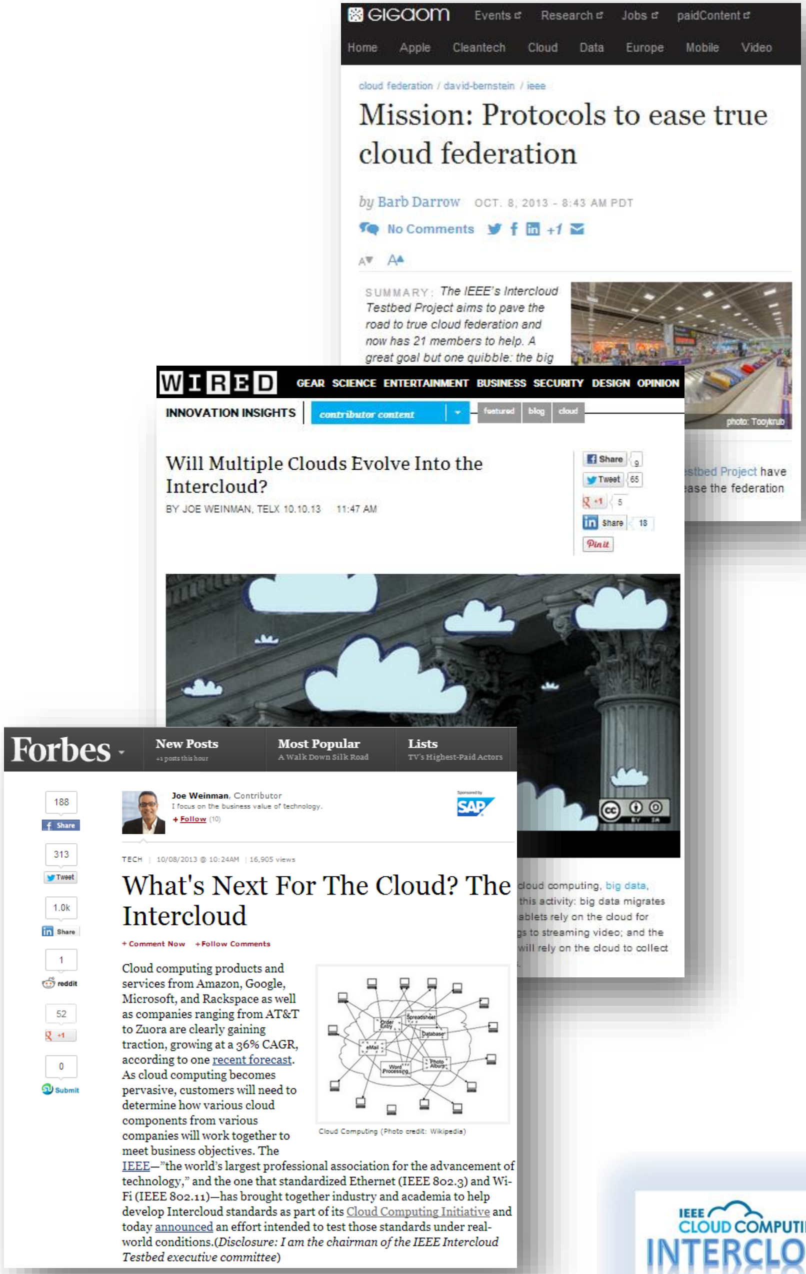
IEEE Intercloud Testbed

Design Workshop
May 8 and 9, 2014

OBJECTIVE FOR THIS WORKSHOP
BACKGROUND
PORTABILITY
INTEROPERABILITY – TELCO & INTERNET
INTEROPERABILITY – CLOUD
ARCHITECTURAL OVERVIEW
USE CASES
NAMESPACE AND GOVERNANCE
IDENTITY/TRUST
CONVERSATIONAL SUBSTRATE
TRANSPORT/SERVICES FRAMEWORK
SEMANTIC DIRECTORY
RESOURCE MATCHER/SOLVER
FEDERATING API
BEARER NETWORK FABRIC
REPLICATION/SCALING
AUDIT
TESTBED PROCESS
GET INVOLVED



Intercloud

WORKSHOP OBJECTIVE

Workshop Objective

- Understand proposed “theory of operation”
  - A mental model of how the system is supposed to work *
  - Do You think this will work?
  - Where Doubtful or Vague, Can we Improve/Make it Work?
- How
  - Thought process of key design themes
  - Analogies to other systems. Learnings from the Greats.
  - Example/Plausible Implementation Approaches for Whole System
- OK I “got it” now what
  - Complete/Improve/Revise so we really do have Example/Plausible Implementation Approaches
  - Postulate: The now “New and Improved” Example/Plausible Implementation Approaches are a good place to start the Experimental/Iterative/Agile/DevOps development process
- So, let’s build this thing

* http://en.wikipedia.org/wiki/Theory_of_operation

Workshop Schedule and Beyond

- Understand proposed "theory of operation"
  - A mental model of how the system is supposed to work *
  - Do You think this will work?
  - Where Doubtful or Vague, Can we Improve/Make it Work?
- How
  - Thought process of key design themes
  - Analogies to other systems. Learnings from the Greats.
  - Example/Plausible Implementation Approaches for Whole System

Day 1 Workshop

- OK I "got it" now what
  - Complete/Improve/Revise so we really do have Example/Plausible Implementation Approaches
  - Postulate: The now "New and Improved" Example/Plausible Implementation Approaches are a good place to start the Experimental/Iterative/Agile/DevOps development process

Day 2 Workshop

- So, let's build this thing

Post Workshop Engineering

Intercloud

BACKGROUND

Cloud Computing is the New Pervasive Ubiquitous Intelligence & Communications Platform for Planet Earth

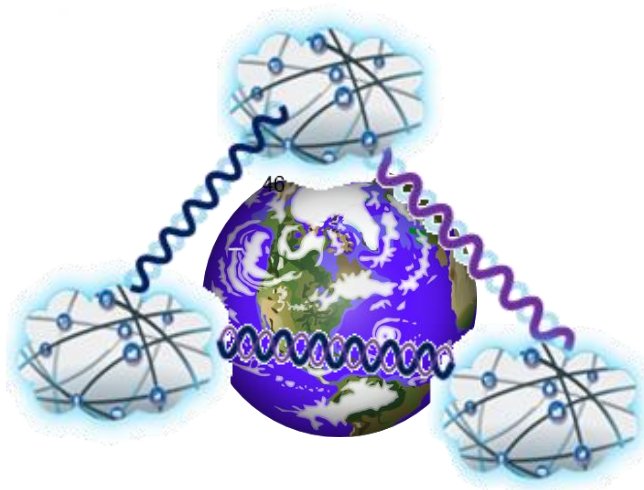
Education

WIKIPEDIA

Relationships

Day to Day Life

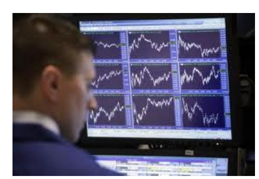
Communications



Entertainment

Infrastructure

Transportation

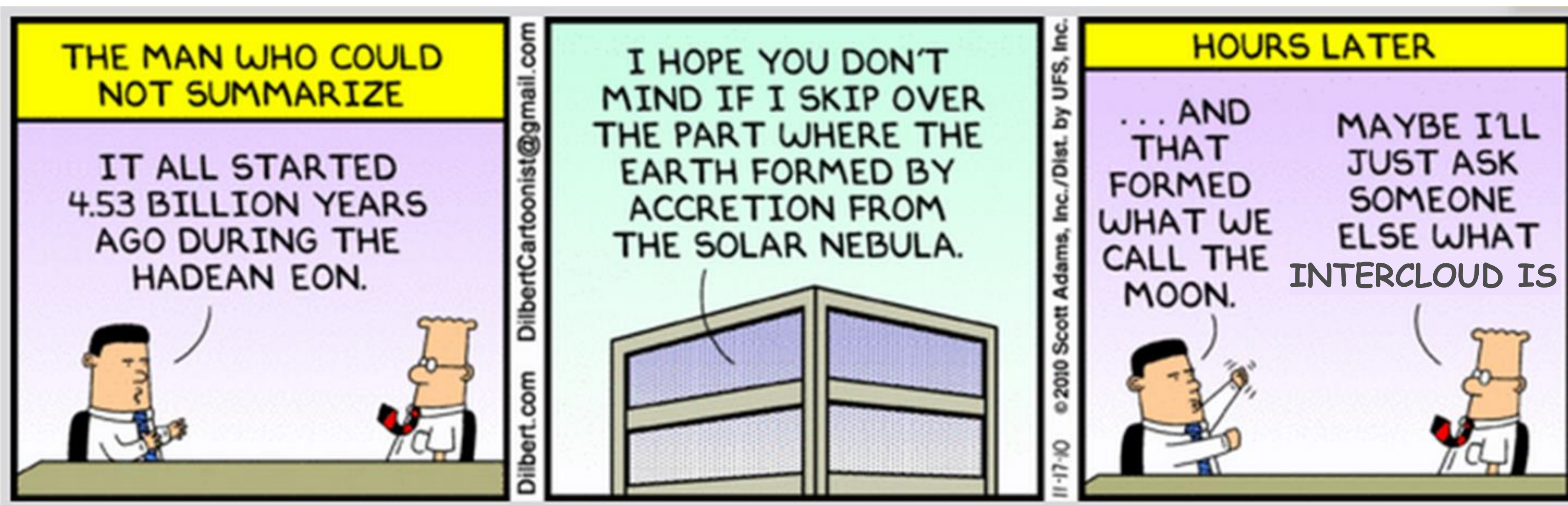


Commerce

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

"History doesn't repeat itself - at best it sometimes rhymes"

- Mark Twain
… with apologies to Dilbert



IEEE CLOUD COMPUTING INTERCLOUD TESTBED

1980 - 1997

1980

1989

1984

1997
Earthlink
hits 1M users,
has IPO

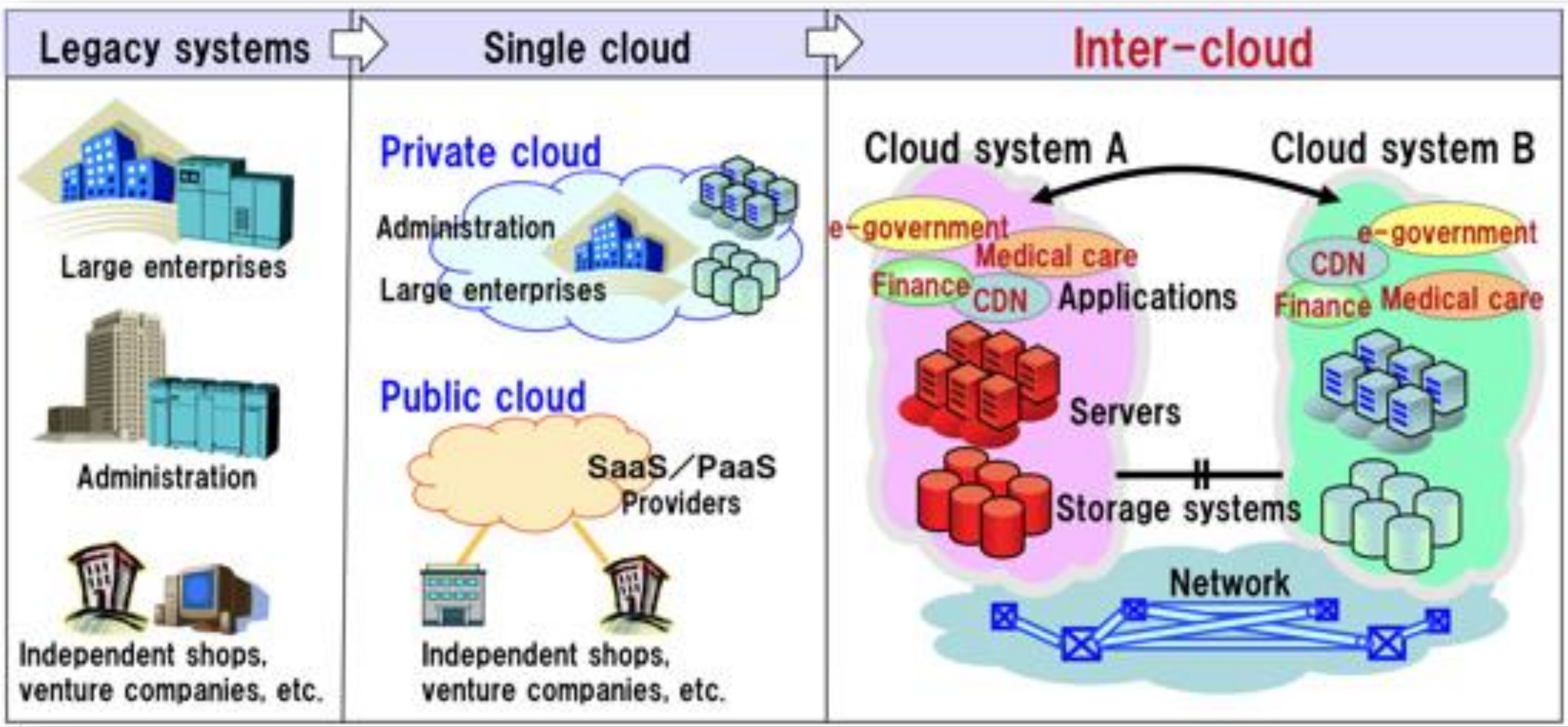
IEEE CLOUD COMPUTING INTERCLOUD TESTBED

It Really is a Déjà Vu!

Does it really take a visionary to see what will happen next?

"I'm seeing a possibility of inter-cloud problems mirroring the Internet problems we had thirty or forty years ago,“, Vint Cerf, Vice President and Chief Internet Evangelist for Google

Simple View of Intercloud



Source: GICTF

The Perfect Storm Begins, 2007

Cloud

LTE/WiFi

Smart Devices

IEEE Intercloud Background

- **2007 – Kevin Kelly, founding executive editor of Wired magazine, and a former editor/publisher of the Whole Earth Catalog, blogs about Cloud Computing and theorizes *"Eventually we'll have the intercloud, the cloud of clouds. This intercloud will have the dimensions of one machine comprised of all servers on the planet"***
- **2009 – Cisco team writes paper "Blueprint for the Intercloud"**
- **2009 – Industry group "Global Intercloud Technology Forum" (GICTF) forms in Japan**
- **2010 – Intercloud Research explodes. First IEEE International Workshop on Cloud Computing Interoperability and Services (InterCloud2010) held in France**
- **2011 – IEEE launches technical standards effort called P2302 - Standard for Intercloud Interoperability and Federation (SIIF)**
- **2012 – "Intercloud" made the Wired Magazine Jargon Watch list**
- **2013 – IEEE announces the IEEE Global Intercloud Testbed initiative. Two dozen cloud and network service providers, cloud-enabling companies, and academic and industry research institutions from the United States, the Asia-Pacific region, and Europe.**

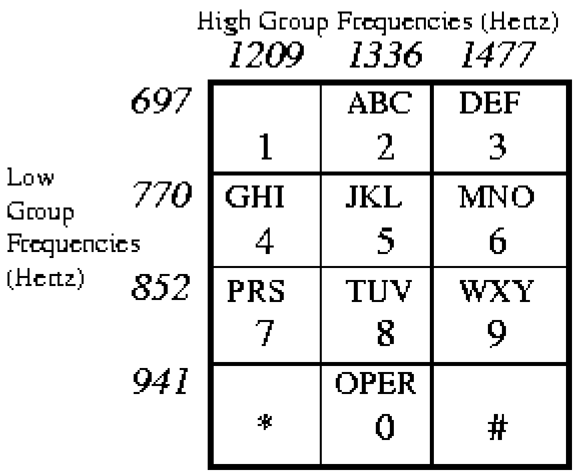
Public Network Federation Trends

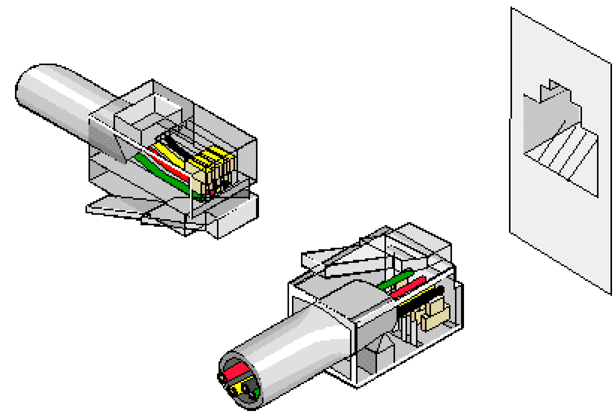
Telephony Federation	Internet Federation	Cloud Federation
Took 100 Years	Took 15-20 Years	Taking 5-10 Years
Formal Standard (ITU) for Protocols	Informal Standard (IETF) for Protocols	De Facto Standards for User Protocols (AWS, GCE)
No Open Source for any Protocols	Open Source for User Protocols (TCP/IP) No Open Source for Federation Protocols (Routing)	Open Source for Everything
Peer to Peer Federation model ✓	Peer to Peer Federation model ✓	Peer to Peer Federation model ✓

Intercloud

PORTABILITY

Successful User to Network Interface Standards





Voice

Data

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Microsoft's Original Cloud Vision was about .NET Portability

"We know about PCs, we know about servers .. (cloud is) a new kind of computer that 20 years from now we'll wonder how we did without"

"things that are fundamentally different tend to be at the application framework level"

"it will initially seem familiar, but .. there are certain aspects of it that feel different .. most applications will not run that way out of the box"

"we've raised the level of abstraction .. to .. a service model where you declare .. the different pieces of your application .. you give it hints .. which pieces need to run close together, which pieces need to run far apart .. want this running in a different datacenter .. and you let the system just deal with it"

Ray Ozzie on Microsoft Azure
October 28, 2008

NETWORKWORLD

News | Blogs & Columns | Clear Choice Tests | Videos | Events |

Security | LANs & WANs | VoIP | Net Management | Wireless | Software | Data Center | SMB

Desktops | NAS | SANs | Servers | Storage Mgmt. | Utility Computing | Virtualization | Green IT | Clear Choice Te

Microsoft will float cloud OS this month

Details of Microsoft's cloud OS will finally be revealed

By John Fontana , Network World , 10/02/2008

Share/Email | Buzz up! | Comment | Print | IT Buyer's Guides

Microsoft at the end of the month will unveil its "Cloud OS," the secretive Ray Ozzie project that provides a virtual Windows operating system platform for the rapid development, deployment, and maintenance of Internet services and applications.

Microsoft will unveil details later this month at its Professional Developers Conference (PDC) and show developers the APIs and plumbing services provided by a utility computing platform code-named Red Dog.

In essence, it is an application development and execution platform that lives on the Internet. Red Dog is similar to Amazon's Elastic Compute Cloud (EC2) and Google's App Engine, both cloud-based application platforms.

Read the latest WhitePaper - Case Study: Golder Associates

Cloud Portability Was Top of Mind

Eucalyptus : A Technical Report on an Elastic Utility Computing Archietcture Linking Your Programs to Useful Systems
UCSB Computer Science Technical Report Number 2008-10

Daniel Nurmi, Rich Wolski, Chris Grzegorczyk
Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitrii Zagorodnov

Computer Science Department
University of California, Santa Barbara
Santa Barbara, California 93106

Abstract

Utility computing, elastic computing, and cloud computing are all terms that refer to the concept of dynamically provisioning processing time and storage space from a ubiquitous "cloud" of computational resources. Such systems allow users to acquire and release the resources on demand and provide ready access to data from processing elements, while relegating the physical location and exact parameters of the resources. Over the past few years, such systems have become increasingly popular, but nearly all current cloud computing offerings are either proprietary or depend upon software infrastructure that is invisible to the research community.

In this work, we present Eucalyptus, an open-source software implementation of cloud computing that utilizes compute resources that are typically available to researchers, such as clusters and workstation farms. In order to foster community research exploration of cloud computing systems, the design of Eucalyptus emphasizes modularity, allowing researchers to experiment with their own security, scalability, scheduling, and interface implementations. In this paper, we outline the design of Eucalyptus, describe our own implementations of the modular system components, and provide results from experiments that measure performance and scalability of an Eucalyptus installation currently deployed for public use.

The main contribution of our work is the presentation of the first research-oriented open-source cloud computing system focused on enabling methodical investigations into the programming, administration, and deployment of systems exploring this novel distributed computing model.

1 Introduction

Scalable Internet services [1, 4, 24, 44] deliver massive amounts of computing power (in aggregate) on demand to large, internationally distributed user communities through well-defined software interfaces. Until recently, however, access to these services has been restricted to human-oriented and simple query-style application programming interfaces (APIs). With few exceptions, an application programmer wishing to incorporate such a service as a software component had little ability to direct and control computation *inside* the service explicitly.

Cloud computing [11, 46] has emerged as a new paradigm for providing programmatic access to scalable Internet service venues. [1] While significant debate continues with regard to the "optimal" level of abstraction that such programmatic interfaces should support (c.f., software-as-a-service versus platform-

[1] The term "cloud computing" is considered by some to be synonymous with the terms "elastic computing," "utility computing," and occasionally "grid computing." For the purposes of this paper, we will use the term "cloud computing" to refer to cloud, elastic, or utility computing but not to grid computing. The difference is explained in Section 4.

1

Cloud Computing Provider, GoGrid, Moves API Specification to Creative Commons Licensing Under a ShareAlike License

In a marked departure from proprietary standards, GoGrid today released its cloudcenter Application Program Interface (API) specification under a CreativeCommons license, enabling developers, system integrators, and other companies to openly copy, modify, distribute and republish this cloud computing API.

San Francisco, CA (Vocus) January 20, 2009

Tweet Like +1 Share EMAIL

GoGrid, the Cloud Computing division of ServePath, LLC today announced the release of its cloudcenter API OpenSpec (open specification) under the Creative Commons ShareAlike license. This allows any Cloud Computing provider to build an API based on this OpenSpec, as well as to modify, share, and republish changes to the specification itself and profit from these efforts.

> “rough consensus and running code.”

This innovative move reaffirms GoGrid and ServePath's continuing commitment to the Open Source and Open License movement as well as setting an example in establishing open standards, transparency and commonality within the Cloud Computing Marketplace. Tools written to this OpenSpec standard will control both the GoGrid cloud platform and the products of other compatible cloud computing providers.

GoGrid is simultaneously releasing the GoGrid cloudcontrol™ command line utility, which exercises the functionality in the cloudcenter API OpenSpec. This utility is released under the Lesser General Public License (LGPL) which enables the broadest adoption while also encouraging reuse and sharing in conjunction with the OpenSpec.

