Tcl and Cloud Computing Automation

Tclcloud, Tclwinrm & Cato

Patrick Dunnigan
Chief Architect, Cloud Sidekick
cloudsidekick.com
@CloudSidekick

- TclCloud - Tcl api for AWS public cloud / Ecualyptus private cloud
- Tclwinrm - Tcl api for the Windows remote management protocol
- Cloud Sidekick Cato - open source framework for cloud and enterprise IT automation

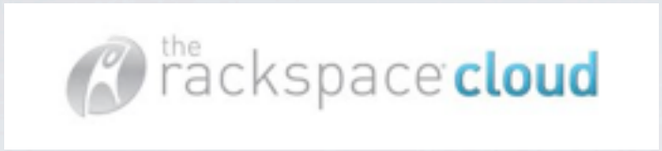
agenda

show of hands

- shared computing resources, no capital expense (public)
- on demand, pay only for what you use
- scales massively
- elastic, grows and shrinks
- api / web service driven

cloud principles

public cloud



hybrid

private cloud

iaas - infrastructure as a service

- Tclcloud - web service cloud api for managing and automating cloud environments

- not abstracted: direct correlation to cloud vendor api actions; a wrapper around the web service client

- currently supports Amazon AWS, Eucalyptus

- near future releases: VMWare vCloud, Cloud.com, Opsource, etc.

- home: https://github.com/cloudsidekick/tclcloud

Tclcloud

- bootstrap: run virtual servers (EC2 instances), terminate instances
- network: (EC2 security groups, VPC - virtual private cloud, elastic ip), define ELBs (Elastic Load Balancers)
- monitor: EC2 instances, failover database master to slave, applications
- administer: add, remove, snapshot EBS volumes and devices, create AMI images
- high performance computing: run Elastic Map Reduce (Hadoop) jobs

Tclcloud uses

```
package require tclcloud
set access_key DKE65EEFEETGHJJS7E782D
set secret_key f6gbUsddYr62MFJRi93wWHd8el3drcjE5BnTZf

set ::tclcloud::debug 1
set conn [::tclcloud::connection new $access_key $secret_key]

lappend args Filter.1.Name architecture Filter.1.Value.1 x86_64 Filter.2.Name root-device-type Filter.2.Value.1 ebs \
       Filter.3.Name name Filter.3.Value.1 {bitnami-wordpress-3.2.1-1-linux-x64-ubuntu-10.04-ebs}

set result [$conn call_aws ec2 {} DescribeImages $args]

puts $result
$conn destroy
```

- aws api access key
- aws secret key

- set debug for verbose
- define connection object

- tcl list of arguments passed to aws

- DescribeImages api call

- print xml results
- destroy connection object

```
<?xml version="1.0" encoding="UTF-8"?>
<DescribeImagesResponse xmlns="http://ec2.amazonaws.com/doc/2011-07-15/">
   <requestId>e57a67e7-4ae7-453d-907d-c4ab6b33c38f</requestId>
   <imagesSet>
      <item>
         <imageId>ami-3f7dbd56</imageId>
         <imageLocation>979382823631/bitnami-wordpress-3.2.1-1-linux-x64-ubuntu-10.04-ebs</imageLocation>
         <imageState>available</imageState>
         <imageOwnerId>979382823631</imageOwnerId>
         <isPublic>true</isPublic>
         <architecture>x86_64</architecture>
         <imageType>machine</imageType>
         <kernelId>aki-427d952b</kernelId>
         <name>bitnami-wordpress-3.2.1-1-linux-x64-ubuntu-10.04-ebs</name>
         <description>BitNami Wordpress 3.2.1-1  EBS AMI</description>
         <rootDeviceType>ebs</rootDeviceType>
         <rootDeviceName>/dev/sda1</rootDeviceName>
         <blockDeviceMapping>
            <item>
               <deviceName>/dev/sda1</deviceName>
               <ebs>
                  <snapshotId>snap-1c85127c</snapshotId>
                  <volumeSize>10</volumeSize>
                  <deleteOnTermination>true</deleteOnTermination>
               </ebs>
            </item>
         </blockDeviceMapping>
         <virtualizationType>paravirtual</virtualizationType>
         <hypervisor>xen</hypervisor>
      </item>
   </imagesSet>
</DescribeImagesResponse>
```

