Defining, Securing, and Standardizing Cloud Computing

Lee Badger and Chris Johnson

Sep. 28, 2010

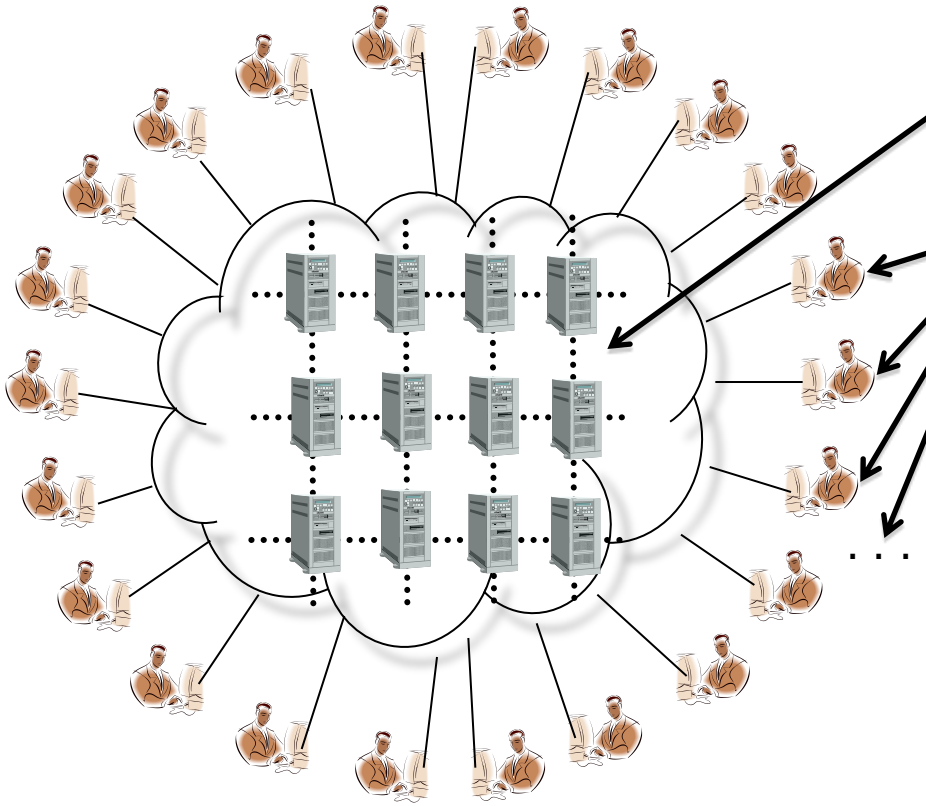
Outline

1. Brief review of clouds.
2. A few security issues in the cloud.
   - virtualization
3. SCAP
4. Introduction to Standards Acceleration to Jumpstart Adoption of Cloud Computing (SAJACC).

Note: Any mention of a vendor or product is NOT an endorsement or recommendation.

1 Brief review of clouds.

Cloud Computing



Computers in a network, providing service.

Users with network access.

Convenient remote computer rental.

In any quantity.

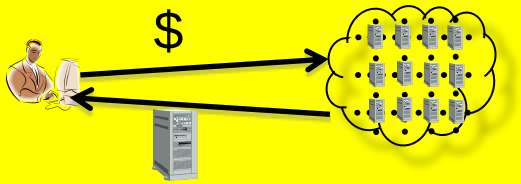
"Feels" local.

A technical or business innovation?

NIST Working Cloud Definition (1 of 3)

5 Key Characteristics

1 On-demand self service



renting takes minutes

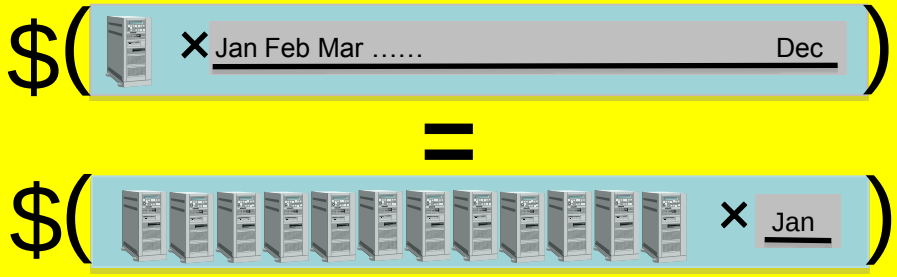
2 Ubiquitous network access

anywhere / any device

3 Metered use

conserve resources

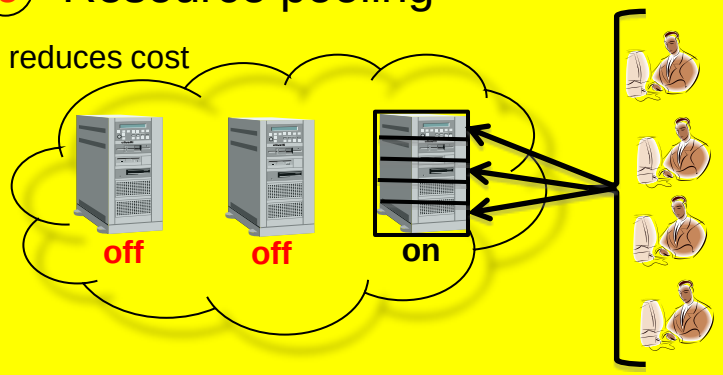
4 Elasticity



rent it in any quantity

5 Resource pooling

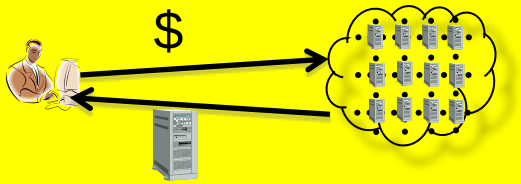
reduces cost



NIST Working Cloud Definition (1 of 3)