This release is in the spirit of the original IETF mission statement (RFC3935) that specifies the creation of standards based on "rough consensus and running code." GoGrid actively encourages the support and interest of the Cloud Computing Community in further developing this OpenSpec and will be revising and working towards updating the cloudcenter API and the its OpenSpec in a manner that fosters the community, interoperability and ease of use.

IEEE CLOUD COMPUTING
INTERCLOUD TESTBED

OpenStack History

Eucalyptus Re-Brands as Open Source, competing with OpenStack

EUCALYPTUS

UCSB Professor Takes Eucalyptus, and makes Closed Source Start-Up

Eucalyptus Systems

amazon web services™

UCSB open sources AWS clone as Eucalyptus.

Eucalyptus

UC SANTA BARBARA engineering

UCSB Creates a simple "clone" of AWS EC2 and S3. Repeatedly asks Amazon for permission to clone AWS API. Amazon ignores request but quietly removes copyrights from XML definitions.

NASA Attempts to back contribute Improvements; Eucalyptus Systems refuses them

NASA Takes Eucalyptus, Uses it, Improves it, especially Compute Part

Rackspace has good Cloud Storage but less strong, and not AWS compatible compute. Teams with NASA, donates it's Storage to OpenStack

rackspace HOSTING

OpenStack is Formed

openstack™ CLOUD SOFTWARE

NASA

??

NASA is obligated to return improvements to US Citizens, must place code somewhere in Public Domain

NASA Donates improved Eucalyptus Compute to OpenStack

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

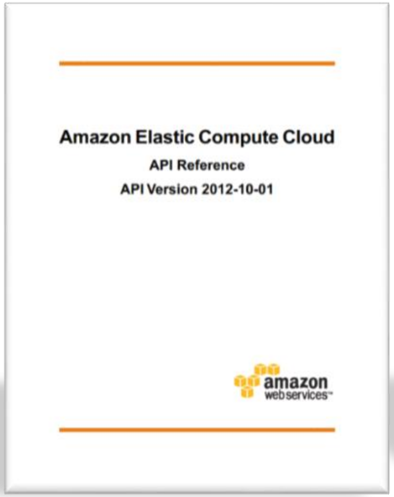
OpenStack Compatibility
"All the APIs"

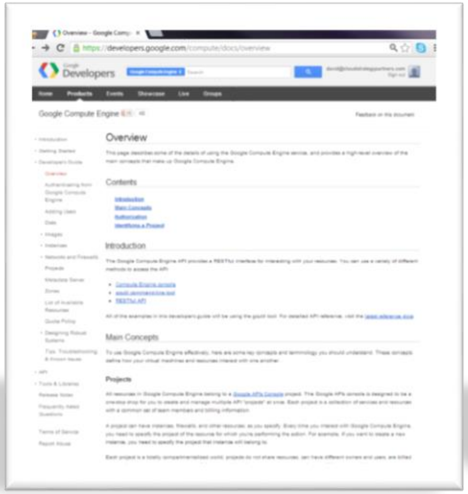
THE RACKSPACE OPEN CLOUD POWERED BY OPENSTACK

amazon web services™

Google Compute Engine

Occi Open Cloud Computing Interface





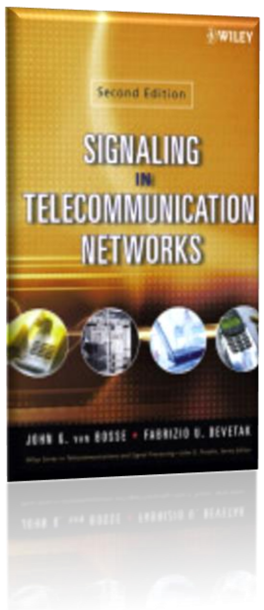
IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Intercloud

TELCO AND INTERNET INTEROPERABILITY

Signaling is a Network to Network Interface

In telecommunications, a network-to-network interface (NNI) is an interface which specifies signaling and management functions between two networks. An NNI can be used for interconnection of Signaling (e.g. SS7) or Internet Protocol Routing (IP)

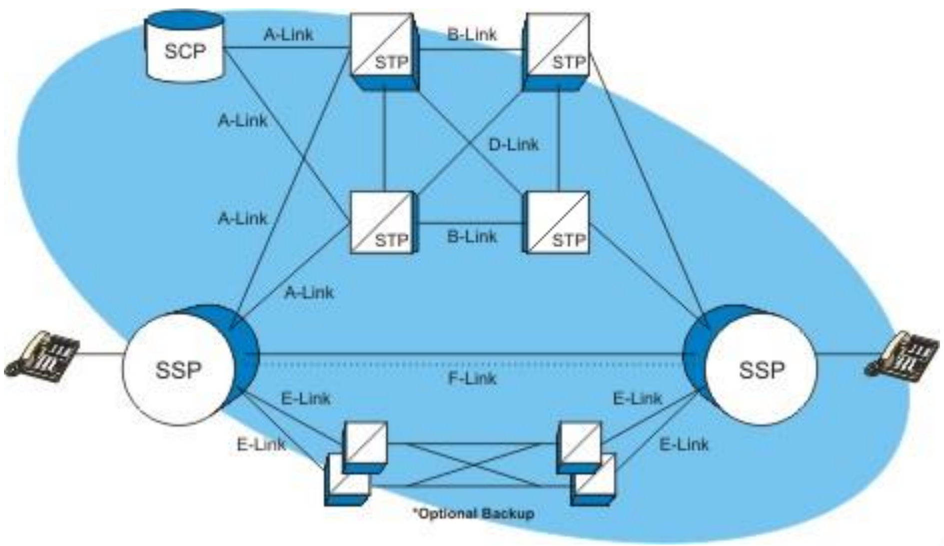


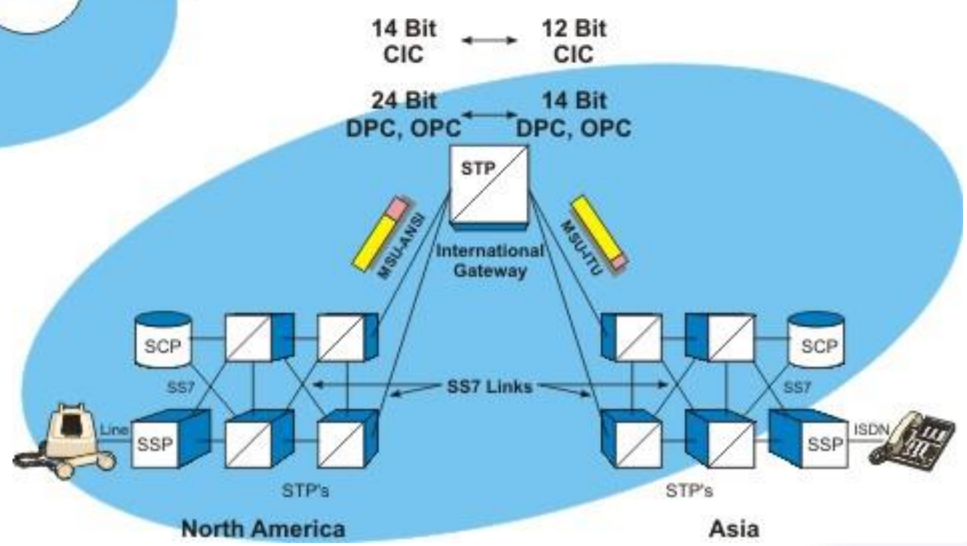
- Bearer Traffic
  - The content or payload
- Bearer Network
  - Carries Bearer Traffic
- Signals
  - Control information about the Bearer Network
- In-band signaling
  - Control information flows on the Bearer Network
- Signaling Network
  - Control information flows on a network separate from the Bearer Network
- CCS (common channel signaling)
  - Control Information regarding multiple Bearer Networks flow on a network separate from the Bearer Network

Signaling Networks Scale and have Transparent Federation → Intercloud should use a CCS/Signaling Network + Bearer Network

http://en.wikipedia.org/wiki/Network-to-network_interface

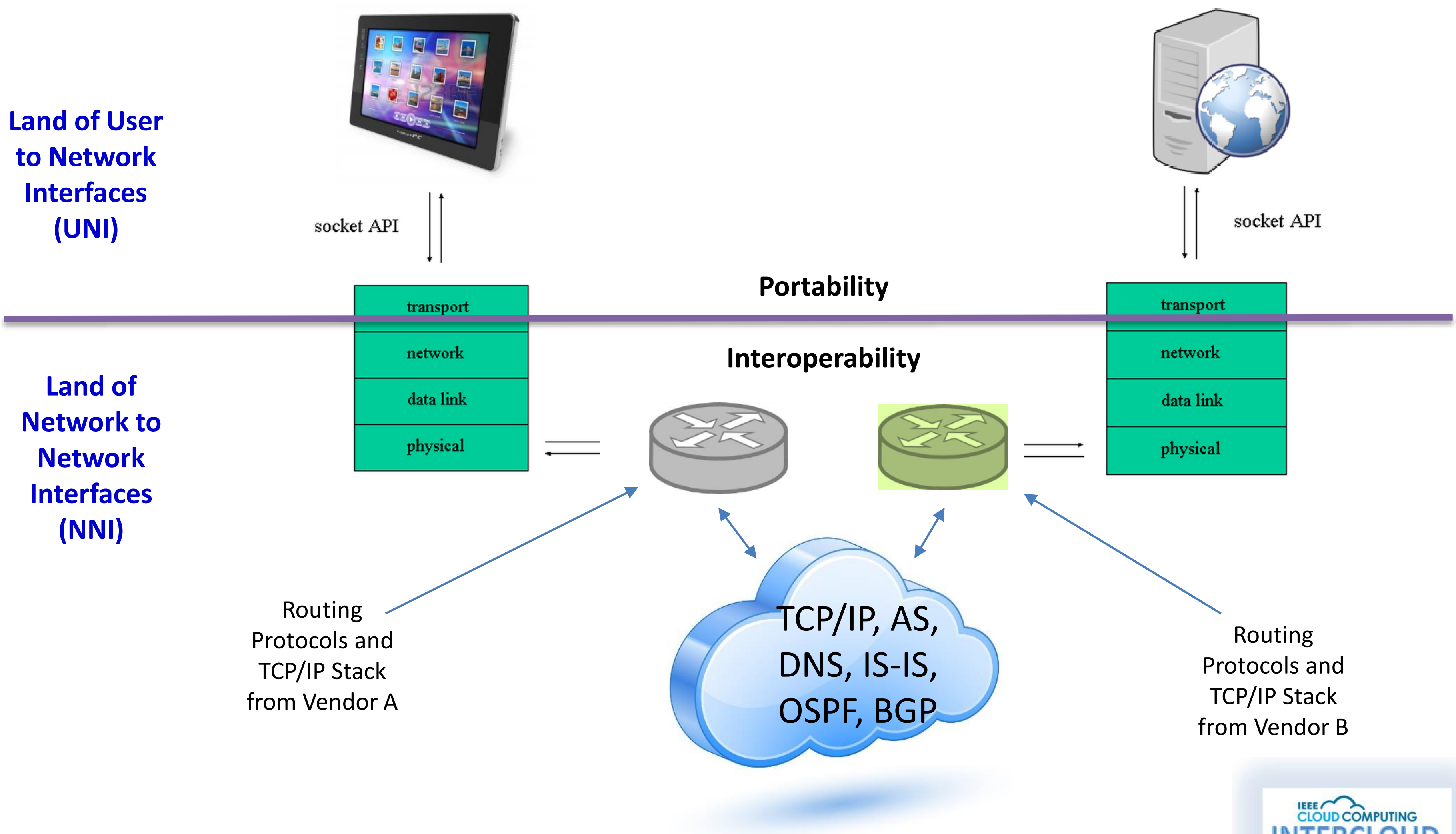
SS7 / IN Elements



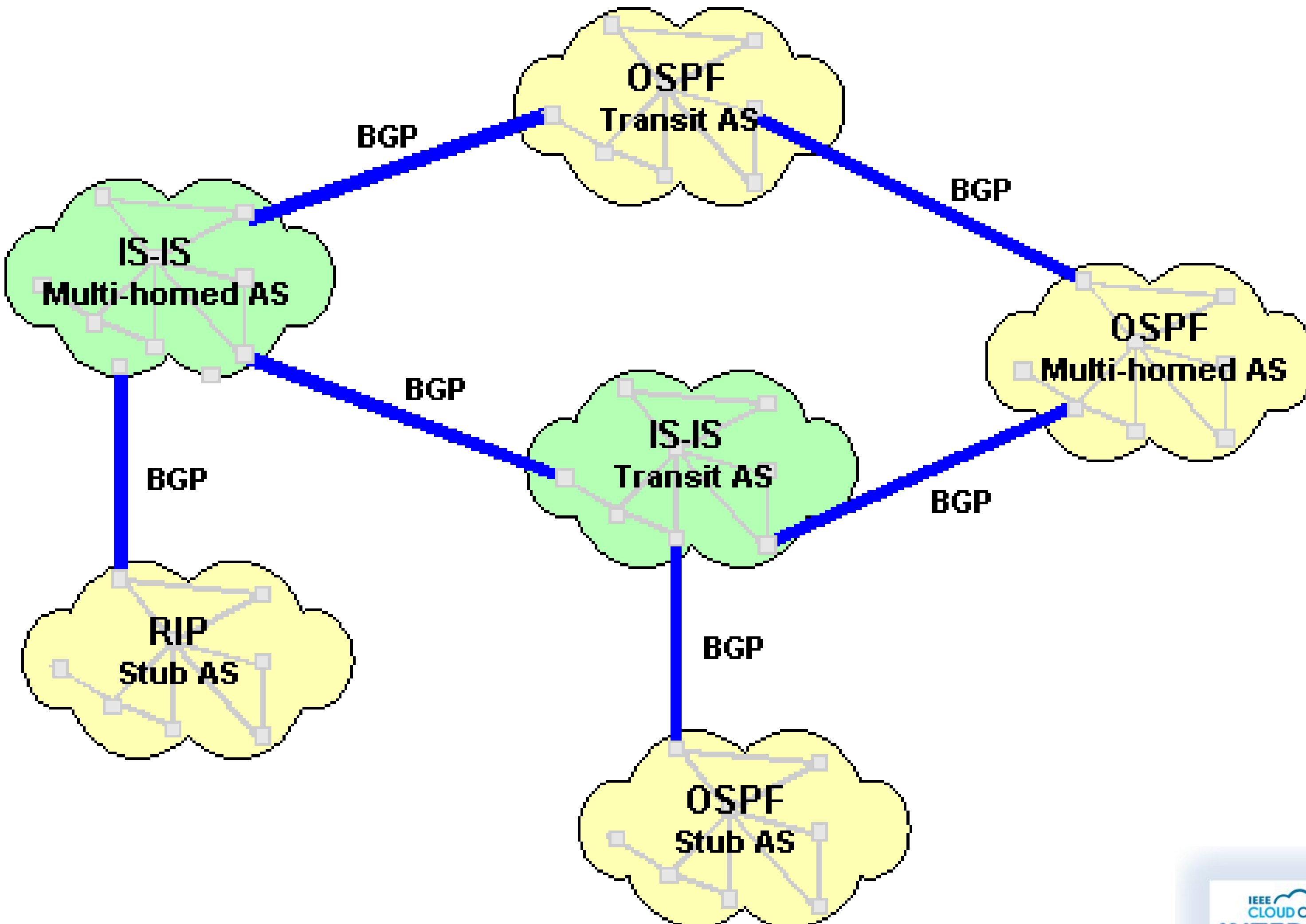


A 5ESS SS7 Capable Telco Switch

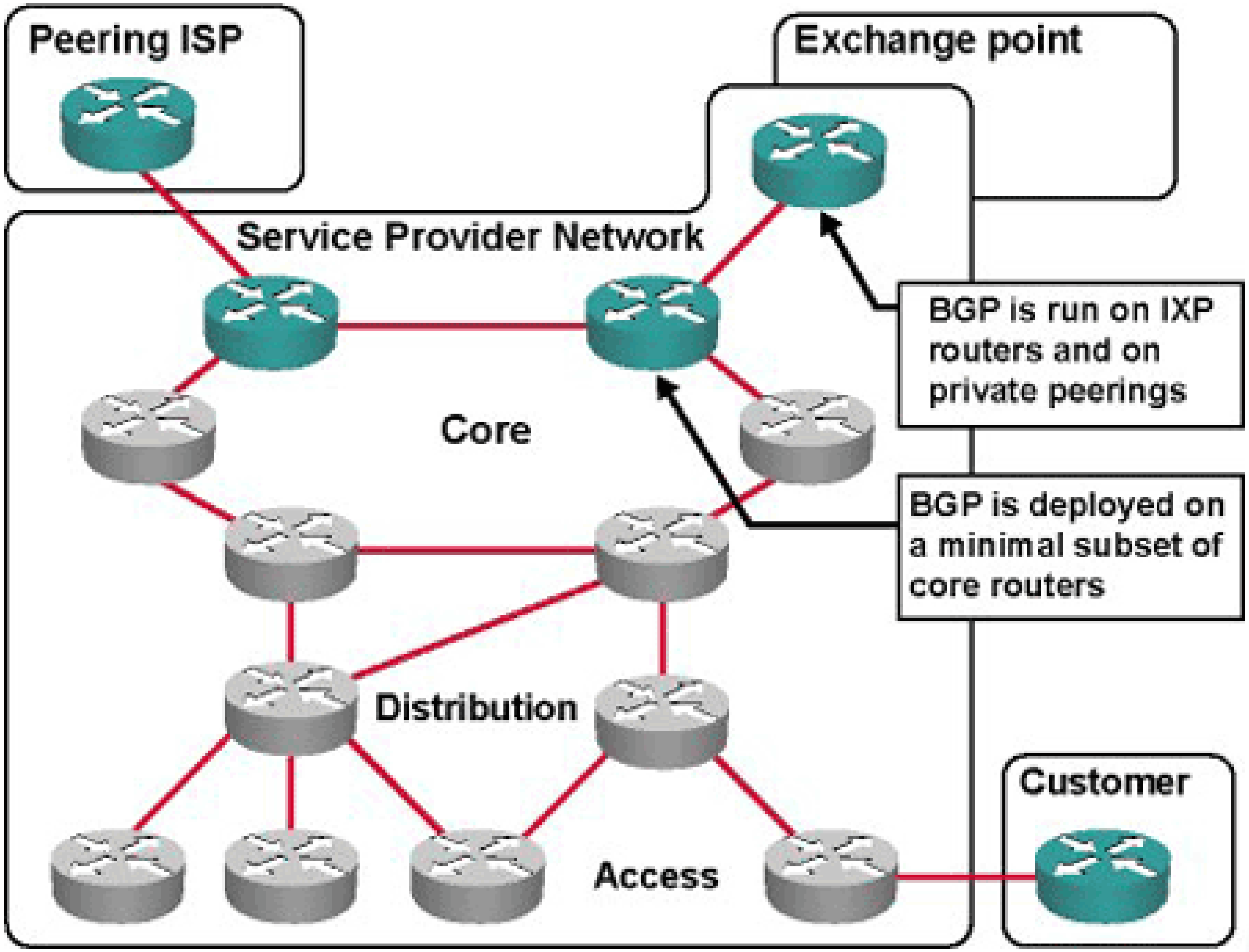
Portability and Interoperability in the Internet



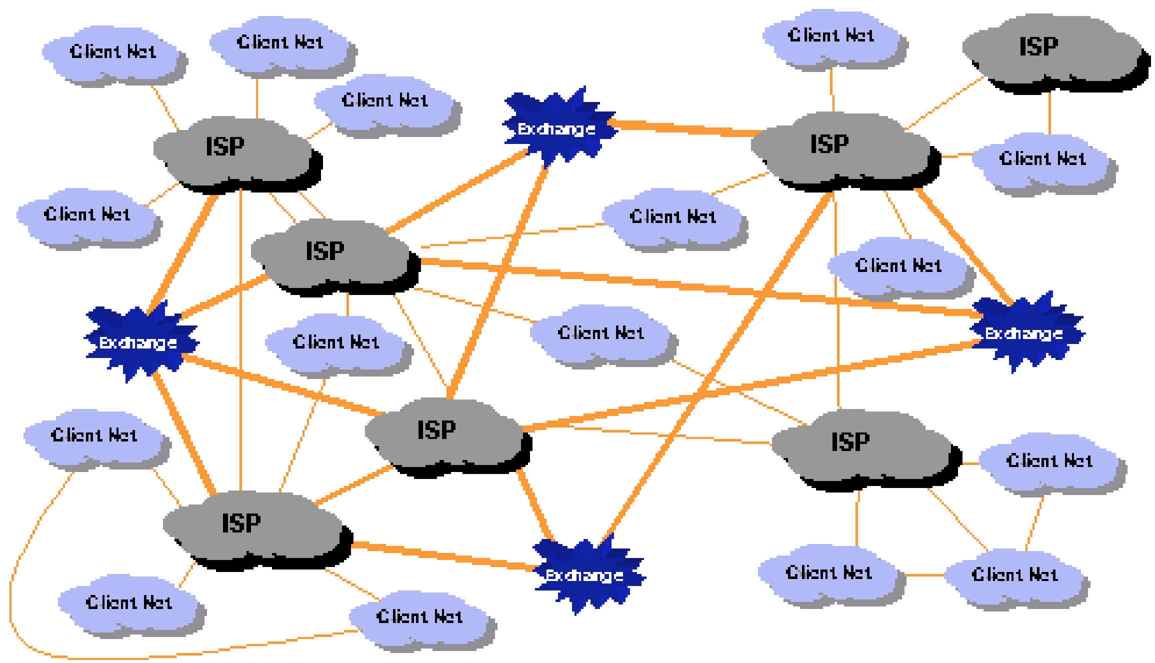
Routing Protocols in the Internet



ISP's and Routing Protocols



Multiple ISP's in the Internet

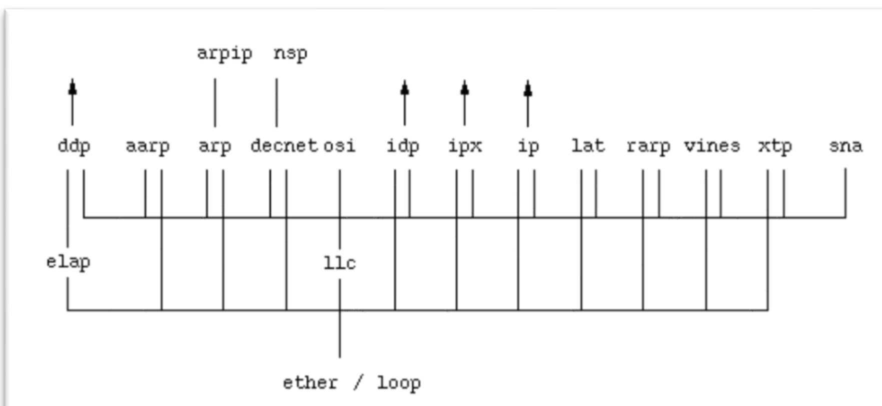


Intercloud

CLOUD INTEROPERABILITY

IEEE Intercloud Federation Concept

When Networks Connect using
UNI

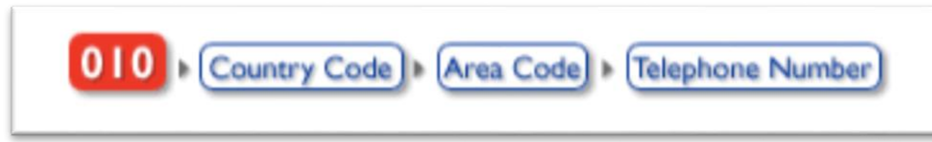


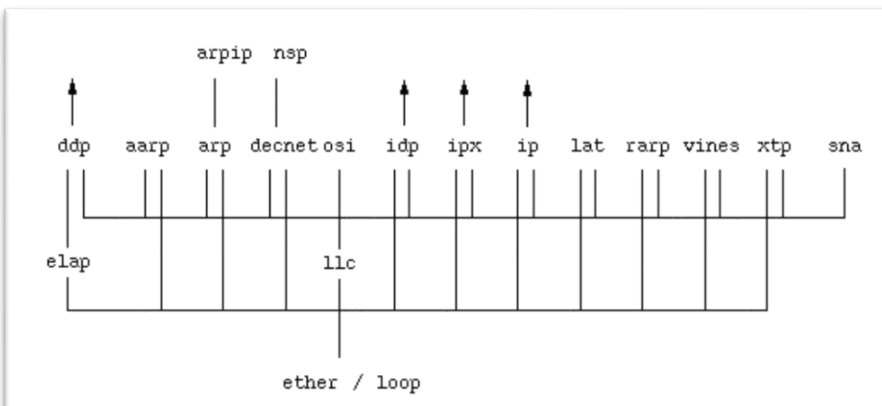
Today, this is what Cloud Computing looks like

