

Curb Your Innovation

Peter Krey
Krey Associates, Inc.
Open Networking Summit
October 17-19, 2011 Stanford University

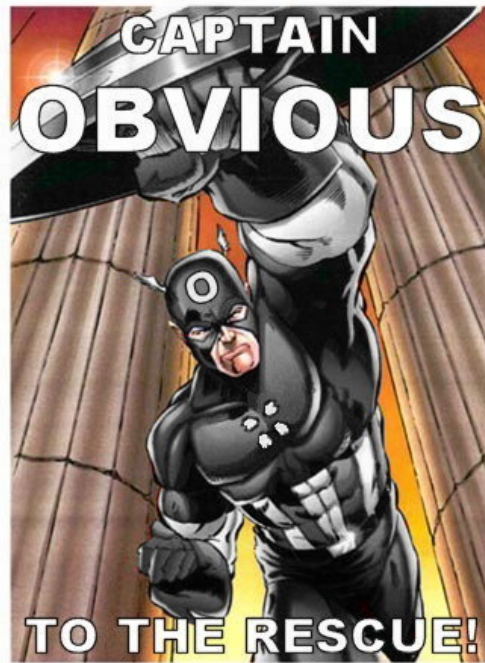
OPEN NETWORKING SUMMIT
OCTOBER 17-19, 2011 @ STANFORD UNIVERSITY'S LKS CENTER



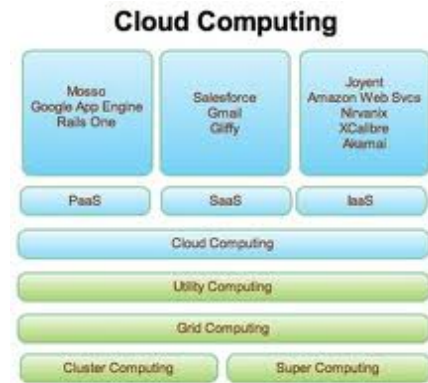
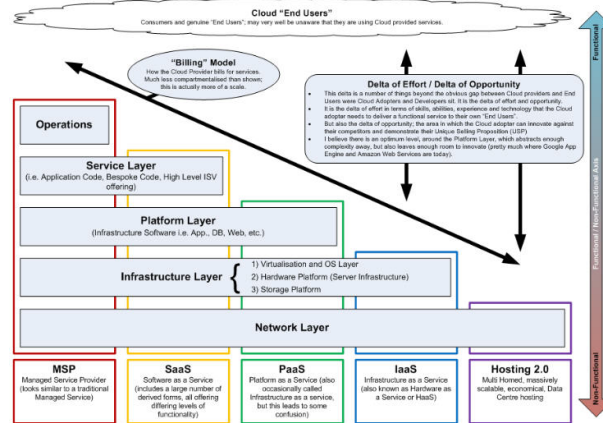
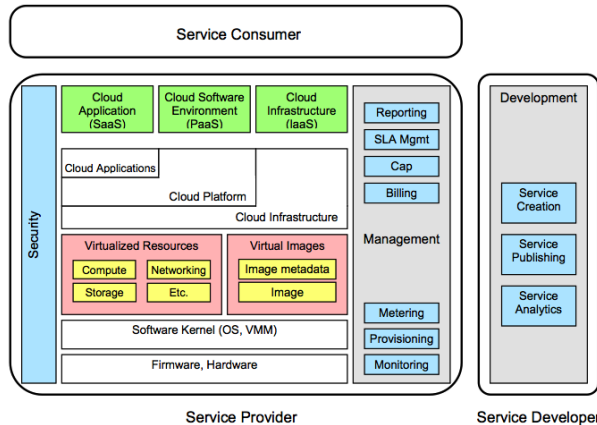
Overview

- Intro - Who Is Peter Krey ?
- Context & Overview
 - POV: Cloud Computing As Starting Point
- Cloud Equals ...
- Cloud Metrics Sampler
- The New Grand Canyon
- Network Infrastructure As Code
- Shawshank Redemption
- OpenFlow Enabled Open Source Apps
- SDN Metrics Sampler (WIP)
- Killer Capex & Opex
- Summary ... Q&A

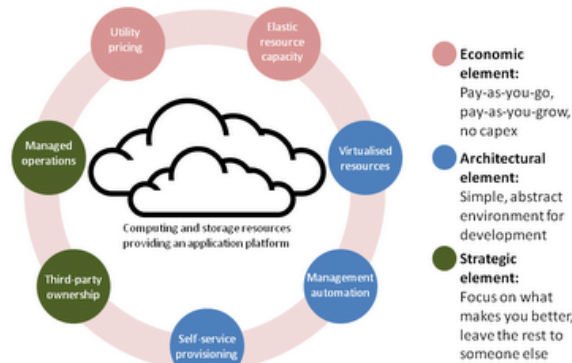
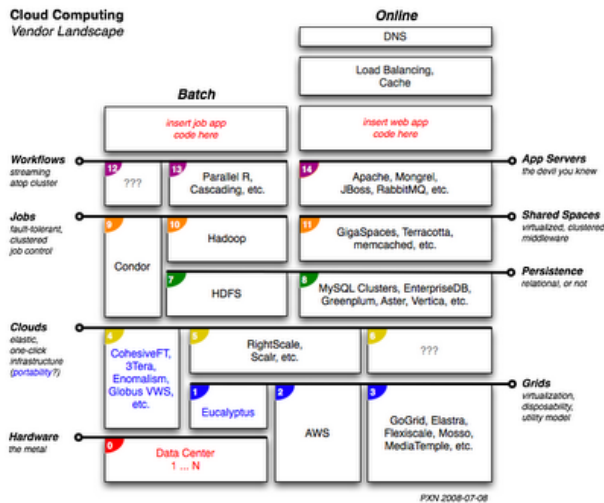
Who Is Peter Krey ?



