SOFTLAYER®
an IBM Company

你好
(nǐ hǎo)

image credit: http://goo.gl/3bYXWr

SoftLayer eCPI for BOSH

PIVOTAL LABS

dr.max @maximilien (SVL team)
Zhou (Tom) Xing and
Hua (Edward) Zhang (China team)

ibm cloud labs
maximilien.org

v0.3.14
March 14th, 2015

π day

3/14/15 9:26 am

agenda

- **introduction**
  - new **µBOSH** (micro BOSH) CLI
  - external BOSH Cloud Provider Interfaces (**eCPIs**)
- **SoftLayer eCPI**
  - considerations and goals
  - overview of approach
  - demo, test, and code
  - what next? and future work

introduction

- BOSH is **THE** cloud shell for Cloud Foundry
  - agnostic (supports any cloud), e.g., AWS, OpenStack, SoftLayer, Google CC, Azure
  - rolling deployments, resilient, flexible, supports non-connected deployments, …
  - allows repetitive, predictable, and consistent deployments of large cloud software systems
- BOSH used for **all Cloud Foundry deployments** (development, testing, and prod)
- BOSH is **not perfect**
  - monolithic, in that one package for all components (director, agent, CPIs, CLIs, etc)
  - written in Ruby (all parts) which is a great language but not appropriate for some tasks
  - can be complicated, reflecting the breath of its usage and cloud software systems

µBOSH

- µBOSH (micro BOSH) is the bootstrap mechanism
- new rewrite in Golang and streamlined of the CLI
- key **features** and **design points**

- one binary, yeah!
- support all of the old BOSH micro commands
- needs external CPIs to work
- supports **one VM** deployments (for now)

μBOSH (*cont.*)