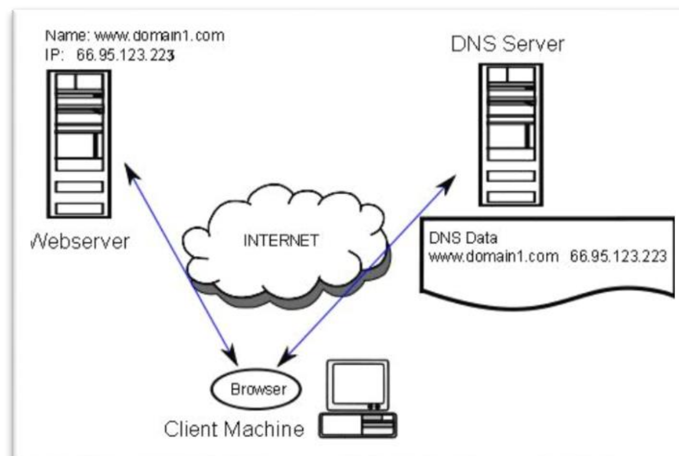
IEEE Intercloud Federation Concept

When Networks Connect using UNI

When Networks Federate using NNI Signaling

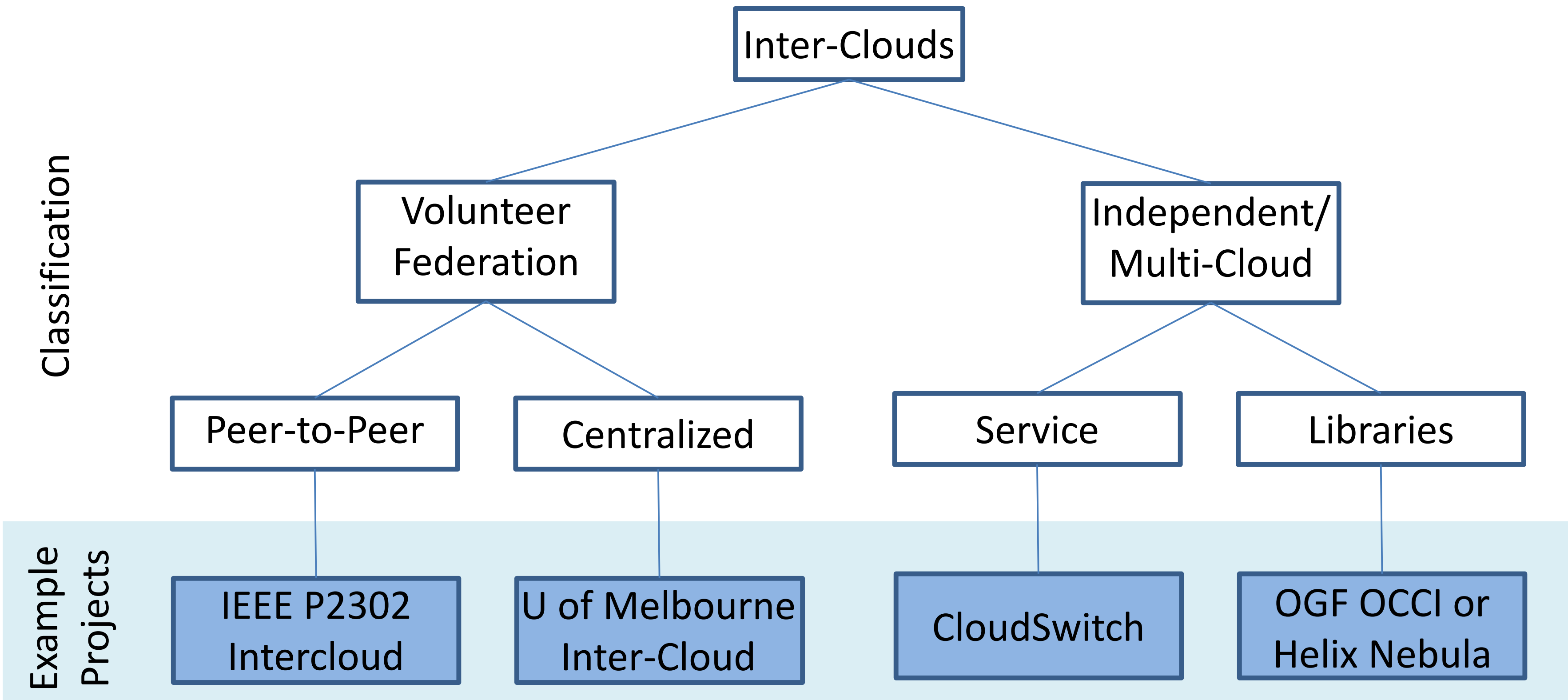






Today, this is what Cloud Computing looks like, instead of this

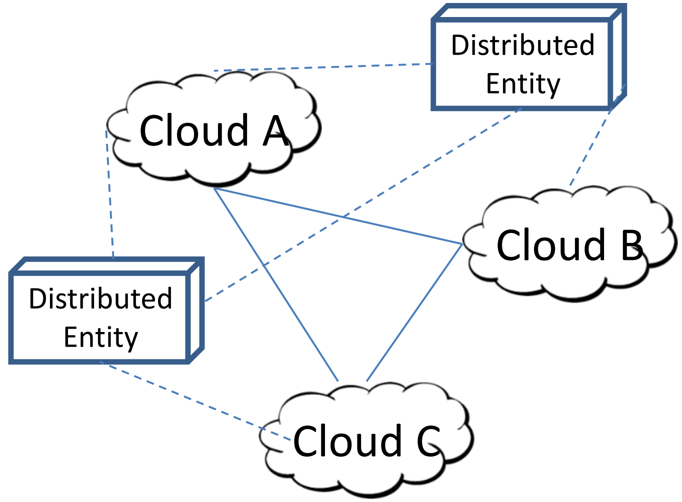
Architectural Classification of Interoperable Clouds



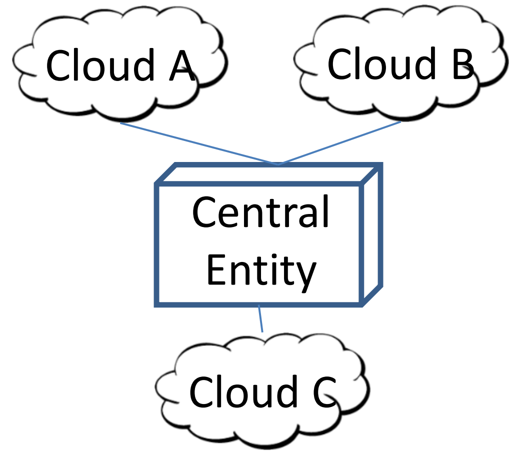
Inter-Cloud architectures and application brokering: taxonomy and survey; Nikolay Grozev and Rajkumar Buyya

Topologies - different cloud interoperability

Federations

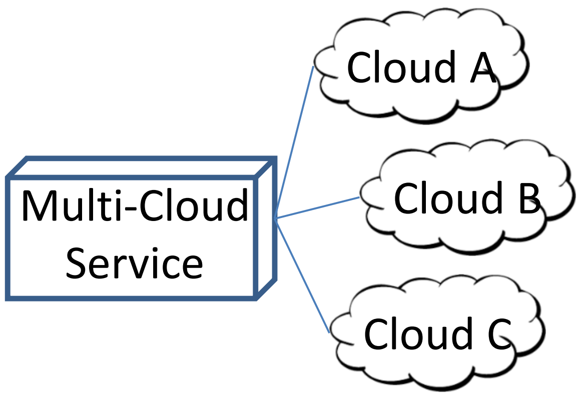


Peer-to-Peer Inter-Cloud Federation: Clouds collaborate directly with each other but may use distributed entities for directories or brokering

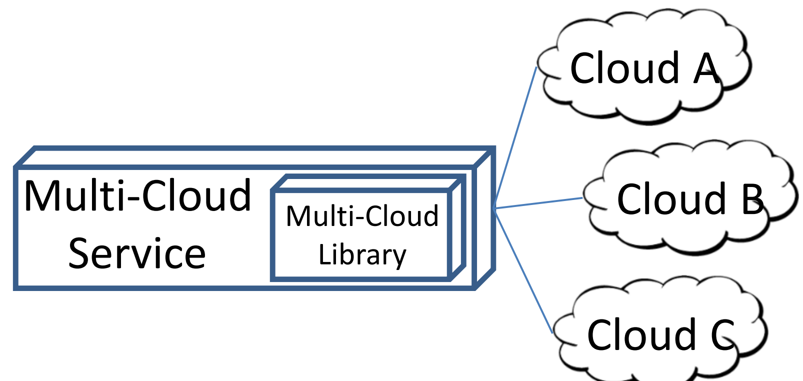


Centralized Inter-Cloud Federation: Clouds use a central entity to facilitate resource sharing

Multi-Clouds

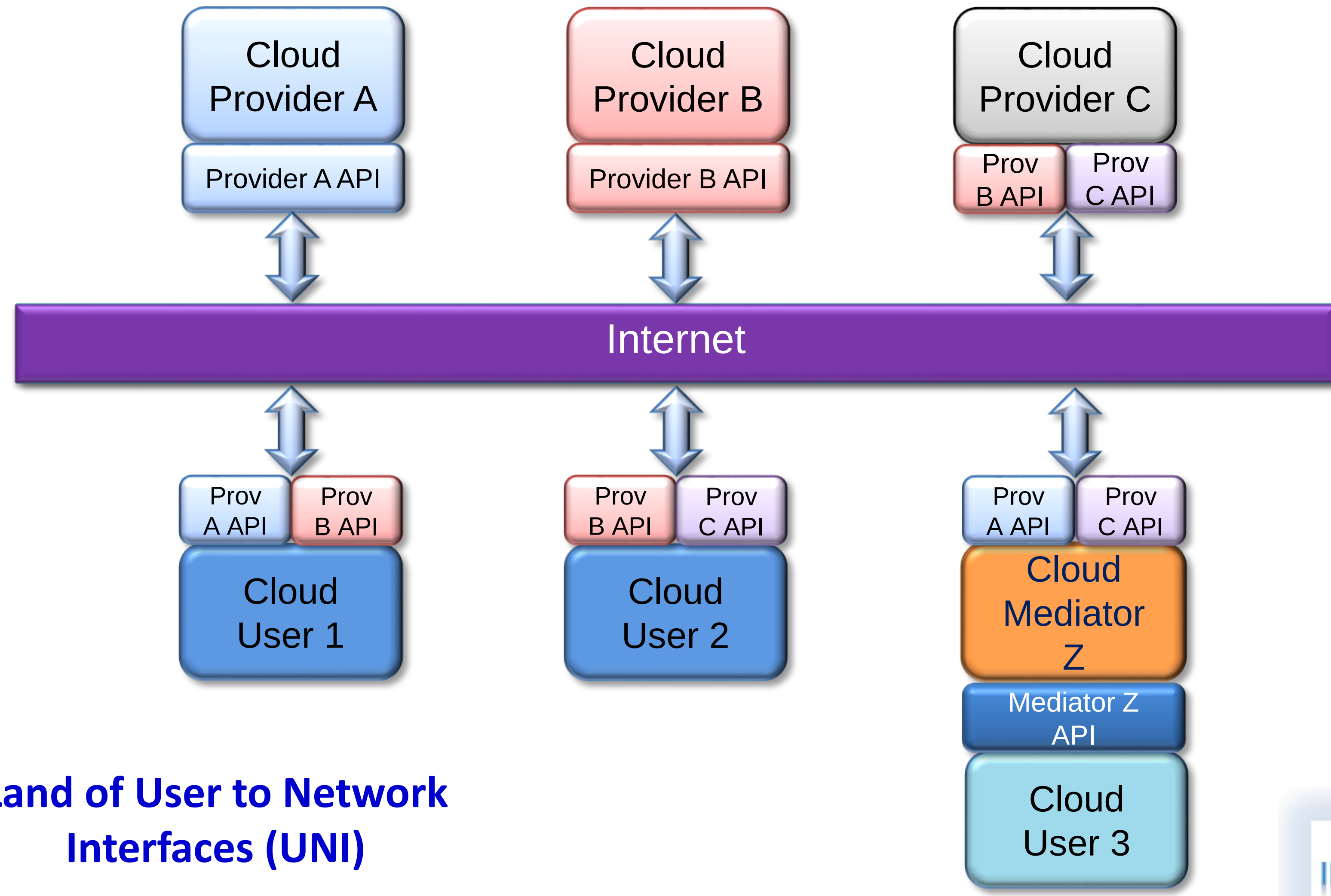


Multi-Cloud Service: Clients access Multiple clouds through a service



Multi-Cloud Library: Clients develop their own Brokers by using a unified cloud API as a library

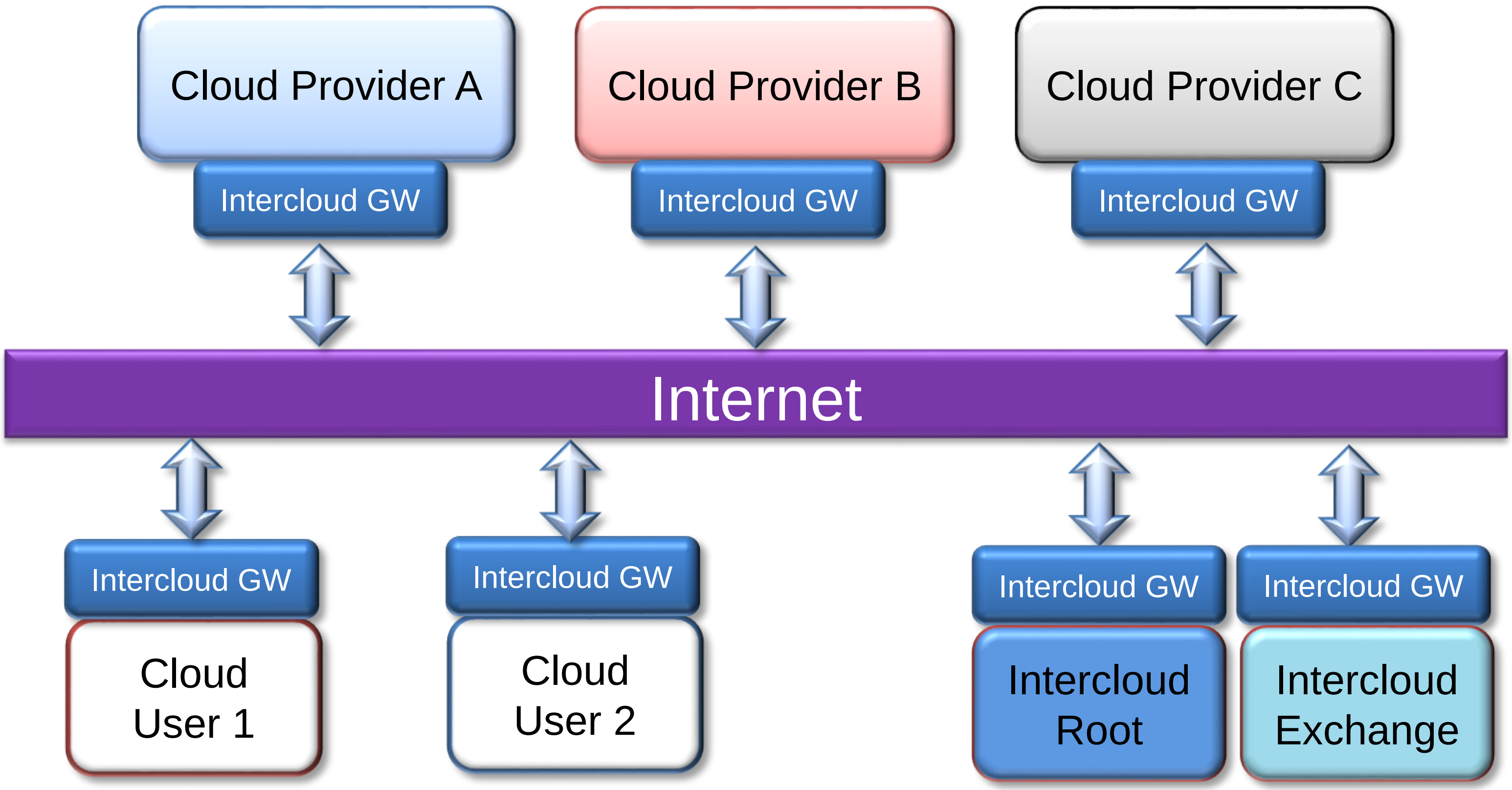
The “Multicloud” Approach



Land of User to Network Interfaces (UNI)

Some Multicloud Examples

The Intercloud Approach



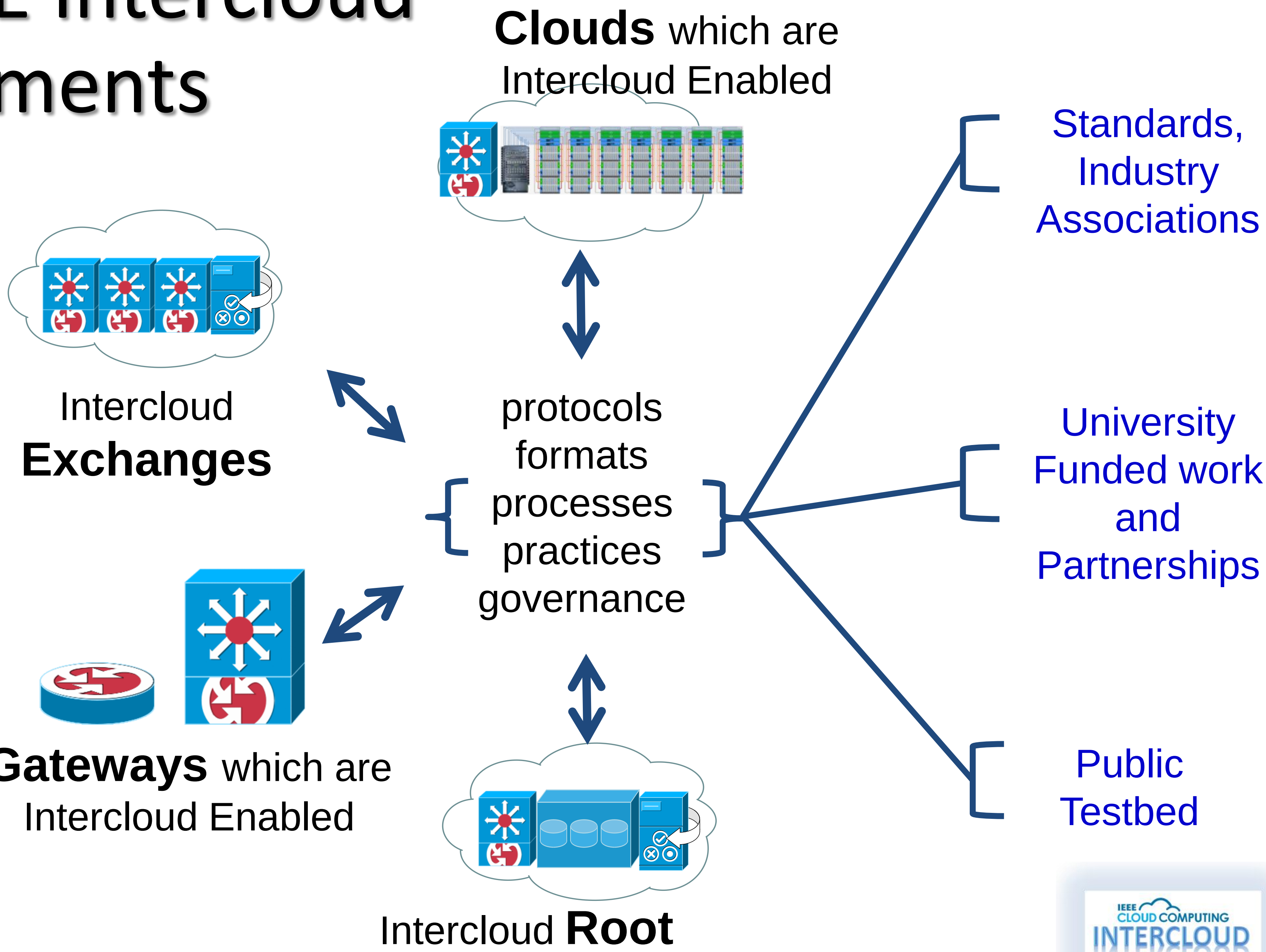
Land of Network to Network Interfaces (NNI)