What is Cloud Computing?



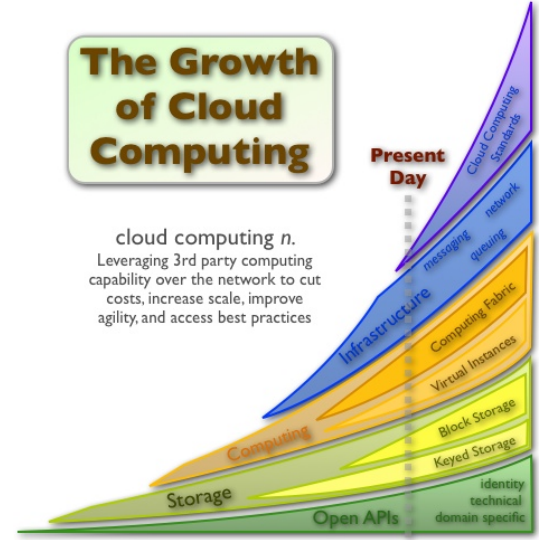
Cloud Computing Vendor Landscape



The Growth of Cloud Computing

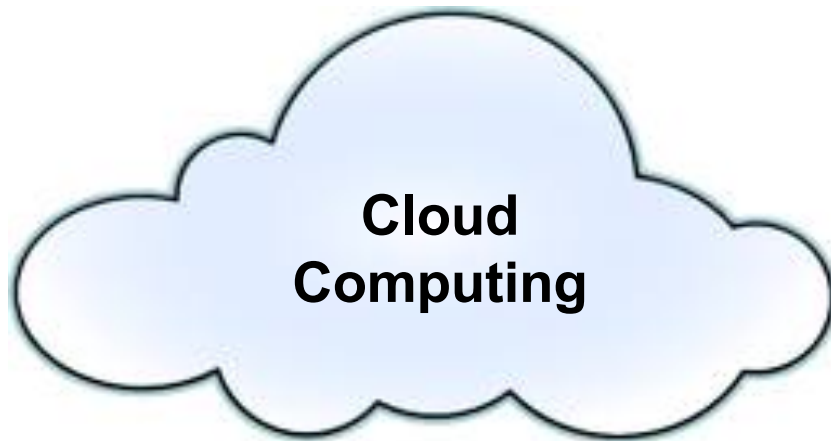
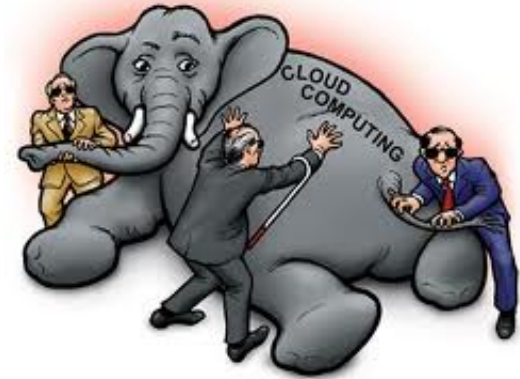
Present Day

cloud computing *n*. Leveraging 3rd party computing capability over the network to cut costs, increase scale, improve agility, and access best practices



From <http://blogs.zdnet.com/Hinchcliff>

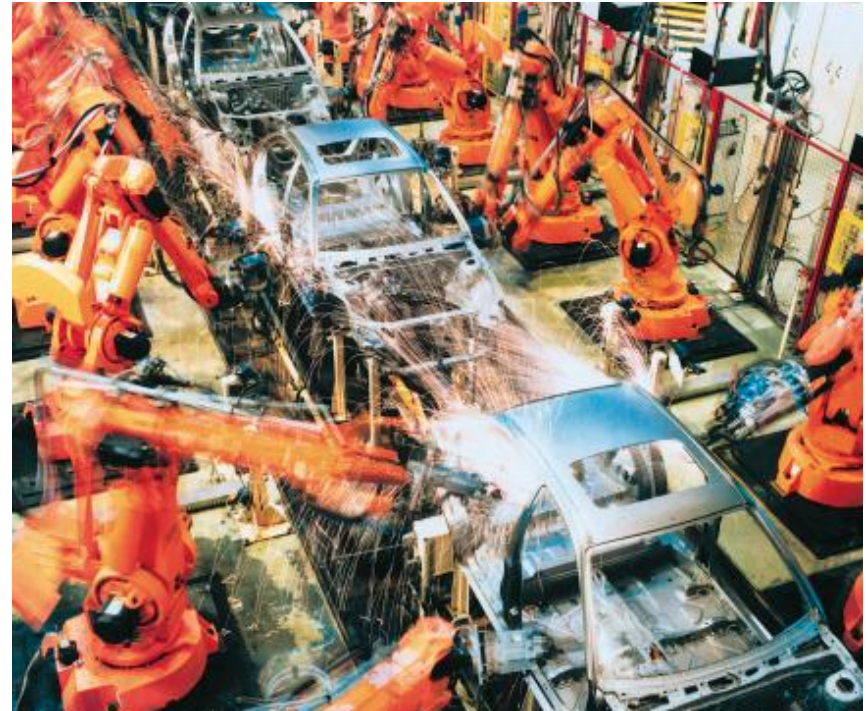
Cloud Computing Is ...