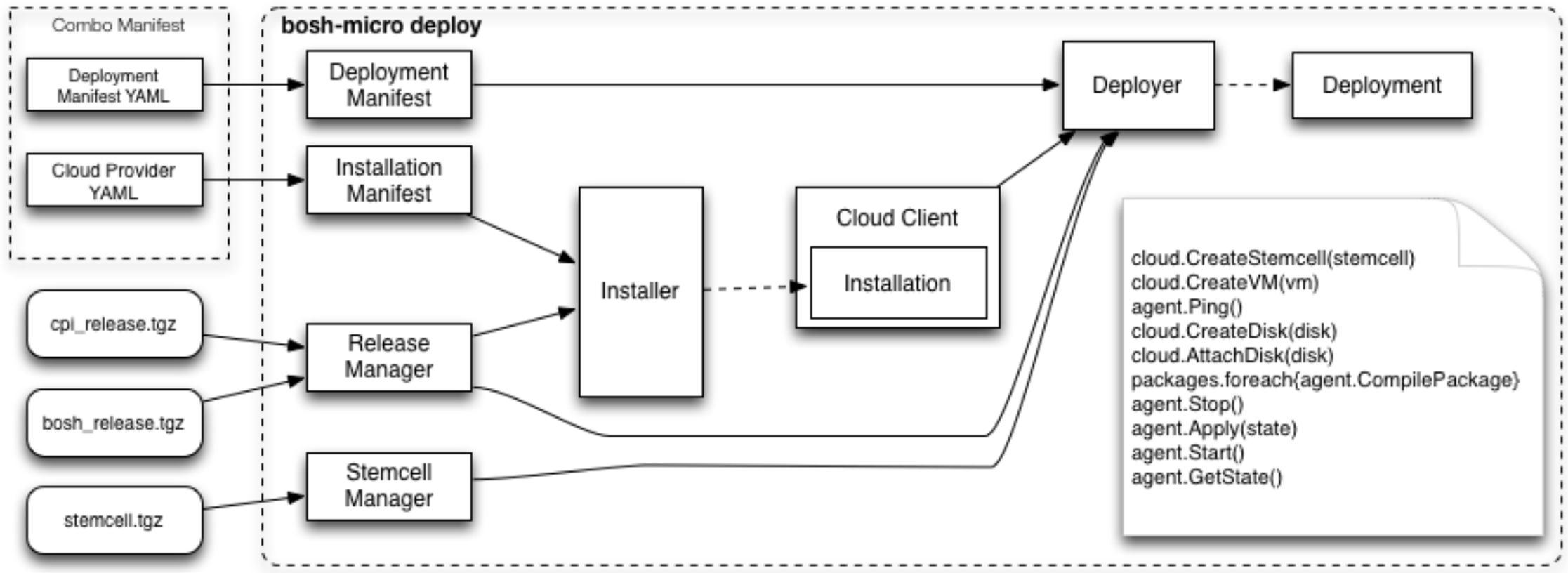
Create deployment manifest

This file will be used by the BOSH Micro CLI to deploy Micro BOSH.

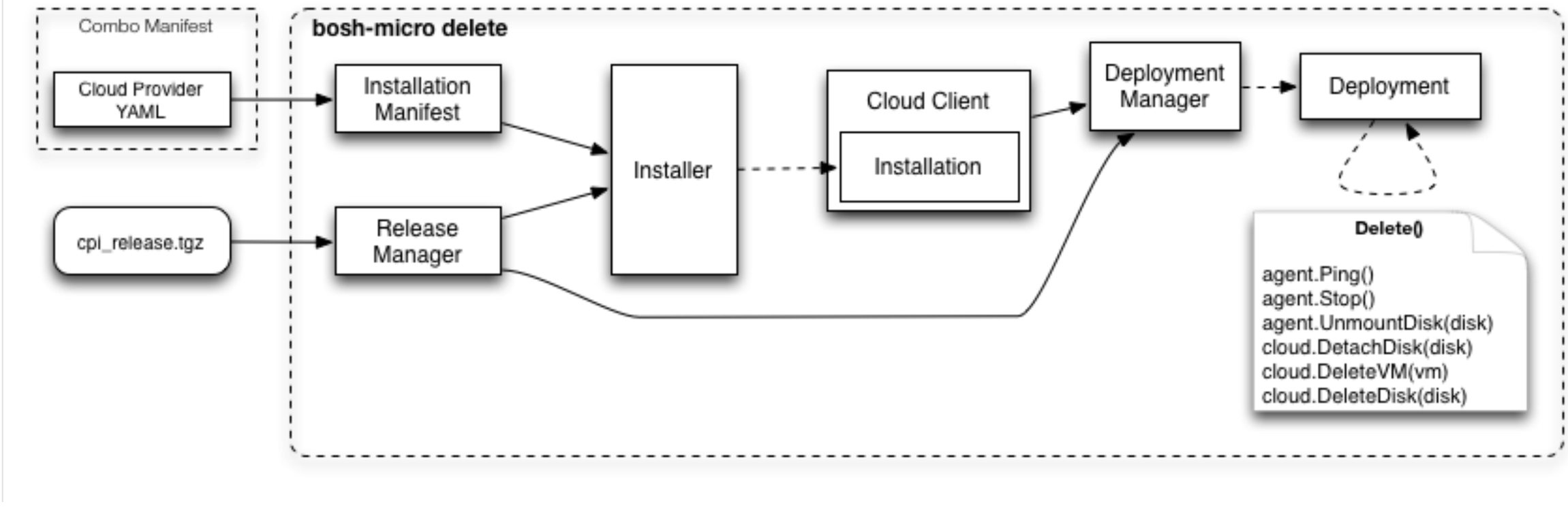
Example deployment manifest

```
---
name: micro-bosh

networks:
- name: default
  type: dynamic
  cloud_properties:
    subnet: AWS_SUBNET_NAME
- name: vip
  type: vip

resource_pools:
- name: default
  cloud_properties:
    instance_type: AWS_INSTANCE_TYPE
    availability_zone: AWS_AVAILABILITY_ZONE

cloud_provider:
  ssh_tunnel:
    host: MICRO_BOSH_IP
    port: 22
    user: ssh-user
    password: ssh-password
  registry:
    username: registry-user
    password: registry-password
    port: 6901
    host: localhost
  mbus: https://mbus-user:mbus-password@MICRO_BOSH_IP:6868
  properties: # properties that are saved in registry by CPI for the agent
    blobstore:
      provider: local
      path: /var/vcap/micro_bosh/data/cache
    registry:
      username: admin
```