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Intercloud

ARCHITECTURAL OVERVIEW

IEEE Intercloud Elements

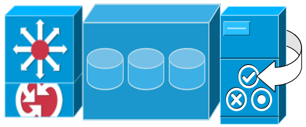


Intercloud Gateway

- Software or Appliance
  - Open Source and Adapted to Each Cloud Platform
- Supports "Common Channel Signaling" profile of Intercloud protocols and standards
  - Naming
  - Identity and Trust
  - Conversation Substrate
  - Services Transport
- Supports Cloud OS specific Federation API's and Bearer Network "Drivers"
  - Federation APIs
    - Remote Compute – Simple Remote VM Lifecycle Protocol (SRVM)
    - Storage Remoting – Simple Remote Object (SROB)
    - Storage Replication – Simple Storage Replications Protocol (SSRP)
  - Bearer Network Drivers
    - IPSEC VPN/VPC, MPLS VPN/VPC, 802.1q VLAN/VPC, RDMA/OFA Infiniband
    - TCP, UDT, and others (eg, MPI, SDP/rsockets)

Intercloud Exchanges

- Software on a Cloud Platform
  - Open Source
- Supports CCS profile of Intercloud protocols and standards via Gateway
  - Naming
  - Identity and Trust
  - Conversation Substrate
  - Services Transport
- Transient Copy of Semantic Resources Directory
- Exchange Extents
  - Trust Levels for Remote Exchange Use via Exchange Extent Policies Mechanisms
  - Replication of Semantic Resources Directories via Exchange Extent Policies Mechanisms
- Solver / Optimized Matching System
  - Supply / Demand Algorithms
- Audit Records



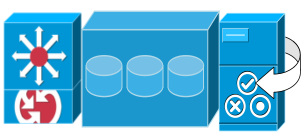
Intercloud Roots

- **Software on a Cloud Platform**
  - **Open Source**
- **Supports CCS profile of Intercloud protocols and standards via Gateway**
  - **Naming**
  - **Identity and Trust**
  - **Conversation Substrate**
  - **Services Transport**
- **Stable Copy of Semantic Resources Directory**
- **Root Extents**
  - **Trust Levels for Peer Roots via Root Extent Policies Mechanisms**
  - **Replication of Semantic Resources Directories via Root Extent Policies Mechanisms**

Reference Intercloud Components

- CS Namespace
- Conversational Substrate (XMPP)
- Transport/Services (Web Sockets)
- Trust/Identity

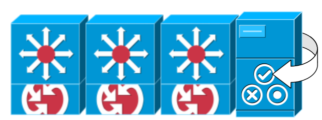
- Replication (BitTorrent)
- Semantic Directory (Ontology, RDF)



Intercloud Root

- CS Namespace
- Conversational Substrate (XMPP)
- Transport/Services (Web Sockets)
- Trust/Identity

- Replication (BitTorrent)
- Semantic Directory (Ontology, RDF)
- Solver (Hadoop/Sparql)
- Auditing



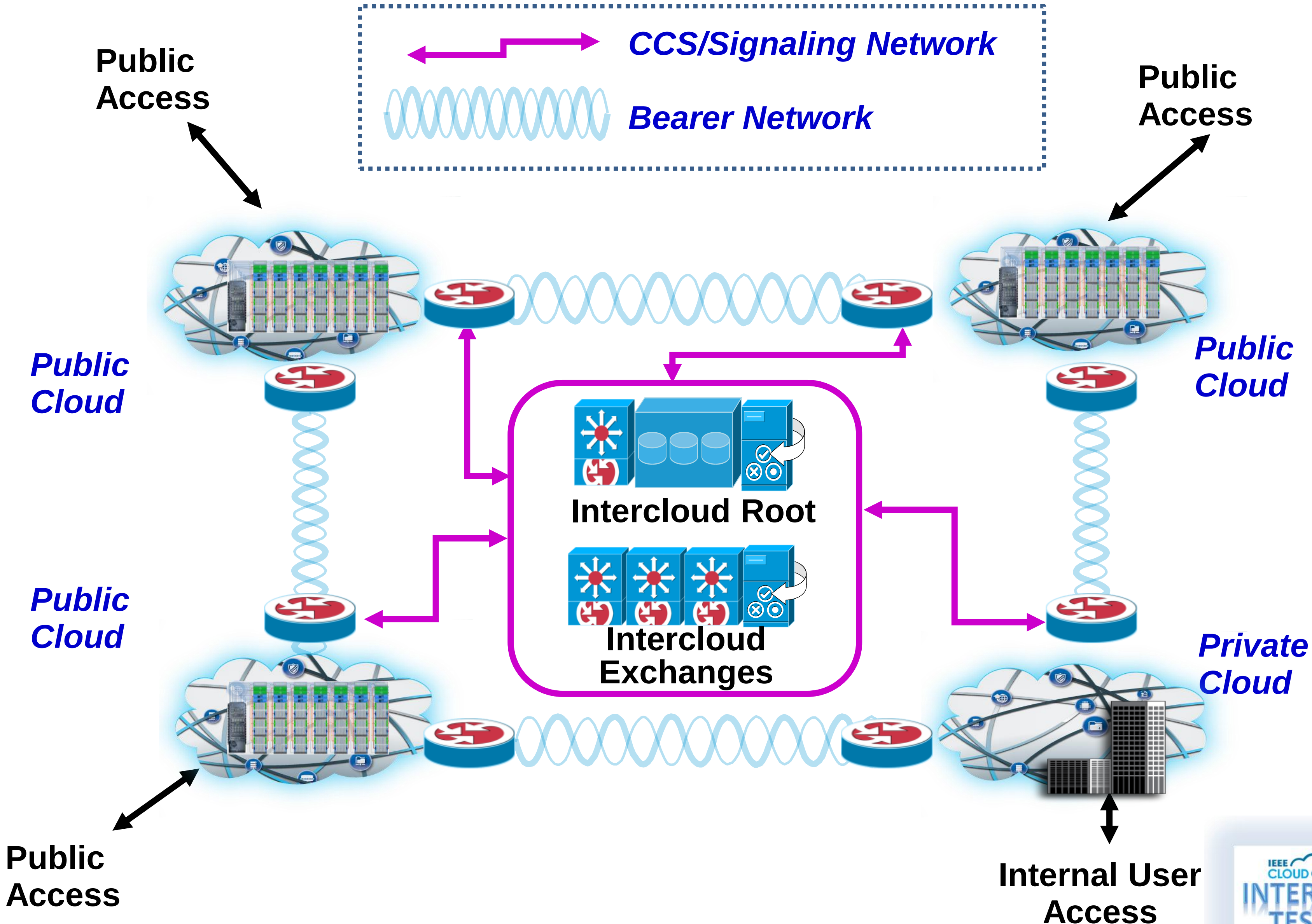
Intercloud Exchanges

- CS Namespace
- Conversational Substrate (XMPP)
- Transport/Services (Web Sockets)
- Trust/Identity

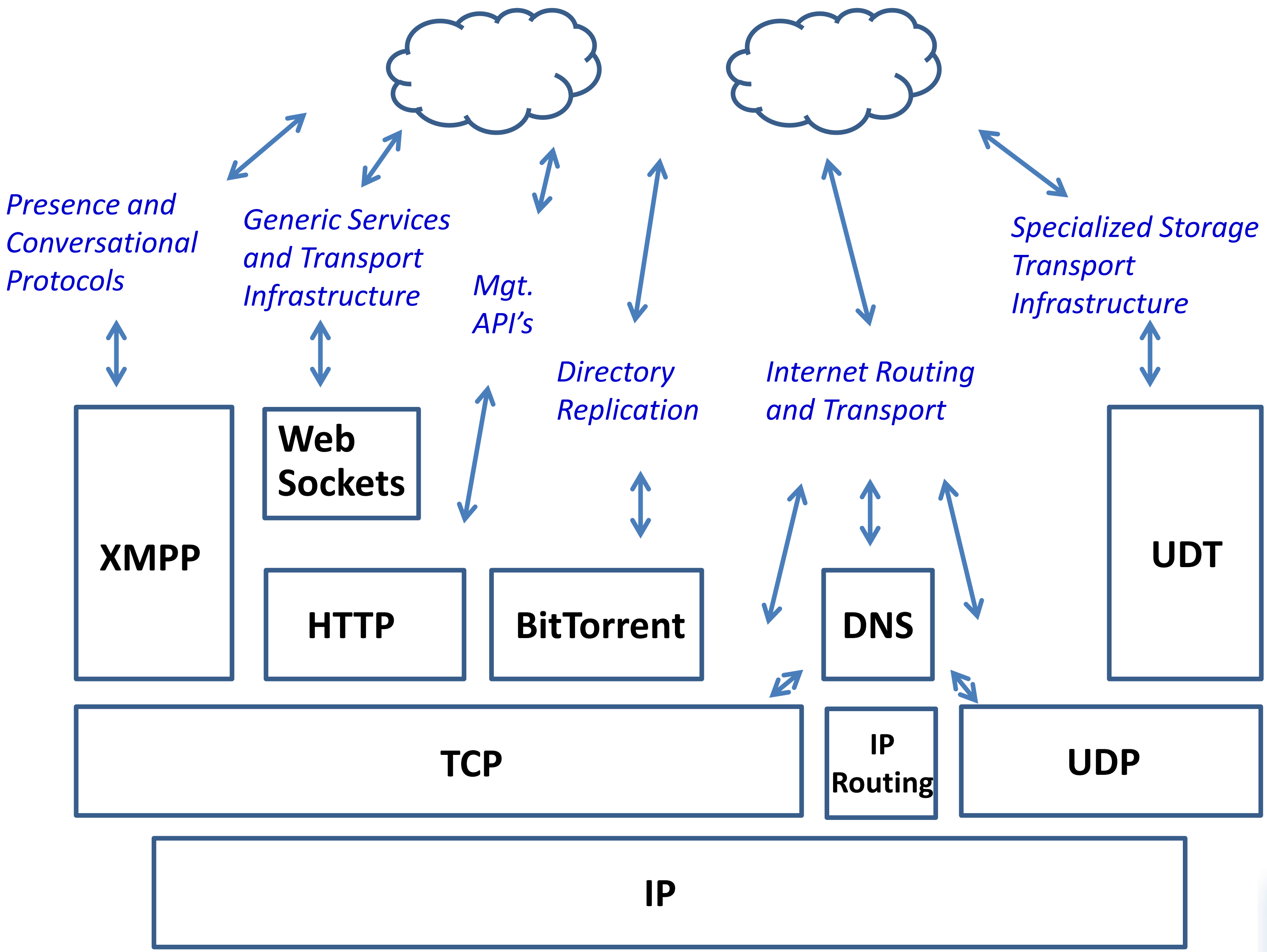
- Federating API
- Federating Transport
- Federating Implementation

Intercloud Gateway

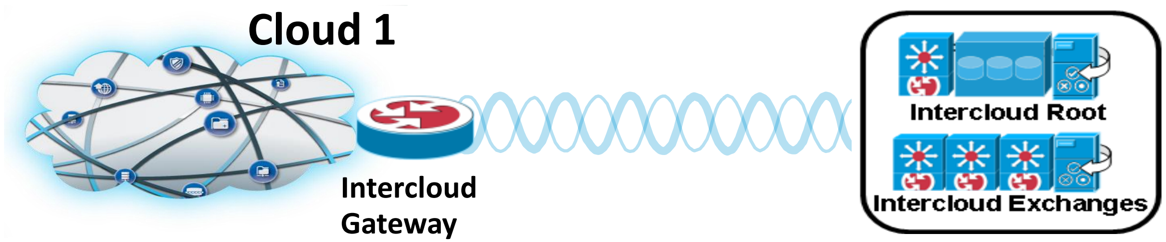
Reference Intercloud Topology



Intercloud Protocols Taxonomy



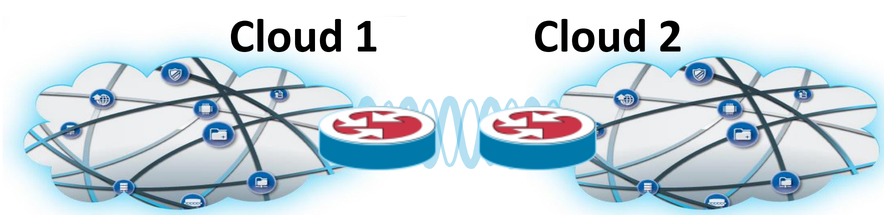
Intercloud Control Flow



Cloud 1 authenticates
→ XMPP, SAML 2.0

Cloud 1 queries “Cloud Computing Catalog”
→ XMPP, RDF/SPARQL, OWL

Cloud 1 determines the service description of another cloud that meets it’s constraints of requirements
→ XMPP



Cloud Exchange facilitates dialog between two clouds
→ XMPP based Presence & Interactive Chat Protocol

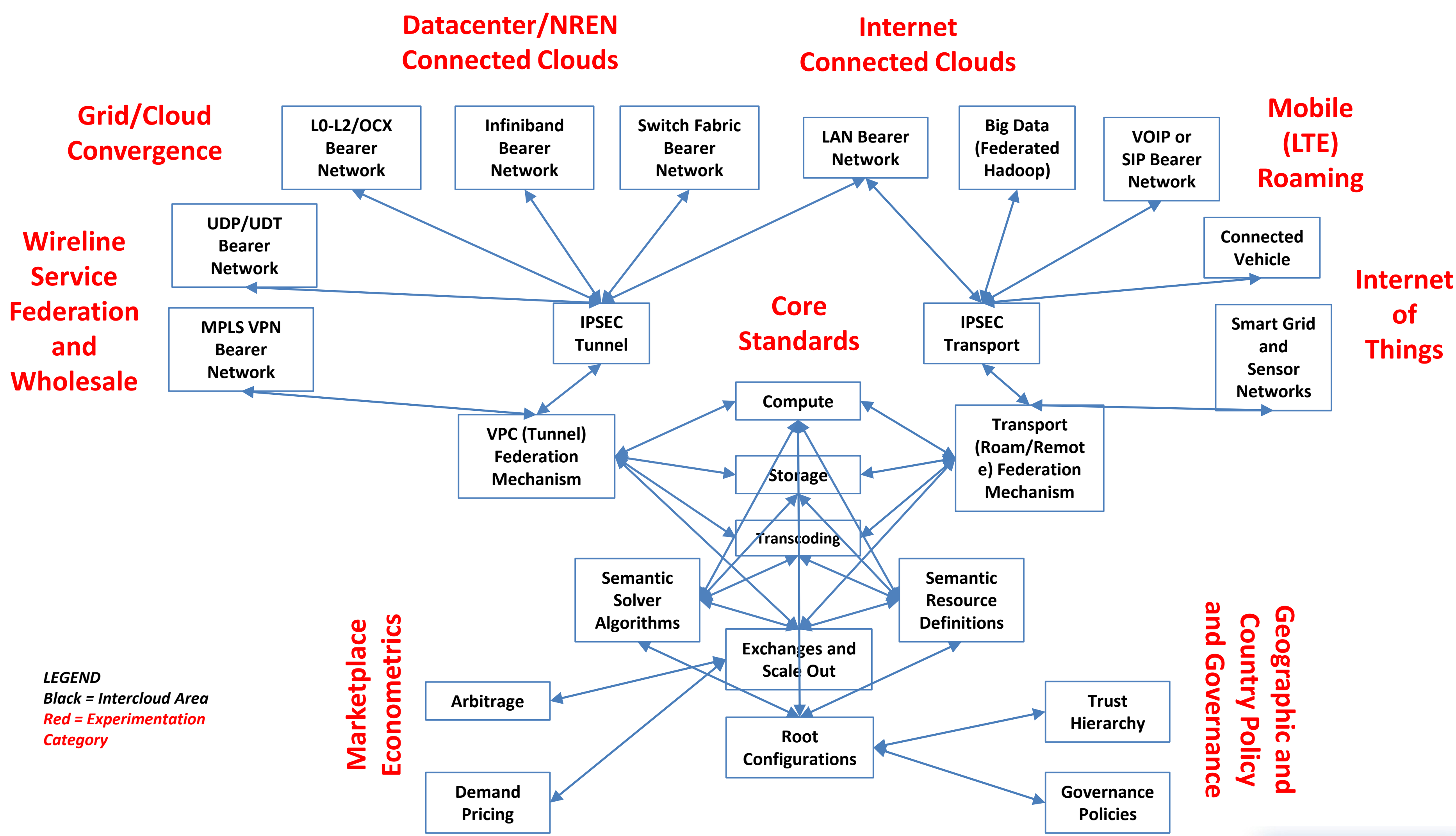
Two Clouds finalize on the Storage Replication and Access Protocols
→ XMPP

Cloud 1 starts using Cloud 2 Storage as part of the overall Federated Storage Architecture
→ Metering, SLA

Intercloud

USE CASES

Experiments Mind Map



IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Experiments Details Part 1

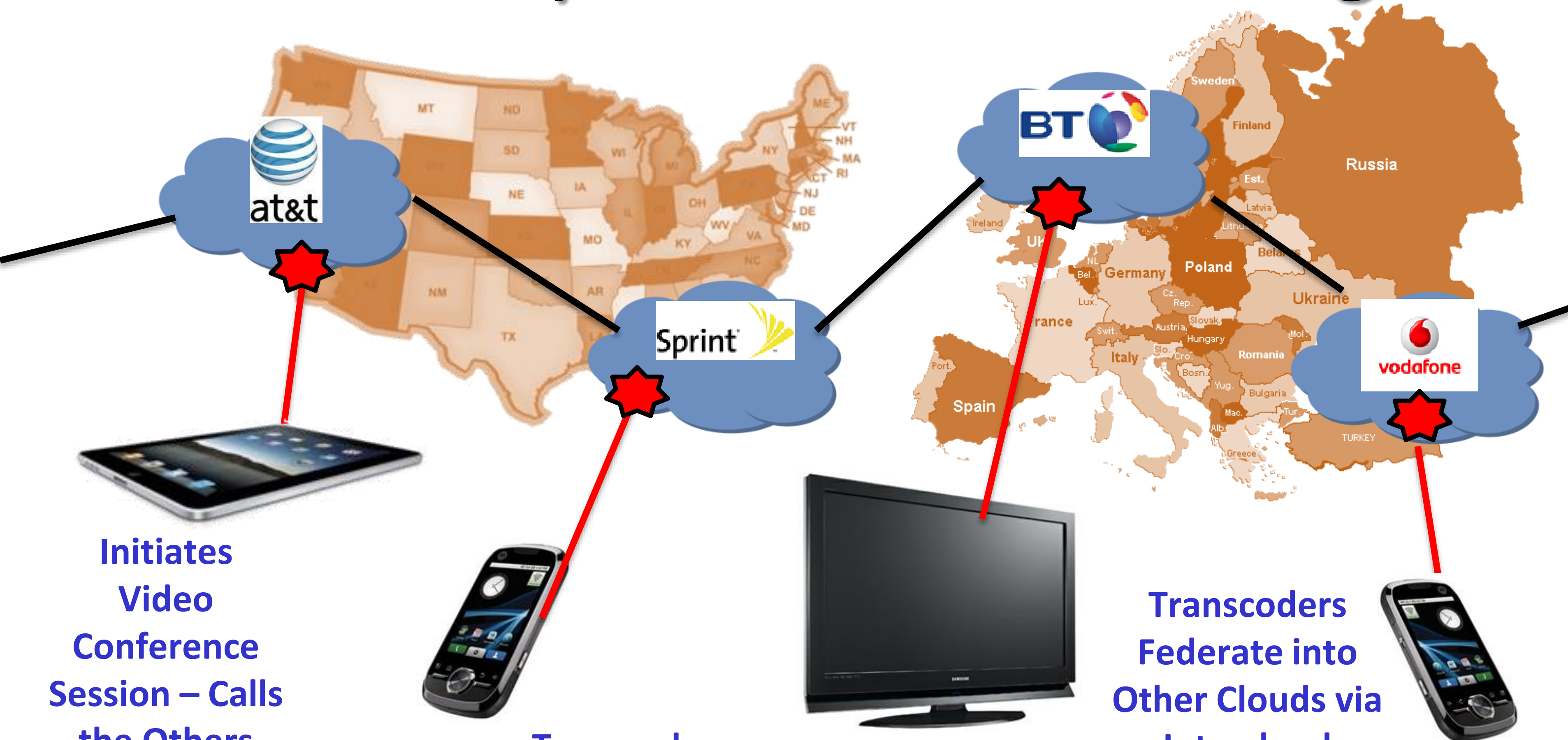
Experiment Name	Industry / Commercialization	Description
Wireline Service Federation / Intercloud MPLS Integraton	Telecommunications Carrier	Allows the inclusion of Wholesale Virtual Storage and Compute integrated with Wholesale MPLS Virtual Network Offering. Remote resources are made available in the requester's address space through the Virtual Private Cloud Mechanism (VPC). This mechanism is currently anticipated to use SDN and the most logical SDN to use in this case in OpenContrail which uses an MPLS based SDN control protocol.
Wireline / NREN Service Federation / Intercloud Regional Storage Replication	Telecommunications Carrier / Cloud Service Provider	Allows the replication of storage between providers with high bandwidth, high latency (long distance high speed fiber for example) by transiting VPC based VPN base Storage federation over for example an IPSEC tunnel but instead of using TCP with it's known poor high bandwidth, high latency performance using a UDP based protocol such as UDT. Applies well to NREN based interconnects.
Open Cloud Exchange style (L0-L2) based Intercloud Federation	Research Clouds, Grid/Cloud Convergence	Allows the use of LAN in-Datacenter or Metro-E connected clouds to federate directly (with or without IPSEC tunneling, depending on address space and security needs) using shared Ethernet transport (so-called Open Cloud Exchange style). Attractive way to stretch Grid/HPC applications adapted to Clouds proper, across more than one cloud without using Multi-cloud techniques.
Infiniband based Intercloud Federation	Research Clouds, Grid/Cloud Convergence	Speculative Experimentation on using Infiniband to connect trusted clouds for shared memory pooling as a type of Intercloud Federation. Requires investigation of shard memory type as semantic resource and also requires HPC/Grid type Clouds to interconnect intimately via Infiniband. As Infiniband extends over Ethernet may lead to a novel way to dynamically construct shared memory spaces across Intercloud.
Distributed Switch Fabric based Intercloud Federation	Storage Service Provider feature of Virtual Portable Data	Experimentation with so-called "Powered by Peak Extended Switch Fabric" style network where Storage Service Provider co-locate swith multiple cloud providers in a datacenter and connects storage to clouds via shared distributes switch fabric. Data in Storage Service Provider can virtually appear to move/be available to multiple Cloud Platforms with consistency ensured through Intercloud Federation.
LAN based Intercloud Federation (requester address space aka VPC)	Telecommunications Carrier / Cloud Service Provider	Canonical case of Intecloud Federation of Storage and Compute using overlay IPSEC Virtual Network. Remote resources are made available in the requester's address space through VPC. This mechanism is currently anticipated to use SDN. OpenContrail (which uses an MPLS based SDN control protocol) is the only open source SDN may not be the only choice given MPLS is not used, might want to use OpenFlow.
LAN based Intercloud Federation (provider address space aka Roaming)	Telecommunications Carrier / Cloud Service Provider	Variation on canonical Intercloud Federation using overlay IPSEC Transport (not Tunneling) mode. Same as above case but Remote resource are made available in the providers address space. This is a roaming use case where the requester knows they are running on foreign resources (nevertheless automatically federated to them), there is application specific code to address this. SDN is utilized but inverted.
LAN based Intercloud Federation for Big Data	Hybrid and/or public cloud applicable	Intercloud Federation as in the Roaming use case with application specific code (in this case, distributed Map Reduce/Hadoop) to control and sweep back the multiple Big Data nodes forming a cloud-spanning Intercloud platform for limitless Big Data