=



Enterprise IT...



... vs “West Coast”

Why Cloud Computing ?



Rob Carter, CIO at FedEx

"... it's also difficult to be the long pole in the tent every time we want to go do something."

Michael Heim, CIO of Eli Lilly

"... now it takes days. It's hard to underestimate the value of letting scientists work at their own pace ..."

Chris Perretta, CIO State Street

"Our goal is to optimize the time from idea to solution ... the idea is when the stopwatch starts."

"Really, it's about the customer -- when does their stopwatch start, not mine."

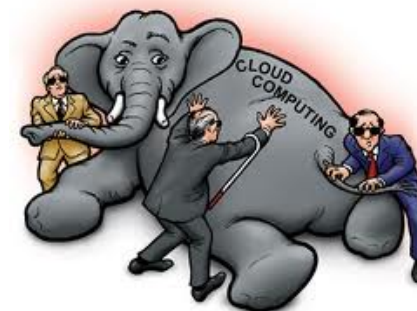


Key Cloud Metrics Sampler



Metric	Description	Metric / UOM
<i>Degree of Service Provider And/or IT Mgt Process Automation</i>	Exactly How Automated is the Service Acquisition & Delivery Process ?	Low / Medium / High
<i>Degree of End-user Self-service and Automation</i>	<ul style="list-style-type: none"> • Web Based Portal Front-end • Service Acquisition & Deployment Workflow • Command Line Tools • Developer And/or Sys Admin Api's • Language Neutral Interface (SOAP, REST, Etc) 	Yes, No, Partial (%)
<i>Service Delivery Level</i>	How Long Must an End-user or Developer Wait for a Cloud Service to Be Delivered ?	Seconds, Minutes, Hours, Days, Months, Qtrs ?
<i>Degree of Infrastructure H/W & S/W Independence</i>	<ul style="list-style-type: none"> • Prop vs. Non-Prop Form Factors: 1U, 2U, etc, Versus Blades • Standards Based Sys Mgt (I.E., IPMI) • Prop vs. Non-Prop Hypervisors (VMWare Versus Xenserver, KVM) • Automation & Toolset Hypervisor Dependency 	Low, Medium, High
<i>Infrastructure Services Automation Categories</i>	<ul style="list-style-type: none"> • Compute • Storage • Networking 	Yes, No, Partial (%)

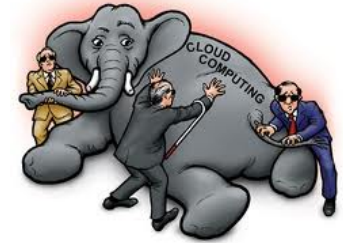
Cloud = Not Enterprise IT ...



- Point. Click. Deploy. 100% Automated *Everything* !
 - *Delivered in Minutes ... Not Months. Near Instant Time-to-Market*
- Developer & End-User Self-Service, API Direct Automation Access
 - *Web Portal Front-end To Underlying Automated Provisioning Processes*
 - *Zero Management Engagement And/or Service Provider Interaction*
- Economics Driven By Multi-Tenant, Virtualized, Shared Infrastructure
 - *Virtualized Servers, Shared Storage & Networking and Other Core Infrastructure*
 - *Managed High Utilization of Shared Resources*
- Developer As Customer & SysAdmin Direct Automation Access
 - *Command Line Interface (CLI) and ReST Multi-language Interface*
- Grid-Like, Scale-Out Infrastructure
 - *Enables Smoother Scalability Path*
 - *H/W & S/W Vendor Independent, Volume Driven Economics*
 - *Quasi Elasticity and Dynamic Scaling*










Cloud OSI-Like Layers (WIP)



- **Infrastructure as a Service (IaaS)**
 - Users & Developers Access Via Web-portal, CLI And/or REST Based Interfaces.
 - Users & Developers Self-deploy and Run Arbitrary Software, Incl. OS & Applications Within VM's.
 - All Basic Computing Infrastructure Deployed & Delivered As 100% Fully Automated Services (e.g., AWS)
 - Infrastructure Platform Virtualization of Servers, OS As VM's, Networking, and Storage (Mountable & Byte Object Based)
 - Data-center Space and Other Fundamental Infrastructure All Incl. Within Fully Outsourced Service
 - Versus Purchasing, Building, and Operating Infrastructure, Utilized As Pay-as-Used Variable Service
- **Platform as a Service (PaaS)**
 - App Dev Platform to Create & Deploy New Applications (E.G., Tomcat / Spring, Google App Engine)
 - Deploys Onto Cloud IaaS, Applications Created Using Programming Languages, Development Platform and Tools Supported by the PaaS
 - Facilitates Deployment of Applications Without the Cost and Complexity of Buying and Managing the Underlying Hardware and Software Layers.
- **Software as a Service (SaaS)**
 - Web-based Applications & Software Services (e.g., GMail, Google Apps, Salesforce.com, WebEx)
 - Accessible From Various Client Devices Through a Thin Client Interface Such As a Web Browser
 - Provided Over Internet, Eliminating the Need to Install & Run the Apps on the Customer's Own Infrastructure, Simplifying Maintenance and Support