µBOSH (*cont.*)


Combo Manifest
Deployment Manifest YAML
Cloud Provider YAML
bosh-micro deploy
Deployment Manifest
Installation Manifest
cpi_release.tgz
bosh_release.tgz
stemcell.tgz
Release Manager
Stemcell Manager
Installer
Cloud Client
Installation
Deployer
Deployment
cloud.CreateStemcell(stemcell)
cloud.CreateVM(vm)
agent.Ping()
cloud.CreateDisk(disk)
cloud.AttachDisk(disk)
packages.foreach{agent.CompilePackage}
agent.Stop()
agent.Apply(state)
agent.Start()
agent.GetState()

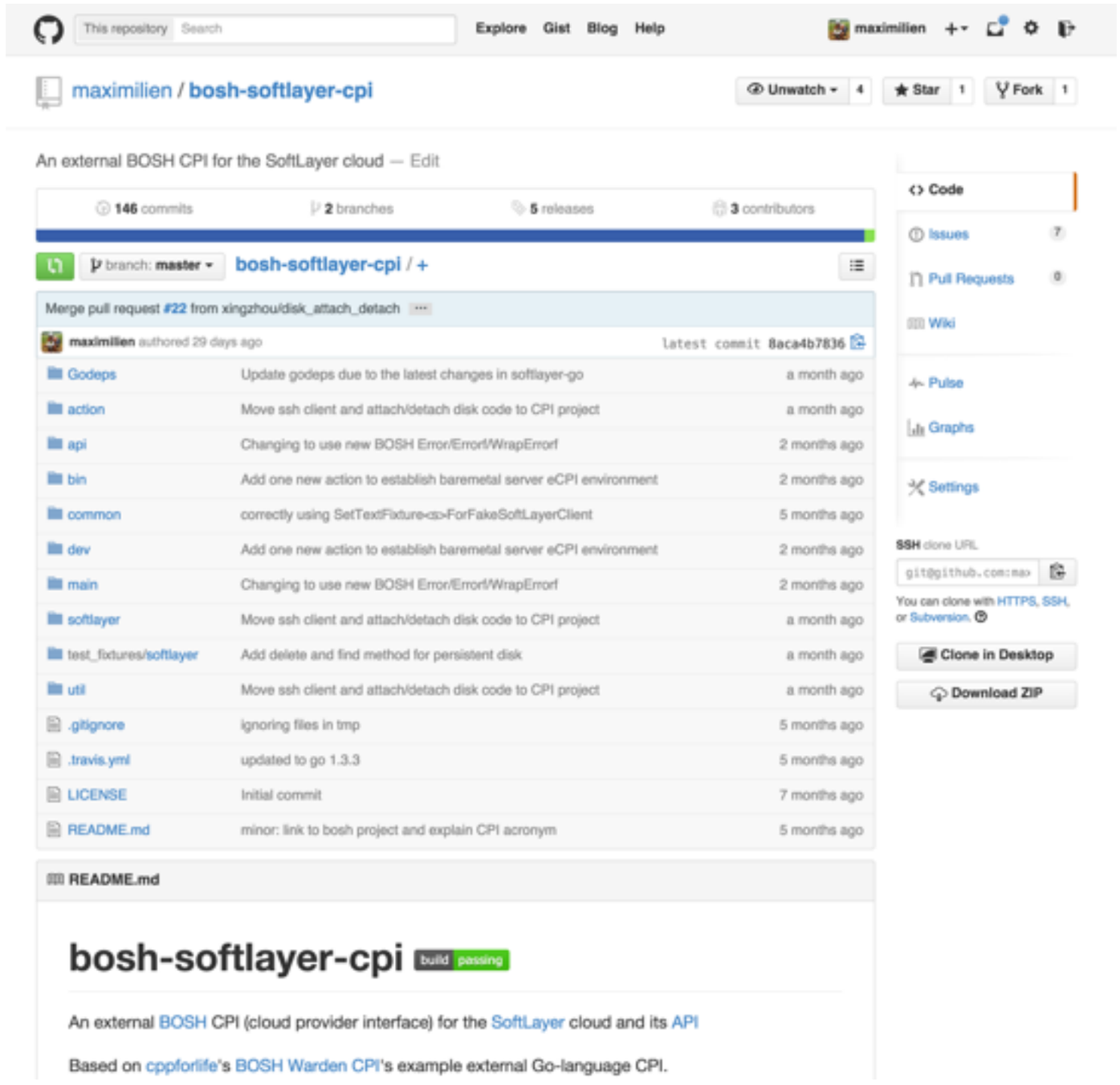
µBOSH (*cont.*)