Experiments Details Part 2

Experiment Name	Industry / Commercialization	Description
VOIP or SIP / Distributed Transcoder based Intercloud Federation for Voice & Video; Fixed & Mobile variations	OTT Internet Phone apps, or LTE Network Substrate	Intercloud Federation as in the Roaming use case except using additional layer of VOIP or SIP protocols to set up multiple cooperating endpoints for voice or video calls. Use case requires development of transcoder as a federated resource. Applications specific code from call set up to adaptive transcoder provisioning. Enablement of Location Based information into Intercloud Requester data structures if access to the back of a Carrier LTE network can be had. So called specific code can be placed in user space or an investigation of the Intercloud XMPP substrate with the LTE SIP substrate will be in order to more firmly mobile enable the very design core of Intercloud Federation
Intercloud Federation for Internet of Things, Connected Vehicle Example	Automobile with Carrier Partner	Intercloud Federation as in the Roaming use case, leveraging the mobile integration, experimenting with applications in Connected Vehicle. Self driving via Local Cloud and Carrier control is a must. Extension to safety features including multiple auto dynamics (traveling in packs, collision avoidance) as well as augmented reality (Intercloud Federated Deep Web) for concierge, shopping, and advertising.
Intercloud Federation for Internet of Things, Smart Grid/Sensors Example	Utilities companies, smart homes, Carrier Partner	Intercloud Federation as in the Roaming use case, but where the smart elements (meters/sensors) are static location. Because broadband networks are asymmetric (much slower upload), in commercial deployments of smart grids/meters, carriers provide lower cost off hour network access (Wired, SMS, 3G/4G data) for upload. This causes small time windows for uploads and processing. It is well known that in many sensor applications burst windows are major processing challenges. Most sensor processing uses event processing models (Storm/Kafka, or in-memory sale-out DBs like VoltDB) to process sensor inputs. This experiment examines burst window dynamic Intercloud federation for supporting burst window capacities of event processing models.
Core Intercloud Standards Framework	Financial Commoditization of Cloud	In order to commoditize Cloud Resources suitable for automated exchange based commerce, both object and method standard must be put into place, eg, semantic resource ontologies and core federation transports and mechanisms. This effort formalizes these in the context of automated commodity exchange mechanism.
Geographic Governance Models	Policies of Countries w.r.t. Intercloud Governance; Cloud as National Asset	There are various Trust Hierarchy/Proxy models for Intercloud, starting with “local” and “foreign” exchange (think local and long distance telephone call model) all the way to geographical or economic/political. This experiment models the effects and impacts of Governance/Policy Experimentation for Societal and Sustainability outcome analysis
Commodities Trading Enablement	Financial Commoditization of Cloud	Simple “Peering” as exchange (original Internet model) is surely not how cloud resources will be federated. This experiment involves enablement of the core Intercloud Platform with ability for multiple extension and algorithmic experimentation to enable automated commodities of cloud resources through Intercloud Federation. The result will be creation of new Industries/Business Models of Clouds.
Intercloud Advanced Econometrics	Financial Commoditization of Cloud	This experiment builds on the Commodities Trading Enablement experiments to model several so called exotic Economic Model Experiments including Trading, Arbitrage, Derivatives, Hedging, and Volatility

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

One Intercloud Use Case
Multi-Party Video Conferencing



Initiates Video Conference Session – Calls the Others

Transcoders are made to run near end devices

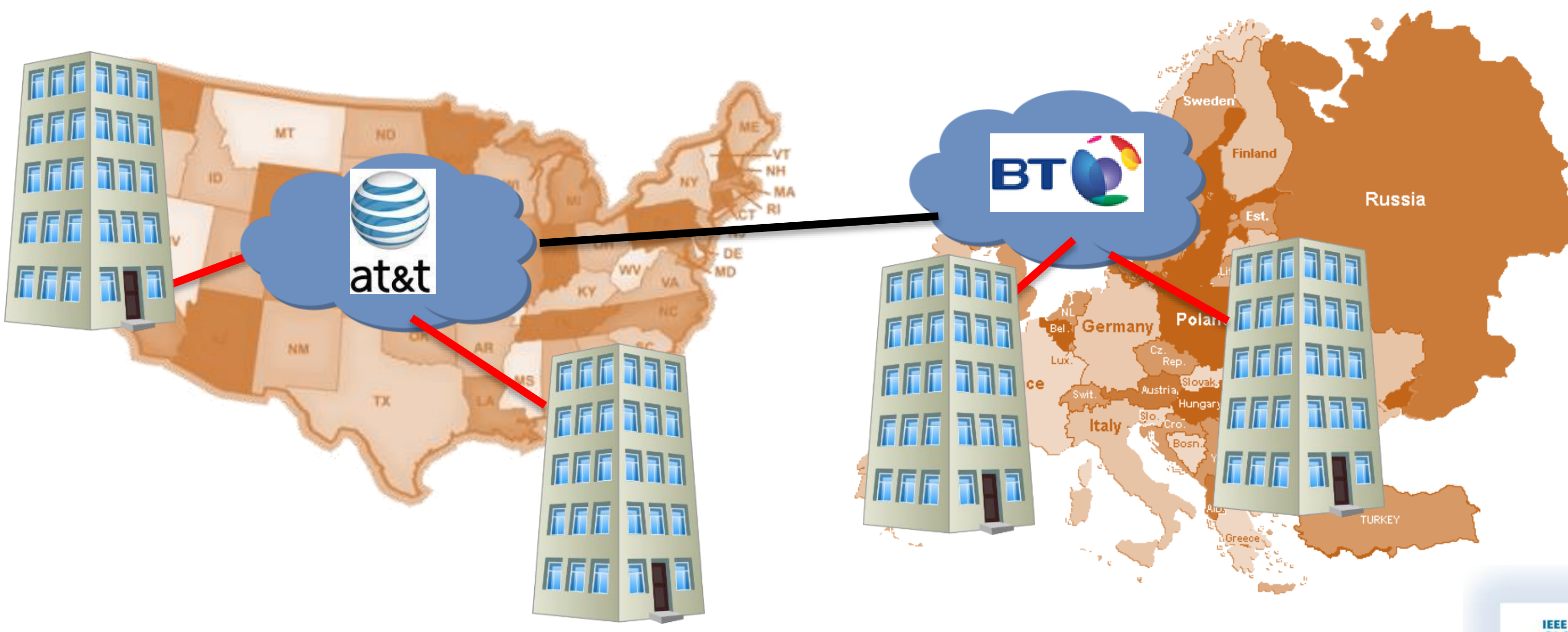
Transcoders Federate into Other Clouds via Intercloud

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Another Intercloud Use Case
Wholesale Computing/Storage with MPLS

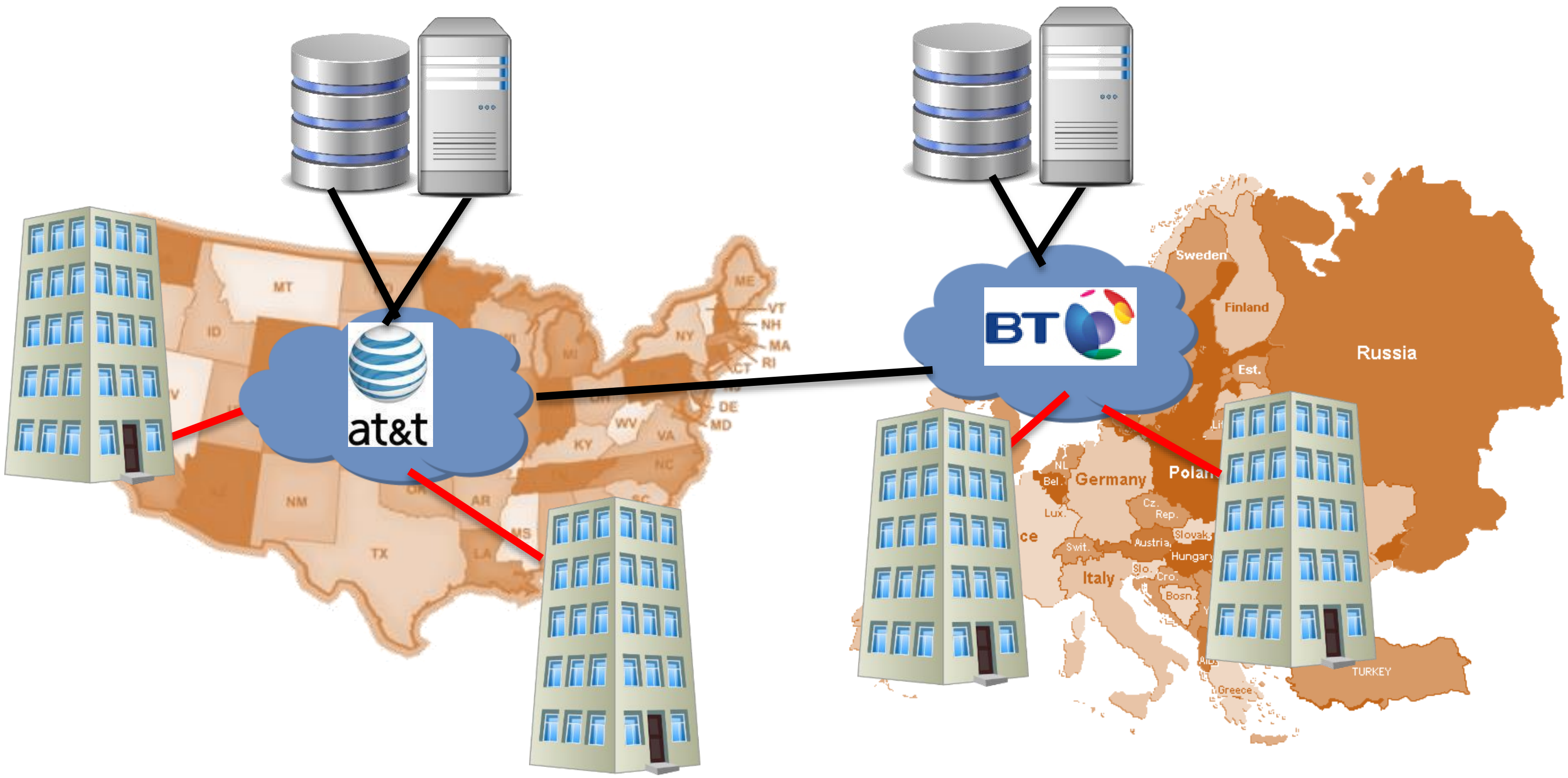
US Carrier provides VPN to multi-location Corporation via MPLS using it's own network infrastructure

US Carrier provides "US VPN" to multi-location Corporation via MPLS via Wholesale of partner network



Another Intercloud Use Case

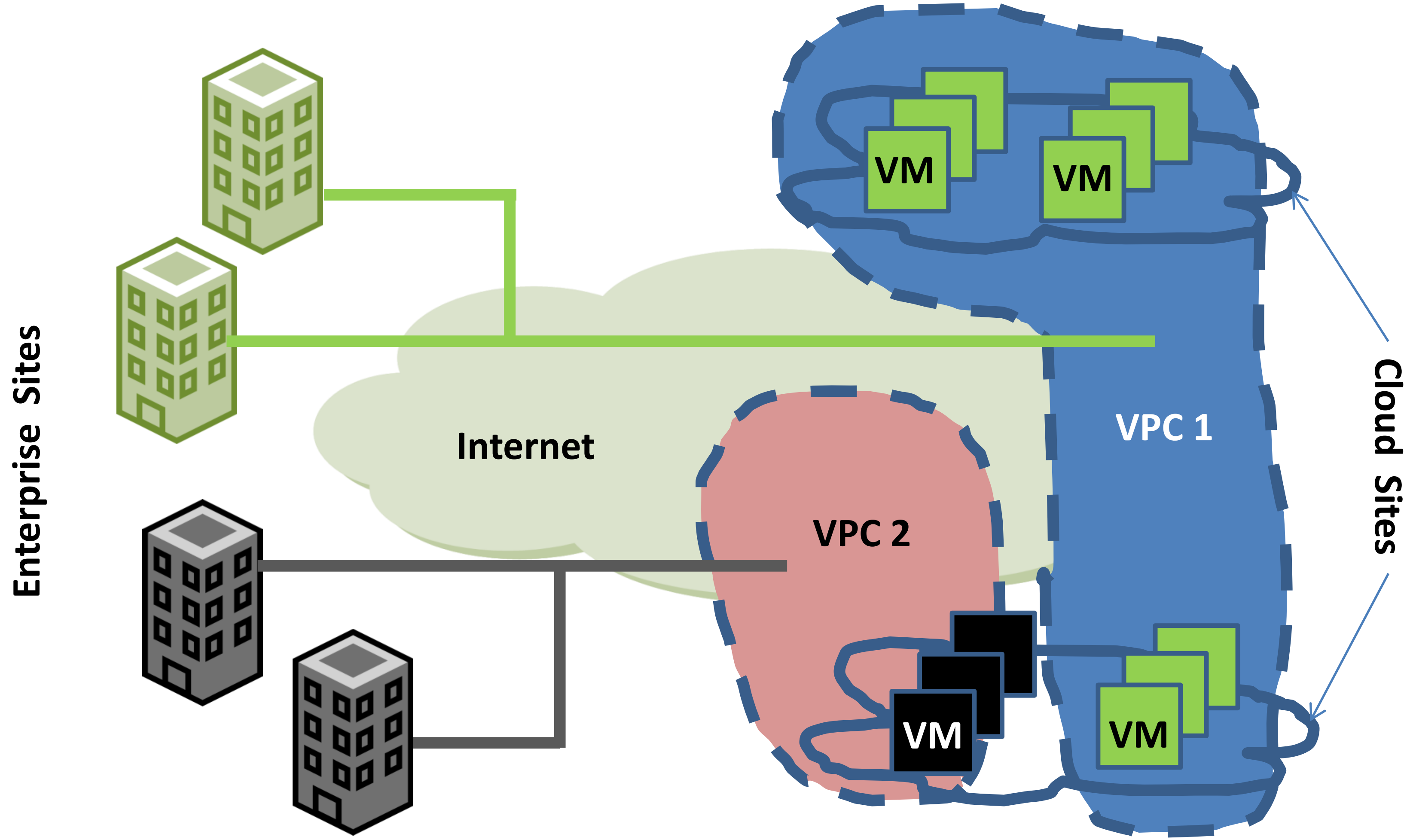
Wholesale Computing/Storage with MPLS



Cloud Services such as Compute and Storage can ALSO be Wholesaled by US Carrier through the MPLS VPN in area where they don't operate infrastructure

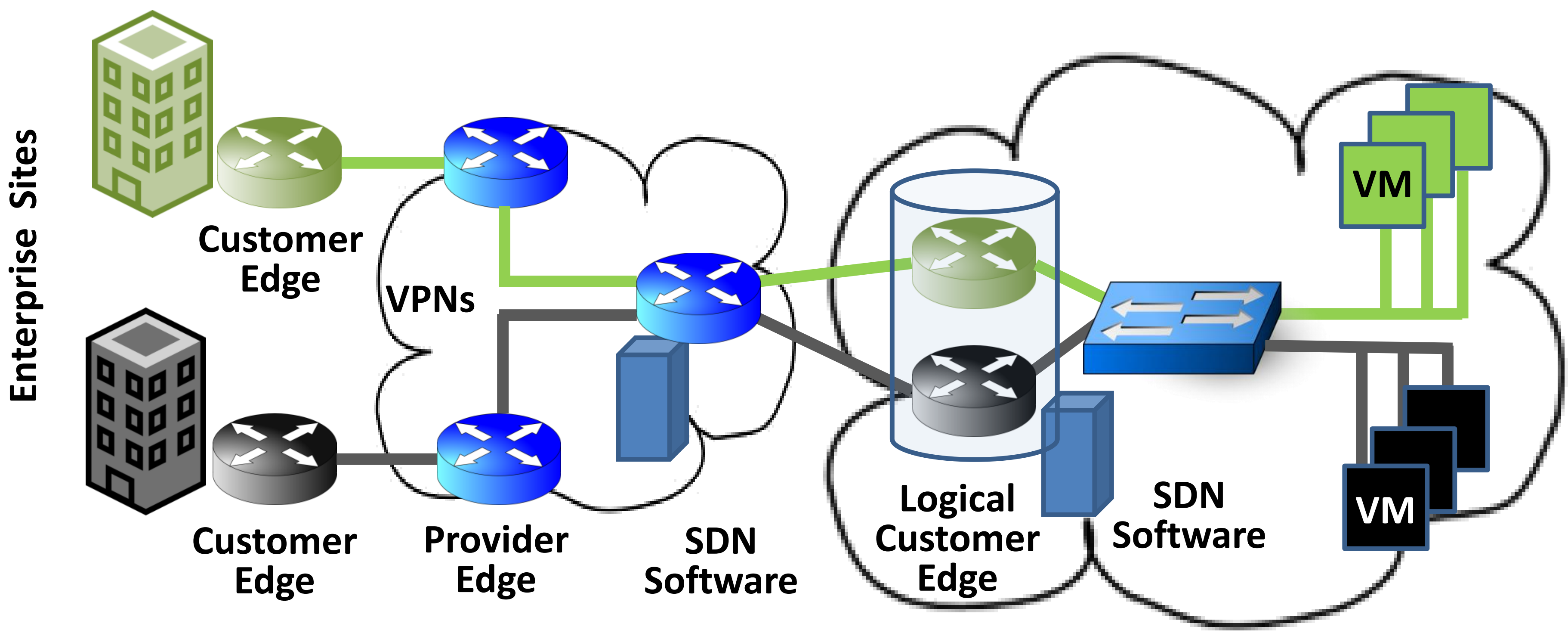
IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Multiple VPC Federation

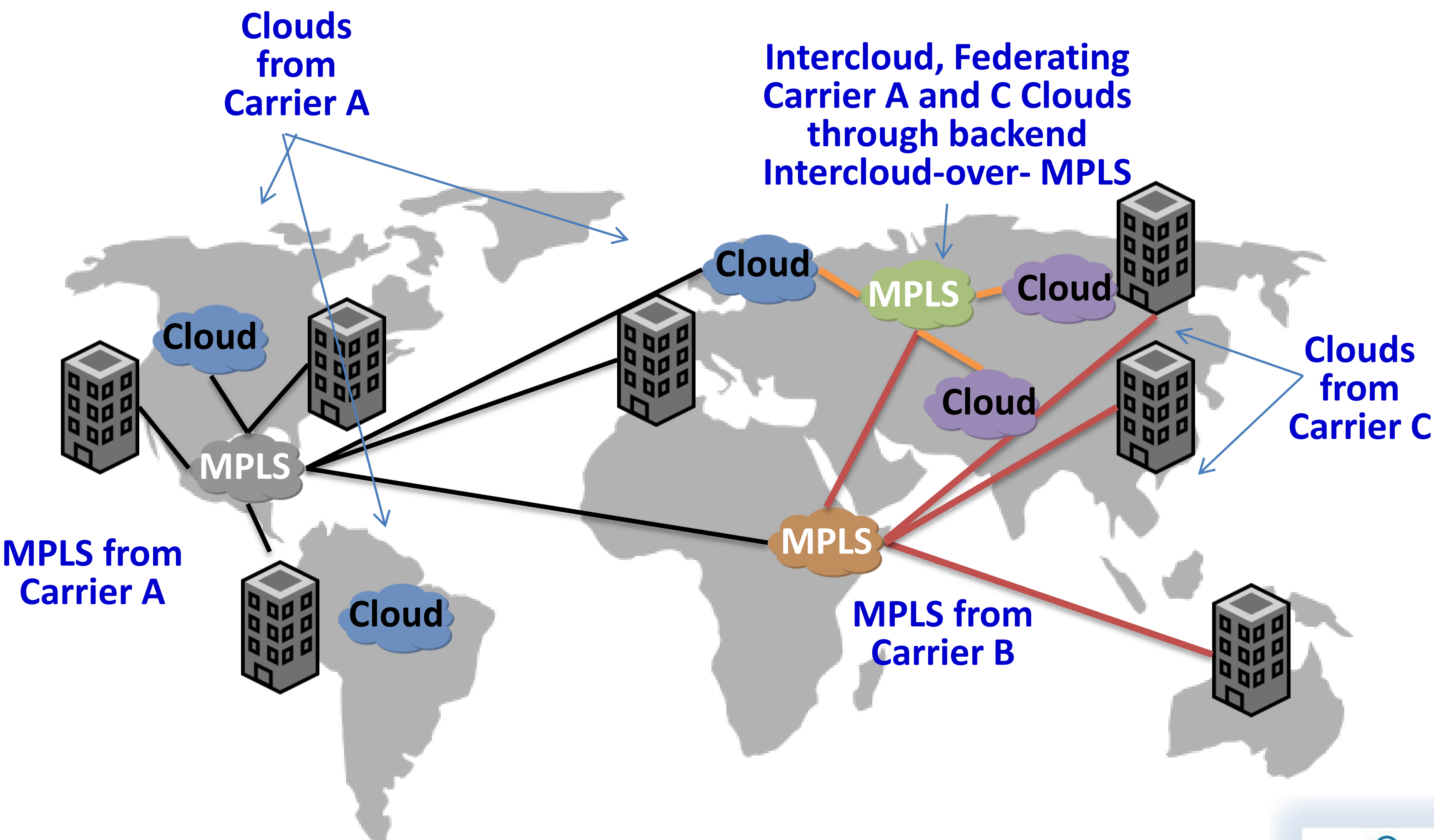


Two VPCs isolate resources within the cloud sites and securely link them to enterprise networks

