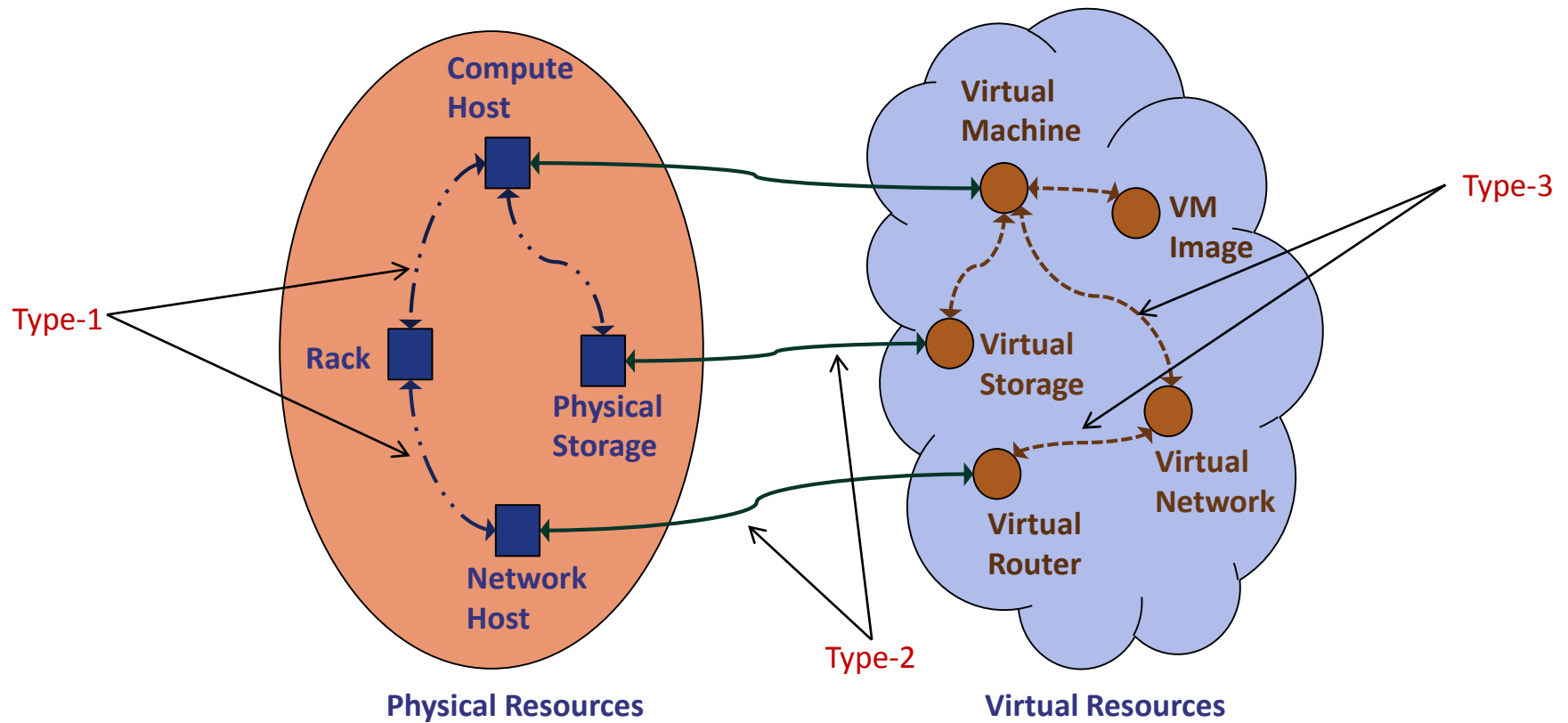


Constraints Specification for Virtual Resource Orchestration in Cloud IaaS

Khalid Bijon, Ram Krishnan and Ravi Sandhu
University of Texas at San Antonio

5th ACM Conference on Data and Applications Security and Privacy (CODASPY 2015)

- **Introduction**
- **Motivation**
- **Goal**
- **Methodology**
- **Enforcement (in Cloud IaaS)**
- **Implementation (in OpenStack)**
- **Conclusion**



- Three Different Mapping Types
- Shared Responsibility
- Only Consider Type-3 Mappings
- Complex Management Process

■ Inefficient and Tedious Management Plane

- ◆ Manual Identification
- ◆ User Centric (**unnecessary indirection**)