Combo Manifest
Cloud Provider YAML
cpi_release.tgz
bosh-micro delete
Installation Manifest
Release Manager
Installer
Cloud Client
Installation
Deployment Manager
Deployment
Delete()
agent.Ping()
agent.Stop()
agent.UnmountDisk(disk)
cloud.DetachDisk(disk)
cloud.DeleteVM(vm)
cloud.DeleteDisk(disk)

BOSH eCPIs

- interface between **BOSH and targeted cloud** (IaaS)
- amounts to implementing about **13 methods** (create_vm, delete_vm, create_disk, attach_disk, detach_disk, and so on)
- some of the methods get passed properties which allow customization
- externalizing CPIs from BOSH code base allows
  - different evolution of BOSH and CPIs (no need to fork!)
  - experimentation for new CPI features (via new properties)

BOSH eCPIs (*cont.*)



BOSH eCPIs (*cont.*)

```
49
50      return concreteFactory{
51              availableActions: map[string]Action{
52                      // Stemcell management
53                      "create_stemcell": NewCreateStemcell(stemcellFinder),
54                      "delete_stemcell": NewDeleteStemcell(stemcellFinder),
55
56                      // VM management
57                      "create_vm":          NewCreateVM(stemcellFinder, vmCreator),
58                      "delete_vm":          NewDeleteVM(vmFinder),
59                      "has_vm":             NewHasVM(vmFinder),
60                      "reboot_vm":          NewRebootVM(vmFinder),
61                      "set_vm_metadata":    NewSetVMMetadata(vmFinder),
62                      "configure_networks": NewConfigureNetworks(vmFinder),
63
64                      // Disk management
65                      "create_disk": NewCreateDisk(diskCreator),
66                      "delete_disk": NewDeleteDisk(diskFinder),
67                      "attach_disk": NewAttachDisk(vmFinder, diskFinder),
68                      "detach_disk": NewDetachDisk(vmFinder, diskFinder),
69
70                      "establish_bare_metal_env": NewEstablishBareMetalEnv(bmCreator, bmFinder),
71
72                      // Not implemented (disk related):
73                      //   snapshot_disk
74                      //   delete_snapshot
75                      //   get_disks
76
77                      // Not implemented (others):
78                      //   current_vm_id
79                      //   ping
80              },
81      }
82  }
```

SoftLayer eCPIs: what? how? when?