5 Key Characteristics

1 On-demand self service



renting takes minutes

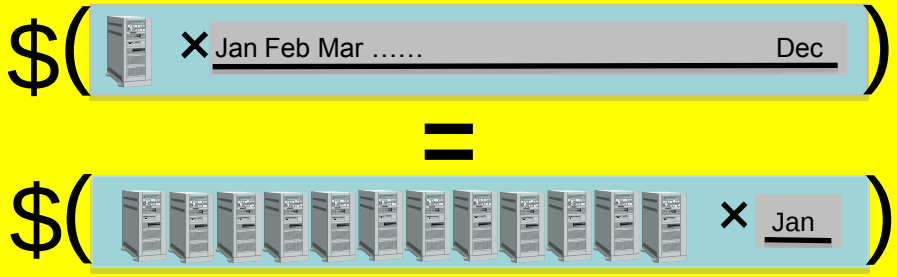
2 Ubiquitous network access

anywhere / any device

3 Metered use

conserve resources

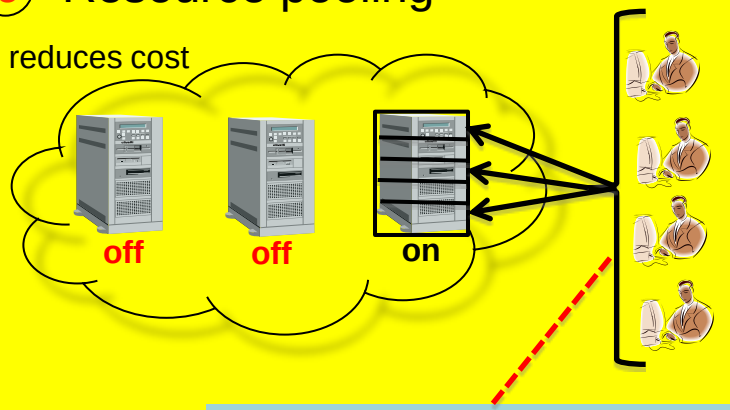
4 Elasticity



rent it in any quantity

5 Resource pooling

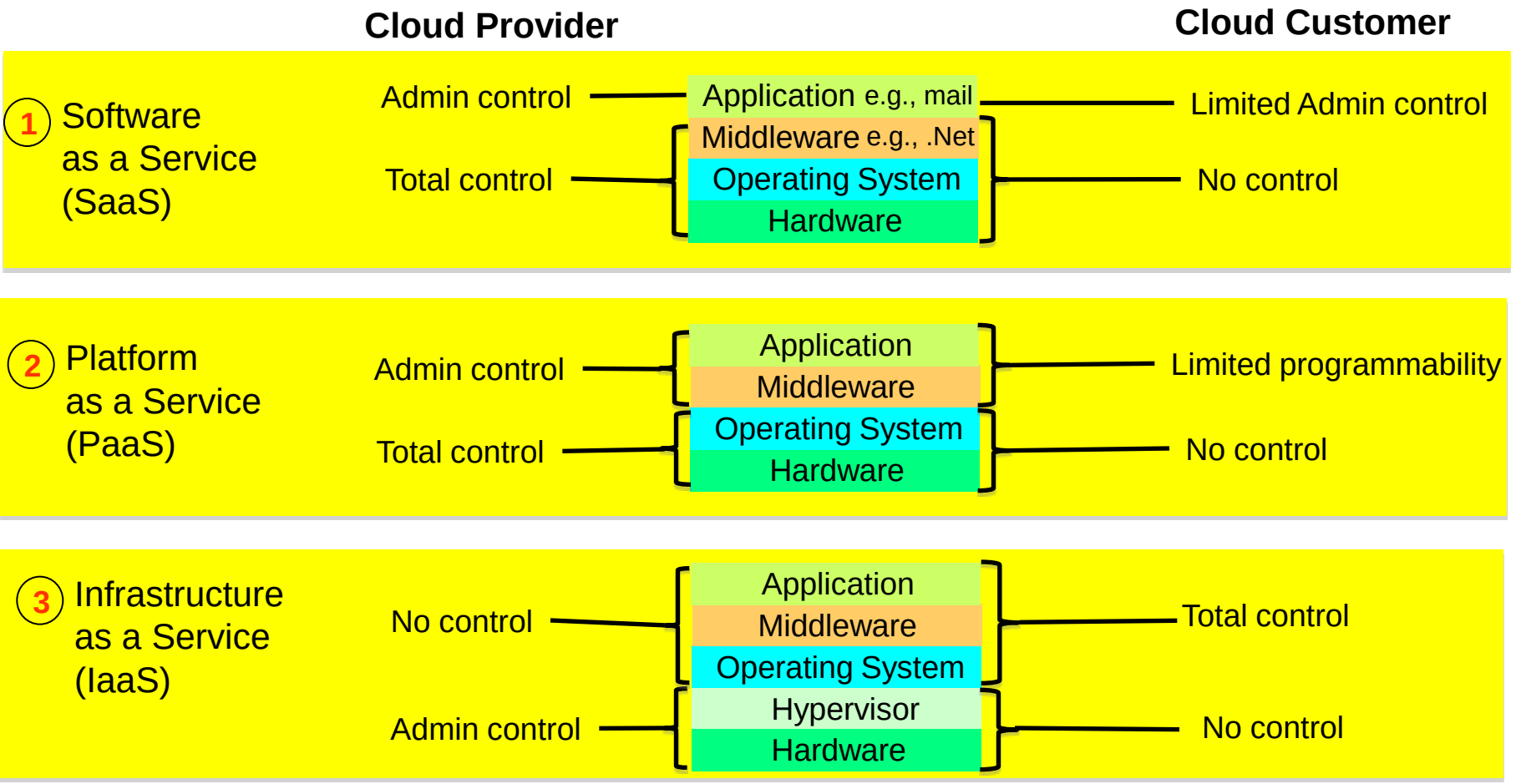
reduces cost



where is my workload?

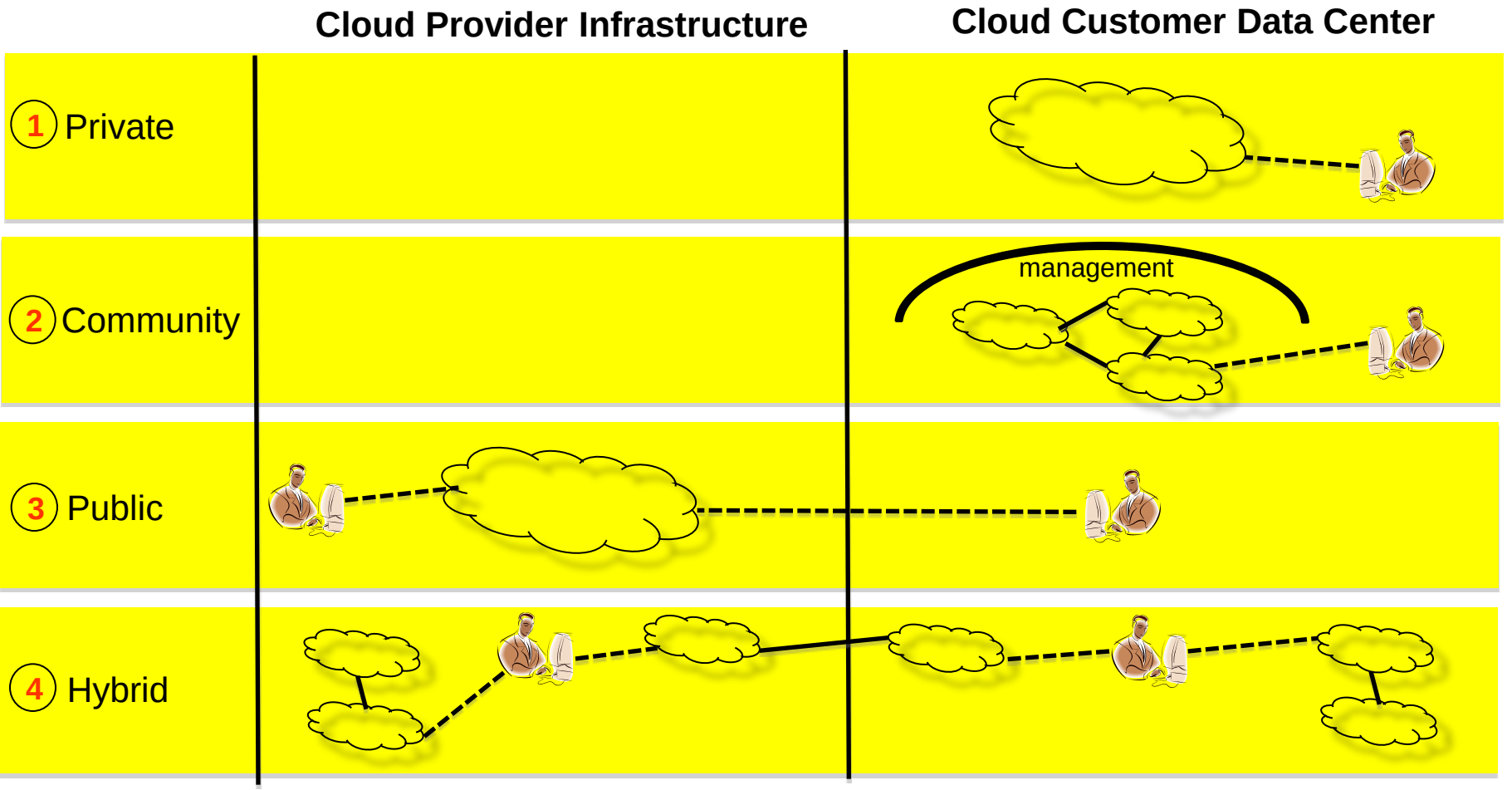
NIST Working Cloud Definition (2 of 3)

