacts to them, for example the command line, Spacewalk, or puppet
Video of add/remove user via command line
Goal: Lightweight low footprint
Starts on demand, no overhead when not in use
Headless, runs in a browser
Make it Stop!
We want cockpit to stop when not in use
Make it Stop!

(when not in use)
So stop your dbus configuration services when not in use
Currently possible, although hard to get rid of all races exiting a dbus service
But with kdbus this is totally supported. So long term we want this to be part of everything
Goal: Ad-hoc
No infrastructure prerequisite
We don't force you to setup some other services or infrastructure before using cockpit
Goal: Use infrastructure well
But if you have infrastructure like a domain, we want to use it properly.
Goal: Domain authentication
Fallbacks for non-domain case will be present
Goal: Server roles
For example if no domain is present we want to help the admin set one up with FreeIPA for example.
Non-goal: Configuration management
* Puppet/Salt and the like are excellent centralized configuration management tools
* Notify admins when a system has them in use
* Ideally avoid changing puppet-managed state
* Help discover how to configure a puppet master
Goal: Opinionated when possible
If there's a best practice, we want to help people discover it. There's a hundred ways to do it the other ways, cockpit doesn't have to do them all.
Non-goal: Yet another API (hint: OpenLMI)
Help make it better
Goal: Pluggable UI
Not monolithic, will have a modular architecture and is extensible
Cockpit is open source

LGPL v2+

Code: github.com/cockpit-project

cockpit-devel@lists.fedorahosted.org
Questions?

http://cockpit-project.org
#cockpit on Freenode

Credits:
Done in pinpoint
d10n2000 on Flickr, bigfez on Flickr, dalbera on Flickr
axeman3d on Flickr, toddmccann on Flickr