- image identifier

- 64 bit

- wordpress image
- elastic block store

- 10 gig image

... continued ...

```
package require tdom

proc strip_namespaces {xml} {
        set xmldoc [dom parse -simple $xml]
        set root [$xmldoc documentElement]
        set xml_no_ns [[$root removeAttribute xmlns] asXML]
        $root delete
        $xmldoc delete
        return $xml_no_ns
}

proc get_xpath_value {xml path} {
        set xmldoc [dom parse -simple $xml]
        set root [$xmldoc documentElement]
        set value [$root selectNodes string($path)]
        $root delete
        $xmldoc delete
        return $value
}

set result [strip_namespaces $result]
set imageId [get_xpath_value $result //imageId]

puts "The image is $imageId"
```

- tdom to parse xml

- a helper procedure using tdom to remove the namespace from xml (make it easier)

- a helper procedure using tdom to get data out of xml (simple example, could be more efficient)

- call the strip namespaces
- get the image id using xpath

- print image id

- use xpath query to extract results from returned data
- tdom works well for this
- xpath tutorial: http://www.w3schools.com/xpath/

samples/sample_tclwinrm.tcl

DescribeImages

get imageId

RunInstances

get instanceId

check status in loop

DescribeInstances

is running?

get external address

wait for boot to finish

http://external.address

end

demo example

- Microsoft WinRM - Windows Remote Management
- SOAP based web service
- administration and automation capabilities without RPC / DCOM
- Server 2008+

- winRS - Windows Remote Shell
- win command line tools standard
- still, linux clients limited

winrm

- Tclwinrm - client-side Tcl extension for interaction with the WinRM service
- remotely manage Windows servers from linux (yay!)
- Windows command line commands, PowerScript
- Microsoft Server Core 2008 R2
- future releases: ssl, digest / kerberos auth
- enables automation in cloud & data center
- https://github.com/cloudsidekick/tclwinrm

Tclwinrm

- install and configure software
- remote administration
- monitoring: server and application
- run central PowerScript library
- did I mention manage Windows from linux?

Tclwinrm uses

```
package require tclwinrm
package require base64

puts "local machine is $::tcl_platform(os)"

set address ec2-107-20-119-132.compute-1.amazonaws.com
set port 5985
set user administrator
set pass p@ssw0rd

set conn [tclwinrm::connection new http $address $port $user $pass]

set script {$strComputer = $Host
        $RAM = WmiObject Win32_ComputerSystem
        $MB = 1048576
        "Win host Installed Memory: " + [int]($RAM.TotalPhysicalMemory /$MB) + " MB"
}
set command "powershell -encodedcommand [::base64::encode -wrapchar "" [encoding convertto unicode $script]]"
set result [$conn rshell $command 120 0]
puts \n$result

set command {dir c:\ }
set result [$conn rshell $command 120 0]
puts $result
exit
```

- base64 required for powershell scripts

- print local os

- port 5985 standard on 2008 R2
- user with winrm connect rights

- define connection object

- a simple powershell script

- base64 encode script
- call remote shell on win server
- print result

- a simple dos command, no encoding
- call remote shell
- print result

```
local machine is Linux

Win host Installed Memory: 615 MB

 Volume in drive C has no label.
 Volume Serial Number is 8E0B-09AF

 Directory of c:\

07/14/2009  03:34 AM    <DIR>          PerfLogs
11/13/2010  12:13 PM    <DIR>          Program Files
11/13/2010  11:49 AM    <DIR>          Program Files (x86)
11/13/2010  04:16 AM                 9 query
06/27/2011  03:35 PM    <DIR>          Users
06/27/2011  03:04 PM    <DIR>          Windows
               1 File(s)              9 bytes
               5 Dir(s)  11,147,808,768 bytes free
```