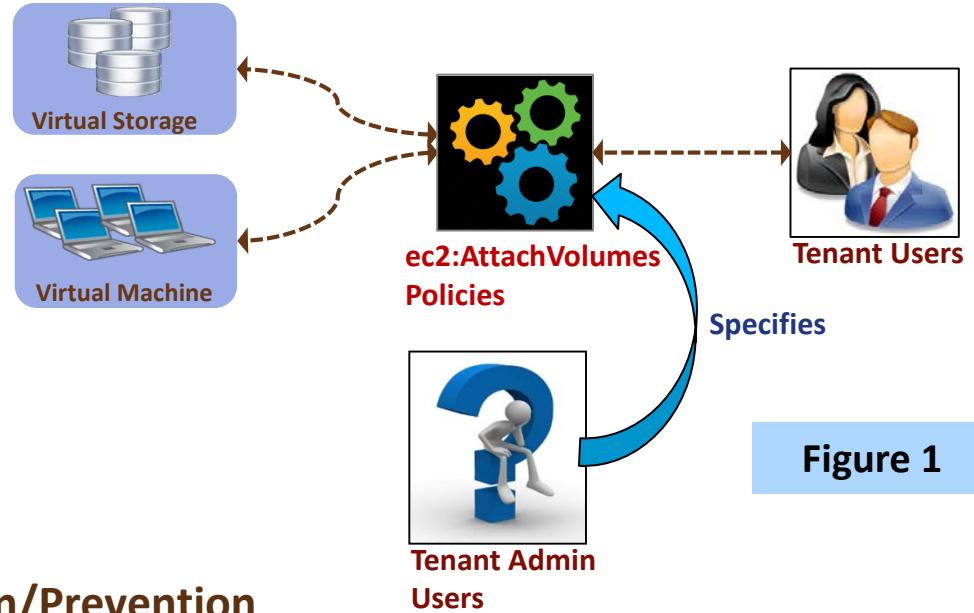


Figure 1

■ No Direct Misconfiguration Detection/Prevention

- ◆ Elevate Security Vulnerability



Manual Detection



Credit: www.iconarchive.com
www.consulting.ky
www.acm.icpc.org

■ Easily Manageable Type 3 Mapping

- ◆ High-level Policy
- ◆ Configure Diverse Requirements



Tenants

■ Establish Direct Relations

- ◆ Keep Users Out of Loop



■ Automatically Prevent Misconfiguration

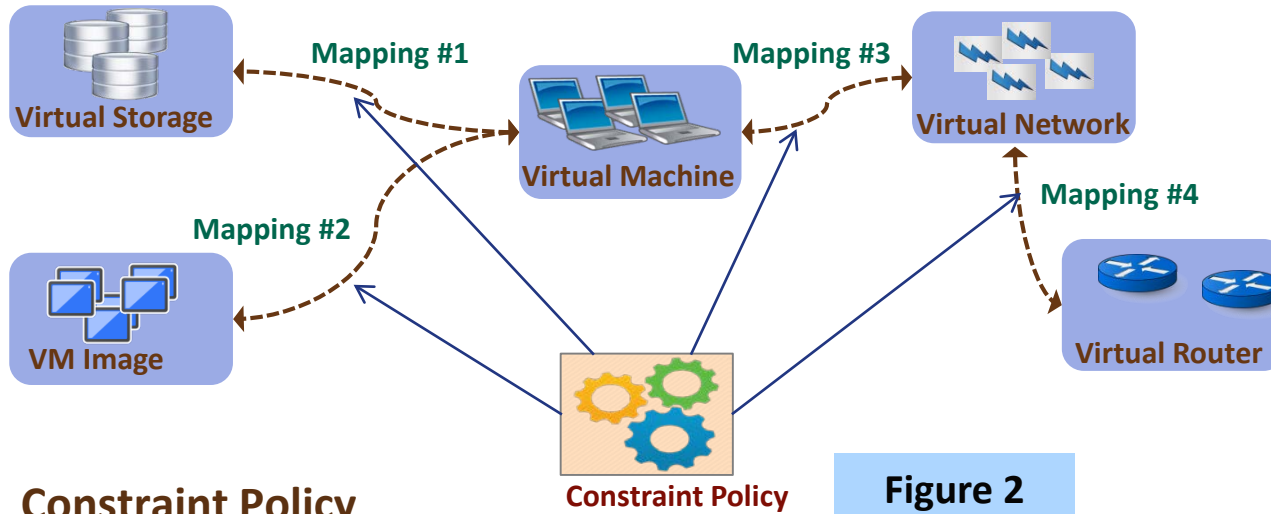


Figure 2

■ **Constraint Policy**

