Open. Together.

OCP SUMMIT

Open Source firmware
automatized testing

Jean-Marie Verdun
CTO
ITRenew

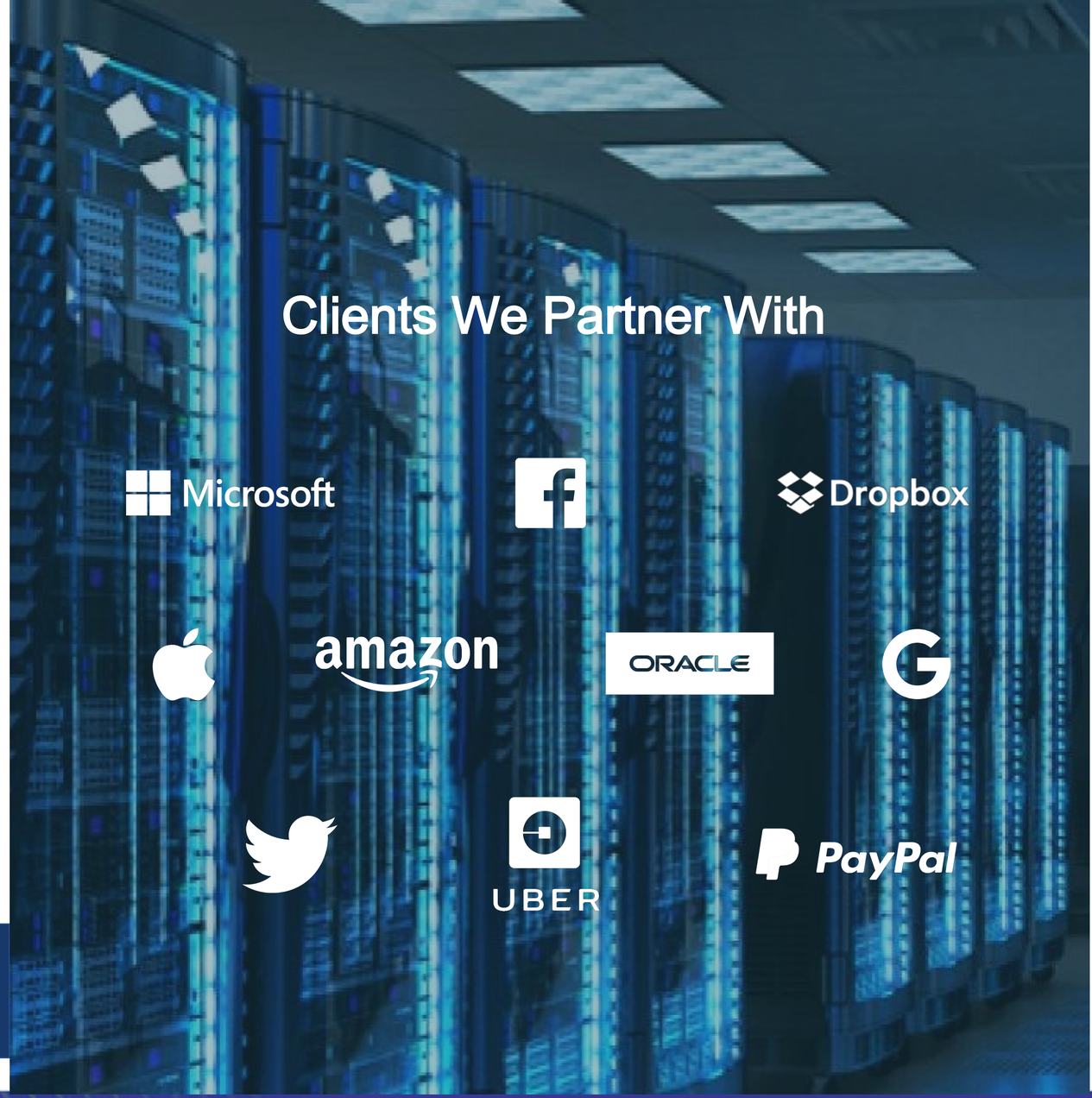
ITRENEW

- Leading provider of circular data center solutions to the hyperscale cloud market: technology, logistics, monetization
- Leading global provider of technology -enabled data center decommissioning services, including data sanitization software and IT asset remarketing
- Manufacturer of Sesame open compute and storage solutions, 1st of its kind suite of recertified and warrantied solutions for global use cases