- local os

- result from powershell script

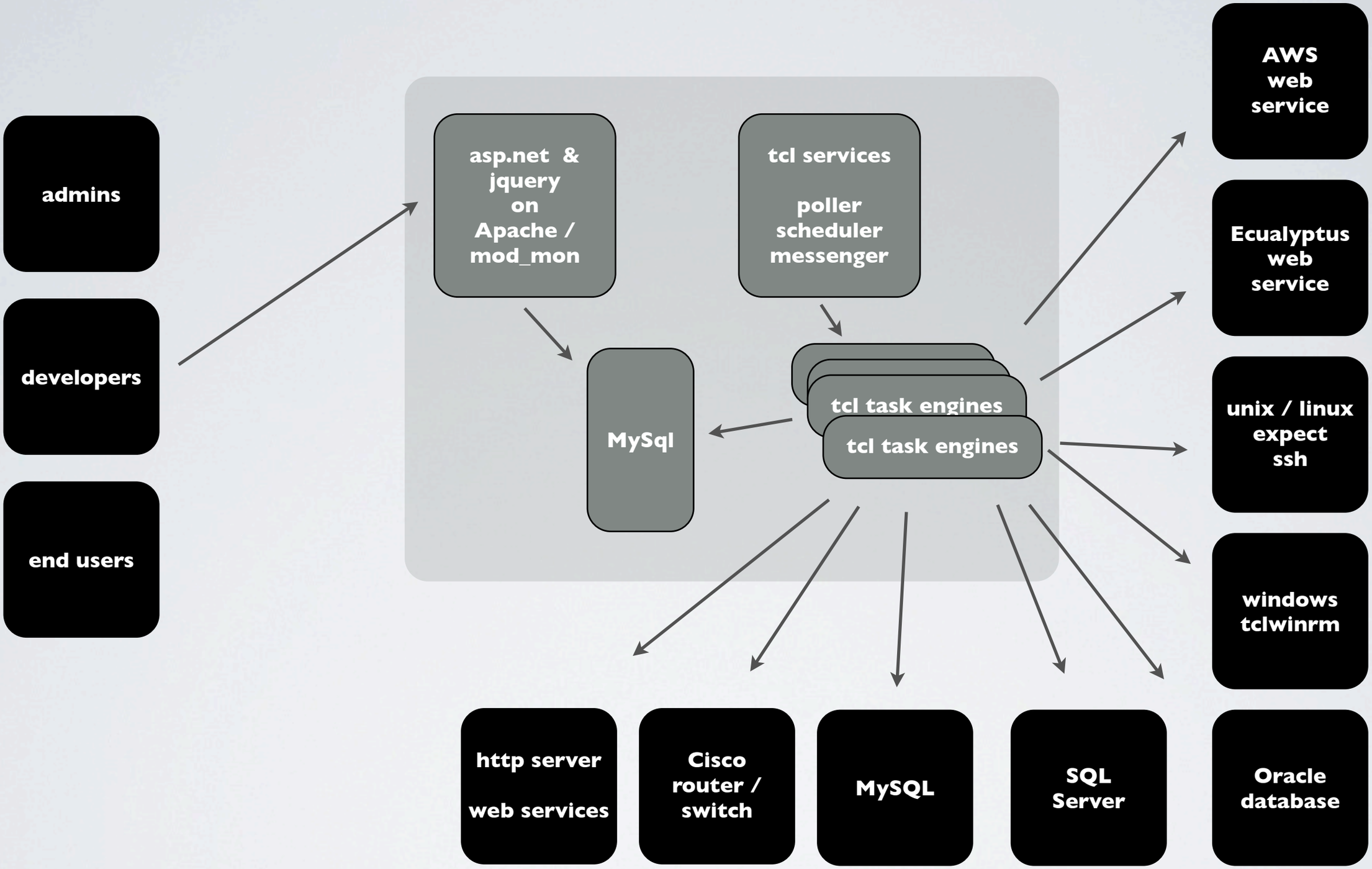
- result from dos command

- open source (Apache 2)
- architecture
  - browser based UI (.net on linux / mono)
  - database (MySql)
  - automation engine (Tcl)
- automation toolkit
- central script repository (Tasks)
- build Tasks drag and drop
- run now or scheduled
- supports cloud apis (Tclcloud), ssh (Expect), databases (OraTcl, mysqltcl, tcltds), windows (Tclwinrm)

Cloud Sidekick - Cato

- starting / stopping virtual servers, bootstrapping
- deploying / upgrading application software
- moving data on databases / filesystems
- monitoring application performance
- release builds / testing
- refreshing data
- configuring networking, "virtual private clouds"
- enabling hybrid clouds (public to private / data center)
- database administration
- scaling applications
- backup filesystems / databases
- enable user self service