IaaS, PaaS, SaaS Commercial Cloud Sampler

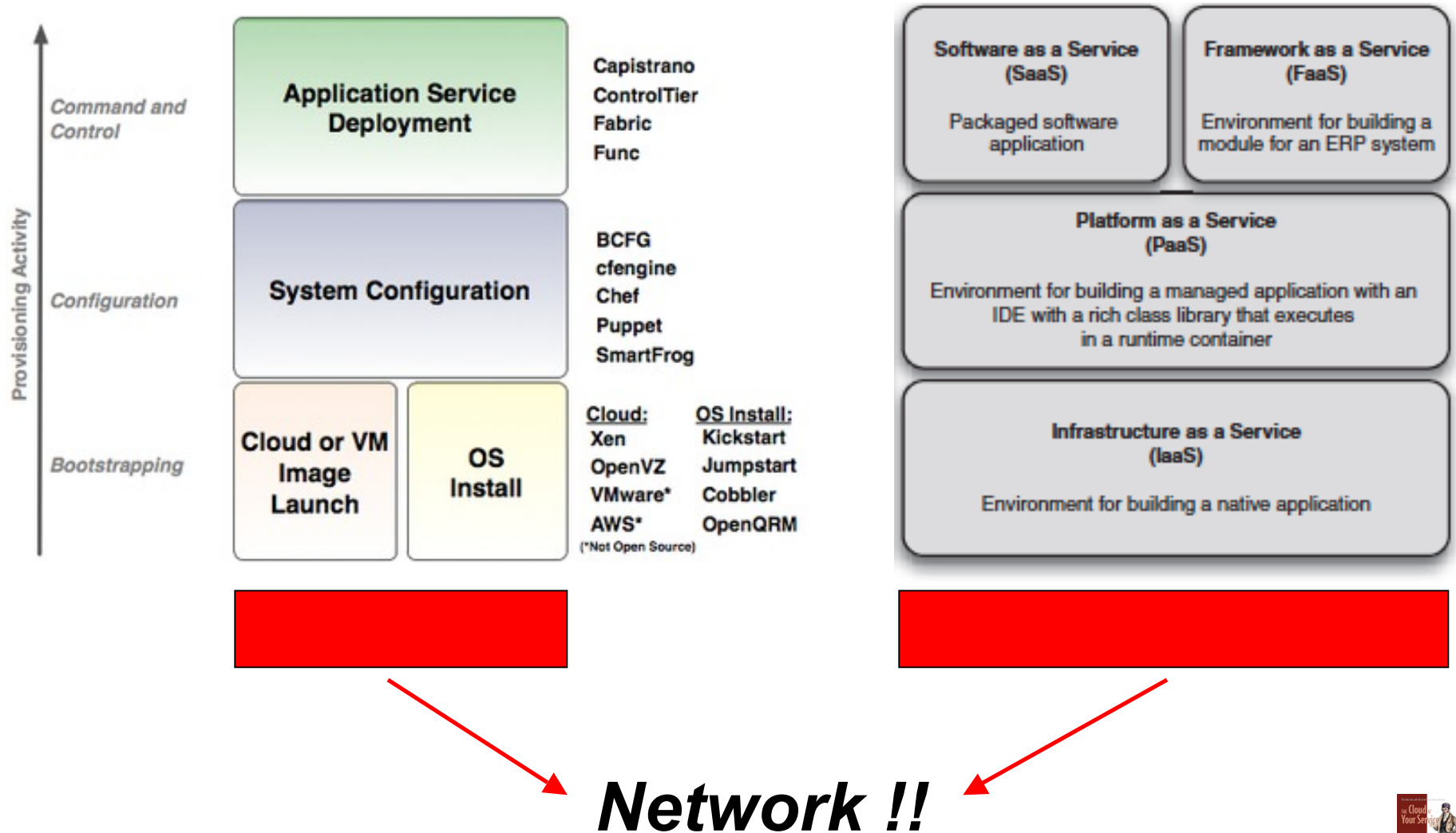
	Amazon	Rackspace	Google	Microsoft	Salesforce
SaaS					
PaaS					
IaaS					



Source: OpenStack.org

What's Missing Here ?

What's Missing Here ?