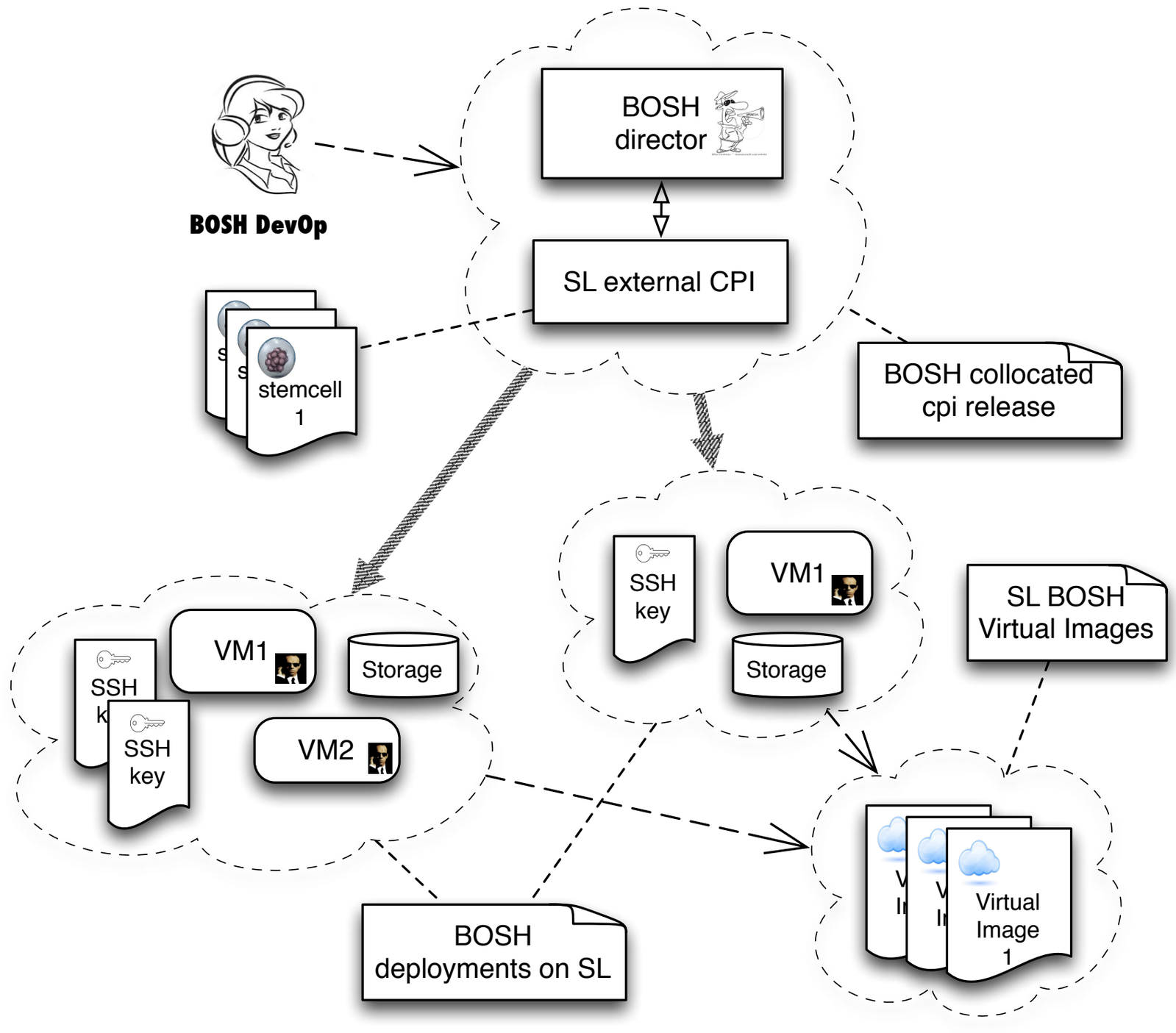
- Cloud provider interface (**CPI**) for SoftLayer (SL)
- Built from **scratch**:
  - 100% in Go language (**Golang**)
  - BOSH **external CPI**
- **Currently**: 4.5 KLOC (SL client) + 5.4 KLOC (CPI) ~= 10**KLOC****

