Mitigating Multi-Tenancy Risks in IaaS Cloud Through Constraints-Driven Virtual Resource Scheduling

Khalid Bijon, **Ram Krishnan**, and Ravi Sandhu

The University of Texas at San Antonio, USA

ACM Symposium on Access Control Models and Technologies

(SACMAT 2015)

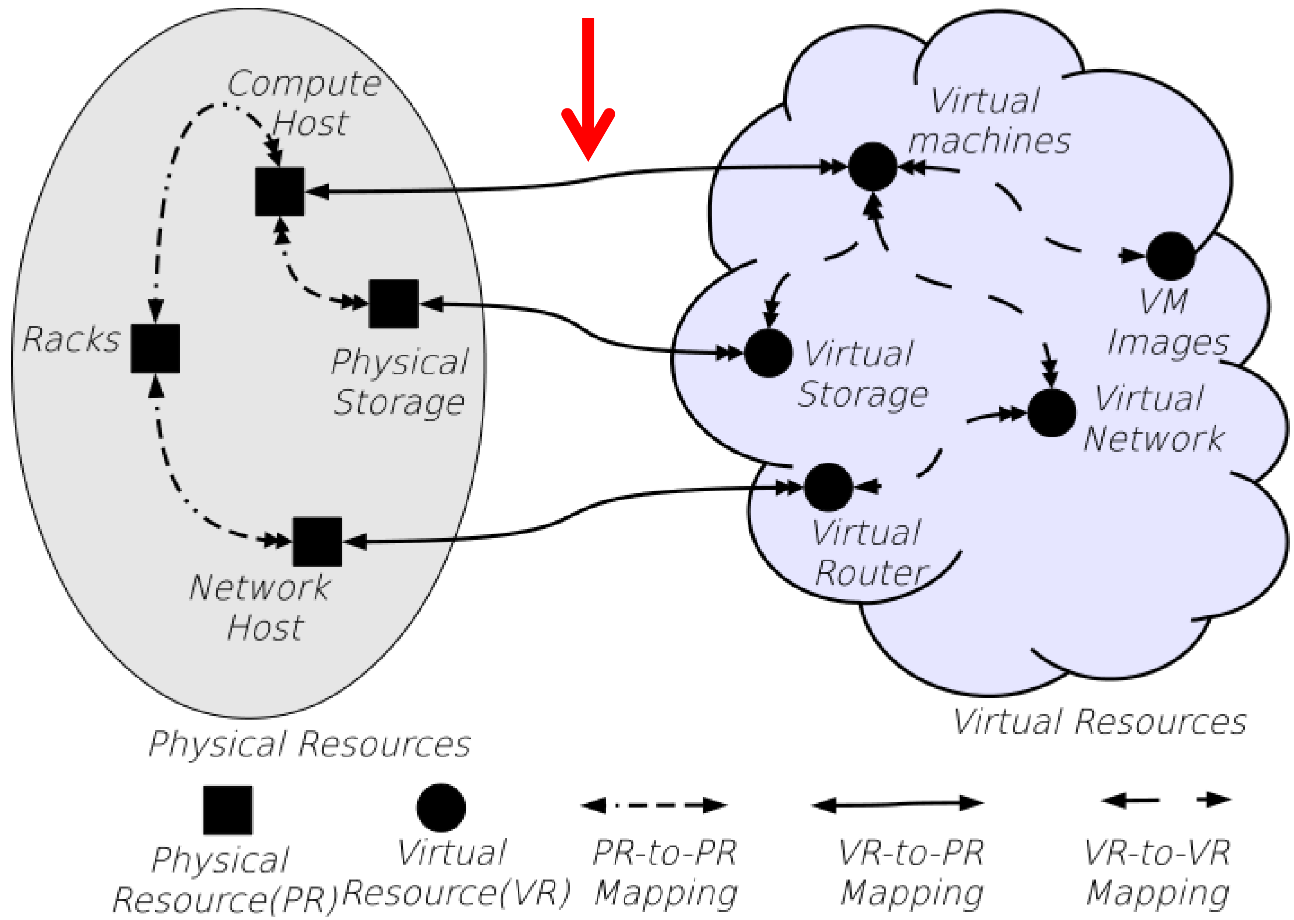
Vienna, Austria

June 1-3, 2015

Cloud Service Models

Software as a Service (SaaS)	*Network accessible software*
Platform as a Service (PaaS)	*App dev environment with cloud characteristics*
Infrastructure as a Service (IaaS)	*Virtualized hardware infrastructure*