- ◆ For Each Type-3 Mappings

■ **Satisfied By**

- ◆ Individual Virtual Resources

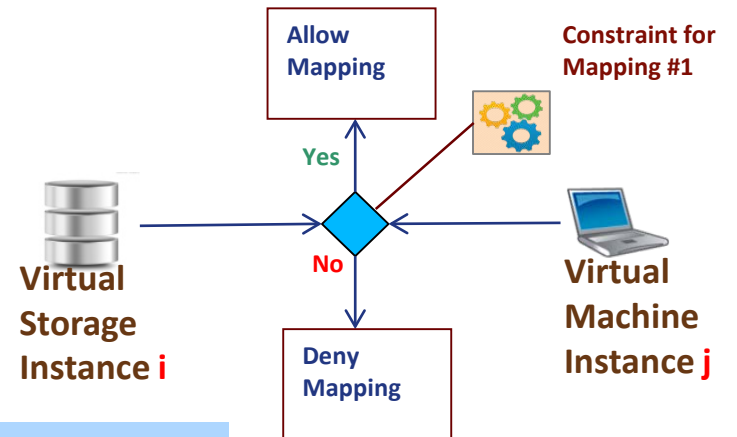


Figure 3

Credit: www.iconarchive.com

- **Attribute Specifies Virtual Resource Properties**

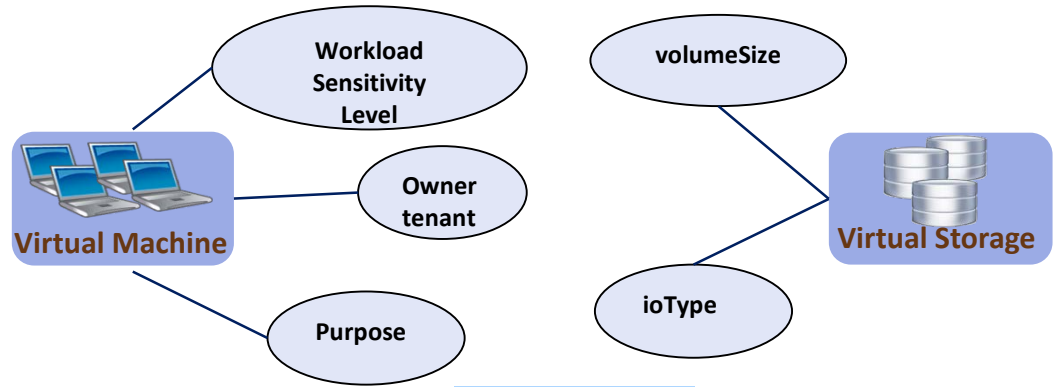


Figure 4

- **A name:value Pair**

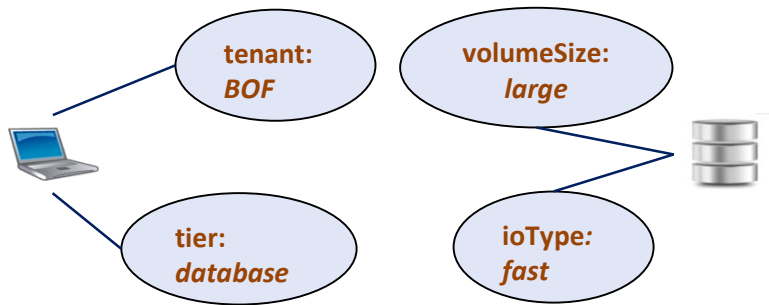


Figure 5

- **Designed as Functions**

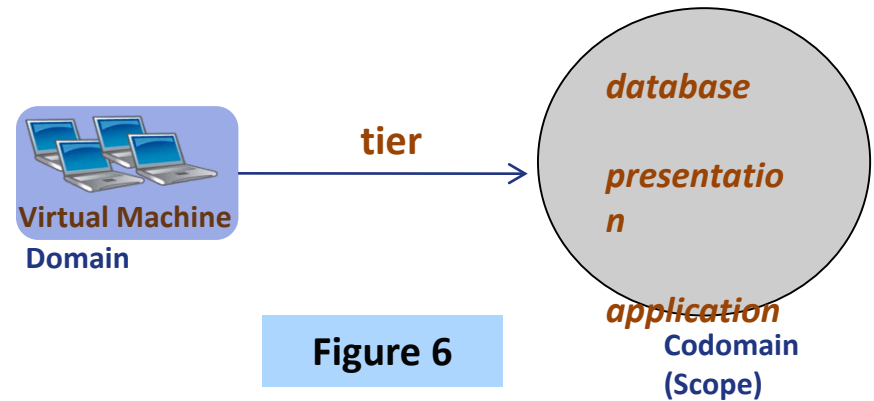


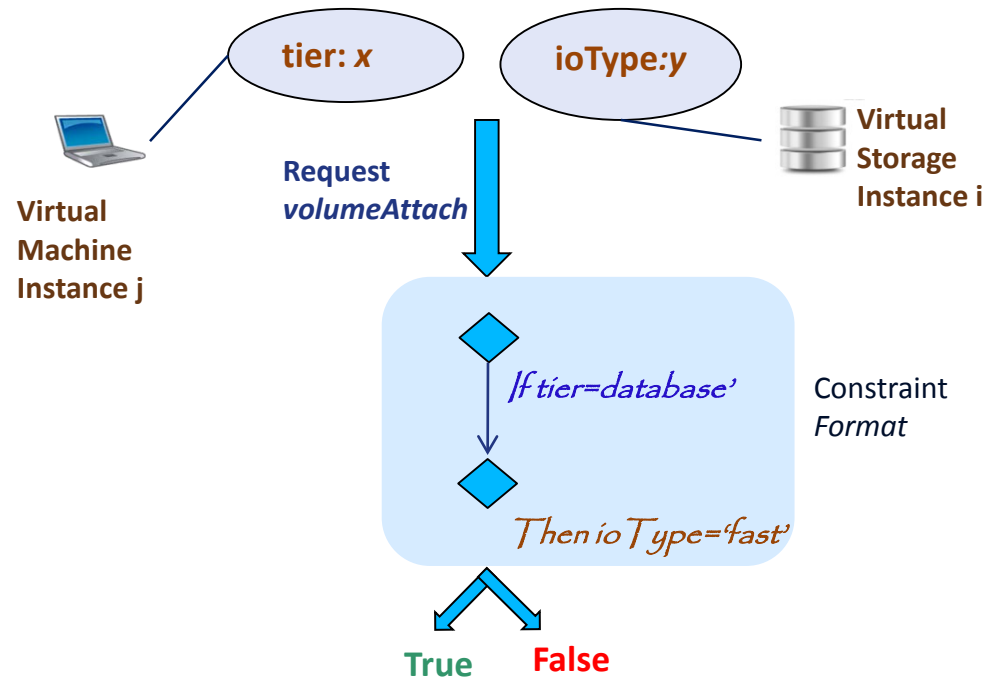
Figure 6