Networking ... The New Grand Canyon

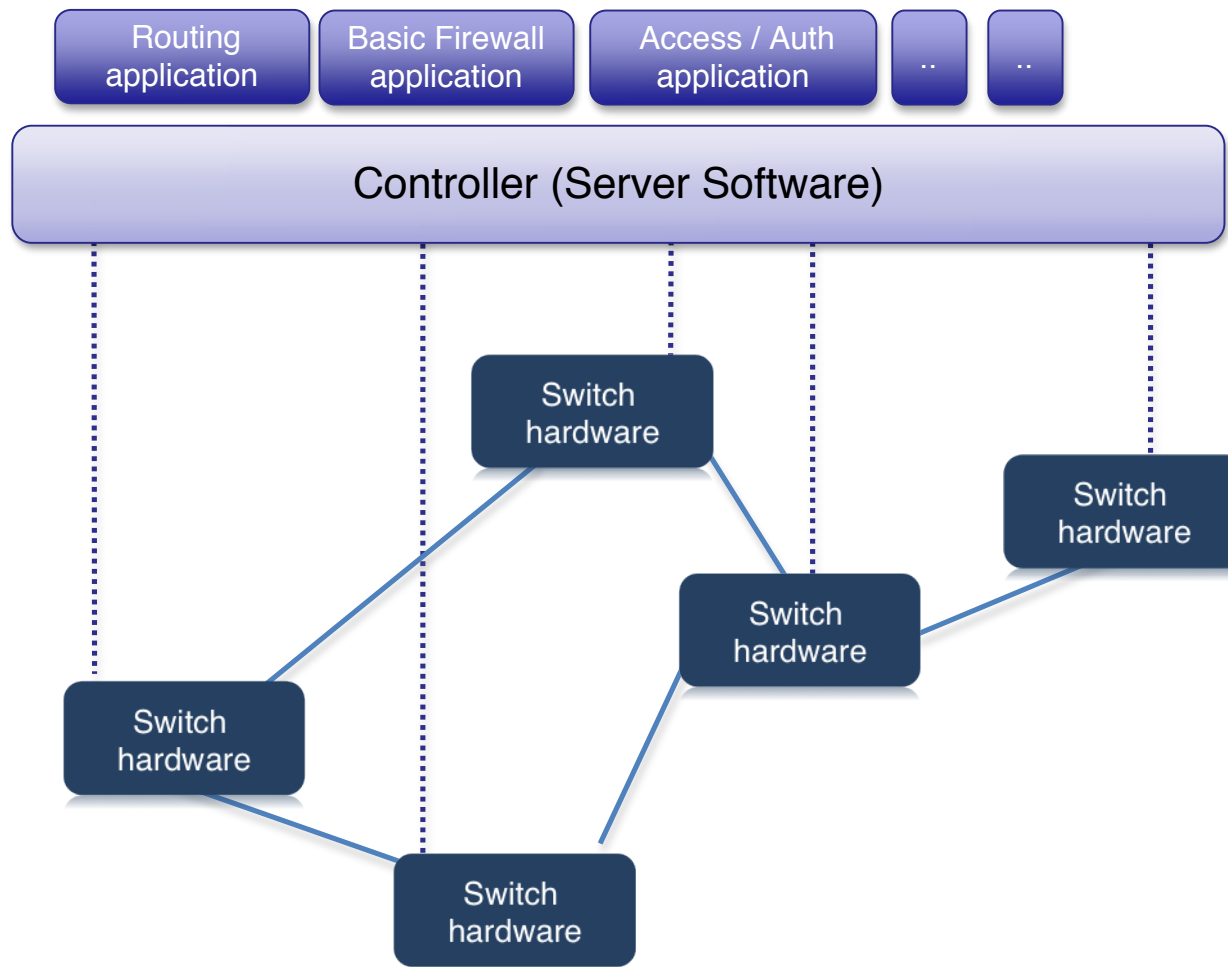


Enter Open Flow ...