IaaS Cloud: Virtual to Physical Mappings

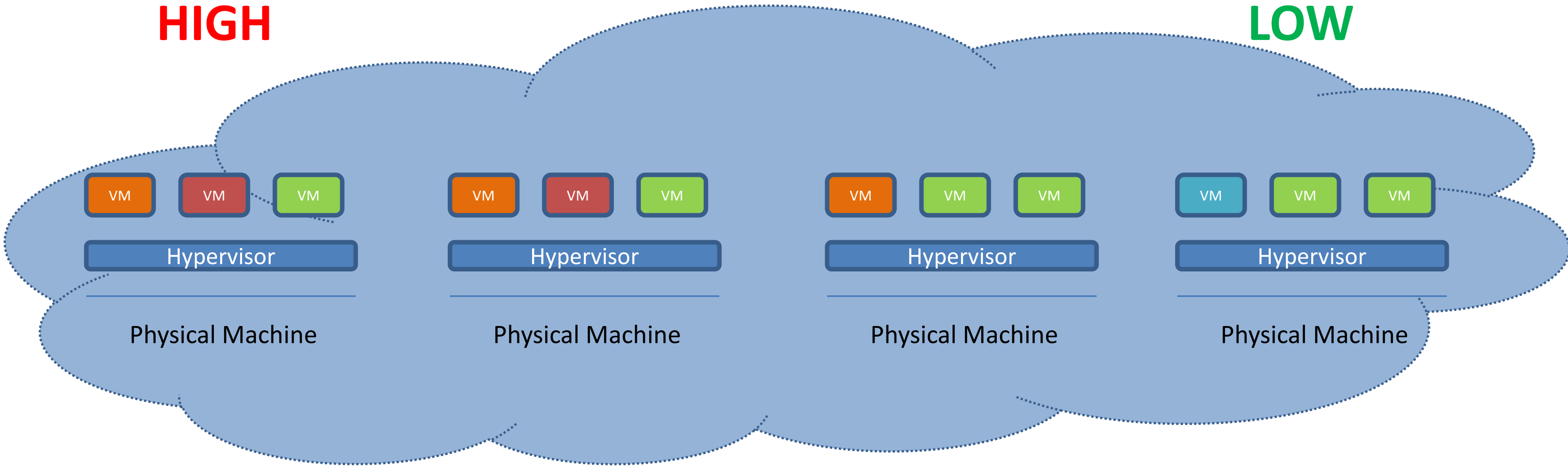


PUBLIC $ $$$ PRIVATE

CLOUD HIGH RISK LOW RISK CLOUD

Level of Physical Resource Sharing

HIGH LOW



Tenant 1: Need 3 VMs

Tenant 2: Need 3 VMs

Tenant 3: Need 2 VMs

Tenant 2: Need 3 VMs

Tenant 4: Need 1 VM

- Multi-tenancy is unavoidable in cloud platforms
  - Hypervisor provides isolation, albeit tricky
  - E.g. Ristenpart et al

UTSA Engineering

I·C·S The Institute for Cyber Security

Constraints-Driven Co-location

- Toward a programmable cloud platform for resource isolation that can satisfy constraints such as:
  - "Do not co-locate *sensitive* VMs with *low-sensitive*"
  - "Do not co-locate *high-availability* VMs in the same rack"
  - "Do not co-locate Exxon VMs with those of BP"
- <u>Must not co-locate</u> vs. must co-locate
  - Scheduling problems

Attribute-Based Conflict Specification for VM Co-location

- Name-value pairs on VMs
  - E.g. sensitivity(vm_1)="high", tenant(vm_2)="Acme"
  - Specified for VMs of each tenant
- Intra-tenant (tenant-specified)
  - Varies from tenant to tenant
  - E.g. "sensitivity", "group", etc.
- Inter-tenant (cloud service provider specified)
  - Available to VMs of all tenants
  - E.g. "tenant", "flavor", etc.