SPLITTED DESKTOP

- Delivering TCO value recovery > $1Billion to date

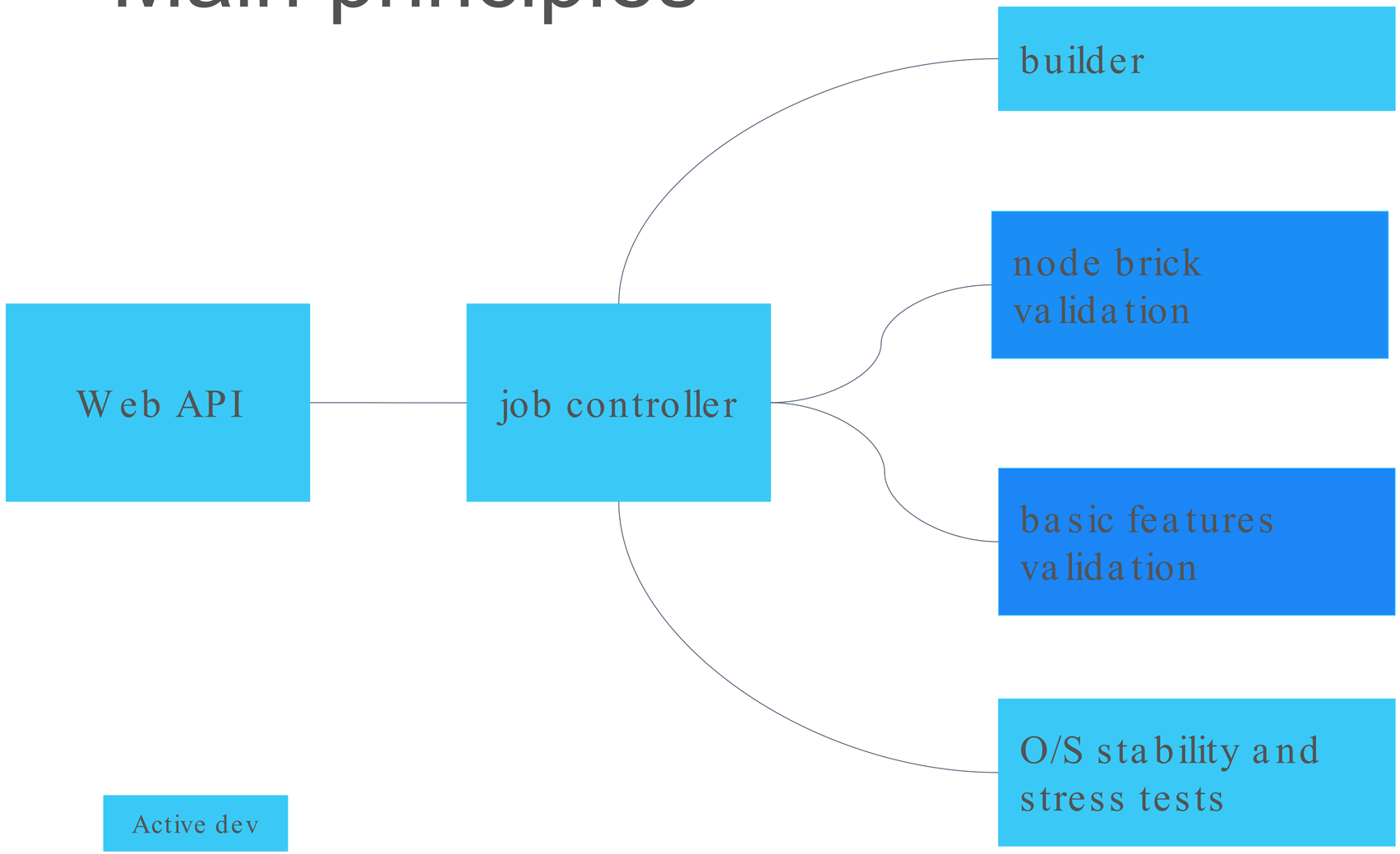
Redefining and powering LTV (a.k.a. not your grandfather's ITAD)



Project goals and genesis

- Build a full open source stack from hardware to operating system
- Provide back end infrastructure which could validate every patches and commits to the following project by running automated testings
    - linuxboot
    - OpenBMC

Main principles



- CI must be used from a Web API. Job Controller takes as an input a job description file which can stop the build at any 4 predefined stage.
- Output is a status and a rom file for a specific board
- Job status can be queried through the API

Implementation

1. Automatic redeployment managed through Ansible scripting
2. Automatic redundancy and scalability based on hardware availability
3. Slurm batch scheduler to manage run queue, unique job ID, and jobs output
4. KVM used to sand box builds into Ubuntu Xenial VMs
5. Basic setup requires:
    b. An ansible master node
    c. A Slurm controller node
    d. A Slurm batch node
    e. All of them sharing the same subnet

Web API

GOALS

1. Get as an input a github repository address with a unique commit ID
2. Provide job control
   - Launch
   - Kill
   - List
3. Provide jobs status feedback
   - Build log file