Cato uses

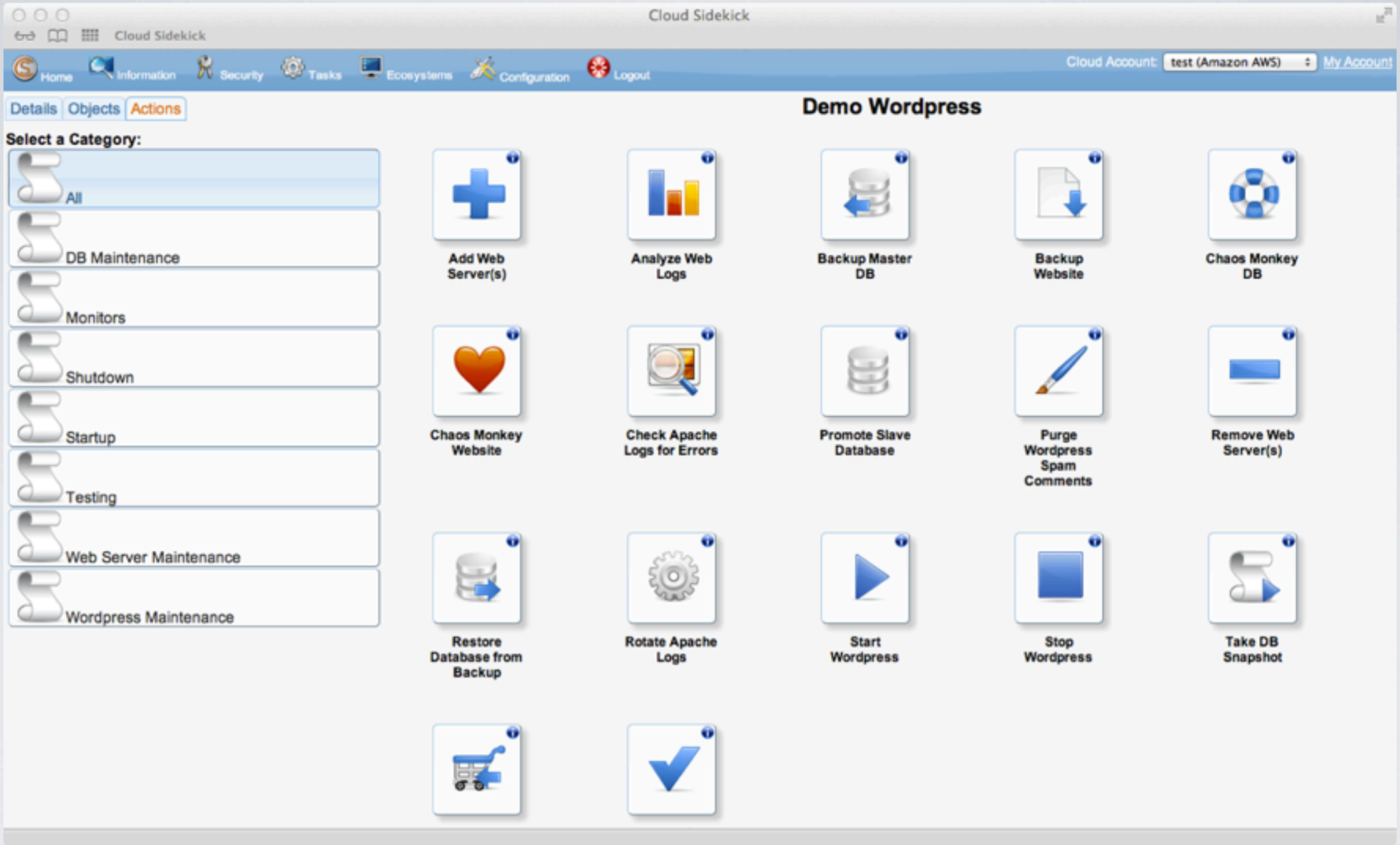


Cato architecture

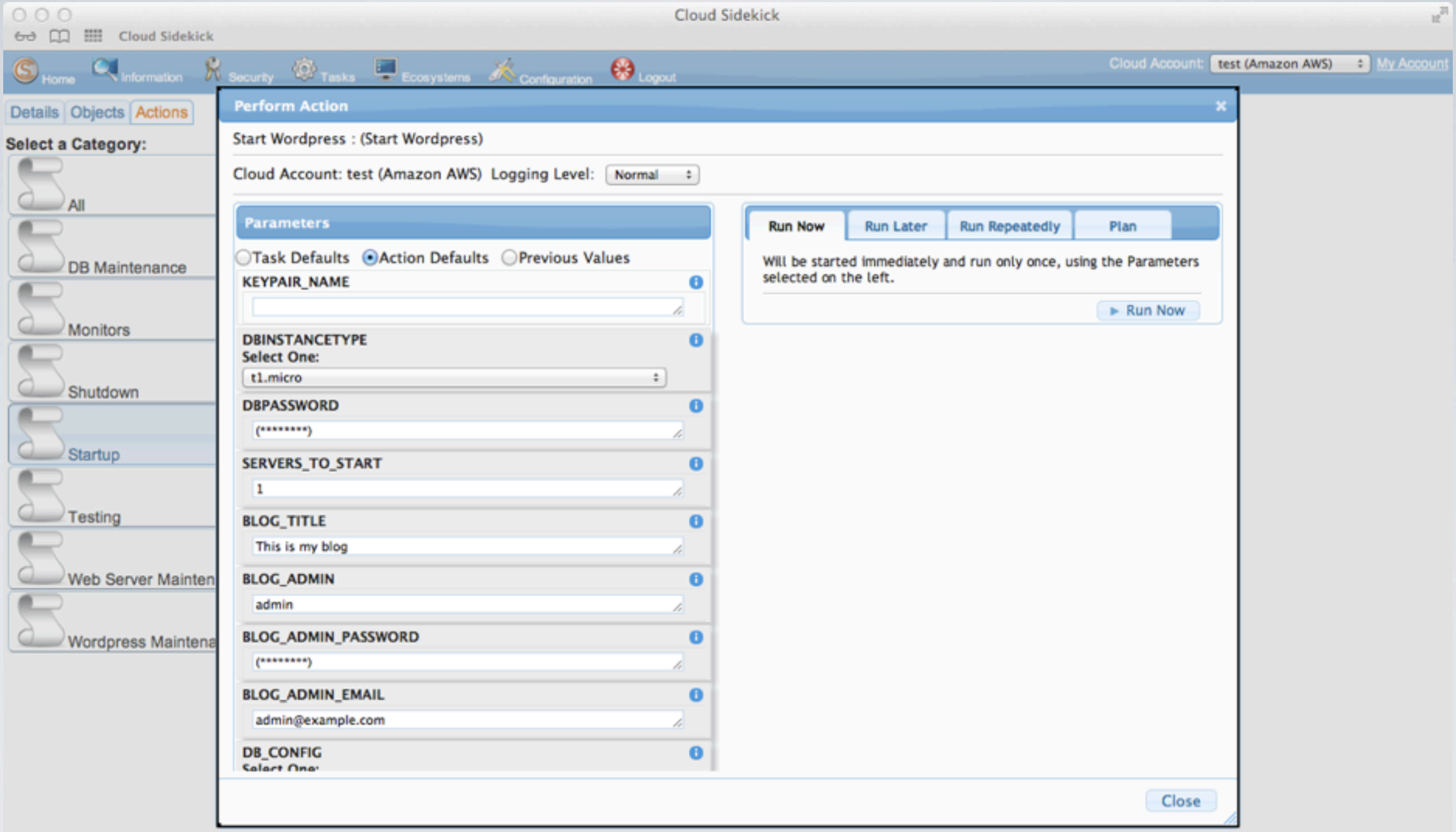
- more clouds apis: Cloud.com CloudStack, VmWare vSphere, Rackspace, Opsource, OpenStack, etc.
- interfaces: rabbitMQ messaging bus, Puppet / Chef integration, RightScale
- packaging apt-get Ubuntu / Debian, rpm Redhat, etc.
- action / task template library, community based