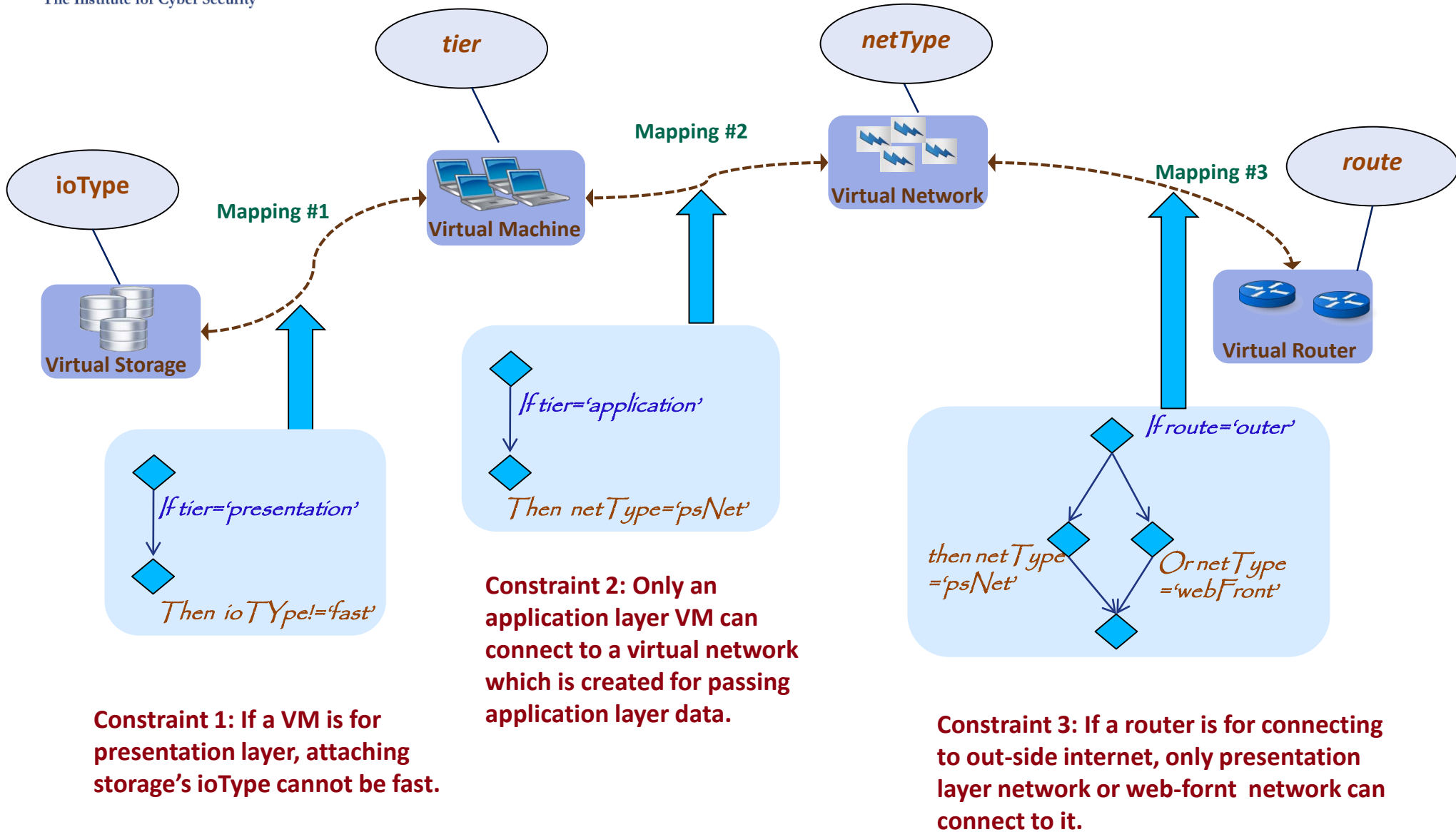
■ A Constraint

- ◆ Logical Formula
- ◆ Compares Certain Attribute Values

■ Simple but Powerful

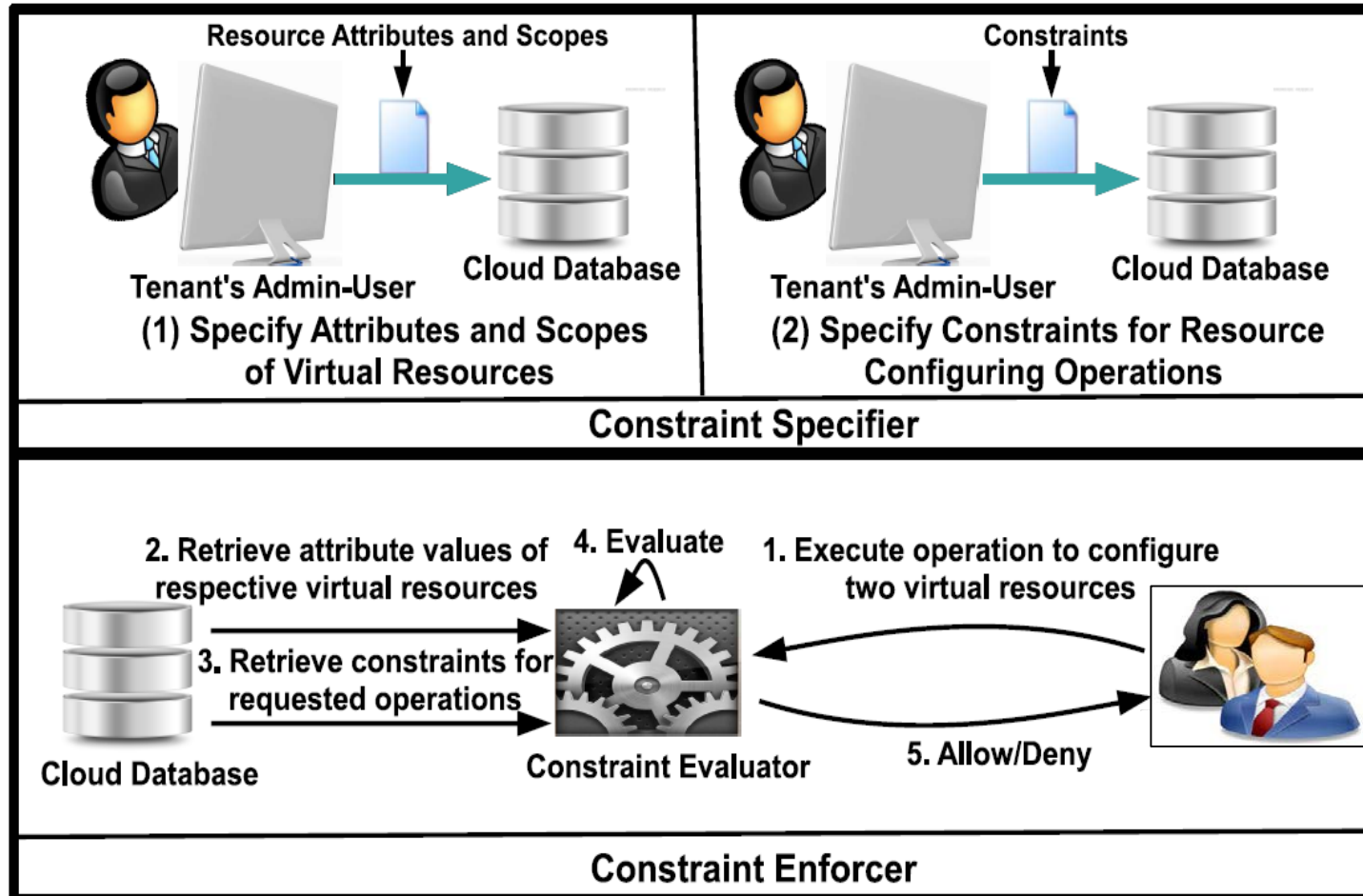
- ◆ Hadoop Cluster
- ◆ 3-tier business application



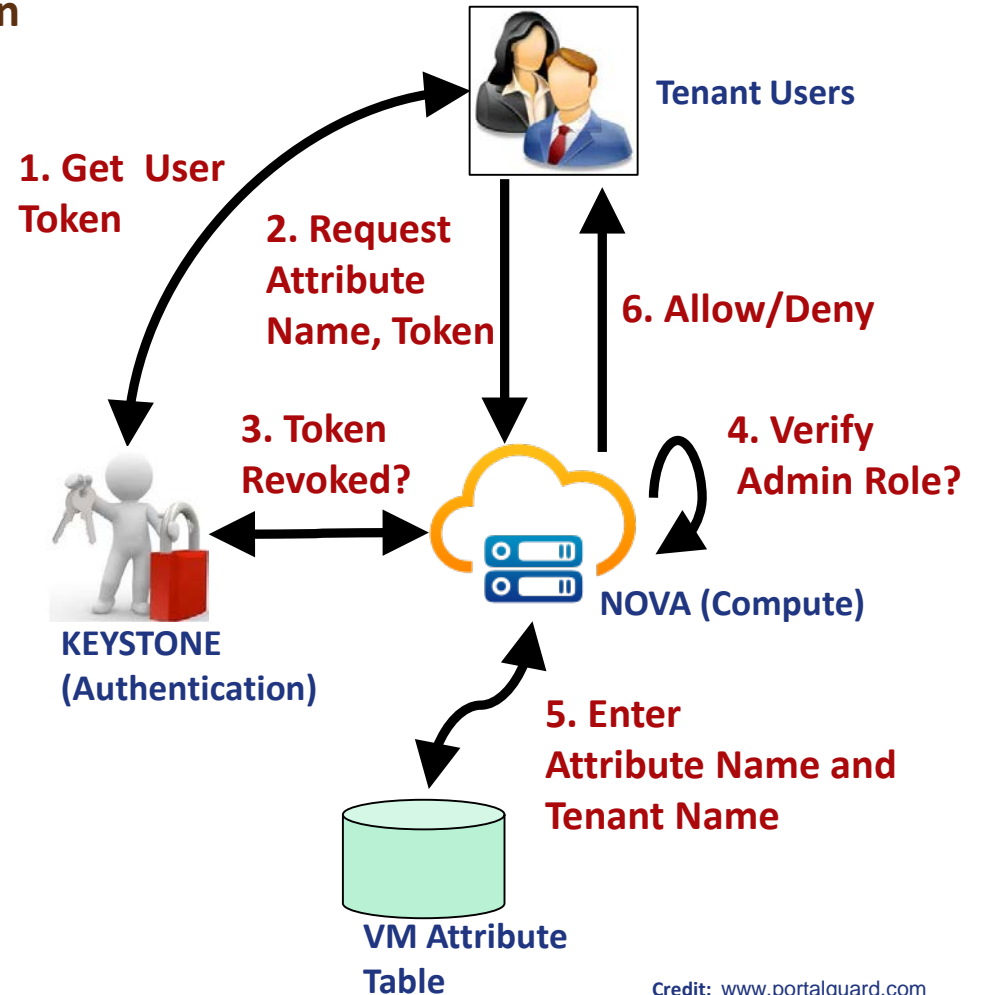


Two Components

- ◆ Specifier and Enforcer




- Implemented in OpenStack
- Execution of “*attribute-creation*” operation
- Similarly,
 - ◆ Attribute-value specification
 - ◆ Constraint Specification
 - ◆ Attribute-value assignment