Sample Attributes for a Tenant

I: Attributes, Scope and Conflict-Set

$\mathbf{ATTR}_{\mathbf{VM}} = \{\textit{sensitivity}, \textit{tenant}\}$

$\mathbf{SCOPE}_{\textit{sensitivity}} = \{\text{high}, \text{low}\}$

$\mathbf{SCOPE}_{\textit{tenant}} = \{\text{tnt1}, \text{tnt2}, \text{tnt3}, \text{tnt4}, \text{tnt5}, \text{tnt6}\}$

$\mathbf{ConSet}_{\textit{sensitivity}} = \{ \{\text{high}, \text{low}\} \}$

$\mathbf{ConSet}_{\textit{tenant}} = \{ \{\text{tnt1}, \text{tnt2}\}, \{\text{tnt4}, \text{tnt6}\}, \{\text{tnt2}, \text{tnt3}\} \}$

UTSA Engineering

I·C·S
The Institute for Cyber Security

Conflict-Free Partitioning of Attributes

Step 1:

II: Conflict-Free Partitions of Scope of Each Attribute

$Partition_{sensitivity}$ = { {high}, {low} }

$\mathrm{Partition}_{tenant}$ = { {tnt1, tnt3, tnt6} , {tnt2, tnt4, tnt5} }

Finding MIN_PARTITION is similar to k-coloring: NP-Complete

Step 2:

III: Conflict-Free Segments of the Values of all Attributes

ConflictFreeATTR = {({tnt1, tnt3, tnt6 },{high}),
({tnt1, tnt3, tnt6 },{low}),
({tnt2, tnt4, tnt5 },{high}),
({tnt2, tnt4, tnt5 },{low})}

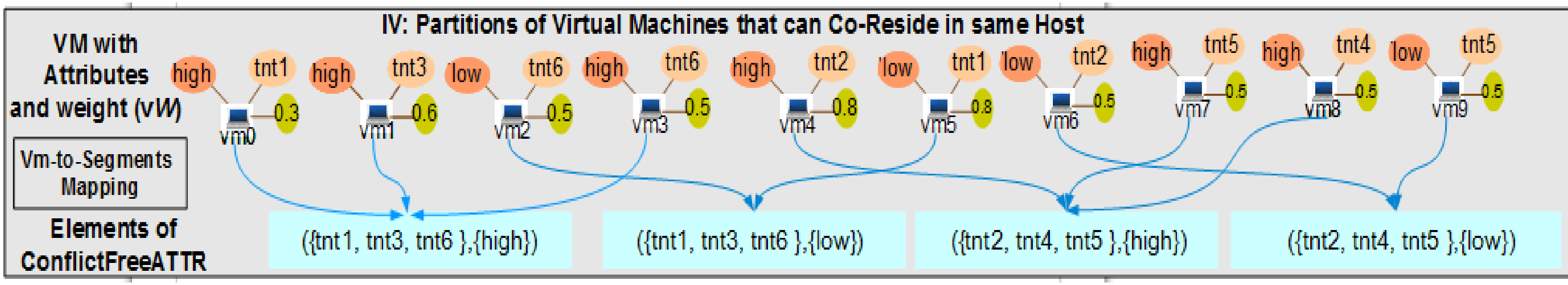
$O(|ATTR_{VM}| \times |PARTITION_{att}|)$