Core Innovation - OpenFlow

Move Applications off the Switch, Into External Controllers



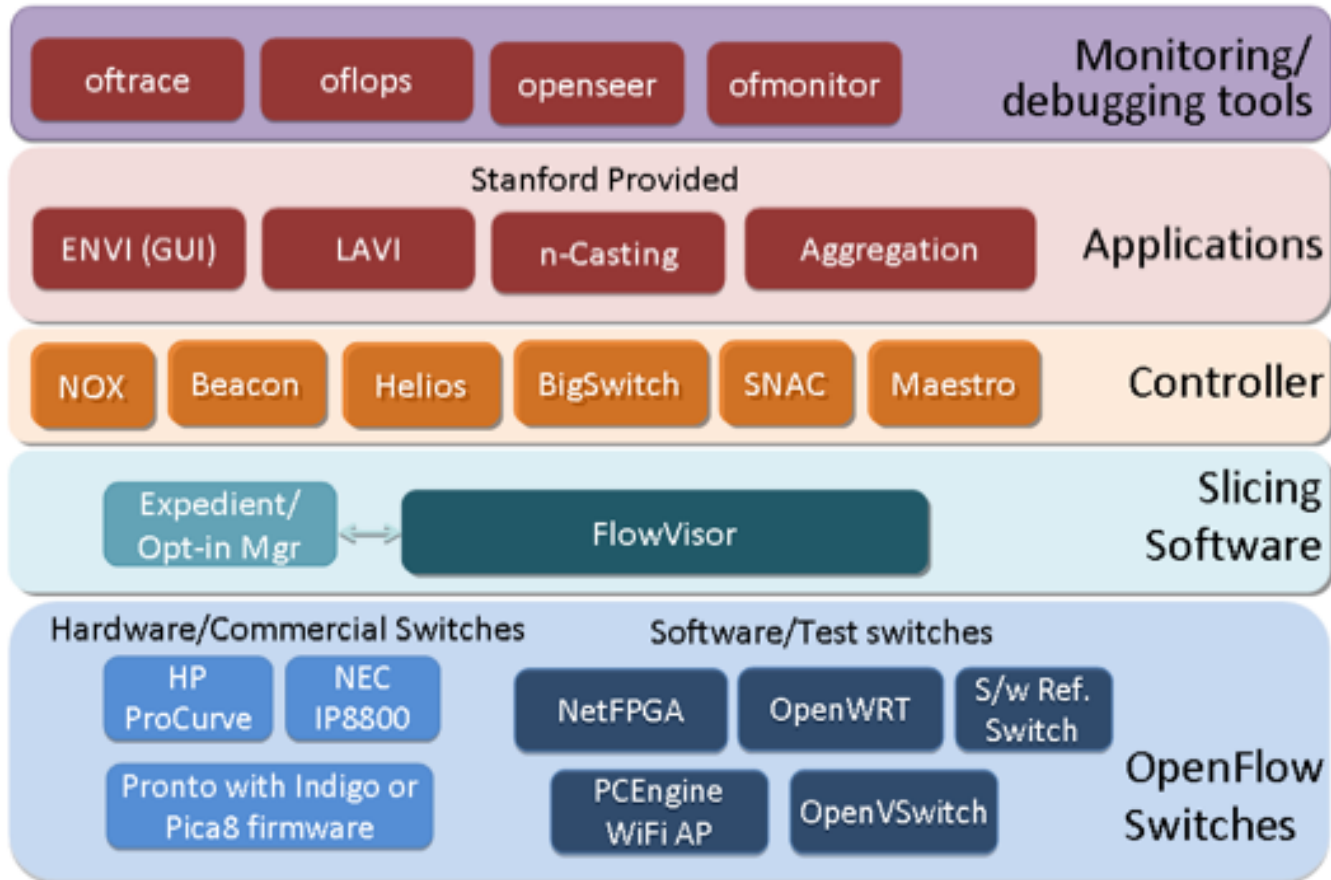
OpenFlow Protocol

Most Core Networking Functions Can Be Built As Server-side Applications on This Framework With Line-rate Performance.

Network Functions Are Decoupled From Underlying Hardware (and Location).

Integration of Applications With External Services Is Vastly Simplified.

OpenFlow Components



OpenFlow SDN

Open Source Sampler



NOX
An OpenFlow Controller

OPEN VSWITCH
An Open Virtual Switch



RouteFlow



The Yale Haskell Group Nettle





SDN ? Hhhmmm ... Let's Take A Look ...

learning_switch.py

```
from nox.coreapps.examples.frenetic_lib import *
from nox.coreapps.examples.frenetic_net import *

def addRule(((switch,mac),packet),(table,flood_dict,policy)):
    # initialize table[switch] if needed
    if not table.has_key(switch):
        table[switch] = {}
        flood_dict[switch] = true_fp()
        policy[switch] = Rule(fld[switch],[flood()])

    # calculate pattern, action, and rule
    pattern = dstmac_fp(mac)
    actions = [forward(inport(header(packet)))]
    rule = Rule(pattern,actions)

    # update pattern in flood_dict[switch]
    flood_dict[switch] = flood_dict[switch] - pattern
    flood_rule = Rule(flood_dict[switch],[flood()])

    if t[switch].has_key(mac):
        # regenerate policy[switch] if mac moved
        policy[switch] = []
        for (pat,acts) in t[switch].values():
            policy[switch].append(Rule(pat,acts))
    else:
        # otherwise just remove flood rule from policy
        del policy[switch][-1]

    # add forwarding rule and new flood rule to policy
    policy[switch].append(rule, flood_rule)

    # save pattern and actions in table[switch][mac]
    table[switch][mac] = (pattern,actions)

    return (table,fld,policy)
```



```
# rules : unit -> E policy
rules_e = None
def rules():
    global rules_e
    if rules_e is None:
        # query: returns first packet from every host (identified by
        # its mac adress) on every switch each time it sends traffic
        # on a different input port.
        q = (Select('packets') *
            GroupBy(['switch','srcmac']) *
            SplitWhen(['inport']) *
            Limit(1))
        # event function: accumulates policy
        ef = (Accum({},{},{}),addRule) >>
            Lift(lambda triple: triple[2])
        rules_e = q >> ef
    return rules_e

def main():
    return rules() >> Register()
```



SDN ? Hhhmmm ...

Another Look ...



```
package net.beaconcontroller.mactracker;

import java.util.Arrays;
import java.util.Set;
import java.util.concurrent.ConcurrentSkipListSet;

import net.beaconcontroller.core.IBeaconProvider;
import net.beaconcontroller.core.IOFMessageListener;
import net.beaconcontroller.core.IOFSwitch;

import org.openflow.protocol.OFMatch;
import org.openflow.protocol.OFMessage;
import org.openflow.protocol.OFPacketIn;
import org.openflow.protocol.OFType;
import org.openflow.util.HexString;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class MACTracker implements IOFMessageListener {
    protected static Logger logger = LoggerFactory.getLogger(MACTracker.class);
    protected IBeaconProvider beaconProvider;
    protected Set<Integer> macAddresses = new ConcurrentSkipListSet<Integer>();

    public IBeaconProvider getBeaconProvider() {
        return beaconProvider;
    }

    public void setBeaconProvider(IBeaconProvider beaconProvider) {
        this.beaconProvider = beaconProvider;
    }

    public void startUp() {
        beaconProvider.addOFMessageListener(OFType.PACKET_IN, this);
    }

    public void shutDown() {
        beaconProvider.removeOFMessageListener(OFType.PACKET_IN, this);
    }

    public String getName() {
        return "mactracker";
    }

    public Command receive(IOFSwitch sw, OFMessage msg) {
        OFPacketIn pi = (OFPacketIn) msg;
        OFMatch match = new OFMatch();
        match.loadFromPacket(pi.getPacketData(), (short) 0);
        Integer sourceMACHash = Arrays.hashCode(match.getDataLayerSource());
        if (!macAddresses.contains(sourceMACHash)) {
            macAddresses.add(sourceMACHash);
            logger.info("MAC Address: {} seen on switch: {}",
                HexString.toHexString(match.getDataLayerSource()),
                sw.getId());
        }
        return Command.CONTINUE;
    }
}
```

<https://openflow.stanford.edu/display/Beacon/Your+First+Bundle>

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    >

    <bean id="mactracker" class="net.beaconcontroller.mactracker.MACTracker"
        init-method="startUp" destroy-method="shutDown">
        <property name="beaconProvider" ref="beaconProvider"/>
    </bean>
</beans>

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:osgi="http://www.springframework.org/schema/osgi"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/osgi
    >

    <osgi:reference id="beaconProvider" cardinality="1..1"
        interface="net.beaconcontroller.core.IBeaconProvider">
    </osgi:reference>
</beans>
```



NOX

An OpenFlow Controller

SDN ? Hhhmmm ... Yet Another Look ...