3 Deployment Models



NIST Working Cloud Definition (3 of 3)

4 Delivery Models



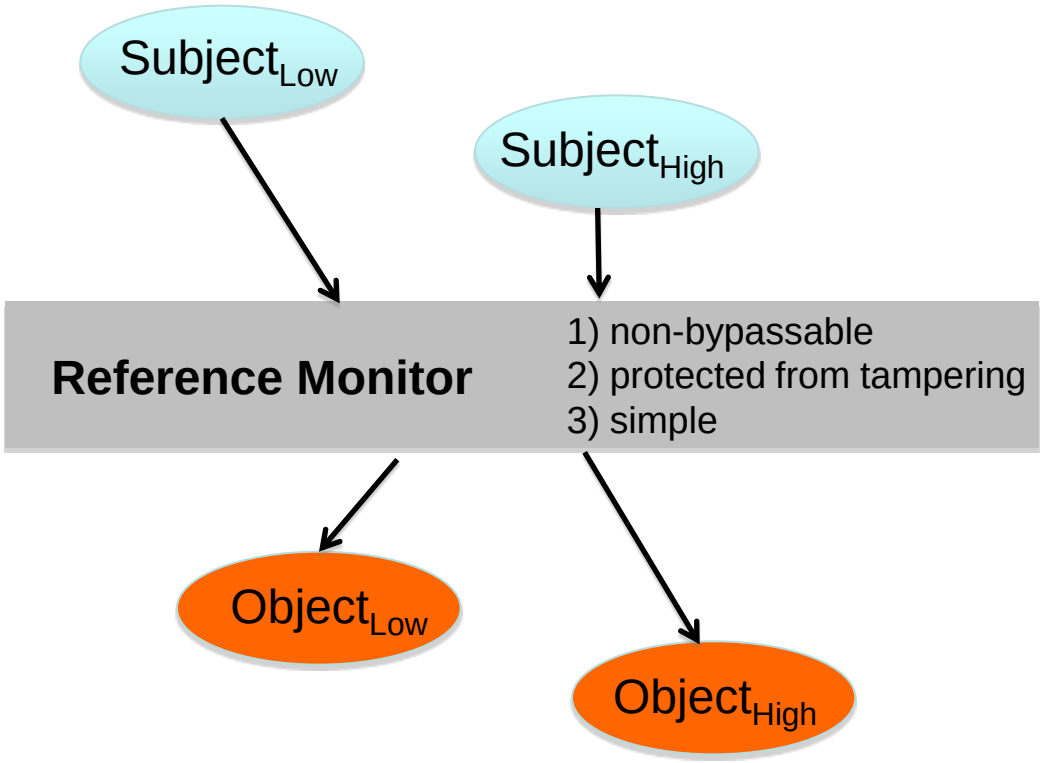
2

A few security issues in the cloud.
- virtualization

What is Security?

- Traditionally, approximately:
  - **confidentiality**: your data not leaked
  - **integrity**: your data or system not corrupted
  - **availability**: your system keeps running
- What does this mean in the cloud?
  - without user physical control
- Some issues
  - with dynamically changing infrastructure
  - key management
  - virtualization

Some Traditional Ideas



Bell/Lapadule (BLP) model
- no read up
- no write down

Biba integrity model
- inverse of BLP rules

Clark/Wilson integrity
- invariant maintenance via transactions

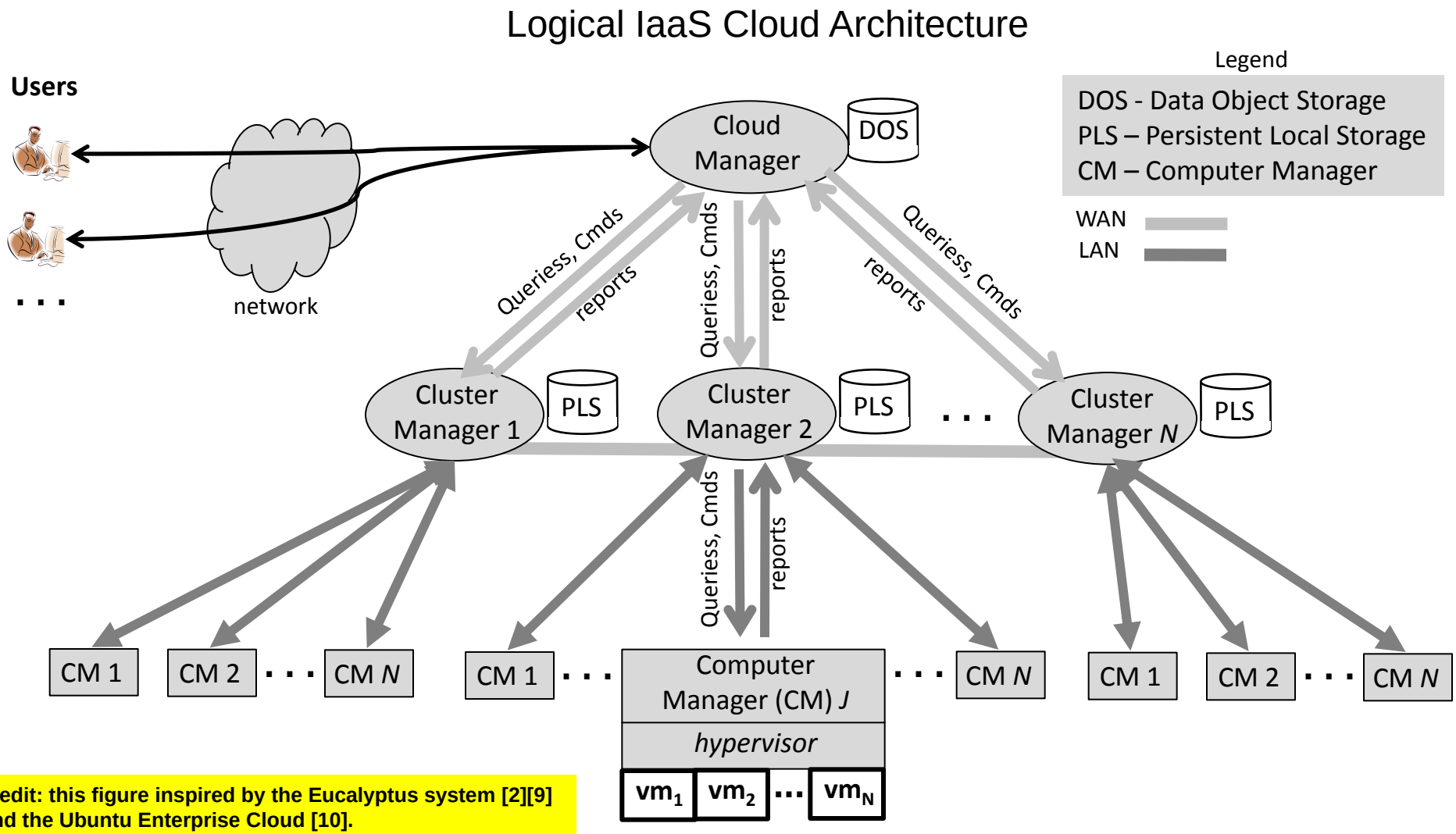
Basic modeling approach:

secure initial state
security-preserving state transition
security-preserving state transition
security-preserving state transition
...

credit: Anderson report from early 1970's (reference monitor).