UTSA Engineering

I·C·S The Institute for Cyber Security

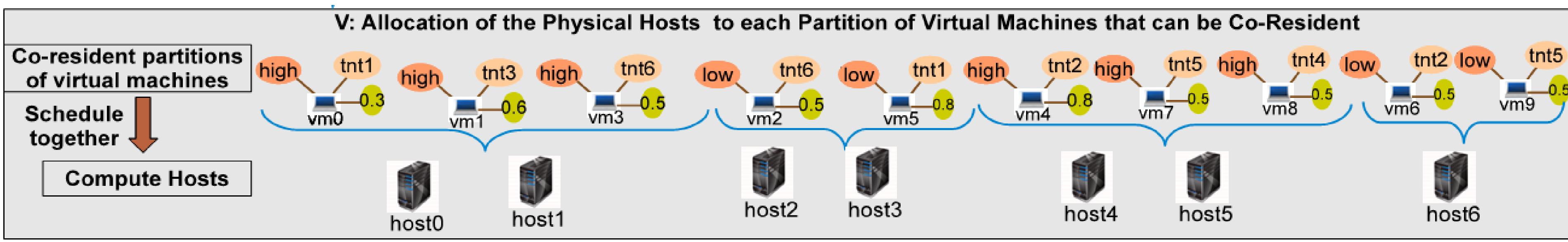
Co-Resident VM Scheduling

Step 3: Partitions of co-resident VMs



$O(|VM| \times |ConflictFreeATTR| \times |ATTR_{VM}|)$

Step 4: Scheduling of co-resident VMs into physical hosts



Similar to bin-packing: NP-Hard

Not a problem introduced by this work

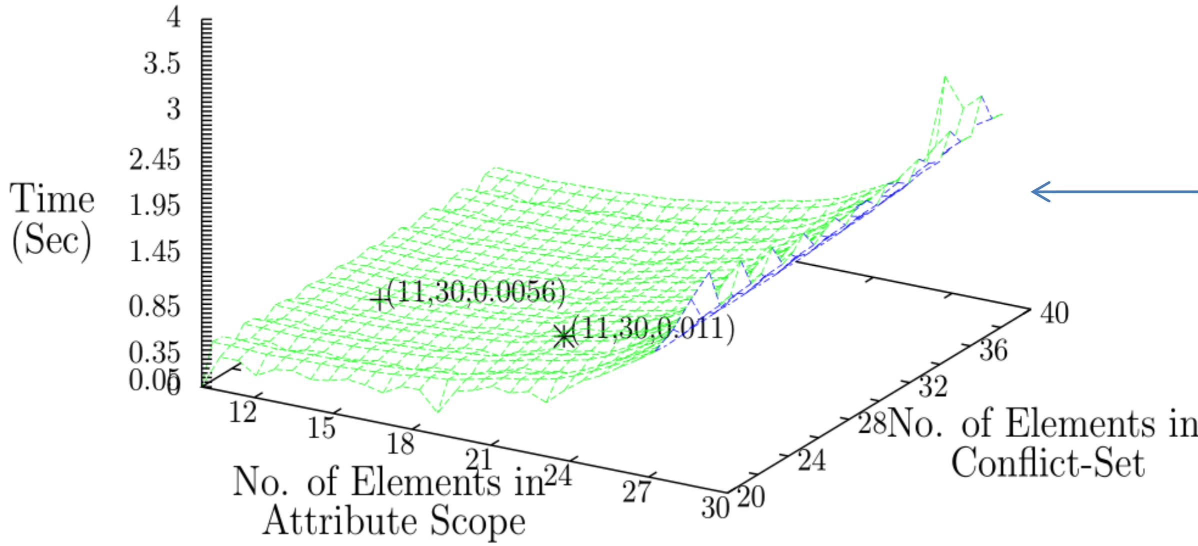
UTSA Engineering

I·C·S The Institute for Cyber Security

Experimental Setup

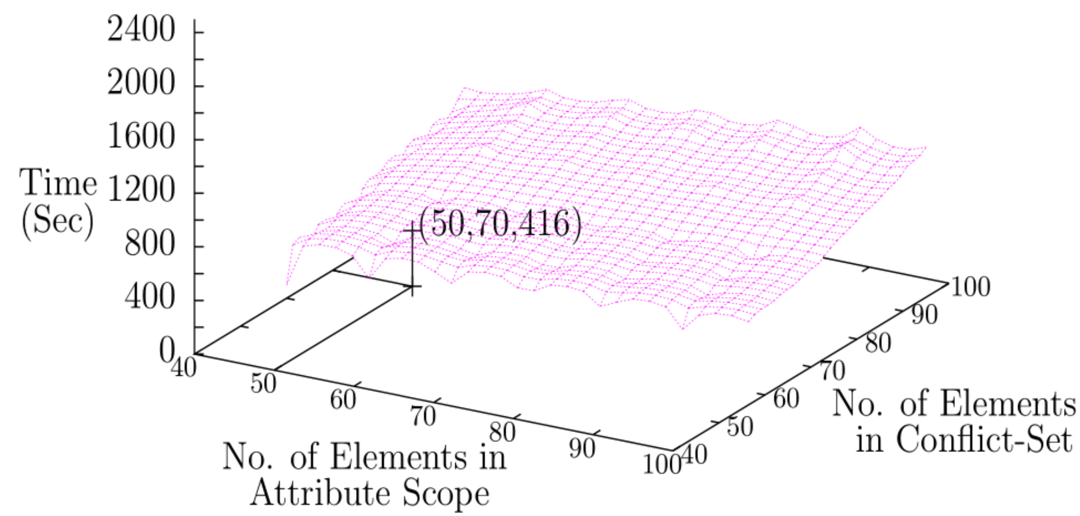
- OpenStack deployed on 5 physical machines
  - Each is a Dell R710 with 16 cores, 2.53 GHz and 98GB RAM
  - Each VM simulated as a physical host to simulate 100s of physical hosts