Cato future

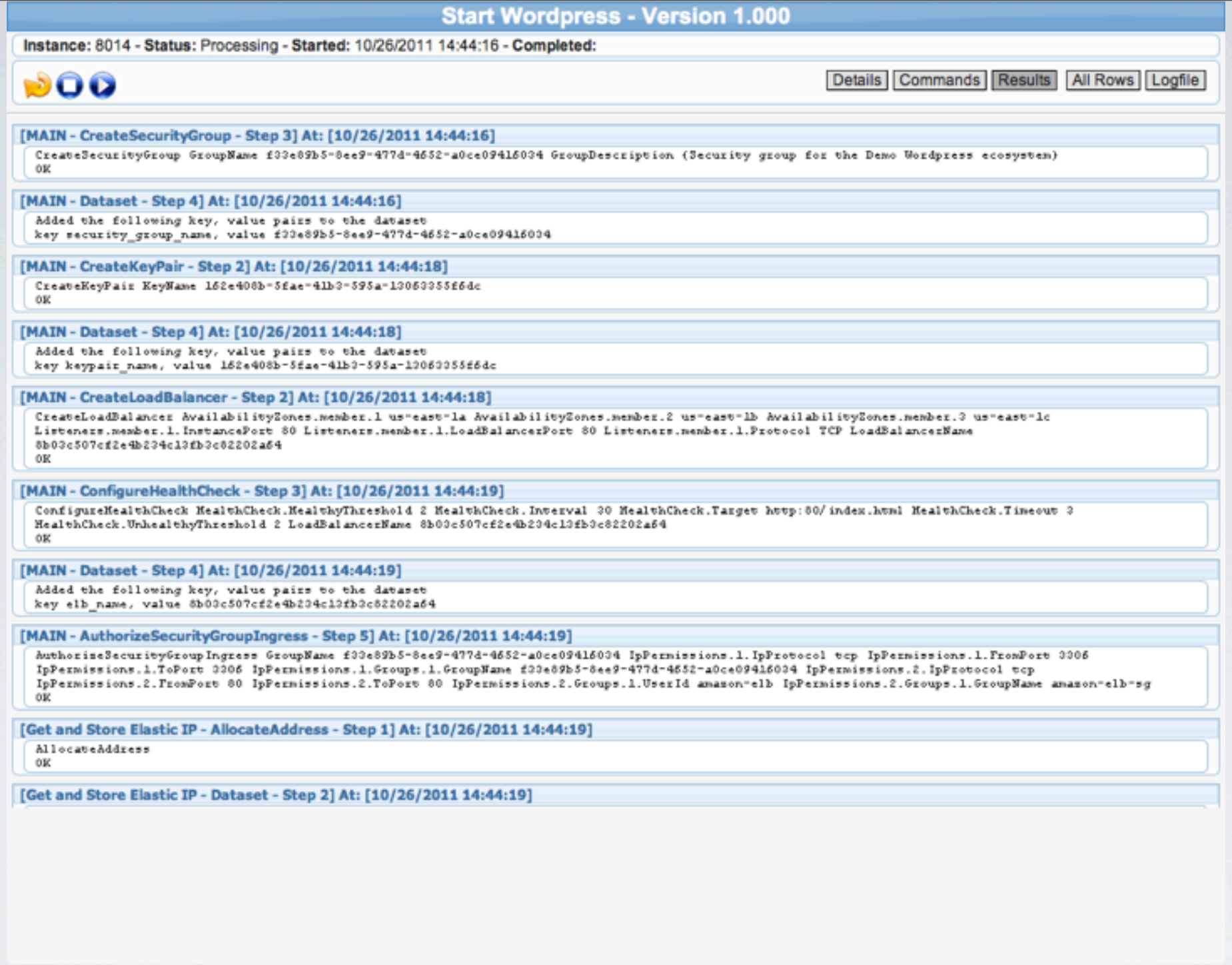
Cato demo



create actions to enable user self service



a user launches an action



action launched, view log in browser...

Cloud Sidekick

Start Wordpress - Version 1.000

Instance: 8014 - **Status:** Processing - **Started:** 10/26/2011 14:44:16 - **Completed:**

Details | Commands | Results | All Rows | Logfile

[MAIN - New Connection - Step 2] At: [10/26/2011 14:44:51] On Connection: [i-6bfb0508]

```
Connecting to (ubuntu@i-6bfb0508)...
ssh connection to host=10.206.98.59, user=ubuntu established.
New connection named CONN1 to asset ubuntu@i-6bfb0508 created with a connection type ssh - ec2
```

[MAIN - Sleep - Step 3] At: [10/26/2011 14:45:33]

```
Sleeping (60) seconds...
```

[MAIN - Command Line - Step 4] At: [10/26/2011 14:46:33] On Connection: [CONN1]

```
sudo mysql -u root -p[redacted]
Welcome to the MySQL monitor.  Commands end with ; or g.
Your MySQL connection id is 34
Server version: 5.1.54-1ubuntu4 (Ubuntu)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL v2 license

Type 'help;' or 'h' for help. Type 'c' to clear the current input statement.

mysql>
```

[MAIN - Command Line - Step 5] At: [10/26/2011 14:46:33] On Connection: [CONN1]

```
create database wordpress;
Query OK, 1 row affected (0.00 sec)

mysql>
```

[MAIN - Command Line - Step 6] At: [10/26/2011 14:46:33] On Connection: [CONN1]

```
grant all privileges on wordpress.* to "wordpress"@"%" identified by "wordpress";
Query OK, 0 rows affected (0.00 sec)

mysql>
```

[MAIN - Command Line - Step 7] At: [10/26/2011 14:46:33] On Connection: [CONN1]