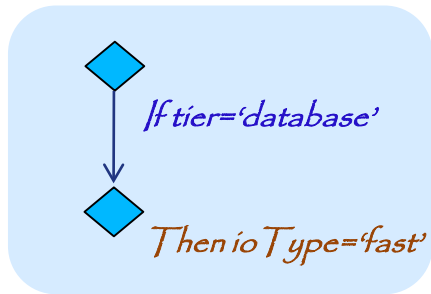


Credit: www.portalguard.com

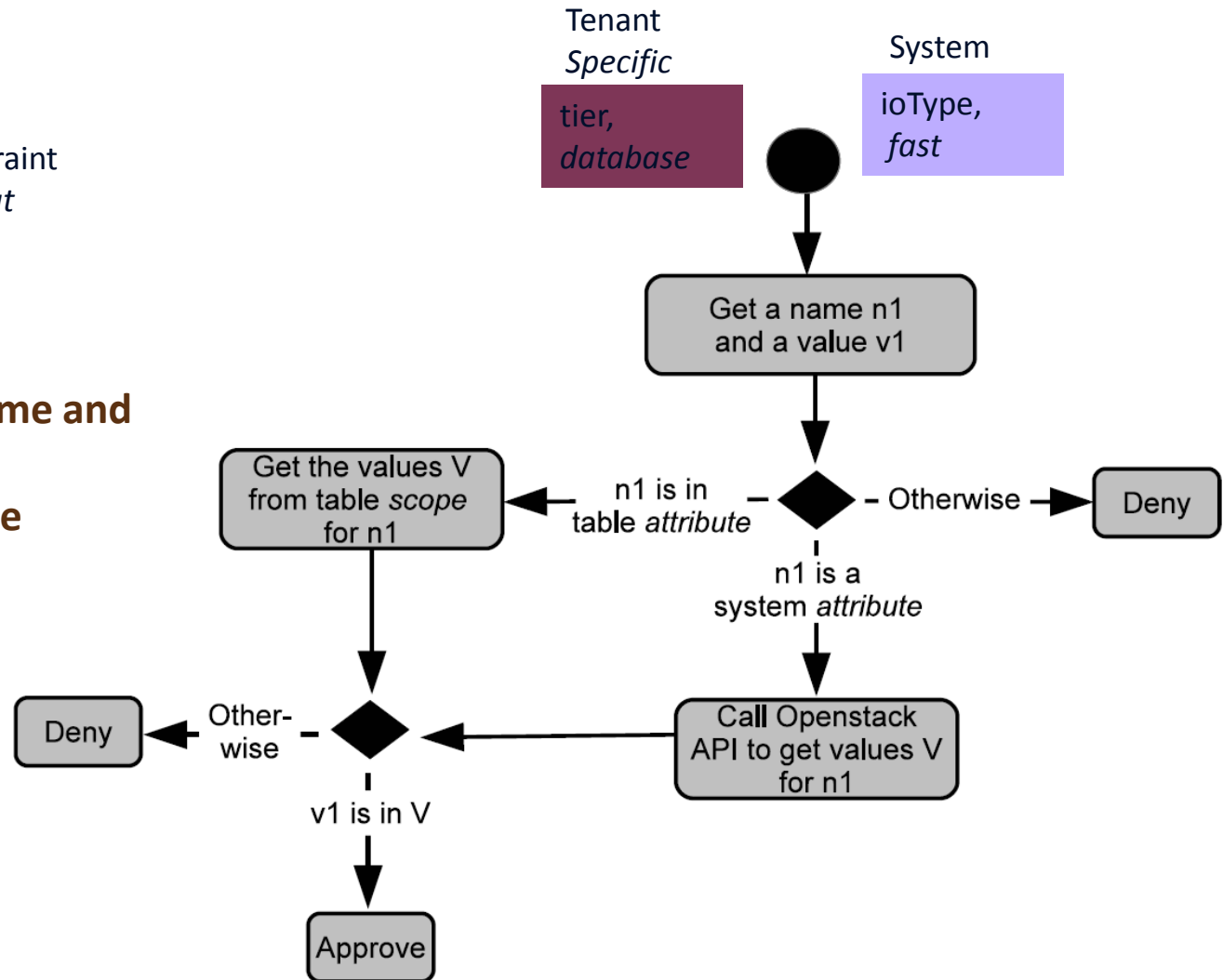
■ API Specification

◆ Rest API

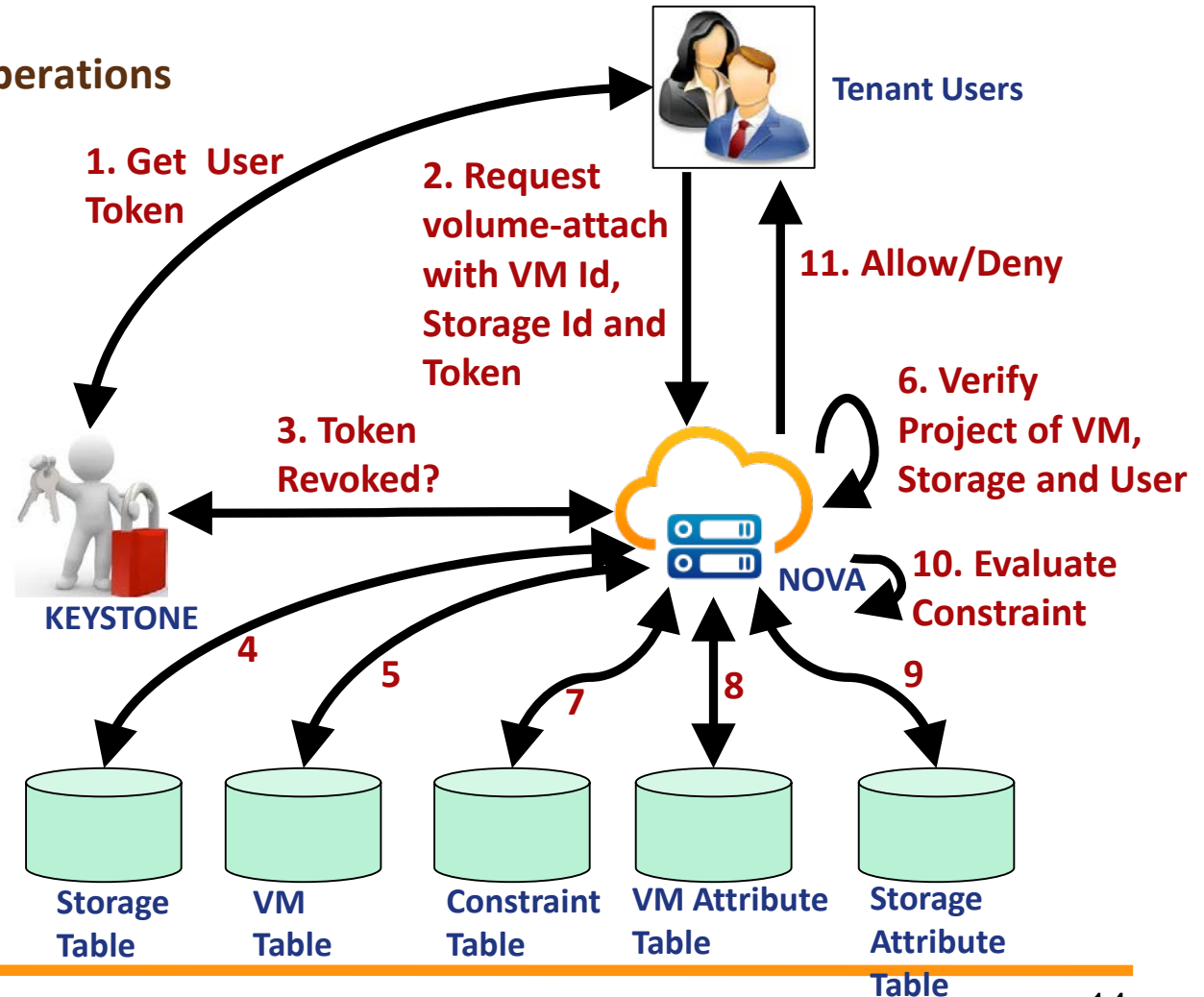
	Name	URL	Type
Attribute Name	att-create	/v2/{tenant_id}/attributes	POST
	att-delete	/v2/{tenant_id}/attributes/{id}	DELETE
	att-list	/v2/{tenant_id}/attributes	GET
Attribute Value	att-value-set	/v2/{tenant_id}/scopes	POST
	att-value-delete	/v2/{tenant_id}/scopes/{id}	DELETE
 Attribute Value Assignment	att-value-get	/v2/{tenant_id}/scopes/	GET