Conflict-Free Partition Using Backtracking



Small-ish scope and conflict set

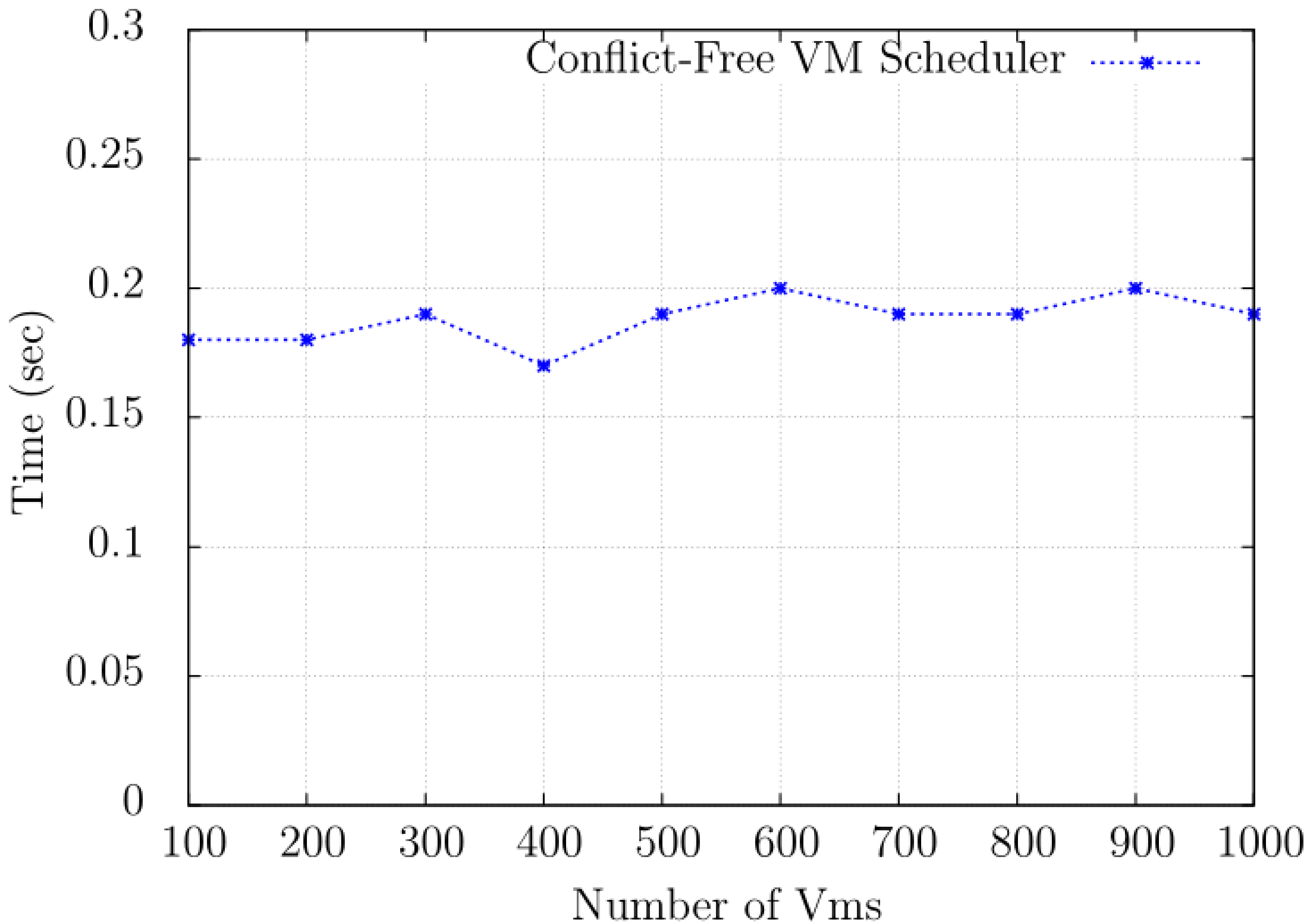
Large scope and conflict set