```
Disposition handler(const Event& e)
{
    return CONTINUE;
}

void install()
{
    register_handler<Packet_in_event>(boost::bind(handler, this, _1));
}
```



```
void timer(){
    using namespace std;
    cout << "One second has passed " << endl;
    timeval tv={1,0}
    post(boost::bind(&Example::timer, this), tv);
}
```

```
timeval tv={1,0}
post(boost::bind(&Example::timer, this), tv);
```

```
post(new Flow_in_event(flow, *src, *dst, src_dl_authed, src_nw_authed, dst_dl_authed, dst_nw_authed, pi));
```

```
def handler(self):
    return CONTINUE

def install(self):
    self.register_handler (Packet_in_event.static_get_name(), handler)
```

```
def timer():
    print 'one second has passed'
    self.post_callback(1, timer)

post_callback(1, timer)
```

```
e = Link_event(create_datapathid_from_host(linktuple[0]), create_datapathid_from_host(linktuple[2]),
              linktuple[1], linktuple[3], action)
self.post(e)
```



Network Infrastructure As Code ?

Jesse Robbins

"Infrastructure as code is a technical domain revolving around building and managing infrastructure programmatically"



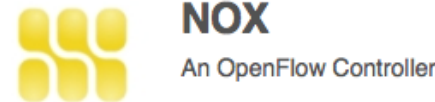
Adam Jacob

"Enable the reconstruction of the business from nothing but a source code repository, an application backup, and bare metal resources"

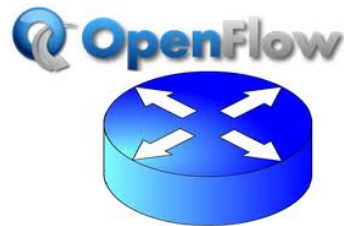


Network Infrastructure As Code !

- OpenFlow SDN
 - Domain Specific Language
- Controller as Network OS
 - Low Level OS: Flow Mods & Packet-Ins
 - Nox: C++, Python
 - Trema: C
 - Nettle: Haskell
 - Beacon: Java/OSGI
 - Floodlight: Java
- Applications Built On Top
 - Learning Switch is Canonical Example
 - Examples in Python, C, Java/OSGI, Java...
 - Frenetic as Networking-Specific Programming Language: Builds on OpenFlow + NOX, Functional Reactive Programming Model
 - REST as API for Networks: OpenFlow as REST API on Top of Floodlight & Beacon

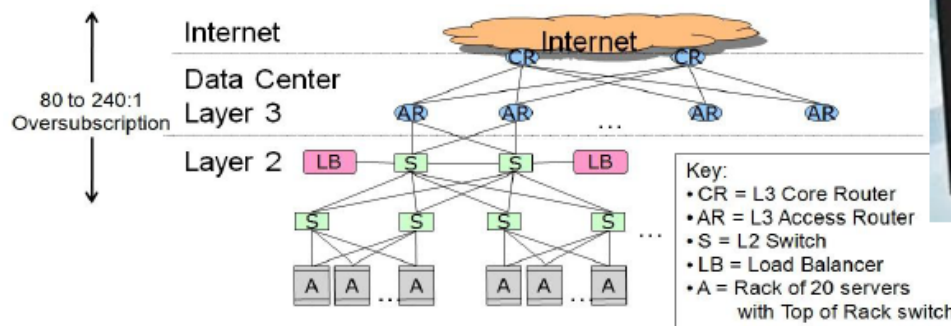
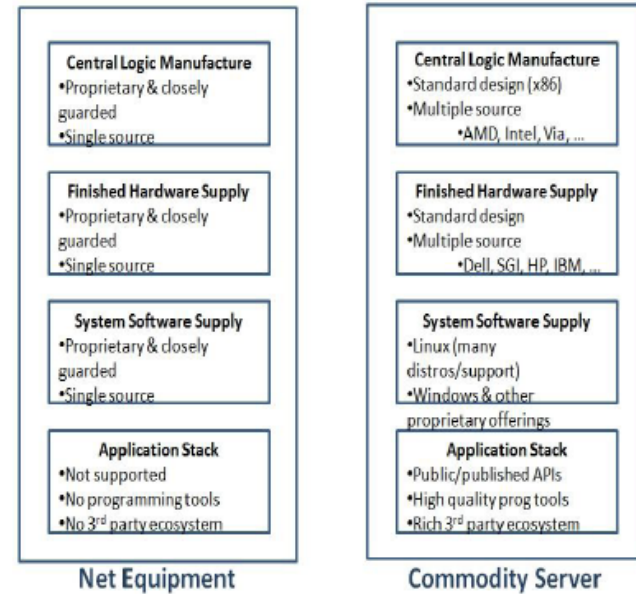


Network Infrastructure As Code !



Emerging Sea Change In Networking

- Current networks over-subscribed
 - Forces workload placement restrictions
 - Goal: all points in datacenter equidistant
- Mainframe model goes commodity
 - Competition at each layer over vertical integ.
- Get onto networking on Moores Law path
 - ASIC port count growth at near constant cost
 - Competition: Broadcom, Marvell, Fulcrum,...



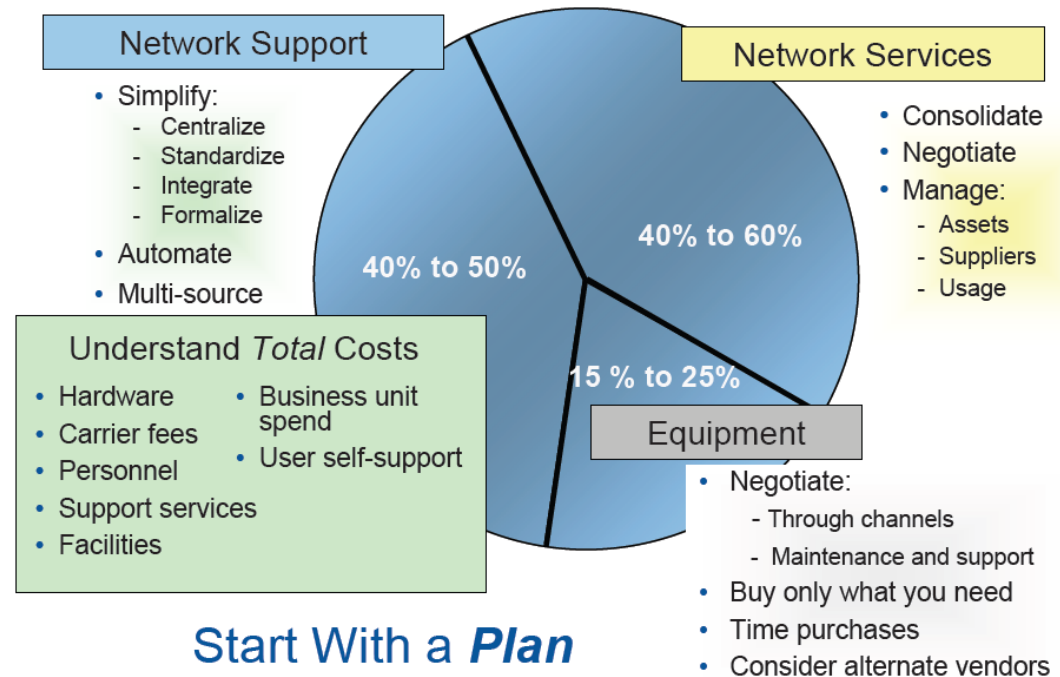
SDN Metrics Sampler (WIP)



Metric	Description	Metric / UOM
<i>Turn around time for a new tenant virtual network</i>	How long does it take to provision a new virtual network (L2 domain and firewall at minimum)	Minutes, hours, days, weeks