Multiple VPC Federation Mechanism



Multi Carrier MPLC/VPN Federation



Intercloud

NAMESPACE AND GOVERNANCE

Namespace Overview

- Proposal/Thoughts
- Need to write a new RFC for Cloud System Number based (CS)
  - Conceptually similar to Autonomous System (AS) Numbering
  - RFC 1771 (16 bit spec) and RFC 1930 (32 bit spec)
- CSNs are 64 bit signed integers
  - Maximum of $7FFF,FFFF,FFFF,FFFF_{16}$ = 9,223,372,036,854,775,807 Cloud Systems
  - Or put another way, every IPv6 endpoint can be a Cloud System
- Reserved Numbers
  - The first and last CSNs of the original 16-bit integers, namely 0 and 65,535, and the last CSN of the 32-bit numbers, namely 4,294,967,295 are reserved and should not be used by operators.
  - CSNs 64,512 to 65,534 of the lower 16-bit CS range, and 4,200,000,000 to 4,294,967,294 of the 32-bit range are reserved for Private Use (following RFC 6996), meaning they can be used internally but should not be announced to the global Internet.
  - All other CSNs are subject to assignment via Governance / Registration Authority

Namespace Details

- Representation of CS Numbers
  - Conceptually similar to ASN scheme as defined in RFC 5396
  - Textual representation of "csplain" (simple integer form)
- URN CSN Designation
  - <nnnn>.csn.intercloud.net
  - Eg, 4.csn.intercloud.net is CSN 4
- Possible XMPP Mapping using DNS
  - XMPP naming is very flexible, depending on the services at the target end to figure out what exactly is being asked for
  - XMPP supports names in inbox form such as foo@example.com, but also <foo>@example.com, ::foo::@example.com, foo@example.com/service, or service@foo@example.com
  - Utilize Name Authority Pointer ("NAPTR") DNS Resource Record 35 (RFC 3403) to map CSN URN to CSN XMPP Service Endpoint
  - Using an example of Megacloud Inc. with CSN 4, a NAPTR query of 4.csn.intercloud.net might be mapped to xmpp://intercloud.megacloud.net which resolves to the IP address of the Conversational Service of Intercloud Gateway of Megacloud, which can then be actually used

Governance

- Essentially Reproduce the Interoperable Global Trust Federation (IGTF) http://www.igtf.net/
- Why not just use IGTF?
  - IGTF represents university and Govt. research facilities and project
  - Do not want Intercloud to have instant peer status with every research Grid and Supercomputer at Day 1.
  - Likely the Intercloud will go commercial. Might drive different policies/governance
- IEEE Testbed to Create a function called Intercloud Trust Federation (ITF)
  - Lift IGTF Charter in whole
  - Work With IEEE Standards Association Registration Authority (Ethernet MAC Address people)
- ITF Runs Initial Infrastructure
  - Reference Root (with Certificate Authority)
  - CSN Registration Authority
  - Global DNS Liasion (for NAPTR lookups)

Governance Documents

- Intercloud Trust Federation (ITF) Common Charter
- **Authentication Profile for End-Entity Users - requirements on traditional X.509 PKI CAs (long-term credentials to end-entities, who posses and control their key pair and their activation data).**
- **Authentication Profile for Issuing Authorities - requirements on issuance of traditional X.509 PKI CAs. (long-term credentials to issuers, who posses and control their signing key, key pair and their activation data).**
- **Authentication Profile for SLCS End-Entity Users - requirements on short-lived X.509 PKI CAs (short-lived or SLCS credentials to end-entities, who posses and control their key pair and their activation data).**
- **Trust Anchor Distribution - a list of trust anchors, root certificates and related meta-information for all the accredited authorities**
- **CA Server Technical Interface Guidelines.**

Intercloud

IDENTITY/TRUST

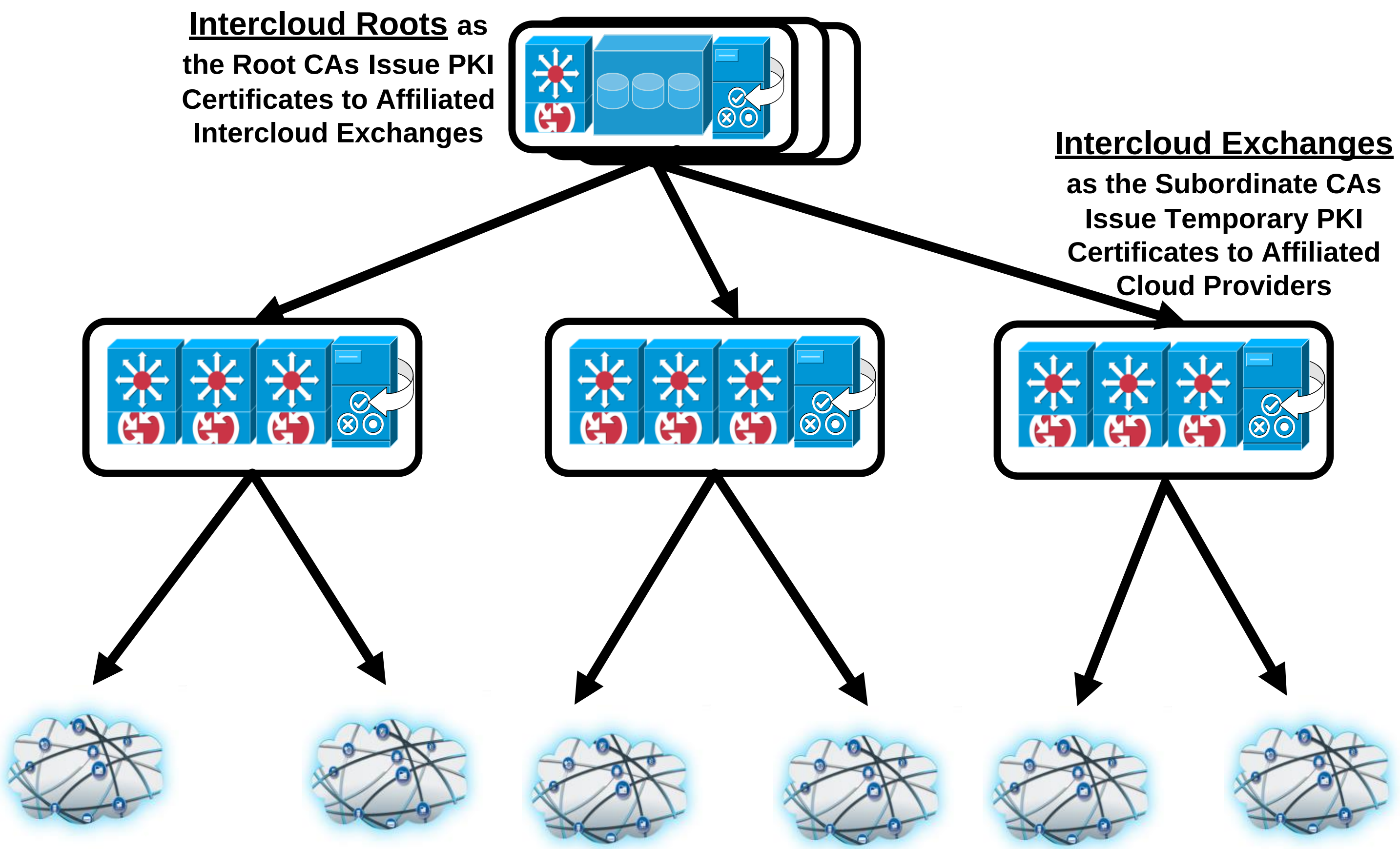
Certificate Authority Server

- **Something like OpenCA or Dogtag**
  - https://pki.openca.org/projects/openca/
  - http://pki.fedoraproject.org/wiki/PKI_Main_Page
- **Requirements**
  - Certificate issuance, revocation, and retrieval
  - Certificate Revocation List (CRL) generation and publishing
  - Certificate profiles
  - Simple Certificate Enrollment Protocol (SCEP)
  - Local Registration Authority (LRA) for organizational authentication and policies
  - Encryption key archival and recovery

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

PKI Certificates

Intercloud PKI Certificates Topology

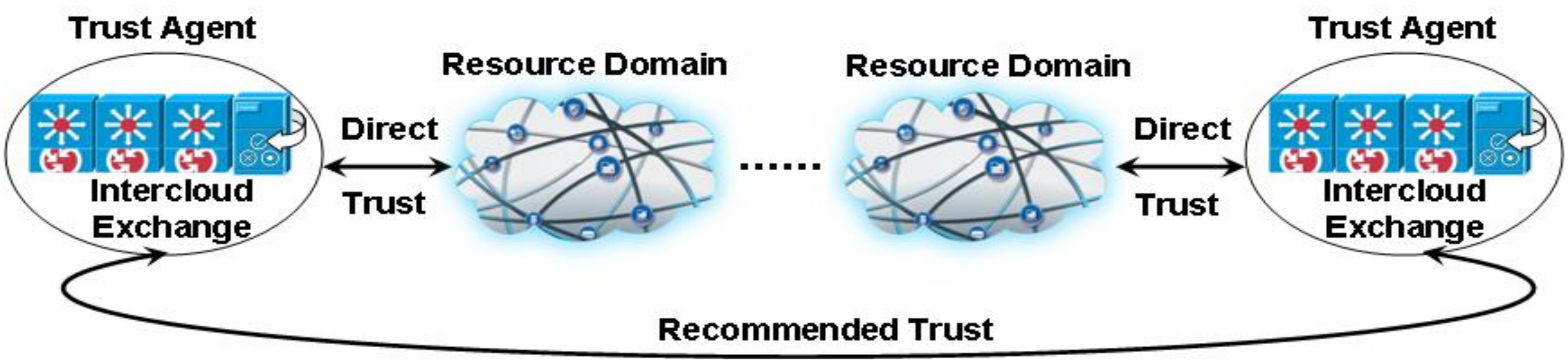


Cloud Providers use Temporary PKI Certificates as part of the Delegation Process – Acting on behalf of Originating Cloud Provider

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

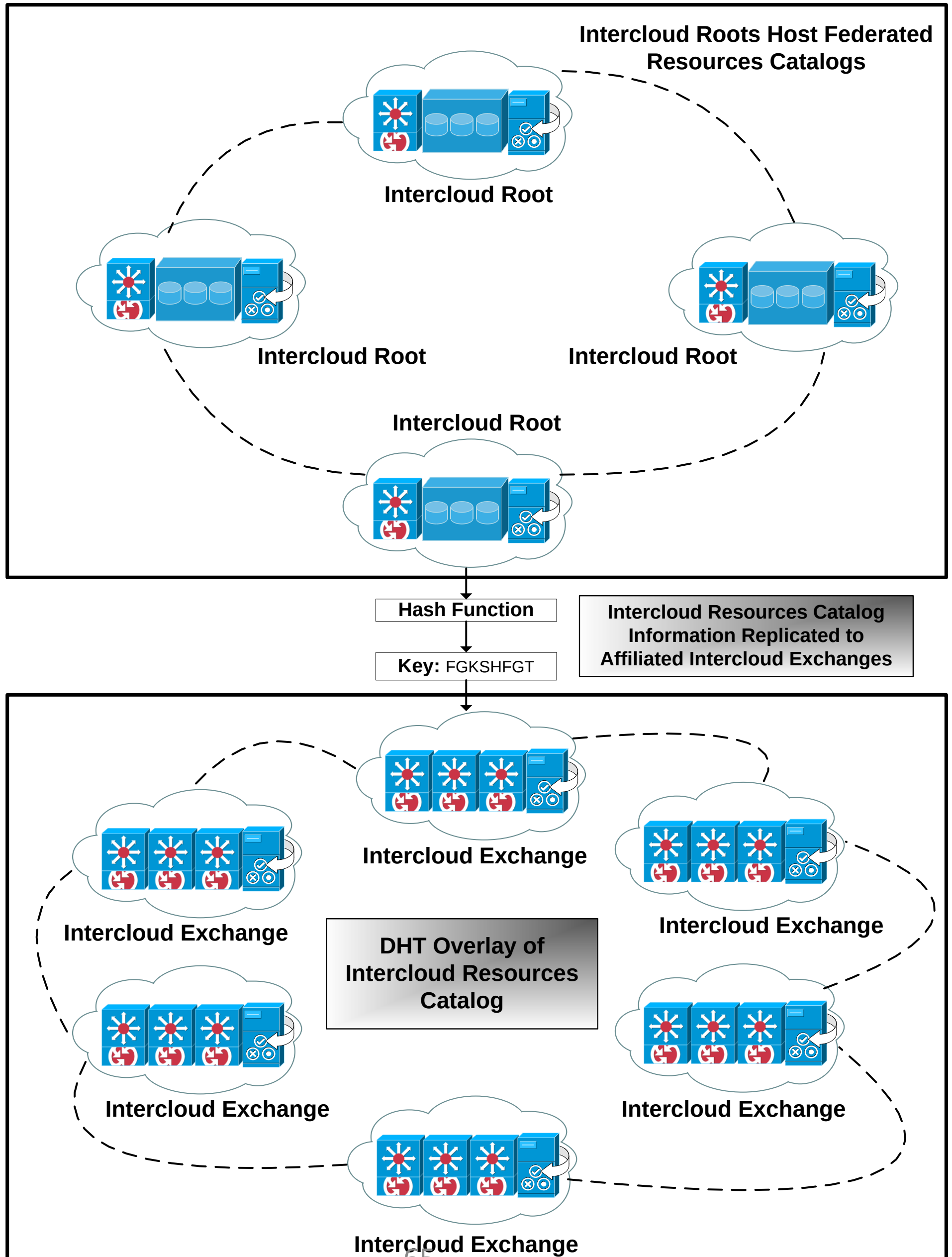
Trust Management – Extended PKI

- **Proposed Intercloud Trust Management Model**
  - From Intercloud topology perspectives, Intercloud Roots will provide static PKI CA root like functionality and Intercloud exchanges will be responsible for the dynamic "Trust Level" model layered on top of the PKI certificate based trust model
  - Exchanges are the custodians/brokers of "Domain based Trust" systems environment for their affiliated cloud providers
  - Cloud providers rely on the Intercloud exchanges to manage trust



Intercloud Trust Management Model

Federated Roots

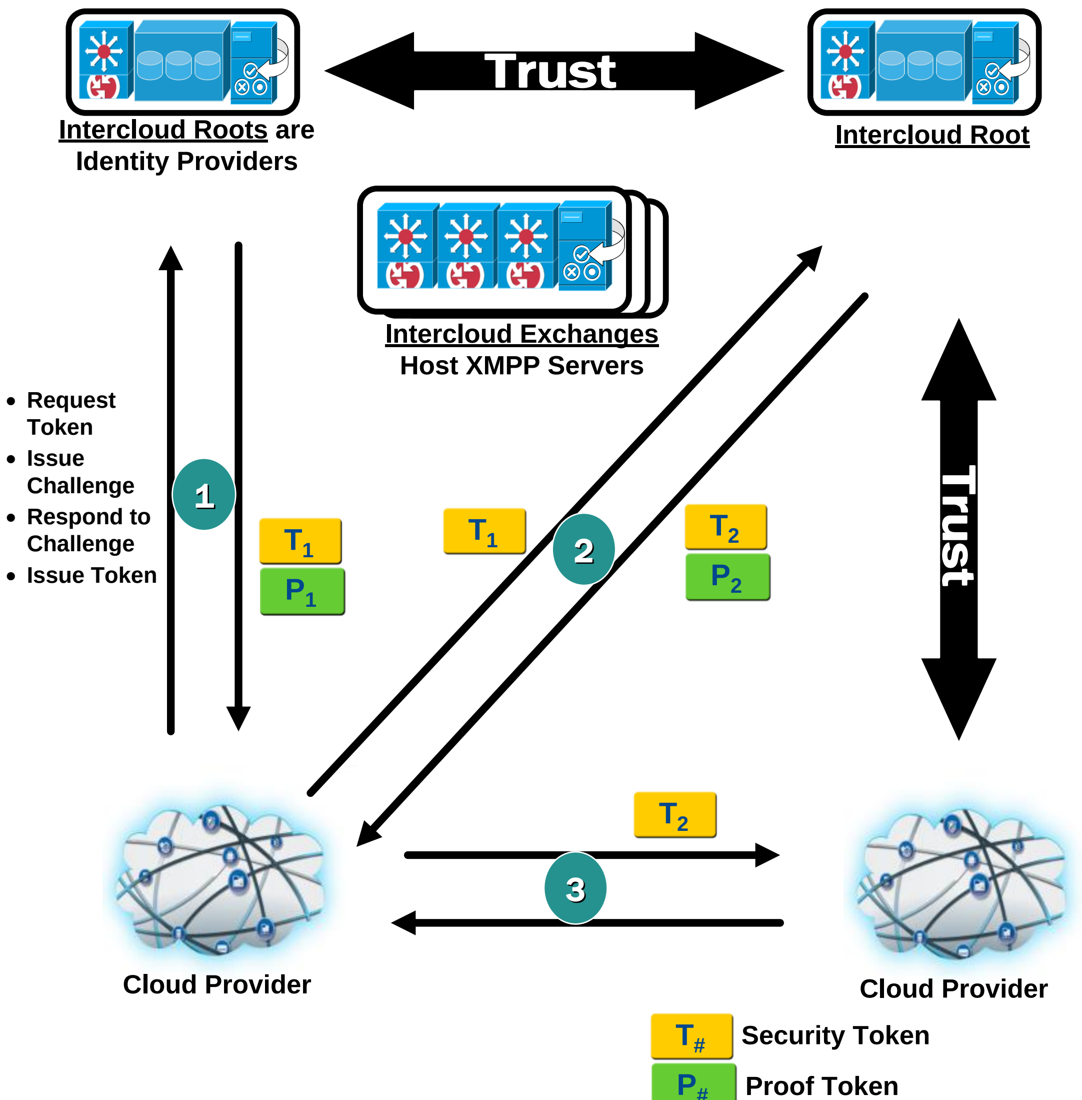


XMPP method securing XML stream

- Propose a channel encryption method which makes use of the Transport Layer Security (TLS) protocol
- Clouds use TLS to secure the streams prior to attempting the completion of SASL based authentication negotiation
- SASL has a method for authenticating a stream by means of an XMPP-specific profile of the protocol
- SASL provides a generalized method for adding authentication support to connection-based protocols.
- Following authentications methods are supported by XMPP-specific profile of SASL protocol: “DIGEST-MD5”, “CRAM-MD5”, “PLAIN”, and “ANONYMOUS”
- SAML provides authentication in a federated environment.
- Currently, there is no support for SAML in XMPP-specific profile of SASL protocol.
- However, there is a draft proposal published that specifies a SASL mechanism for SAML 2.0 that allows the integration of existing SAML Identity Providers with applications using SASL

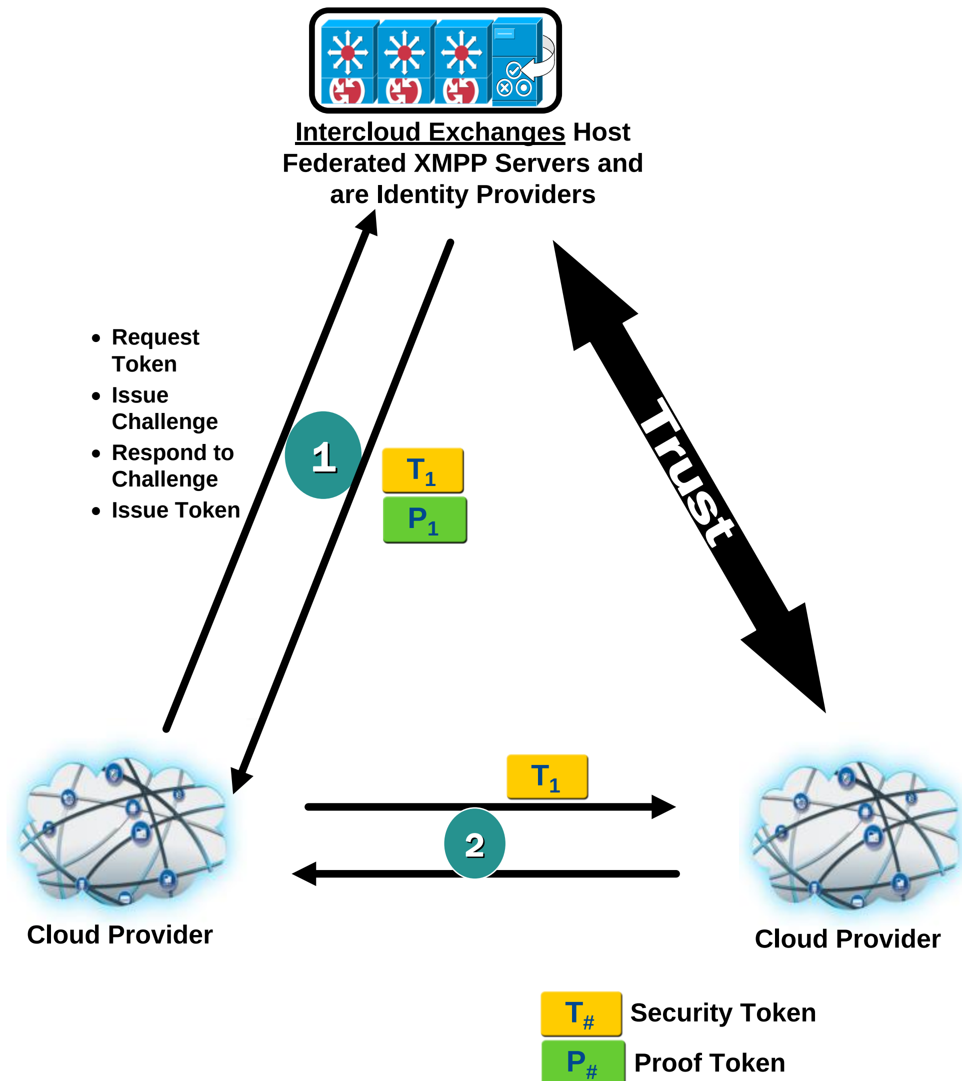
Intercloud Echange/Root

Inter "Intercloud Root" and Inter "Intercloud Exchange" Collaboration Scenario



Intercloud Exchange/Exchange

Intra "Intercloud Exchange" Collaboration Scenario



IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Intercloud

CONVERSATIONAL SUBSTRATE

Suitability of XMPP as a Conversation Transport

- An Intercloud Protocol for presence and messaging needs to exist which can support the 1-to-1, 1-to-many, and many-to-many Cloud to Cloud use cases
- Extensible Messaging and Presence Protocol (XMPP) is exactly such a protocol
- HTTP protocol is synchronous; XMPP are asynchronous
- XMPP root services would be located in the Intercloud Root in the topology explained above
- XMPP servers support encrypted communication (SASL (Simple Authentication and Security Layer) and TLS (Transport Layer Security) with the option to restrict XMPP servers to accept only encrypted client-to-server and server-to-server connections

Why XMPP instead of SIP?