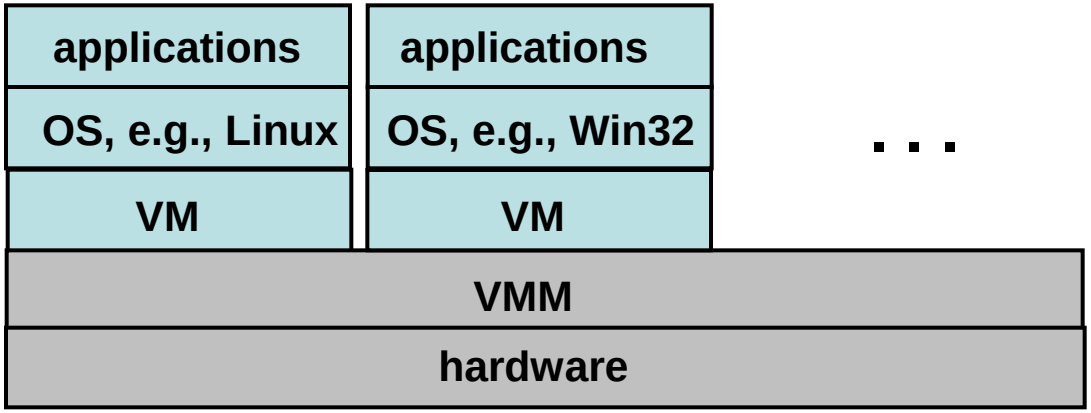
Clouds Might **Contain** Reference Monitors

(but it's a different situation)



credit: this figure inspired by the Eucalyptus system [2][9] and the Ubuntu Enterprise Cloud [10].

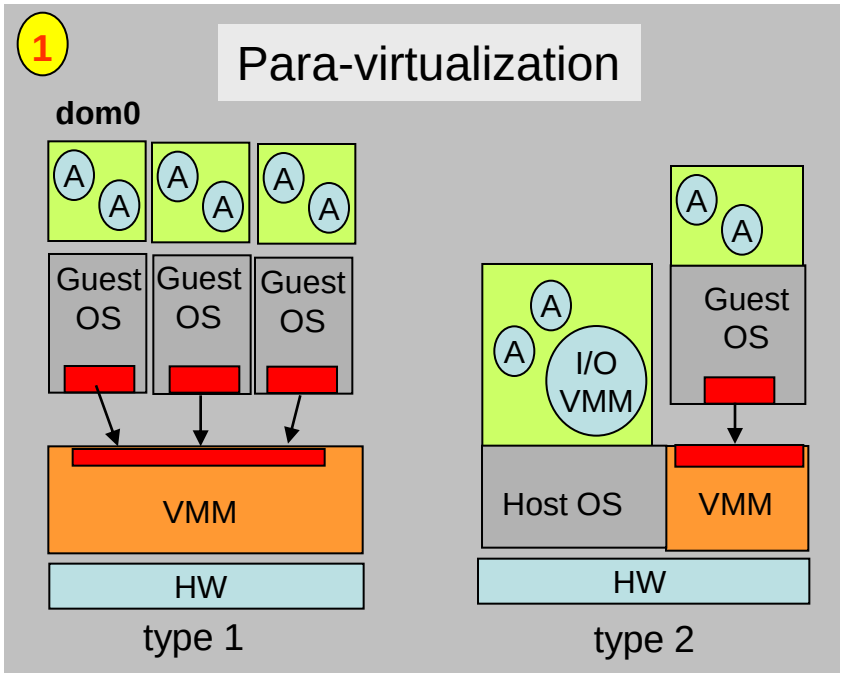
Hardware Virtualization

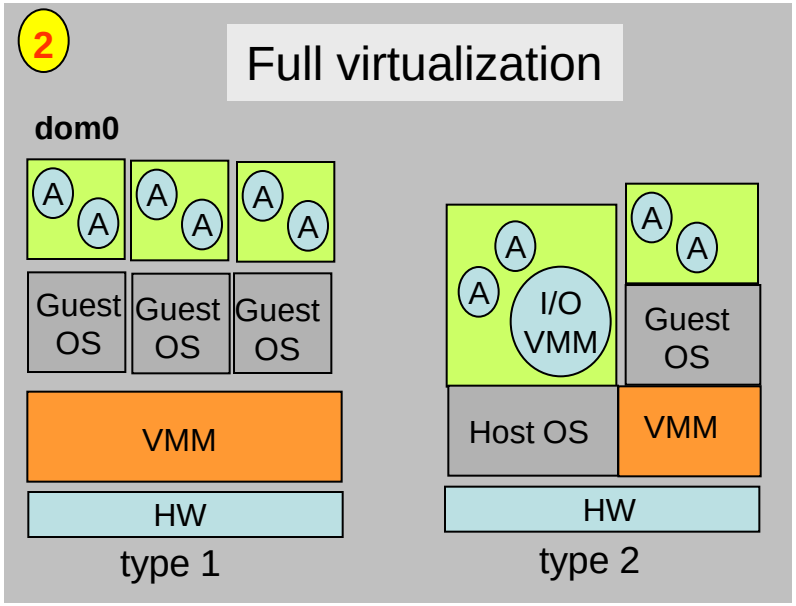


- A simple picture!
- But implementation is complex.
- Virtual Machines (VMs) can be:
  - suspended/copied/moved/lost/recovered.

Hardware Virtualization

(Box View)