** 1000s of line of code using cloc OSS tool

considerations and goals

- **SL API is massive**, lots of services and datatypes
- **No existing Golang clients for SL** are available
- Created a **simplified Golang client** for SL
  - only implement services and datatypes needed for CPI
  - easily (manually) expendable to support more (to all) of SL APIs
- Use BOSH "**light stemcells**" as pointer to SL VGDTB
- Reuse packer-bosh project for provisioning BOSH agent for testing
- New bosh-softlayer-stemcell project for new SL stemcells

overview of approach - architecture diagram



overview of approach - *light stemcell - development*

- Uses **packer-builder-softlayer** and **packer-bosh** OSS projects
- Steps before **creating light stemcell:**

1. use an existing SL base virtual image
2. create instance of that virtual image
3. provision BOSH agent and artifacts using **packer-bosh**
4. capture instance as new SL *Virtual Image*

- Create **light stemcell using Virtual Image ID and UUID** from 4)

overview of approach - softlayer-go

- Create **minimal Golang client** for SoftLayer cloud

  1. `Client` interface and implementation to invoke SL via raw HTTP calls

  2. `GetSoftLayer_Account_Service()`, `GetSoftLayer_Virtual_Guest_Service()`, … methods

  3. each service interface implements methods as exposed in SL service, e.g., `CreateObject(…)`

  4. `data_types` package for SL data types, e.g., `Virtual_Guest`, `Virtual_Image`, …

  5. only implement SL services and data types needed for CPI

- **Naming convention follow SL**, except when clashing with Golang

overview of approach - bosh-softlayer-cpi

- Follows new BOSH **external CPI model**
- Inspired by **bosh-warden-cpi** from Pivotal
  - Golang CLI executable with commands and actions
  - actions match each of the 13 methods for any BOSH CPI
  - pre- and post- processing of actions
- Uses **light stemcell** wrapping **SL Virtual Image ID** and **UUID** for cloud properties
- Create **co-located BOSH release with CPI** to deploy
- **CPI release is own project** for isolation and history: bosh-softlayer-cpi-release

demo, test, and code

demo (*cont.*)

```
Default
    softlayer: Started installing dependencies > Package wget
    softlayer: Finished installing dependencies > Package wget
    softlayer: Started installing dependencies > Package gdb
    softlayer: Finished installing dependencies > Package gdb
    softlayer: Started installing dependencies > Package sysstat
    softlayer: Finished installing dependencies > Package sysstat
    softlayer: Started installing dependencies > Package rsync
    softlayer: Finished installing dependencies > Package rsync
    softlayer: Started installing dependencies > Package iptables
    softlayer: Finished installing dependencies > Package iptables
    softlayer: Started installing dependencies > Package tcpdump
    softlayer: Finished installing dependencies > Package tcpdump
    softlayer: Started installing dependencies > Package traceroute
    softlayer: Finished installing dependencies > Package traceroute
    softlayer: Started updating bosh agent > Placing binaries
    softlayer: Finished updating bosh agent > Placing binaries
    softlayer: Started updating bosh agent > Placing configuration files
    softlayer: Finished updating bosh agent > Placing configuration files
    softlayer: Started updating bosh agent > Registering monit service
    softlayer: Finished updating bosh agent > Registering monit service
    softlayer: Started updating bosh agent > Registering agent service
    softlayer: Finished updating bosh agent > Registering agent service
==> softlayer: Preparing for capturing the instance image. Image snapshot type is 'standard'.
==> softlayer: Getting block devices for instance (id=273954fc-5af2-46ee-b6c4-9311c083c308)
==> softlayer: Will caputure standard image using these block devices: [10514910]
==> softlayer: Waiting for image (2fc2e5eb-397d-429d-9d1d-9753f7051eaa) to finish its creation...
==> softlayer: Waiting for the instance to have no active transactions before destroying it...
==> softlayer: Destroying instance...
==> softlayer: Deleting temporary ssh key...
Build 'softlayer' finished.

==> Builds finished. The artifacts of successful builds are:
--> softlayer: ams01::2fc2e5eb-397d-429d-9d1d-9753f7051eaa (packer-ubuntu-2015-01-12T04:14:19Z)

 |2.1.5| wifi-3-220 in ~/workspace/bosh-softlayer-cpi-release/stemcell
±  |master ✗| → bosh deploy
```