- Based on our research we have concluded that XMPP is a much better and a suitable fit for Intercloud computing environment
- XMPP software stack is more lightweight than SIP/SIMPLE stack
- Open Source is Widely Available
- Additional features can very easily be added via extensions to the XMPP protocol which makes it flexible
- XML foundation of XMPP greatly simplifies integration with existing Internet centric environments
- In XMPP all messages go through a server, which allows the server to mediate, log and audit messages
- A key requirement for enabling Intercloud Exchanges as market makers and be able to practice arbitrage
- SIP/SIMPLE, on the other hand, is a peer-to-peer based standard

Parts of XMPP to use

- **XMPP part of the portable gateway code**
- **Complete at least the XMPP-Core (RFC 6120) and XMPP-IM (RFC 6121) Profiles, as far as a Client goes**
- **Later maybe**
  - **XMPP-ADDR (RFC 6122), and XMPP-E2E (RFC 3923).**
  - **The roles and exact strategy for XMPP-JRN (RFC 4854) and/or XMPP-ENUM (RFC 4979) and/or XMPP-JRI (RFC 5122) need to align with the namespace design component.**
  - **In other words we need to merge the CS Names proposal with XMPP (JID) Naming.**

XMPP Encryption & Authentication

- **XMPP includes a method for securing the XML stream from tampering and eavesdropping. This channel encryption method makes use of the Transport Layer Security (TLS) protocol, along with a “STARTTLS” extension that is modeled after similar extensions for the IMAP and POP3 protocols.**
- **Clouds use TLS to secure the streams prior to attempting the completion of SASL based authentication**
- **Currently, the following authentications methods are supported by XMPP-specific profile of SASL protocol: “DIGEST-MD5”, “CRAM-MD5”, “PLAIN”, and “ANONYMOUS”.**
- **There is a draft proposal published that specifies a SASL mechanism for SAML 2.0 that allows the integration of existing SAML Identity Providers with applications using SASL.**

Excerpt of data flow for a Cloud securing a stream to an Intercloud Root

Step 1: Cloud starts stream to Intercloud Root:

```
<stream:stream
       xmlns='jabber:client'
       xmlns:stream='http://etherx.jabber.org/streams'
       to='intercloudexchg.com'version='1.0'>
```

Step 2: Intercloud Root responds by sending a stream tag to client:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    id='cloud1_id1'
    from='intercloudexchg.com'
    version='1.0'>
```

Step 3: Intercloud Root sends the STARTTLS extension to Cloud:

```
<stream:features>
  <starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'>
    <required/>
  </starttls>
</stream:features>
```

Cloud securing a stream to an Intercloud Root. cont

Step 4: Cloud sends the STARTTLS command to Intercloud Root:

```
<starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
```

Step 5: Intercloud Root informs Cloud that it is allowed to proceed:

```
<proceed xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
```

Step 5 (alt): Intercloud Root informs Cloud that TLS negotiation has failed and closes both stream and TCP connection:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-tls'/>
</stream:stream>
```

Step 6: Cloud and Intercloud Root attempt to complete TLS negotiation over the existing TCP connection.

..cont

Step 7: If TLS negotiation is successful, Cloud initiates a new stream to Intercloud Root:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    to='intercloudexchg.com'
    version='1.0'>
```

Step 7 (alt): If TLS negotiation is unsuccessful, Intercloud Root closes TCP connection.

Step 8: Intercloud Root responds by sending a stream header to Cloud along with any available stream features:

```
<stream:stream
    xmlns='jabber:client'
    xmlns:stream='http://etherx.jabber.org/streams'
    from='intercloudexchg.com'
    id=' cloud1_id2'
    version='1.0'>
<stream:features>
  <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
    <mechanism>DIGEST-MD5</mechanism>
    <mechanism> CRAM-MD5</mechanism>
    <mechanism>PLAIN</mechanism>
    <mechanism>ANONYMOUS</mechanism>
    <mechanism>EXTERNAL</mechanism>
    <mechanism>SAML20</mechanism>  </mechanisms>
</stream:features>
```

..cont

Step 9: Cloud continues with SASL based authentication negotiation.

Step 10: Cloud selects an authentication mechanism:

```
<auth xmlns='urn:ietf:params:xml:ns:xmpp-sasl' mechanism='SAML20'/>
```

Step 11: Intercloud Root sends a BASE64 [28] encoded challenge to Cloud in the form of an HTTP Redirect to the SAML assertion consumer service with the SAML Authentication Request as specified in the redirection URL.

Step 12: Cloud sends a BASE64 encoded empty response to the challenge:

```
<response xmlns='urn:ietf:params:xml:ns:xmpp-sasl'> = </response>
```

Step 13: The Cloud now sends the URL to the local Intercloud Gateway which is passed back to the Cloud who sends the AuthN XMPP response to the Intercloud Root, containing the subject-identifier and the "jid" as an attribute.

Step 14: Intercloud Gateway informs Cloud of successful authentication:

```
<success xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```

Step 14 (alt): Intercloud Gateway informs Cloud of failed authentication:

```
<failure xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
<temporary-auth-failure/>
</failure>
</stream:stream>
```

Intercloud

TRANSPORT/SERVICES FRAMEWORK

Services Framework

- XMPP Based

Or

- Web Sockets Based

XMPP Services Framework

- **First, we must consider how to construct a Services Framework layer on top of XMPP**
- **One Idea: XEP**
- **We leverage a series of XMPP extensions (XEP series)**
- **Extension XEP-0244 provides a "services" framework on top of base XMPP, named IO Data, which was designed for sending messages from one computer to another**
- **It provides a transport for remote service invocation and attempting to overcome the problems with SOAP & REST. A reference implementation for the IO Data XEP, XMPP Web Services for Java (xws4j), is already in place and available**

XMPP Service Invocation

- Once the Cloud has now secured a connection to the Intercloud root, it can look for a suitable other Cloud to interoperate.
- It will either interoperate through an Intercloud Exchange, or directly Cloud to Cloud, as the case may be.
- The way a Cloud would find the appropriate services is by leveraging a catalog of available resources published in a directory residing in the Intercloud Root.
- For the Intercloud, we use this technique to specify resources such as storage, computing, and all the other possible services which Cloud both expose and consume in a Catalog.
- The Catalog uses RDF - a way to specify such resources, and SPARQL is a query/matching system for RDF

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

XMPP based Presence & Dialog

- **The requesting cloud has found a target cloud with which to interwork**
- **It must now turn directly to the target cloud and dialog with it.**
- **The code sample is based on Google AppEngine XMPP JAVA API set**

```
// ...
        JID jid = new JID("user@cloud2.com");
        String msgBody = "Cloud 2, I would like to use
your resources for storage replication using AMQP over
UDT protocol.";
        Message msg = new MessageBuilder()
            .withRecipientJids(jid)
            .withBody(msgBody)
            .build();

        boolean messageSent = false;
        XMPPService xmpp =
XMPPServiceFactory.getXMPPService();
        if (xmpp.getPresence(jid).isAvailable()) {
            SendResponse status =
xmpp.sendMessage(msg);
            messageSent =
(status.getStatusMap().get(jid) ==
SendResponse.Status.SUCCESS);
        }
```

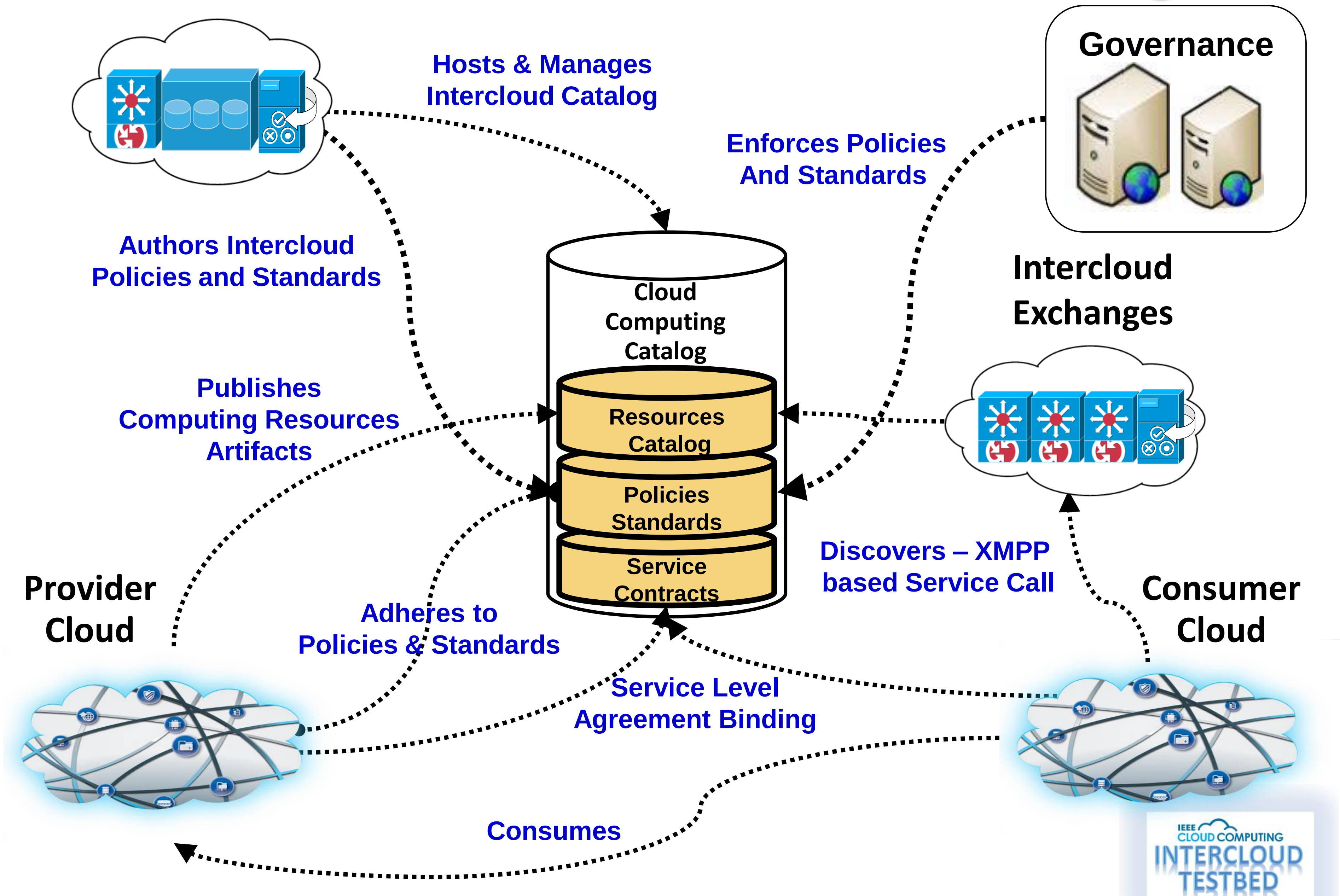
Creating a Web Sockets Services Framework

- **Not as well thought out yet as XMPP XEP**
- **But is imagined to be WebSockets. As described in RFC 6455**
- **Protocol consists of an opening handshake followed by basic message framing, layered over TCP**
- **Goal of this technology is to provide a mechanism for cloud to cloud two-way payload communication that does not rely on opening multiple HTTP connections.**

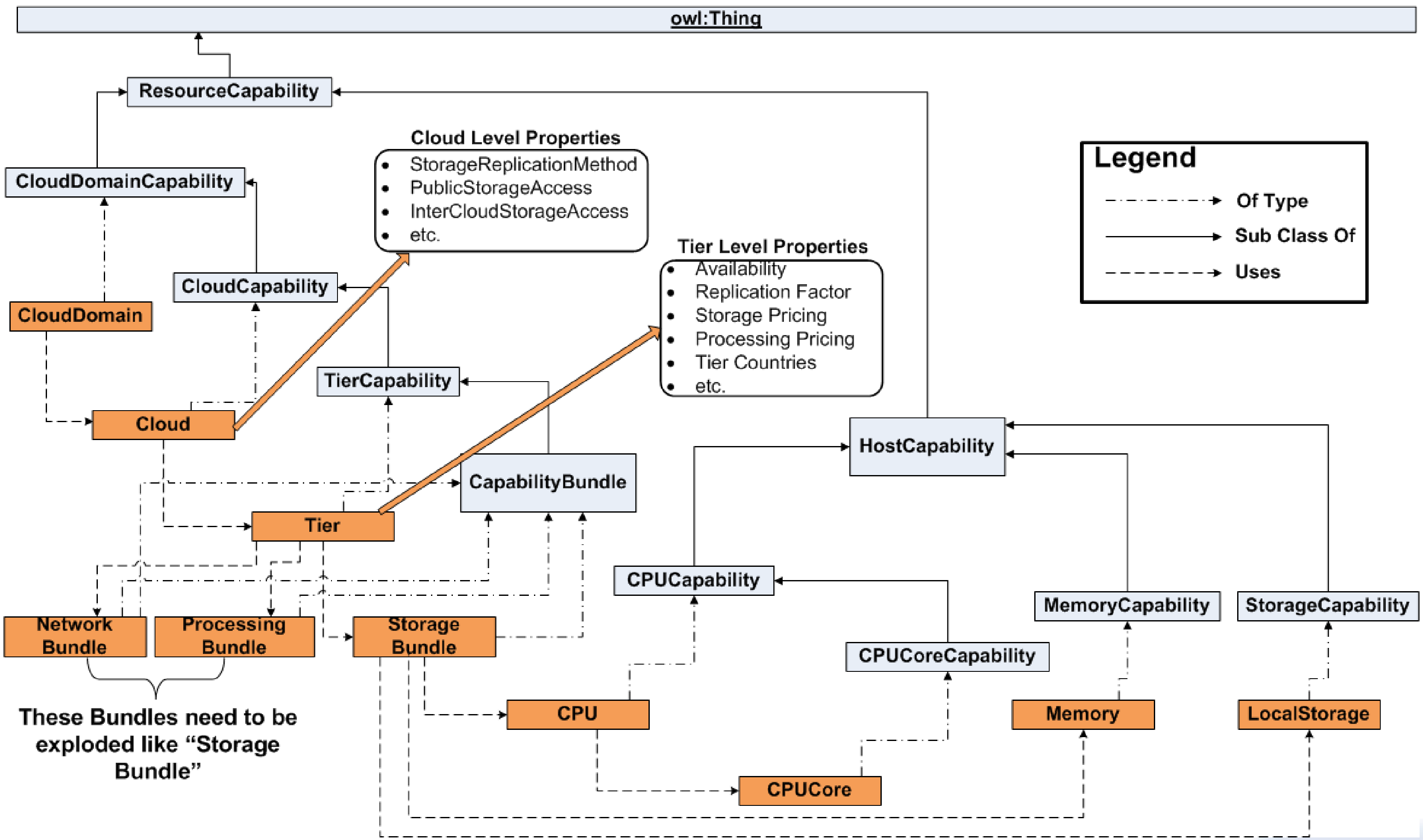
Intercloud

SEMANTIC DIRECTORY

Intercloud Resources Catalog



Intercloud Resources Ontology



IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Proposed Implementation

D1.2 – Cloud ontology and Cloud resources representations

mOSAIC

SEVENTH FRAMEWORK PROGRAMME

D1.2 – Cloud ontology and Cloud resources representations

Grant Agreement number:	256910
Project Title:	mOSAIC - Open-Source API and Platform for Multiple Clouds
Project start date:	01/9/2010
Duration	30 months

Deliverable number:	D1.2
Deliverable title:	D1.2
Due date of deliverable:	31.08.2011
Actual submission date:	31.08.2011
Last update:	31.08.2010
Editors:	Francesco Moscato
Participants:	SUN – IEAT
Workpackage:	WP1
Deliverable Nature:	Report
Dissemination Level:	PP: Restricted to other programme participants (including the Commission Services)
Version:	1.0
Total Number of Pages:	434
File name:	FP7-256910-D1.2-1.0

- **Contributed Technology to P2302**
- **Extensive Implementation**
- **Open Source**

IEEE CLOUD COMPUTING INTERCLOUD TESTBED

Intercloud

RESOURCE MATCHER/SOLVER

Invocation of a SPARQL query over the XMPP connection to the Intercloud Root

(uses IO Data XEP, XMPP Web Services for Java (xws4j):

```
<iq type='set'
    from='user@cloud1.org'
    to='service.intercloudexchg.com'
    id='cloud1_id1'>
    <command xmlns=
      'http://jabber.org/protocol/commands'
      node='constraint_catalog_resources'
      action='execute'>
      <iodata xmlns=
      'urn:xmpp:tmp:io-data' type='input'>
      <in>
      <constraints xmlns='http://www.csp/resOntology'>
        <constraint>
         <attribute>availabilityQuanity </attribute>
         <value>99.999</value>
        </constraint>
        <constraint>
         <attribute>replicationFactor</attribute>
         <value>5</value>
        </constraint>
        <constraint>
         <attribute>tierCountries</attribute>
         <value>JAPAN</value>
        </constraint>
        <constraint>
         <attribute>StorageReplicationMethod
         </attribute>
         <value>AMQP</value>
        </constraint>
        <constraint>
```

Code snippets of N-Triples based ontology semantic model

(a human-friendlier alternative to RDF/XML)

Step 1: In our ontology example, “CloudDomain” is an instance of class “CloudDomainCapability”. It consists of three resources “Cloud.1”, “Cloud.2” & “Cloud.3”:

```
<http://cloud/domain>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/#cloud.1>.

<http://cloud/domain>
<http://www.csp/resOntology#hasCapab
<http://cloud/domain/#cloud.2>.

<http://cloud/domain>
<http://www.csp/resOntology#hasCapab
<http://cloud/domain/#cloud.3>.

<http://cloud/domain> <http://www.w3
rdf-syntax-ns#type>
<http://www.csp/resOntology#Clouddon

<http://cloud/domain> <http://www.w3
schema#label> "Cloud Computing
domain"^^<http://www.w3.org/2001/XMI
```

Step 2: “Cloud.1”, in turn, consists of tier instances “tier.1”, “tier.2” & “tier.3”:

```
<http://cloud/domain/#cloud.1>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/cloud.1#tier1>.

<http://cloud/domain/#cloud.1>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/cloud.1#tier2>.

<http://cloud/domain/#cloud.1>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/cloud.1#tier3>.
```

N-Triples Code snippets, cont.

Step 3: Each of these cloud instances has associated properties such as "StorageReplicationMethod", "InterCloudStorageAccess" etc. etc. These properties are, in turn, used for determining if the computing resources of a cloud provider meet the preferences & constraints of the requesting cloud's interest and requirements:

```
<http://cloud/domai
<http://www.csp/res
<http://cloud/domai
Method>.

<http://cloud/domai
<http://www.csp/res
<http://cloud/domai
Access>.

<http://cloud/domai
<http://www.csp/res
<http://cloud/domai

<http://cloud/domai
<http://www.csp/res
<http://cloud/domai
```

Step 4: Computing resources are logically grouped together as bundles and exposed as standardized units of provisioning and configuration to be consumed by another cloud provider/s. These bundles are "StorageBundle", "ProcessingBundle" & "NetworkBundle". Each "Tier", in turn, consists of instances of resource bundles such as "StorageBundle" etc. Each "Tier" also has its own associated properties depicting preferences and constraints:

```
<http://cloud/domain/cloud.1#tier1>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/cloud.1/bundle/#storage1>.

<http://cloud/domain/cloud.1#tier1>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/cloud.1/bundle/#processing1>.

<http://cloud/domain/cloud.1#tier1>
<http://www.csp/resOntology#hasCapability>
<http://cloud/domain/cloud.1/bundle/#network1>.
```

Using SPARQL Protocol And RDF Query Language

a very powerful SQL-like language for querying and making semantic information machine process-able

Structure:

```
PREFIX: Prefix definition (optional)
SELECT: Result form
FROM: Data sources (optional)
WHERE: Graph pattern (=path expression)
    •FILTER
    •OPTIONAL
```

Example:

```
PREFIX geo: <http://www.geography.org/schema.rdf#>
SELECT ?X ?Y
FROM <http://www.geography.org>
WHERE {  ?X geo:hasCapital ?Y.
            ?Y geo:areacode ?Z }
ORDER BY ?X
```

Figure 5. Structure & Example of SPARQL Query

SPARQL Query for Resource Match

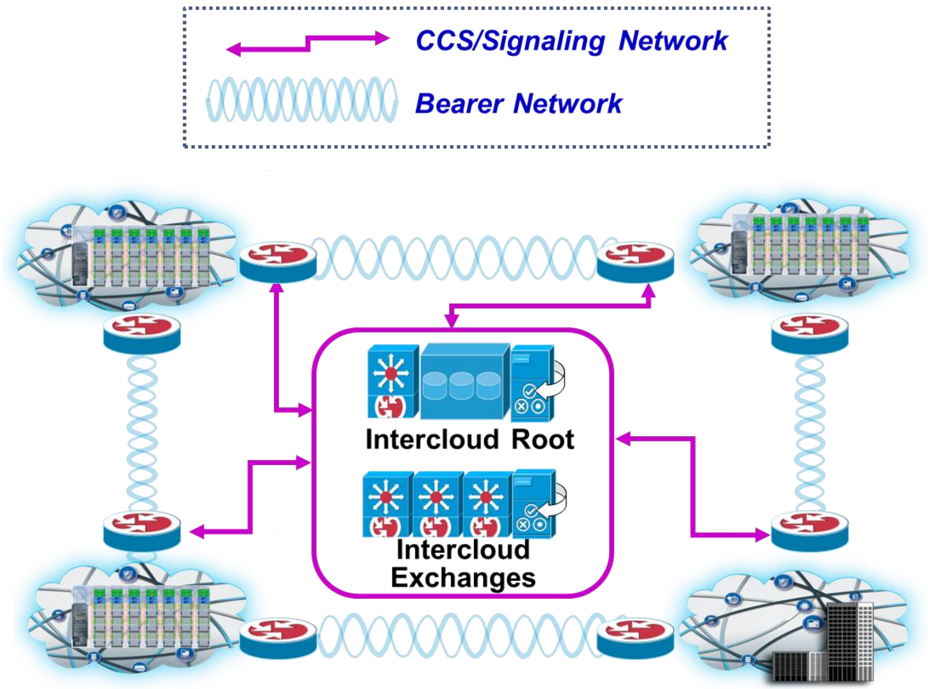
```
PREFIX  xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT  ?cld1 ?cld2 ?cld3 ?cld4 ?cld5
WHERE   { ?cld1
<http://www.csp/resOntology#availabilityQuanity> ?availa
bilityQuanity .
  ?cld2
<http://www.csp/resOntology#replicationFactor> ?replicat
ionFactor .
 ?cld3
<http://www.csp/resOntology#tierCountries> ?tierCountrie
s .
  ?cld4
<http://www.csp/resOntology#StorageReplicationMethod> ?S
torageReplicationMethod .
  ?cld5 <http://www.csp/resOntology#
InterCloudStorageAccess > ?InterCloudStorageAccess .

FILTER ( ?availabilityQuanity = 99.999 )
FILTER ( ?replicationFactor = 5)
FILTER ( ?tierCountries = "Japan")
FILTER ( ?StorageReplicationMethod = "AMQP")
FILTER ( ?InterCloudStorageAccess = "NFS")
    }
```