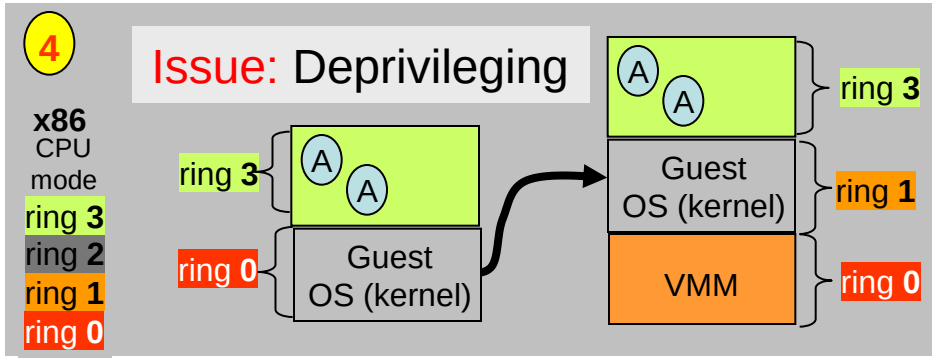




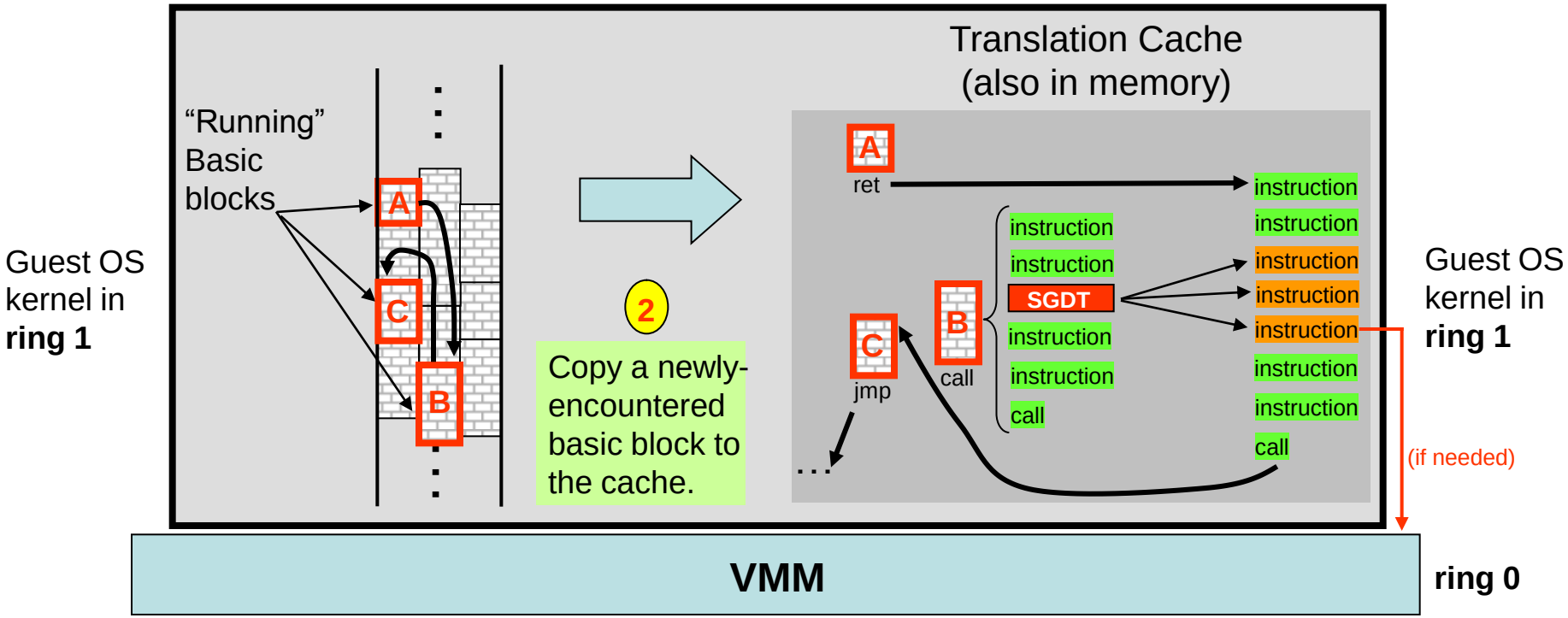
3

Terminology

Guest OS : runs only on VMM
Host OS : runs only on HW
Domain : virtual machine on VMM
Hypervisor : virtual machine monitor



Making x86 Virtualizable
Using Binary Translation



1. Identify the "next" block by scanning instructions for a jump/call/etc (that ends a basic block).

3. Binary translate any prohibited instruction into a sequence that emulates it "safely."

4. Run/rerun translated block at full speed.

Technique used by VMware, in 1999.

Making x86 Virtualizable

Using Extra Hardware

Intel 64

- Intel version of **x86-64**
- contains ~**595 instructions**.
- Hardware extensions make the instruction set virtualizable

Floating Point

Data	17
Arithmetic	26
Compare	14
Transcendental	8
Constants	7
Control	20
State management	2
	94

SIMD

MMX	47
SSE	62
SSE2	69
SSE3	13
SSSE3	32
SSE4	54
	277

General Purpose

Data transfer	32
Arithmetic	18
Logical	4
Shift/rotate	9
Bit/byte	23
Control transfer	31
String	18
I/O	8
Enter/leave	2
Flag control	11
Segment register	5
Misc	6
	167

VT-x Extensions **12**

Safe mode **1**

System **34**

64-bit mode **10**

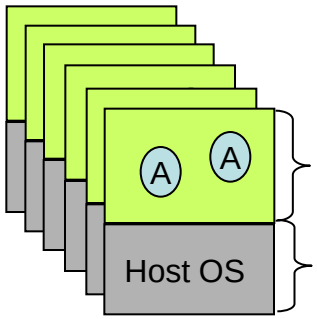
Intel Virtual Machine Extensions (VMX)

Original structure

CPU mode
ring **3**
ring **2**
ring **1**
ring **0**

Legacy software runs in the expected rings, hopefully unaware.

"there is no software-visible bit…indicates…VMX non-root operation", Intel 64 manual.



ring 3

ring 0

VMX non-root

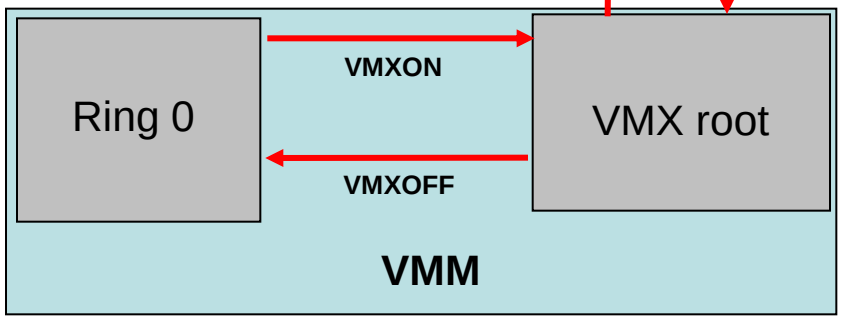
Deprivileged (very configurable).

VMXLAUNCH
VMXRESUME

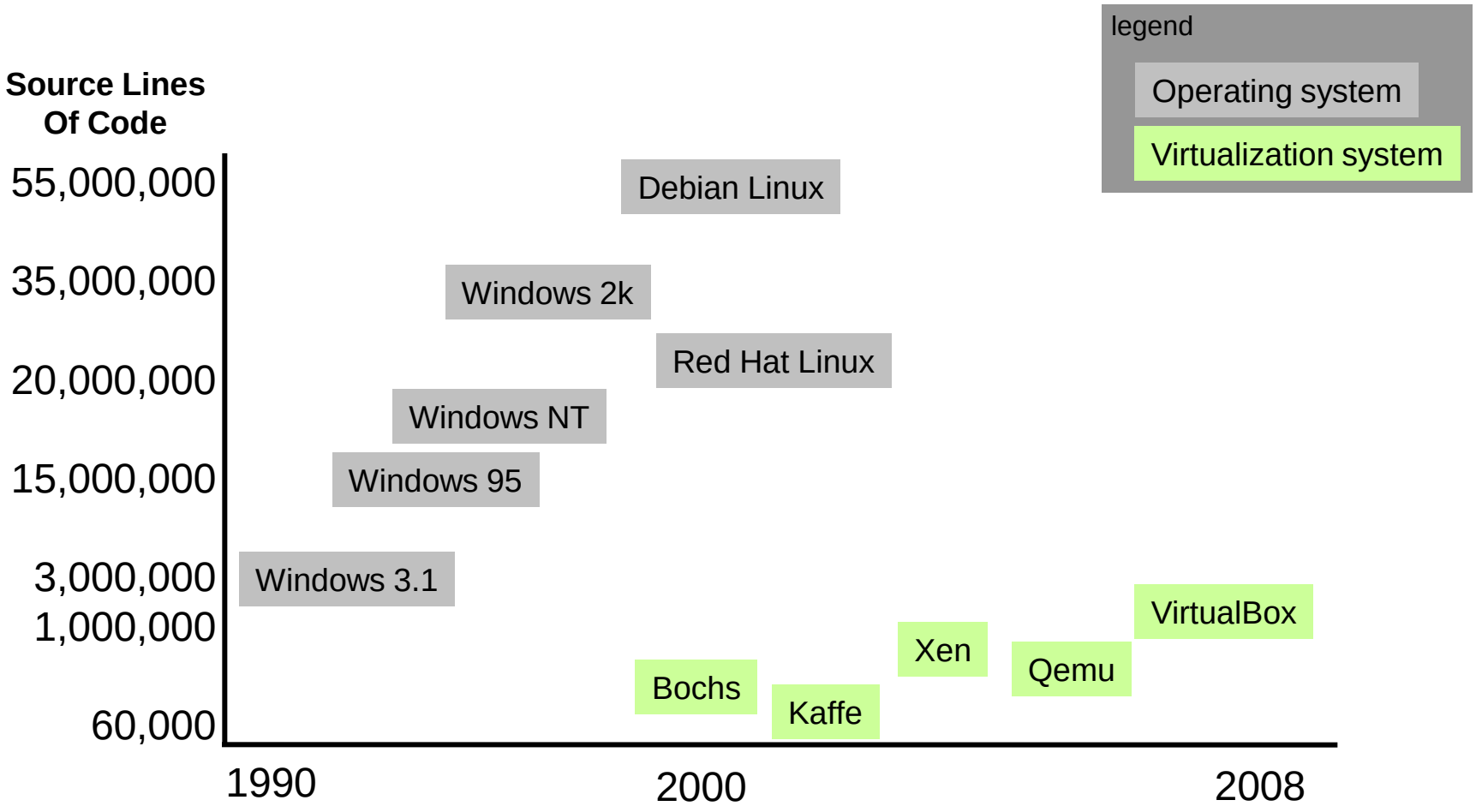
VMXCALL
"side effects"