demo

```
---
name: bosh-softlayer-ubuntu
version: '1235'
bosh_protocol: 1
sha1: 3402bea34a41d5771901287ad8075d41c1a2c11a
cloud_properties:
  name: bosh-softlayer-ubuntu
  version: '1235'
  infrastructure: softlayer
  architecture: x86_64
  virtual-disk-image-id: 380298
  virtual-disk-image-uuid: "2fc2e5eb-397d-429d-9d1d-9753f7051eaa"
  datacenter-name: "ams01"
~
~
~
~
~
```

demo

```
Default

Processing deployment manifest
------------------------------
Getting deployment properties from director...
Compiling deployment manifest...
Please review all changes carefully

Deploying
---------
Deployment name: `dummy.yml'
Director name: `bosh-in-softlayer'
Are you sure you want to deploy? (type 'yes' to continue): yes

Director task 5
  Started preparing deployment
  Started preparing deployment > Binding deployment. Done (00:00:00)
  Started preparing deployment > Binding releases. Done (00:00:00)
  Started preparing deployment > Binding existing deployment. Done (00:00:00)
  Started preparing deployment > Binding resource pools. Done (00:00:00)
  Started preparing deployment > Binding stemcells. Done (00:00:00)
  Started preparing deployment > Binding templates. Done (00:00:00)
  Started preparing deployment > Binding properties. Done (00:00:00)
  Started preparing deployment > Binding unallocated VMs. Done (00:00:00)
  Started preparing deployment > Binding instance networks. Done (00:00:00)
     Done preparing deployment (00:00:00)

  Started preparing package compilation > Finding packages to compile. Done (00:00:00)

  Started preparing dns > Binding DNS. Done (00:00:00)

  Started creating bound missing vms > default/0
```