	constraint-add	/v2/{tenant_id}/constraints	POST
	constraint-delete	/v2/{tenant_id}/constraints/{id}	DELETE
	constraint-get	/v2/{tenant_id}/constraints	GET
World-Leading meta		/v2/{tenant_id}/servers/{resource_id}/metadata	POST



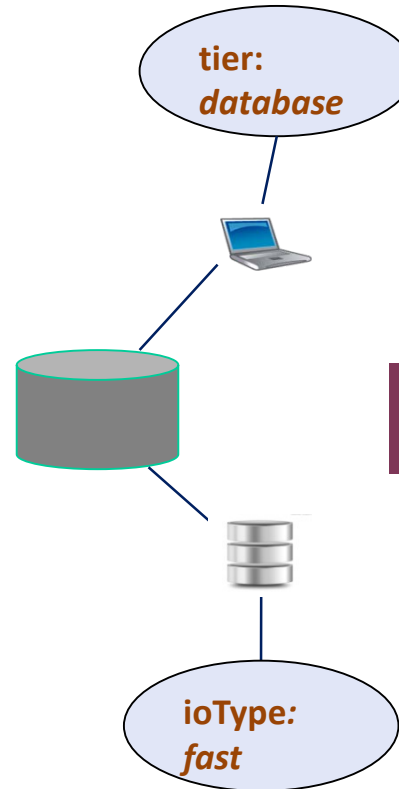
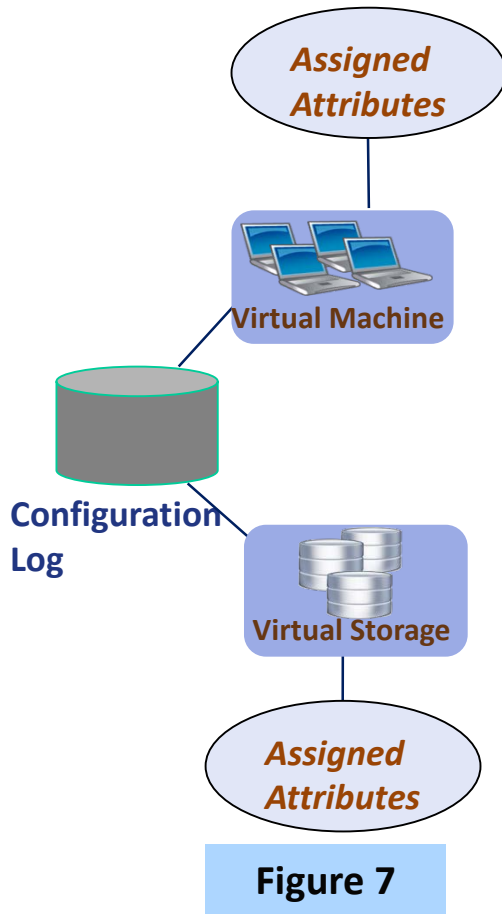
- **Validity of Attribute Name and Value**
- **Tenant-Specific Attribute**
- **System or Inter-Tenant Attribute**



- Implemented in OpenStack
- A Constraint Parser
- Invoked by Resource Mapping Operations (e.g., *volume-attach*)