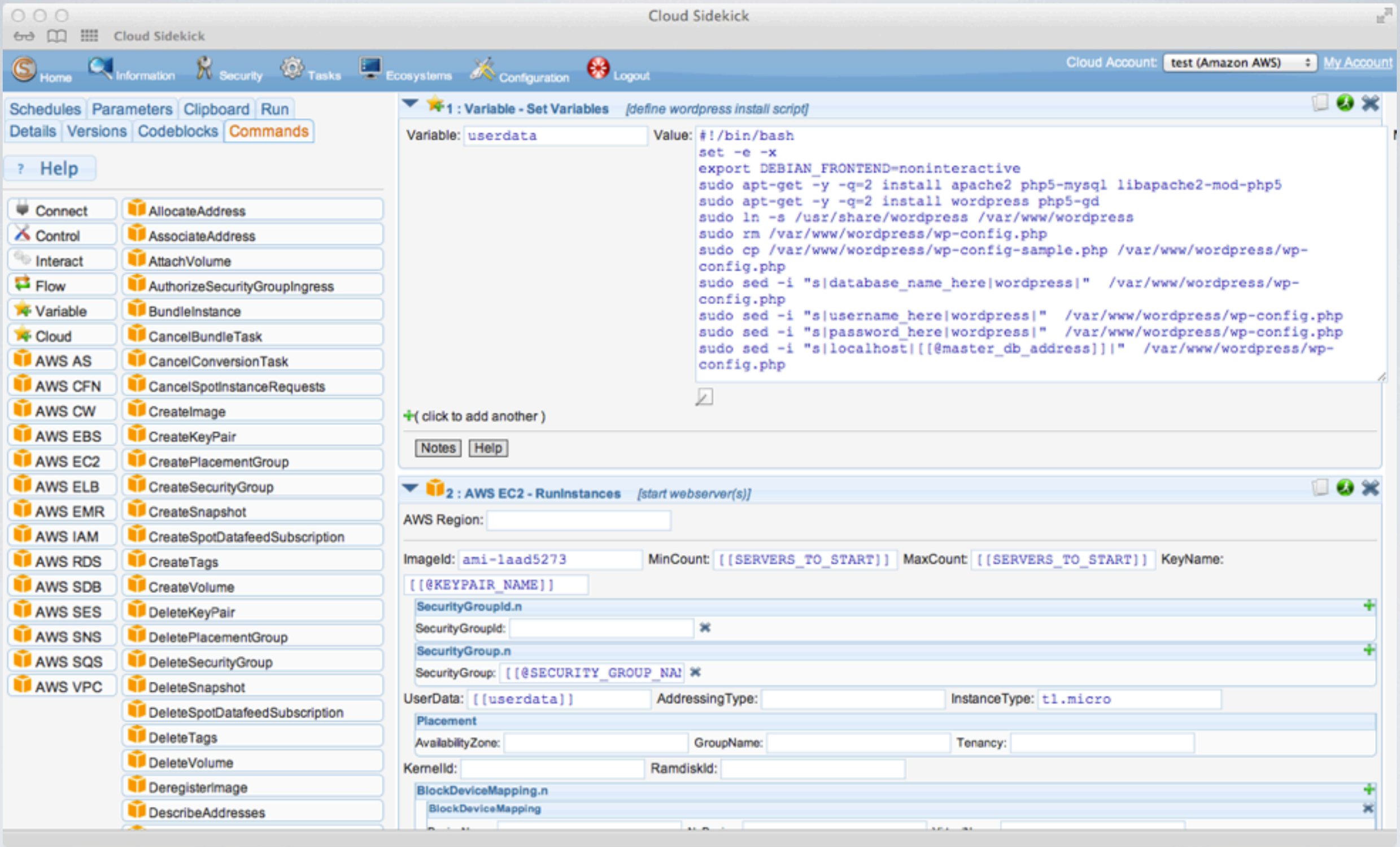
```
flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

[MAIN - Command Line - Step 8] At: [10/26/2011 14:46:33] On Connection: [CONN1]

```
exit;
```

... log continued, linux server started and ssh'ing in



action = task, task edit interface
drag and drop

company site:

http://www.cloudsidekick.com

community site:

http://community.cloudsidekick.com

github:

https://github.com/cloudsidekick/cato

wiki, forum, bugs

http://projects.cloudsidekick.com/projects/cato

check it out, get involved

thanks

Patrick Dunnigan
Chief Architect, Cloud Sidekick
patrick.dunnigan@cloudsidekick.com
twitter @CloudSidekick