Intercloud

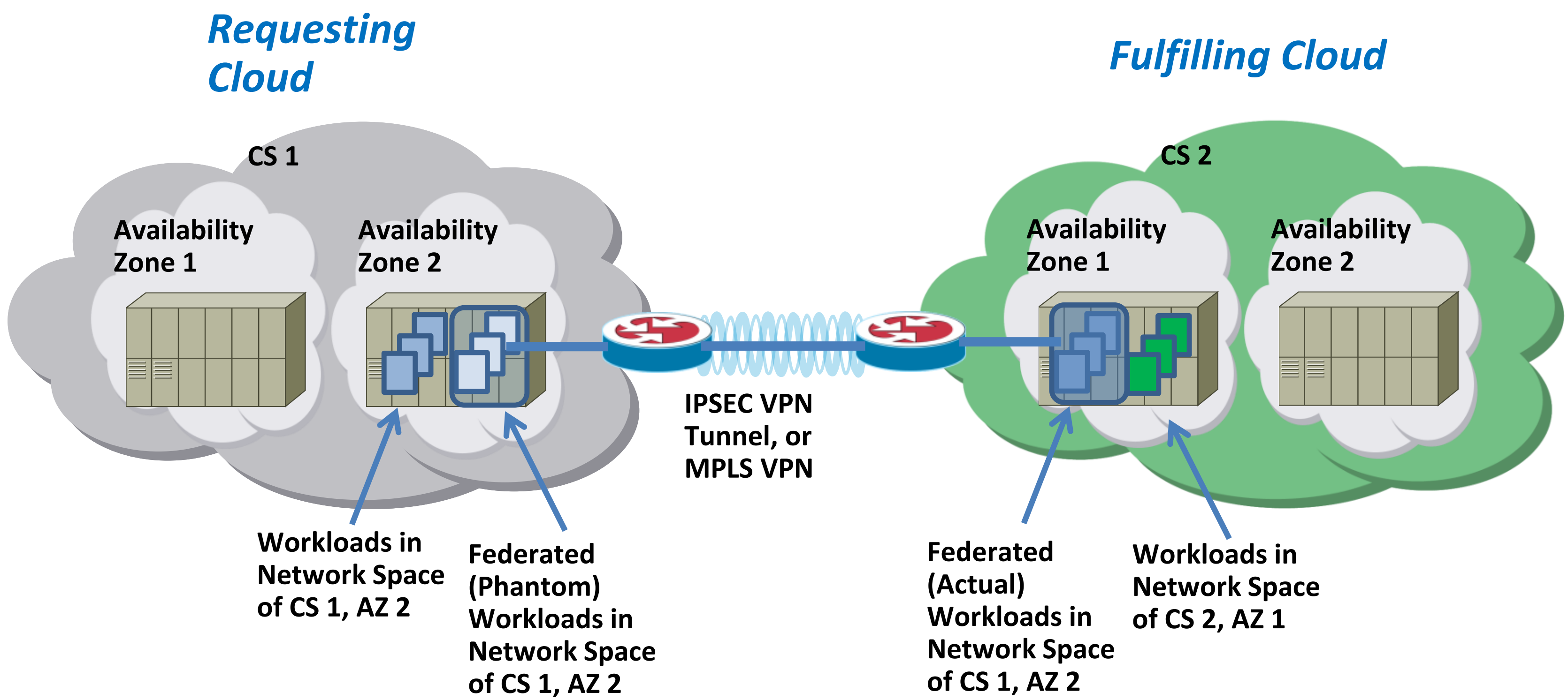
FEDERATING API
BEARER NETWORK FABRIC

CCS / Bearer Network Handoff

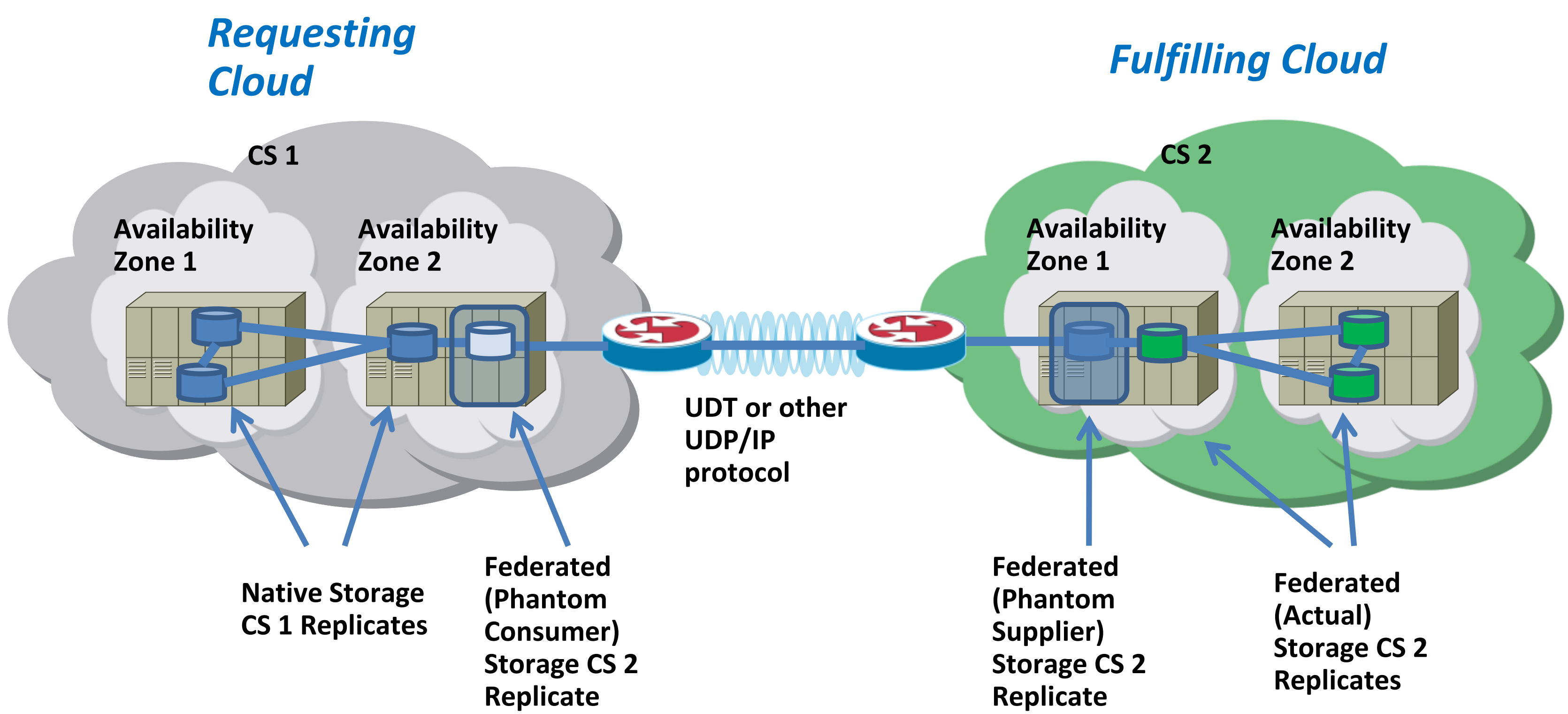
- Clouds have several options for Bearer Network
- Example Bearer Network Drivers in Intercloud Gateway
  - IPSEC VPN/VPC, MPLS VPN/VPC, 802.1q VLAN/VPC, RDMA/OFA Infiniband
  - TCP, UDT, and others (eg, MPI, SDP/rsockets)
- Bearer Network Drivers are registered with Exchange(s)
  - Exchange uses in Solver / Matching Algorithm
  - Services Transport is used to communicate Bearer Network Coordinates ("The Federation API's")



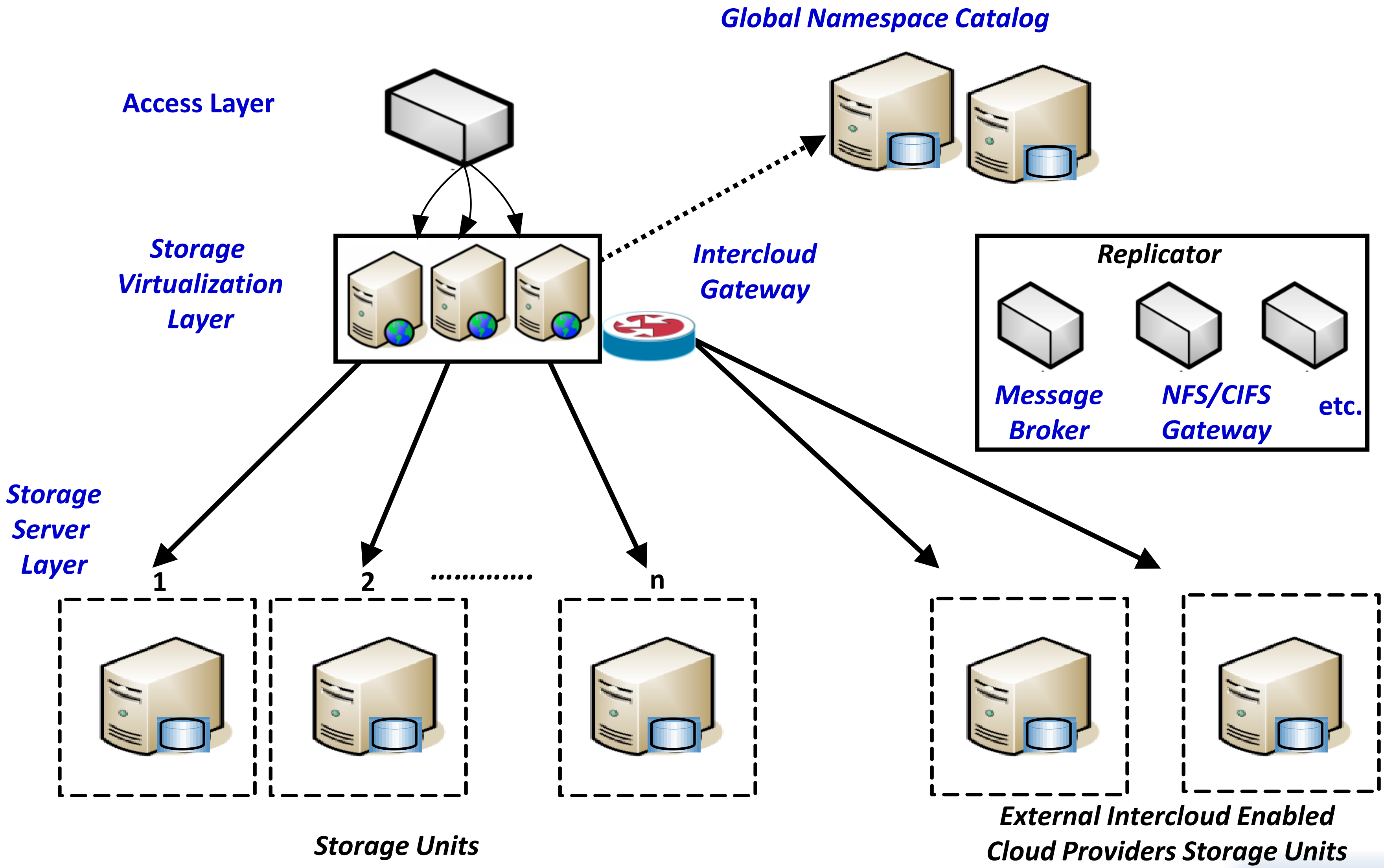
Federation of Workloads –
Looks like Dynamic/SDN VPC (Virtual Private Cloud)



Federation of Storage –
Storage System Replicate Extension



Storage Federation Detail



Intercloud

REPLICATION/SCALING

Replication

- Horizontal of Roots and Exchanges
- Bit Torrent Based?
- Needs Design

Intercloud

AUDIT

Audit

- **The Root servers will support XMPP audit trails.**
- **These implementations will likely use XMPP S2S, but have not been designed yet.**
- **Raw audit traffic will need to be folded and reduced such that conversations relating to decisions of fulfilling federation requests can be reproduced and proven to have matched the request in the most optimal way.**
- **In this way arbitrage will be enabled and trusted.**

Intercloud

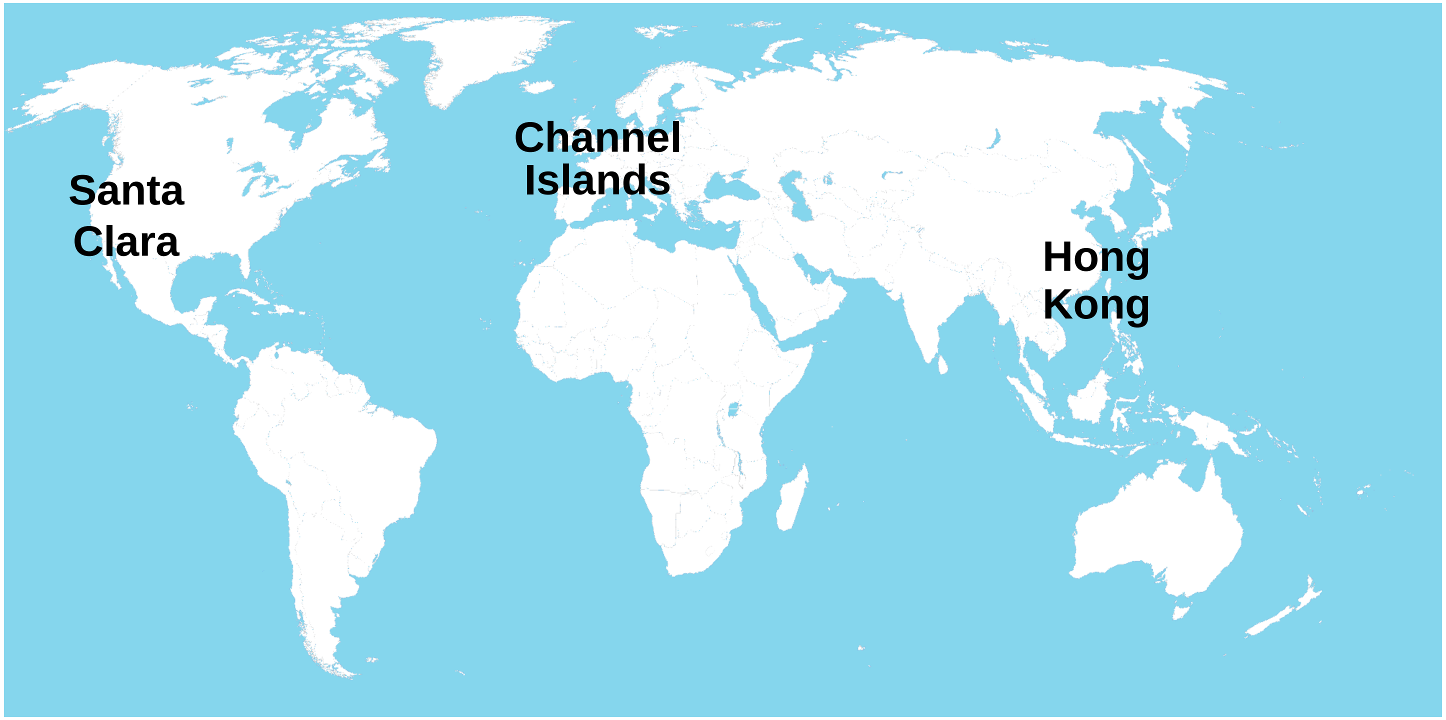
TESTBED

Work Areas

1. Master Technical Design Work
2. Collaboration, Source Code, Specs, Internal and Public Site(s)
3. Namespace Technology
4. Governance Procedures/Structure
5. Identity/Trust Technology
6. Conversational Substrate Technology
7. Transport/Services Framework Technology
8. Semantic Directory Technology
9. Resource Matcher/Solver Technology
10. Federating API and Mechanism Compute
11. Federating API and Mechanism Storage
12. Bearer Network Fabric Technology
13. Replication/Scaling Technology
14. Audit Technology
15. Overall Gateway (per Cloud OS flavor) Package
16. Overall Root System Package
17. Overall Exchange System Package
18. Use Cases
19. Hosted Reference Infrastructure Base Clouds
20. Testbed Infrastructure (Intercloud Software, at first in Labs, then on the Hosted Set up)

Initial Intercloud Network

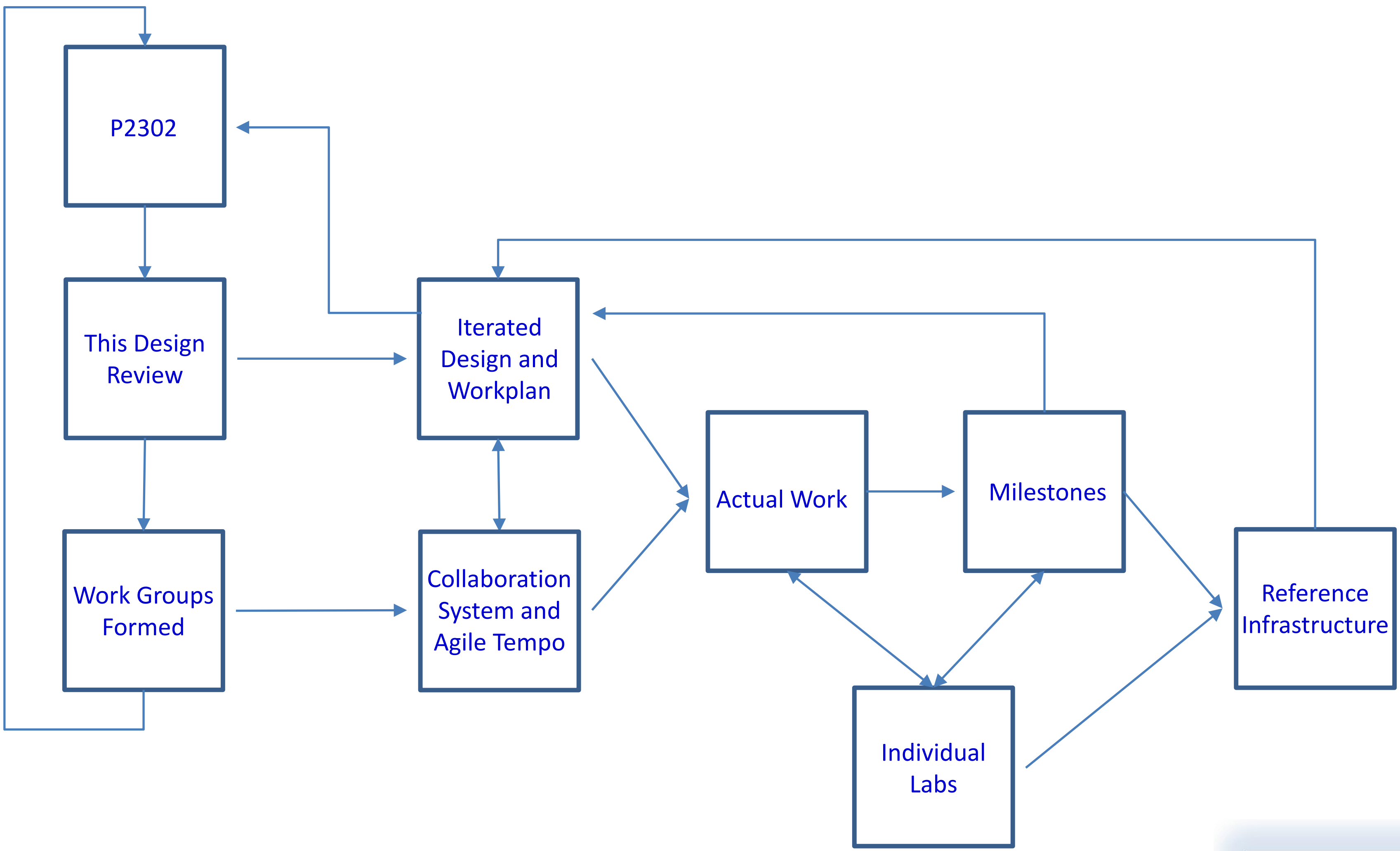
Initial Locations for Reference Roots and Exchanges



Reference Infrastructure

- ***Network***
  - Multi Carrier IP Transit
  - Advertising AS numbers
  - OSPF/BGP
- ***Racked Gear***
  - Machines - 1RU 2 x 6 core or 4 x 4 core, Xeons
  - Eg, Dell R620 or IBM x3530
  - Memory - 24G to 48G
  - Storage - sever internal DAS/SATA, 7200 RPM, minimum 2 x 1TB
  - NICs – 2 x 1G or 10G
  - Wiring - Copper in rack, optical uplink
  - Switches - TOR L2/L3
  - Eg, Dell N4032, Brocade 6650, or Extreme 7100
- ***Clouds***
  - We will build Two Reference Clouds in Each Location
  - One Intercloud Root, and One Intercloud Exchange
  - Racking - One rack is OK in each location – ½ Rack for each Cloud
  - Sizing – 4 to 6 servers for each cloud
  - Power - 2.5KW for rack (est).
- ***Software***
  - CloudOS – OpenStack from Cloudscaling or OpenNebula from C12G
  - Storage – Swift/Cinder for OpenStack, or Ceph for OpenNebula
  - DevOps

Testbed Development Process



Links

david.bernstein@ieee.org
andrew.hughes@ieee.org

http://www.intercloudtestbed.org/

http://cloudcomputing.ieee.org/intercloud
http://www.linkedin.com/groups/IEEE-Cloud-Computing-2856284
https://www.facebook.com/IEEECloudComputing
https://twitter.com/ieeecloud