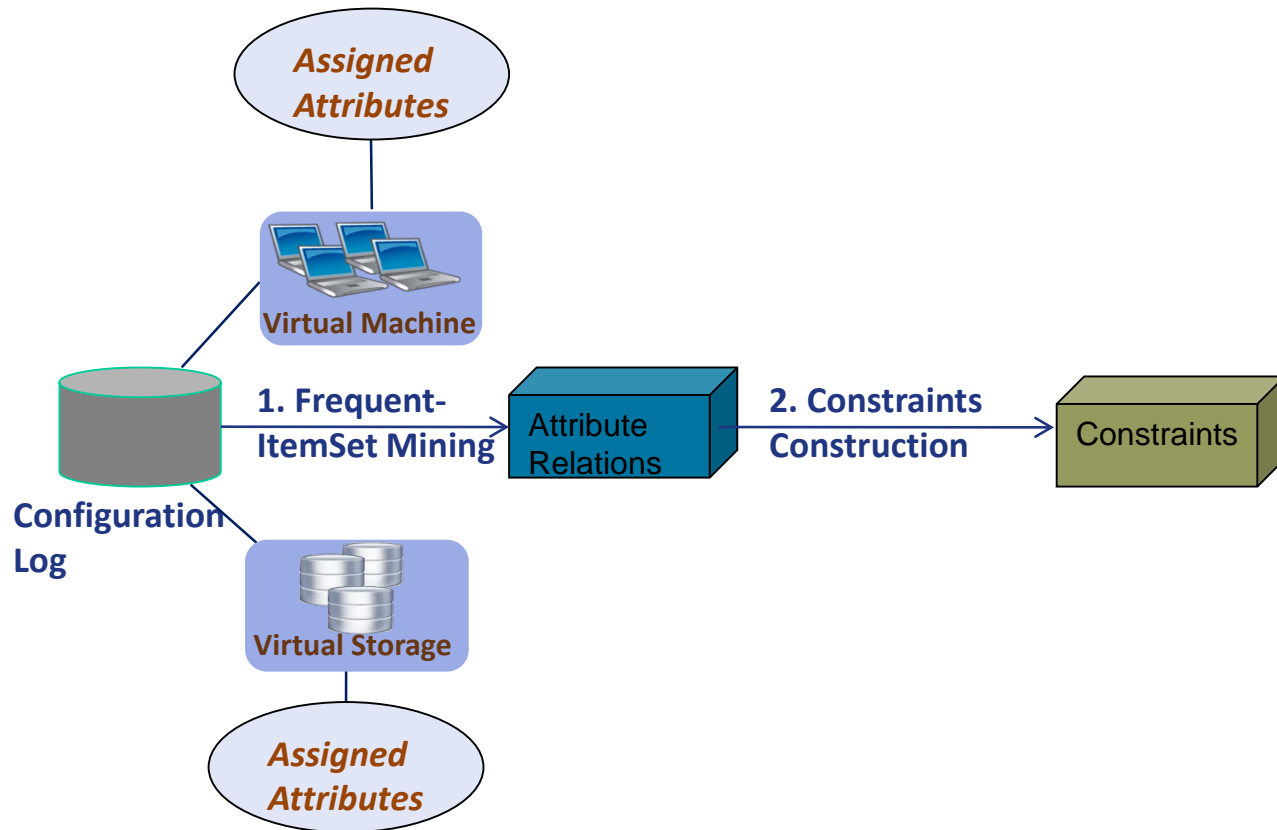
- Helps the tenants to find policy
- From Previous Configurations



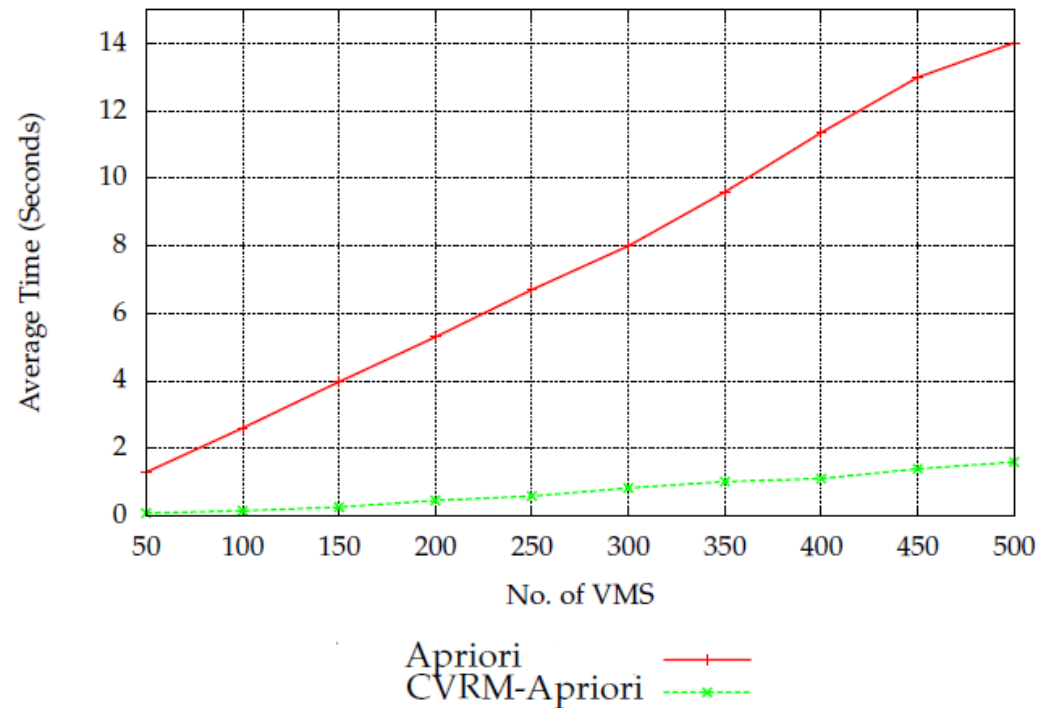
- Construct Relation between values of two attributes

■ Frequent-ItemSet Mining

- ◆ Apriori Algorithm
- ◆ with customization for IaaS (CVRM-Apriori)

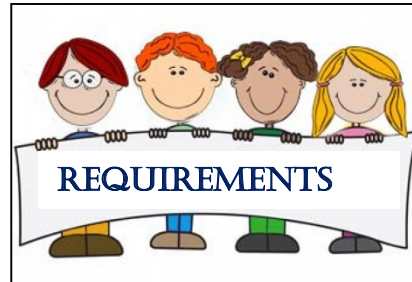


- Policy for VM-Network Connectivity Mapping
- From VM-Network Table (table *virtual_interfaces* in Nova, OpenStack)
- 10 Attributes each with 10 values
- 10 Virtual Networks
- At least three Networks per VM
- Mine relations between every two pair of attribute values



- A Constraint Specification Framework

- Easily manageable and generic



Tenants

- Can be applied for Misconfiguration Prevention

- Also, for detection (flag-generator)

- Automatic Generation of Constraints



- Flag Generator System

- Semantic meaning of mined Attribute Relation

- Improve mining (incorporate noise)

$$\begin{aligned}
 \langle \text{Quantifier} \rangle &::= \forall(\text{vr1}, \text{vr2}) \in \mathcal{R}_{\langle \text{Cls} \rangle}, \langle \text{Cls} \rangle . \langle \text{Stmt} \rangle \\
 \langle \text{Stmt} \rangle &::= \langle \text{Stmt} \rangle \langle \text{connector} \rangle \langle \text{Stmt} \rangle \mid (\langle \text{rule} \rangle) \\
 \langle \text{rule} \rangle &::= \langle \text{Token} \rangle \rightarrow \langle \text{Token} \rangle \\
 \langle \text{Token} \rangle &::= (\langle \text{Token} \rangle \langle \text{connector} \rangle \langle \text{Token} \rangle) \mid (\langle \text{Term} \rangle) \\
 \langle \text{Term} \rangle &::= \langle \text{Attribute} \rangle (\langle \text{resource} \rangle) \langle \text{comperator} \rangle \langle \text{Scope} \rangle \\
 \langle \text{Attribute} \rangle &::= \langle \text{letter} \rangle \mid \langle \text{digit} \rangle \mid \langle \text{Attribute} \rangle \\
 \langle \text{Scope} \rangle &::= \langle \text{letter} \rangle \mid \langle \text{digit} \rangle \langle \text{Scope} \rangle \\
 \langle \text{connector} \rangle &::= \wedge \mid \vee \\
 \langle \text{comparator} \rangle &::= = \mid \neq \\
 \langle \text{Cls} \rangle &::= c_1 \mid c_2 \mid \dots \mid c_n \\
 \langle \text{resoruce} \rangle &::= \text{vr1} \mid \text{vr2} \\
 \langle \text{digit} \rangle &::= 0 \mid 1 \mid 2 \mid \dots \mid 8 \mid 9 \\
 \langle \text{letter} \rangle &::= a \mid b \mid \dots \mid y \mid z \mid A \mid B \mid \dots \mid Y \mid Z
 \end{aligned}$$