- Many instructions cause fault-like VM exits:
  - interrupts
  - I/O events
  - page table management
  - privileged instructions, etc.
- VMM handles faults
- VM exit rate determines performance
- Address translation is complex

CPU State transitions

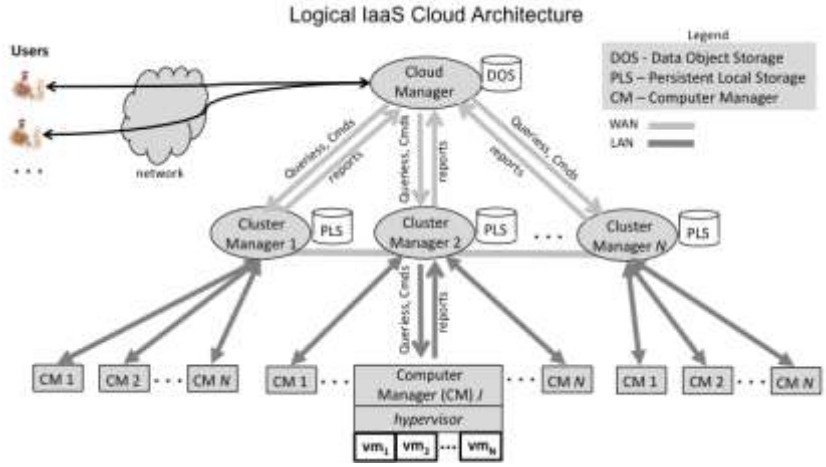


How Complex is Virtualization?



VMM code counts generated using David A. Wheeler's "SLOCCount" tool.
Windows estimate from Bruce Schneier
Linux estimates from Gonzalez-Barahona et al., and David Wheeler

Cloud Computing Security



Lack of Visibility

A number of issues:

complexity	key management
loss of (user) control	trusted platform module
network dependance	automated management
multi-tenancy	compliance
browser-dependence	. . .

3 SCAP

Is there a common thread among these Issues & Challenges?

Lack of visibility into the cloud

- Lack of concrete evidence regarding the security of the cloud environment leads to varying degrees of fear, uncertainty and doubt
- Risk: We can't understand what we can't see
- Control and visibility varies depending on the delivery and deployment model
- Operating on Faith: Trusting absent proof or material evidence

What is needed? - Trust, But Verify

- Ability to express security requirements
- Means of ensuring and reporting compliance
- Technical evidence that demonstrates how requirements are being met
- Metadata about the compliance report and technical evidence collected
- Common, uniform representations that foster interoperability across security products
- Security Automation

Role of Security Automation

Express Security Requirements

- *eXtensible Configuration Checklist Description Format (XCCDF)*
- Standard XML for specifying checklists and for reporting results of checklist evaluation
- Author checklists to assess hypervisors, guest operating systems and applications hosted in the cloud

Role of Security Automation

Common, uniform representations that foster interoperability across security products

Common Configuration Enumeration

Common Vulnerabilities and Exposures

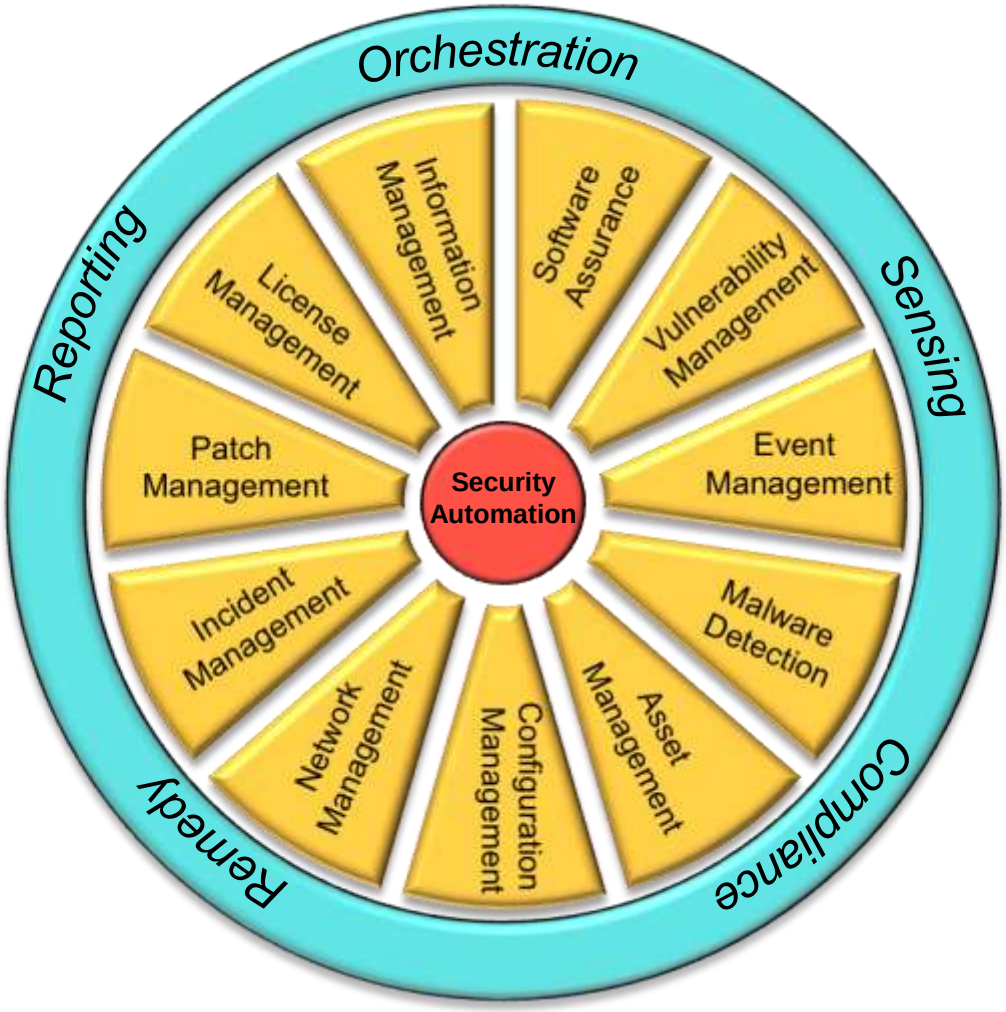
Common Platform Enumeration

Role of Security Automation

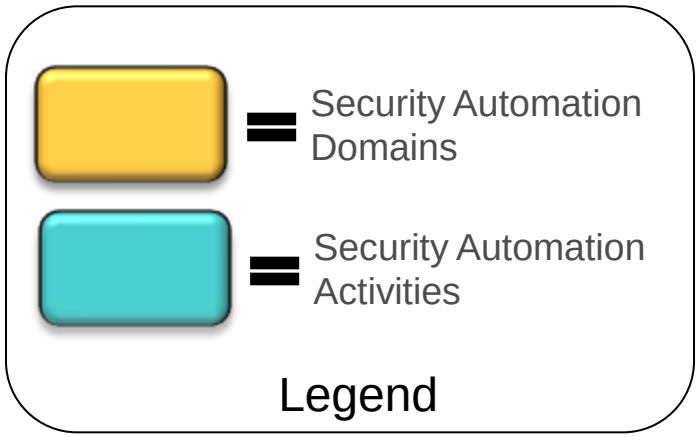
Assess and Report Assessment Details

- *Open Vulnerability and Assessment Language*
- Used to assess low-level machine state
- Able to provide detailed assessment results
- Language expresses the technical details for evaluating security settings