<i>Turn around time to move a server or VM between virtual networks</i>	How long does it take to move a server or VM, e.g. from quarantine to team-5-dev network?	Minutes, hours, days...
<i>Flexibility of topology</i>	Is a software defined network (an L2 domain and firewall at minimum) constrained to a single rack, or can it span multiple racks (L2)? Multiple rows (L3)? Multiple pods (L3+)? Multiple data centers (L3++)?	Distance (both logical and physical) that a software defined network can span before it needs to be cut in half. Note this is the flip side of utilization (i.e. if a virtual network can only span one rack, better keep a lot of that rack empty in case you have to grow that virtual network down the line!)
<i>Ease of automation / integration</i>	How easy is it to hook the software defined network in to your software tool chain (orchestration systems, versioning, inventory)...	Low, Medium, High
<i>Planning complexity due to software defined networking constraints</i>	How much effort is required to work around any short-comings created by the network when rolling out a new application?	Low, Medium, High

Serious Capex & Opex

- I & O App. 60% of Total IT Spend Worldwide (1)
- Networking 20% to 30% of Total Enterprise IT Capex & Opex
- *Some Orgs Even Higher ...*



Gartner - jaypultz april/2009

@50% Total Annual Save

- \$1Bln : \$100M to \$150M / Yr
- \$2Bln : \$200M to \$300M / Yr
- \$3Bln : \$300M to \$450M / Yr

(1) Gartner
<http://www.gartner.com/it/page.jsp?id=1807615>

Summary

- *IT Is Too Darn Slow ! So Automation Is Key ...*
- First Truly Interesting, Innovative, and Disruptive Networking Idea In a Very Long Time ...
- OpenFlow, SDN, and Open Source Networking Apps Are Potentially Huge Game Changers and Enablers ...
- Switching H/W Standardization, Open Developer & Automation Enablement Begin To Bridge The Networking Grand Canyon
- Connection & Integration With Current Developer Tools & Languages, App Server Platforms, ..., etc., Connects & Enables Multiplicative Technology Worlds for the First Time
- Networking Infrastructure as Code is a Potential Killer Infrastructure App Platform

Summary (cont'd)

- The Potential Capex & Opex Impacts are Huge Via H/W Standards, Positive Commoditization, and Automation Support.
- Must Also Heavily Collaborate & Integrate With Other Open Source Projects Incl. Git, Chef, OpenStack, and others.
- Creation of Open Network Compute a la Open Compute ?
- Security Models Must Be Defined, Implemented, and Integrated.
- OpenStack !
- ***Don't Complain OpenFlow About Gaps ... Participate & Contribute !!***

Many Thanks !



peterkrey@gmail.com