UTSA Engineering

I·C·S The Institute for Cyber Security

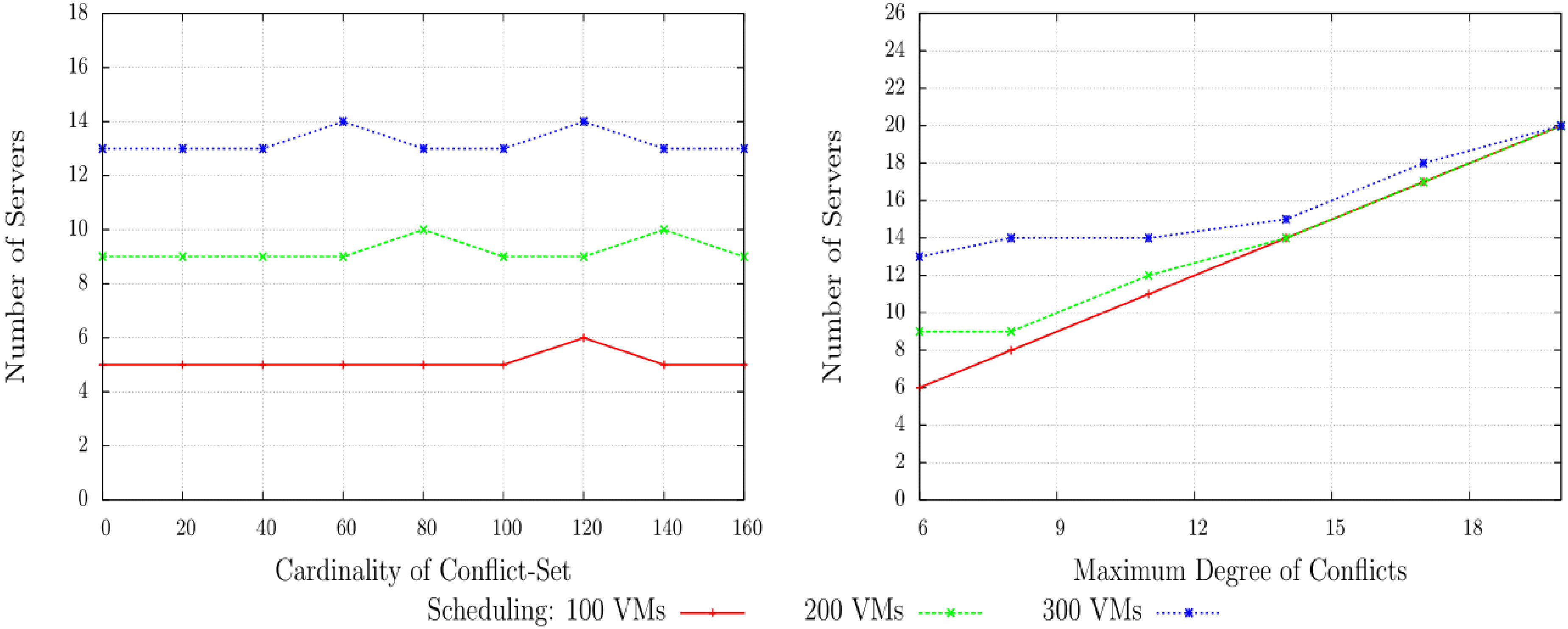
Scheduling Latency After Partitioning