Future Scope of Security Automation Program



- Expose and understand the nuances of these domains and activities within cloud computing environments
- Security Automation specifications are required in each domain/activity area to ensure true interoperability across the IT security landscape.



Additional Thoughts on Automation in the Cloud

Temporal dimension is important

- Persistence
  - Short duration
    - State changes
    - Cloud Resource Provisioning Cycles
  - Monitoring change over the life of a cloud object
- Latency
  - Object is gone before you even knew it was there
  - Latency in assessment and results reporting

Additional Thoughts on Automation in the Cloud

- May require some new thinking on how we describe assets and systems
  - Composition of Assets
  - Clusters
  - Hypervisor and VMs
- Vendor publication of well-documented APIs that allow us to evaluate security state and automated security checklist guidance

4 Introduction to Standards Acceleration to Jumpstart Adoption of Cloud Computing (SAJACC).

Lee Badger
Tim Grance
Dawn Leaf

Important Cloud Computing Requirements

- **interoperability**: clouds work together
- **portability**: workloads can move around
- **security**: customer workloads protected (to the extent possible)

- Well-formulated standards could help, but they take time to evolve.

Short Term Standards Effort

- Until standards mature:
- What is needed is a **process** to test important cloud system requirements --- NIST will provide that.

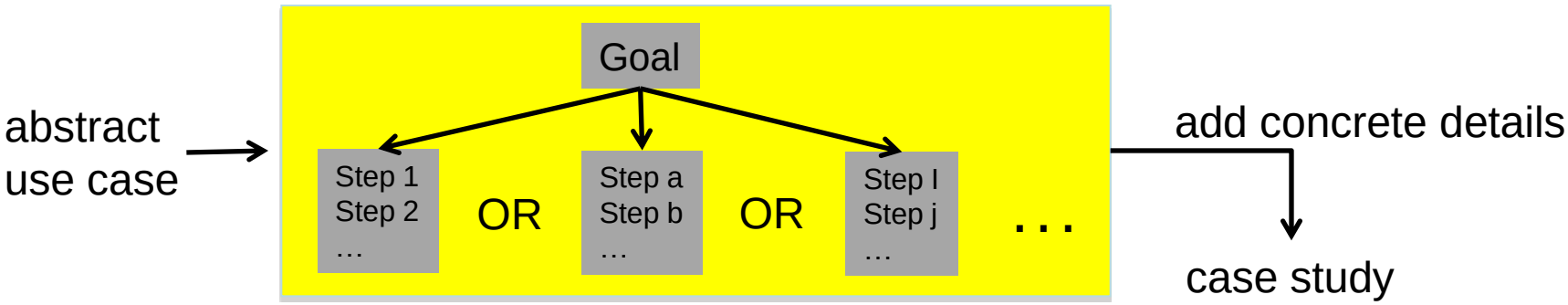
SAJACC

Portable
Interoperable
Secure (as possible)

Standards **A**cceleration to **J**umpstart **A**doption of **C**loud **C**omputing

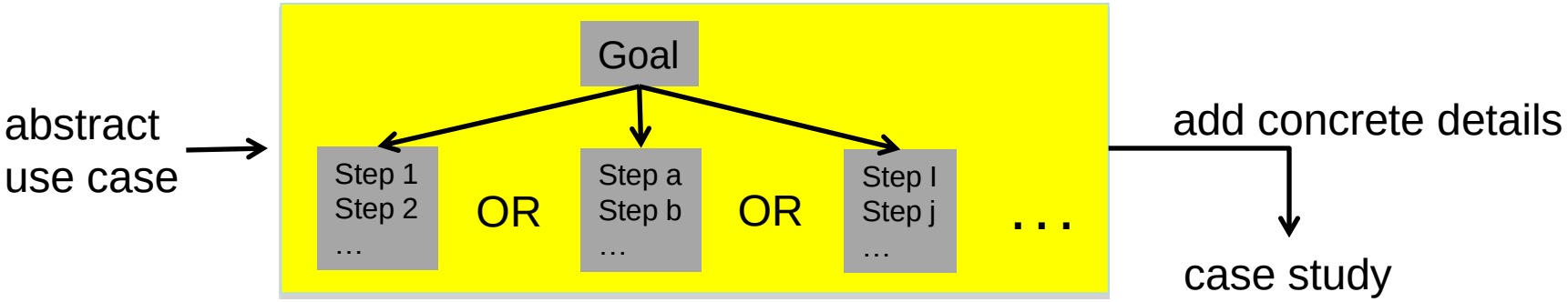
Use Cases

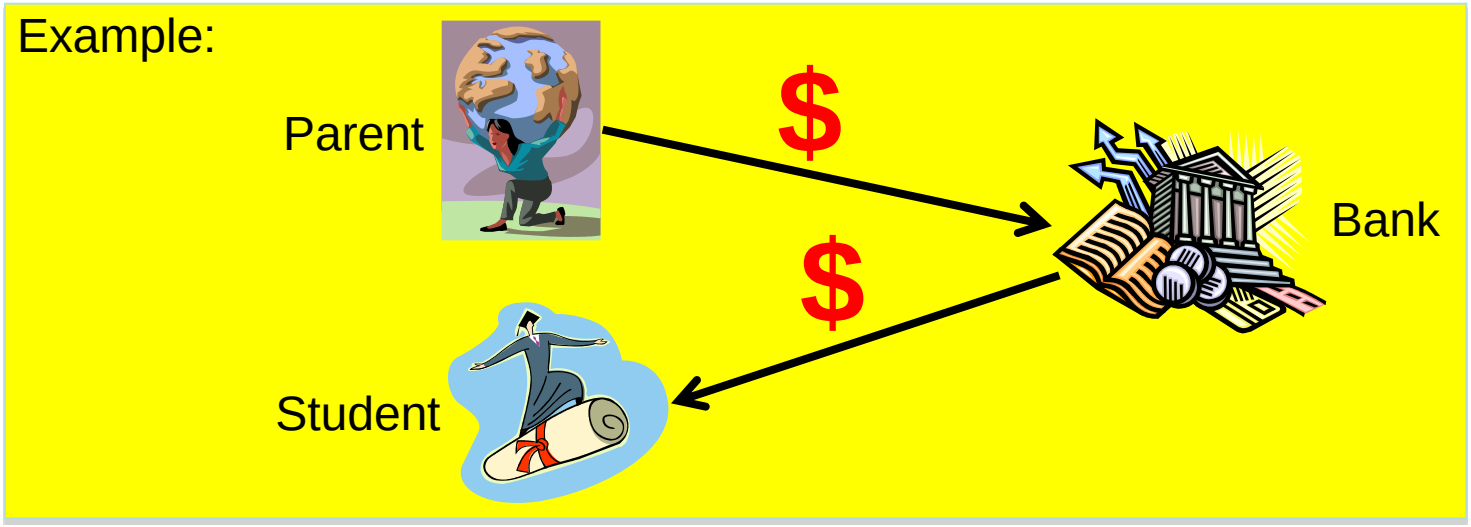
Use Case: a description of how groups of users and their resources may interact with one or more cloud computing systems to achieve specific goals.