Field	Type
id	Integer
name	String(255)
project_id	String(255)

(a) The *attribute* Table

Field	Type
id	Integer
relation_name	String(255)
expression	String(255)
project_id	String(255)

(c) The *constraints* Table

Field	Type
id	Integer
name	String(255)
value	String(255)
project_id	String(255)

(a) The *scope* Table

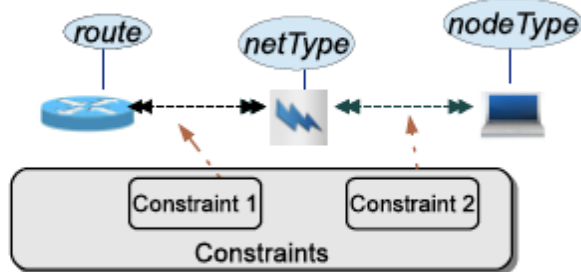
Field	Type
vm_id	Integer
vm_id	String(255)
meta	String(255)

(d) The *instance-metadata* Table

3-Tier Application System Configurations	C: Constraint Specification
<p>A: Virtual Resources, Attributes and Constraints</p>	<p>Constraints for router-network connection mapping:</p> <p>Const 1: <i>If route attribute of a router is outerRoute then only network with netType outerNet can connect to it.</i></p> $((route(router)=outerR) \rightarrow ((netType(network)=outerNet) \vee (netType(network)=psNet)))$
<p>B: Scopes of the Attributes</p> $ATTR_{3\text{-tier}}^{VM} = \{tier, versionVM, status\}$ $SCOPE_{tier} = \{presentation, application, database\}$ $SCOPE_{versionVM} = \{psV1, psV2, app1, dbV1, dbV2\}$ $SCOPE_{status} = \{running, stop, suspended\}$ $ATTR_{3\text{-tier}}^{IMG} = \{tier, versionIMG\}$ $SCOPE_{tier} = \{presentation, application, database\}$ $SCOPE_{versionIMG} = \{psI1, psI2, appI1, dbI1\}$ $ATTR_{3\text{-tier}}^{NET} = \{netType\}$ $SCOPE_{netType} = \{outerNet, psNet, appNet, dbNet\}$ $ATTR_{3\text{-tier}}^{RTR} = \{route\}$ $SCOPE_{route} = \{outerR, psToappR, appTodbR\}$ $ATTR_{3\text{-tier}}^{STR} = \{ioType, volumeSize\}$ $SCOPE_{ioType} = \{regular, fast, fastest\}$ $SCOPE_{volumeSize} = \{regular, large, huge\}$	<p>Constraints for network-vm connection mapping:</p> <p>Const 2: <i>Only presentation layer vm can connect to a psNet network. Similar, constraints can be generated for other layer vms and networks.</i></p> $((netType(network)=psNet) \rightarrow (tier(vm)=presentation))$ <p>Const 3: <i>A network can be removed from a vm if the vm status is stop.</i></p> $((status(vm)=stop))$ <p>Constraints for image-vm connection mapping:</p> <p>Const 4: <i>If tier of IMG is presentation and versionImg is psI1, it cannot be used by presentation VM with versionVM psV2.</i></p> $(((tier(image)=presentation) \wedge (versionImg(image)=psI1)) \rightarrow (((tier(vm)=presentation) \wedge (versionVM(vm) \neq psV2)) \vee (tier(vm) \neq presentation))))$ <p>Constraints for storage-vm connection mapping:</p> <p>Const 5: <i>A presentation vm cannot get fastest ioType storages.</i></p> $((tier(vm)=presentation) \rightarrow (ioType(storage) \neq fastest))$

Hadoop Cluster Configurations

A: Virtual Resources, Attributes and Constraints



B: Scopes of the Attributes

$$\text{ATTR}_{\text{hadoop}}^{\text{VM}} = \{ \text{nodeType} \}$$

$$\text{SCOPE}_{\text{nodeType}} = \{ \text{clientNode}, \text{nameNode}, \text{jobTracker}, \text{mapTask}, \text{reduceTask} \}$$

$$\text{ATTR}_{\text{hadoop}}^{\text{NET}} = \{ \text{netType} \}$$

$$\text{SCOPE}_{\text{netType}} = \{ \text{outerNet}, \text{clientNet}, \text{nameNet}, \text{jobNet}, \text{mapNet}, \text{reduceNet} \}$$

$$\text{ATTR}_{\text{hadoop}}^{\text{RT}} = \{ \text{route} \}$$

$$\text{SCOPE}_{\text{route}} = \{ \text{outerRoute}, \text{nameRoute}, \text{jobRoute}, \text{taskRoute} \}$$

C: Constraint Specification

Constraints for router-network connection mapping:

Constraint 1: *If route attribute of a router is outerRoute then only network with netType outerNet and clientNet can connect to it and if route attribute is taskRoute then it cannot be connected with nameNet, outerNet and clientNet.*

$$\begin{aligned} & ((\text{route}(\text{router}) = \text{outerRoute}) \rightarrow \\ & ((\text{netType}(\text{network}) = \text{outerNet}) \vee (\text{netType}(\text{network}) = \text{clientNet}))) \\ & \wedge ((\text{route}(\text{router}) = \text{taskRoute}) \rightarrow \\ & (((\text{netType}(\text{network}) \neq \text{nameNet}) \wedge (\text{netType}(\text{network}) = \text{outerNet})) \\ & \wedge (\text{netType}(\text{network}) = \text{clientNet}))) \end{aligned}$$

Constraints for network-vm connection mapping:

Constraint 2: *In a nameNet network only nameNode and jobTracker vm can be connected.*

$$\begin{aligned} & ((\text{netType}(\text{network}) = \text{nameNet}) \rightarrow \\ & ((\text{nodeType}(\text{vm}) = \text{nameNode}) \vee (\text{nodeType}(\text{vm}) = \text{jobTracker}))) \end{aligned}$$