INITIAL IMPLEMENTATION

- Written in Go
- Support
  - Job launch
  - Job status query
  - Job log

Job controller

GOALS

1. Allocate and manage build nodes resources
2. Preset build environment
3. Store jobs status
4. Controlled using a hidden file which contains jobs description that will override default values

Job controller

INITIAL IMPLEMENTATION

1. Allocate job through Slurm batch scheduler
2. Setup a virtual machine (based on Ubuntu Xenial) when node is allocated with predefined characteristics using KVM and virsh
   a. VM storage is seating in memory (about 40 GB)
   b. VM have access to the Internet and can run apt command to setup build environment
3. Setup remote access to the VM from the slurm compute node
4. Copy the relevant files into the VM and setup build environment
5. Initiate job execution

Builder

GOAL

- Build a fully functional linuxboot image based on job parameters

INITIAL IMPLEMENTATION Based on osresearch/heads build environment

- Requires initial board ROM
- Can build the linuxboot kernel (currently based on 4.9 but can be changed)
- Can apply various patches to the kernel (BDS, AHCI etc ..) based on board descriptor
- Can build NERF (go based) user environment (with full integrated command control) as well as Heads (busybox/libc based)
- Can control/extract final DXE drivers integrated within linuxboot ROM
- Generate final linuxboot rom
- Heavy Makefile usage (looks like to be a “showstopper” to many Z generation engineers)

Node brick validation

GOAL

- Validate that a newly built image doesn't brick a node (aka that we can talk to the firmware through serial and successfully execute basic command)

INITIAL IMPLEMENTATION

- Based on initial qemu launch of the ROM
- Based on real hardware setup with a ROM emulator connected to the board

Basic features validation

GOALS

- Validate that a newly built image is properly detecting hardware
- Validate that a newly built image is able to install an O/S and boot it through
  - PXEboot or any other network boot capability
  - local boot on AHCI and NVMe storage

Basic features validation

INITIAL IMPLEMENTATION

O/S image installer built for linuxboot

- Ubuntu Xenial netinstaller or local boot kernel fails to boot properly on linuxboot (the kernel hangs)
  i. We regenerate a full ISO image based a valid original Ubuntu Xenial ISO and disable EFI support within the installer kernel
  ii. That image is also pre-configuring console output either to ttyS0 (local serial) or ttyS4 (SoL) on Winterfell machine
  iii. That new image is also bootable through a PXEboot process which is automatically configured on the slurm controller node.

O/S stability and stress test

GOALS

Validate that a newly built image is able to:

Run Linux at full operational mode

i. Properly detect hardware behavior (ACPI, CPU Frequencies management, Power Management, NUMA topology, VMX)

ii. Detect and can manage without error PCIe subsystems
   1. High speed network adapter (10 to 100 Gpbs)
   2. High speed storage box (lightning)
   3. Low speed storage box (knox)

iii. Can run various workload without system error and within an acceptable performance goal
   1. Run the Linpack/pysthone benchmark at speed
   2. Run Networking benchmark at speed
   3. Run bonnie++ at speed
   4. Run Multiple VM implementations of the previous benchmarks

Using the CI as of today

Launching a build

- Setup into a git repo a .ci.yml file which contains 2 basics entries

```
---

script:
  - echo "Print on stdout"
  - echo "Print on stderr" >&2
  - cat /proc/cpuinfo
  - cat /proc/meminfo
  - sudo echo 1
  - cd ubuntu/xenial/
  - pwd
  - sudo ./geniso

artifacts:
  - cpuinfo
  - meminfo
```

script -> Drive the execution workflow

artifacts -> Expected output files

- Execute a protected curl request to http://<my_ci_server>:1234/v1/jobs getting a repo and branch to run from as input parameters

Using the CI as of today

Controlling execution

curl request to http://<my_ci_server>:1234/v1/jobs/<job_id>

Monitoring output

curl request to http://<my_ci_server>:1234/v1/jobs/<job_id>/logs?raw=true

Retrieving results - setting up complex input

The build script shall take care about posting relevant output results into an end user repository or ftp build environment. Input like original ROM file shall be downloaded by the build script into the relevant repository before build is kicked off.

What is next: PRC lab

- Live infrastructure based in California, Open to the community to test and validate firmware build
- Hosted by ITRenew
- Dual racks based on OCP Winterfell/Leopard servers

Call to Action

- Join ITRenew Open Hardware lab in silicon valley (upcoming meetups)
- Contribute to Open Source Firmware initiative
- Provide hardware for testing and build early prototype
- Contact us if you are willing to build adapted designs!
- Download design files on github!

- Want to commit code ?

  https://github.com/linuxboot/linuxboot-ci


Open. Together.
OCP Global Summit | March 14–15, 2019
OCP
SUMMIT