Use Cases

Use Case: a description of how groups of users and their resources may interact with one or more cloud computing systems to achieve specific goals.





A Use Case

Use Case: a description of how groups of users and their resources may interact with one or more systems to achieve specific goals.

Actors: the active entities
Goals: what the use case tries to achieve
Assumptions: conditions assumed true

scope of application

Success Scenario 1 (name, IaaS, PaaS, SaaS) A step-by-step narrative of what happens to achieve the use case goal
Failure Conditions: what might go wrong
Failure Handling: how to deal with known failures

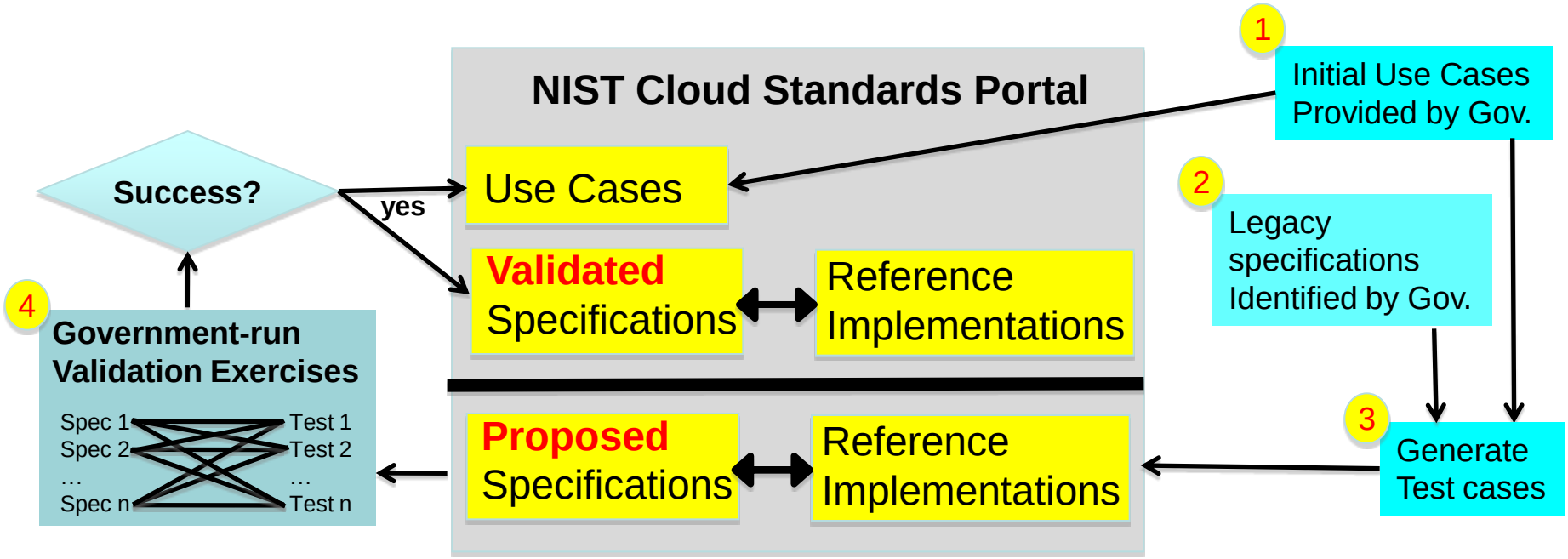
Success Scenario 2 (name, IaaS, PaaS, SaaS) Another narrative
Failure Conditions: what might go wrong
Failure Handling: how to deal with known failures

...

Credit: any source that inspired us

We are using the approach of A. Cockburn, slightly customized Cockburn: www.infor.uva.es/~mlaguna/is1/materiales/BookDraft1.pdf.

SAJACC Flow

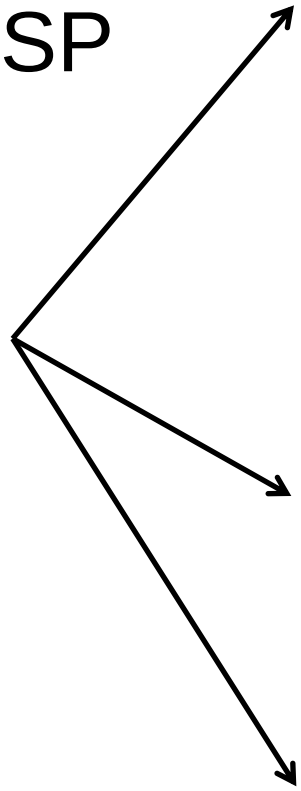


- **specifications, use cases:** provide insight on how clouds can work
- **reference implementations**: enable validation exercises
- **continuously growing portal**: new content added over time
- **publically available**: anyone can access

Use Case SP

For now, a simple taxonomy

22 use cases more on the way



Authors: Jeff Voas, Ramaswamy Chandramouli, Robert Patt-Corner, Robert Bohn, Tom Karygiannis, Tim Grance, Lee Badger.

Credit: various use cases inspired by Amazon, the Eucalyptus project, the DMTF, SNIA, the libcloud project, and by Gaithersburg MD May 2010 use case workshop participants.

References

[1] Amazon Web Services, aws.amazon.com.

[2] “Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Useful Systems”, UCSB Computer Science Technical Report Number 2008-10.

[3] IDC Enterprise Panel, August 2008 n=244

[4] “Interoperable Clouds, A White Paper from the Open Cloud Standards Incubator”, Distributed Management Task Force, Version 1.0, DMTF Informational, Nov. 11, 2009, DSP-IS0101

[5] libcloud, http://incubator.apache.org/libcloud/

[6] “Open Virtualization Format Specification”, DMTF Document Number DSP0243, Version 1.0, Feb. 22, 2009.

[7] “Cloud Storage Use Cases”, Storage Network Industry Association, Version 0.5 rev 0, June 8, 2009.

[8] “Starting Amazon EC2 with Mac OS X”. Robert Sosinski. http://www.robertsosinski.com/2008/01/26 /starting-amazon-ec2-with-mac-os-x/

[9] “The Eucalyptus Open-source Cloud-computing System”, D. Nurmi, R. Wolski, C. Grzegorcyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov, in Proceedings of Cloud Computing and Its Applications, Oct. 2008.

[10] “Ubuntu Enterprise Cloud Architecture”, S. Wardley, E. Goyer and N. Barcet, Technical White Paper, 2009, www.canonical.com

Backup

VMM Implementation Quality Should Not be Assumed

In 2007, Tavis Ormandy subjected 6 virtualization systems to guided random testing of their invalid instruction handling and I/O emulation.

Bochs	QEMU	VMWare	Xen	Anonymous 1	Anonymous 2
178k SLOC	373k SLOC		910k SLOC		

All of the systems failed the tests, most with “arbitrary execution” failures.

Device emulation was a particular area of vulnerability.

For details, see: taviso.decsystem.org/virtsec.pdf

Reference: “An Empirical Study into the Security Exposures to Host of Hostile Virtualized Environments,” by Travis Ormandy. taviso.decsystem.org/virtsec.pdf

Code counts generated using David A. Wheeler's “SLOCCount” tool.