demo

```
Default
D, [2015-01-12T03:41:06.368702 #27269] [task:5] DEBUG -- : (0.000479s) SELECT * FROM "instances" WHERE (("dep
loyment_id" = 1) AND ("job" = 'dummy') AND ("index" = 0)) LIMIT 1
I, [2015-01-12T03:41:06.369007 #27269] [task:5]  INFO -- : ResourcePool `default' - Adding allocated VM (inde
x=0)
I, [2015-01-12T03:41:06.369236 #27269] [task:5]  INFO -- : Binding instance networks
I, [2015-01-12T03:41:06.369433 #27269] [task:5]  INFO -- : Compiling and binding packages
I, [2015-01-12T03:41:06.369543 #27269] [task:5]  INFO -- : Generating a list of compile tasks
I, [2015-01-12T03:41:06.369722 #27269] [task:5]  INFO -- : Job templates `dummy/dummy' need to run on stemcel
l `bosh-softlayer-ubuntu/1234'
I, [2015-01-12T03:41:06.369831 #27269] [task:5]  INFO -- : All packages are already compiled
I, [2015-01-12T03:41:06.369881 #27269] [task:5]  INFO -- : Finished preparing deployment
I, [2015-01-12T03:41:06.369928 #27269] [task:5]  INFO -- : Updating deployment
I, [2015-01-12T03:41:06.370090 #27269] [task:5]  INFO -- : Binding DNS
I, [2015-01-12T03:41:06.370302 #27269] [task:5]  INFO -- : Deleting no longer needed VMs
I, [2015-01-12T03:41:06.370364 #27269] [task:5]  INFO -- : No unneeded vms to delete
I, [2015-01-12T03:41:06.370411 #27269] [task:5]  INFO -- : Deleting no longer needed instances
I, [2015-01-12T03:41:06.370466 #27269] [task:5]  INFO -- : No unneeded instances to delete
I, [2015-01-12T03:41:06.370512 #27269] [task:5]  INFO -- : Updating resource pools
I, [2015-01-12T03:41:06.370627 #27269] [task:5]  INFO -- : Deleting 0 extra VMs
D, [2015-01-12T03:41:06.370678 #27269] [task:5] DEBUG -- : Waiting for tasks to complete
I, [2015-01-12T03:41:06.370762 #27269] [task:5]  INFO -- : Deleting 0 outdated idle VMs
D, [2015-01-12T03:41:06.370822 #27269] [task:5] DEBUG -- : Waiting for tasks to complete
I, [2015-01-12T03:41:06.370923 #27269] [task:5]  INFO -- : Creating 1 missing VMs
D, [2015-01-12T03:41:06.370993 #27269] [task:5] DEBUG -- : Creating new thread
D, [2015-01-12T03:41:06.371131 #27269] [task:5] DEBUG -- : Waiting for tasks to complete
I, [2015-01-12T03:41:06.371680 #27269] [create_missing_vm(default, 0/1)]  INFO -- : Creating missing VM
D, [2015-01-12T03:41:06.372175 #27269] [create_missing_vm(default, 0/1)] DEBUG -- : External CPI sending requ
est: {"method":"create_vm","arguments":["c9fdbb1d-abf0-469a-87c8-f68fd81d4b66","380214",{"domain":"softlayer.
com","startCpus":1,"maxMemory":1024,"datacenter":{"name":"ams01"},"sshKeys":[{"id":136868}]},{"default":{"typ
e":"dynamic","cloud_properties":{},"default":["dns","gateway"],"dns_record_name":"0.dummy.default.dummy.bosh"
}},null,{}],"context":{"director_uuid":"c1719d88-2953-45e0-80a9-b838b3aa1ecb"}} with command: /var/vcap/jobs/
cpi/bin/cpi
D, [2015-01-12T03:41:11.356527 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
D, [2015-01-12T03:41:16.358377 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
D, [2015-01-12T03:41:21.359968 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
D, [2015-01-12T03:41:26.361521 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
D, [2015-01-12T03:41:31.364143 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
D, [2015-01-12T03:41:36.311030 #27269] [task:5-checkpoint] DEBUG -- : (0.000463s) BEGIN
D, [2015-01-12T03:41:36.312559 #27269] [task:5-checkpoint] DEBUG -- : (0.000522s) UPDATE "tasks" SET "checkpo
int_time" = '2015-01-12 03:41:36.309263+0000' WHERE ("id" = 5)
D, [2015-01-12T03:41:36.313355 #27269] [task:5-checkpoint] DEBUG -- : (0.000632s) COMMIT
D, [2015-01-12T03:41:36.365922 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
D, [2015-01-12T03:41:41.368397 #27269] [0x1cfdf2c] DEBUG -- : Renewing lock: lock:deployment:dummy
```

what next?

- Fixing (refactoring)* **bosh-agent** to be generic w.r.t. infrastructure [**done**]
- Complete CPI by adding storage support and testing [**done**]
- Create BOSH release for SL CPI [**done**]
- Deploy using eCPI-enabled microBOSH [**done**]
- Pass all BATs tests [**partially done**]
- Create official BOSH stemcells [**working on it now**]

* I call this refactoring since it's not adding any new functionality but this is major surgery, more than 350 LOC changes

future

- Support and test **SL metal VMs** [**done**]
- Look into supporting **mix pools of VMS** [**partial**]
- Many kinds of **SL storage support** [**partial**]
- Support **Bluemix** in using this new CPI
- Allow Bluemix to **better track** latest **BOSH** & **CF** releases

谢谢你
(thank you)
&
问题?
(xiè xiè, questions?)