UTSA Engineering

I·C·S
The Institute for Cyber Security

#Hosts



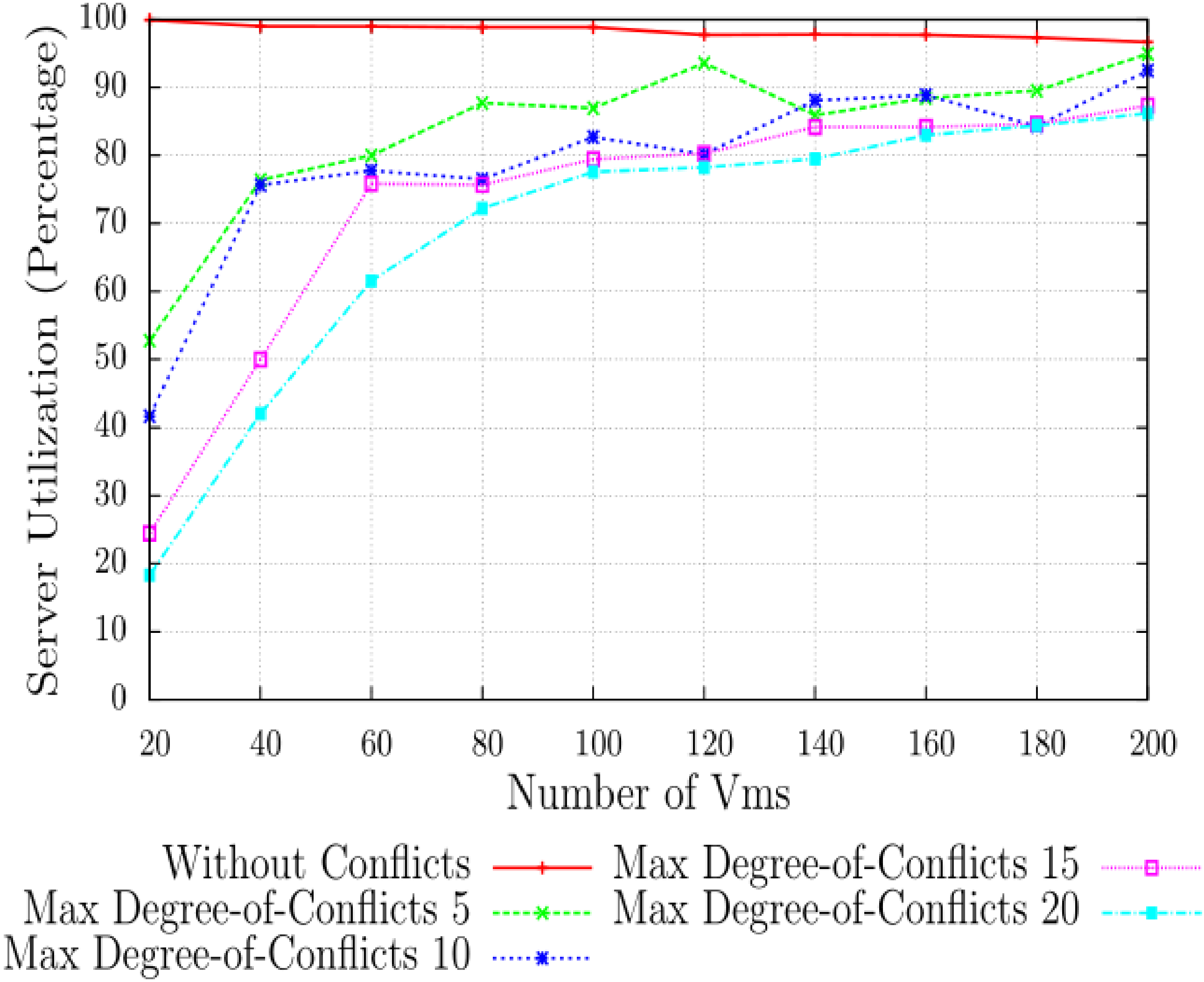
With varying number of elements in Conflict-Set

With varying number of maximum degree of conflicts

UTSA Engineering

I·C·S The Institute for Cyber Security

Host Utilization



UTSA Engineering

I·C·S
The Institute for Cyber Security

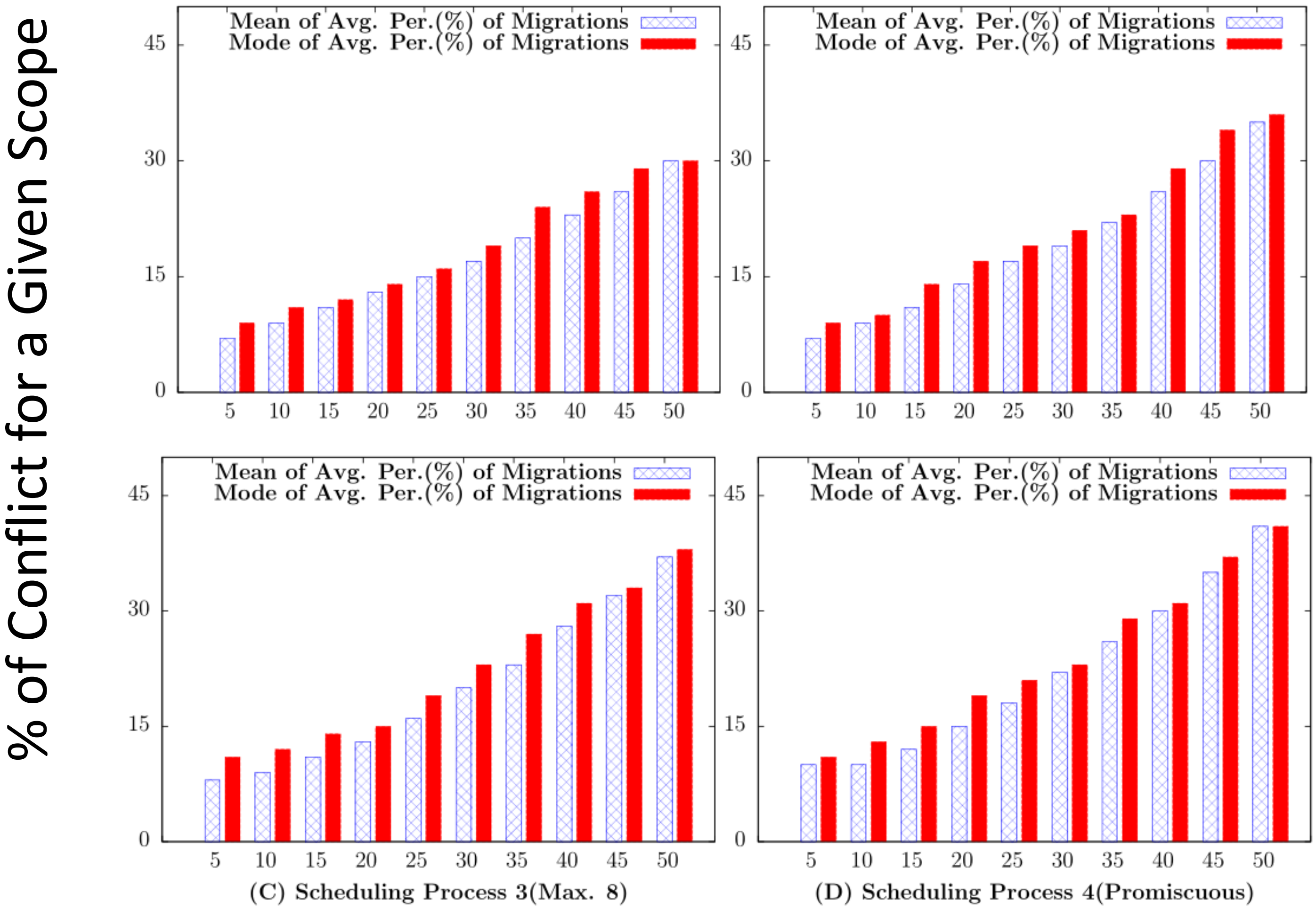
Conflict Changes

- Conflict specification can change over time!
- Changes can be of different types
  - Type 1: remove an element from the $ConSet_{att}$
  - Type 2: add an element to $ConSet_{att}$
    - $PARTITION_{att}$ remains unchanged
  - Type 3: add an element to ConSet*att*
    - $PARTITION_{att}$ changes -> may need to migrate

UTSA Engineering

I·C·S
The Institute for Cyber Security

Migrations



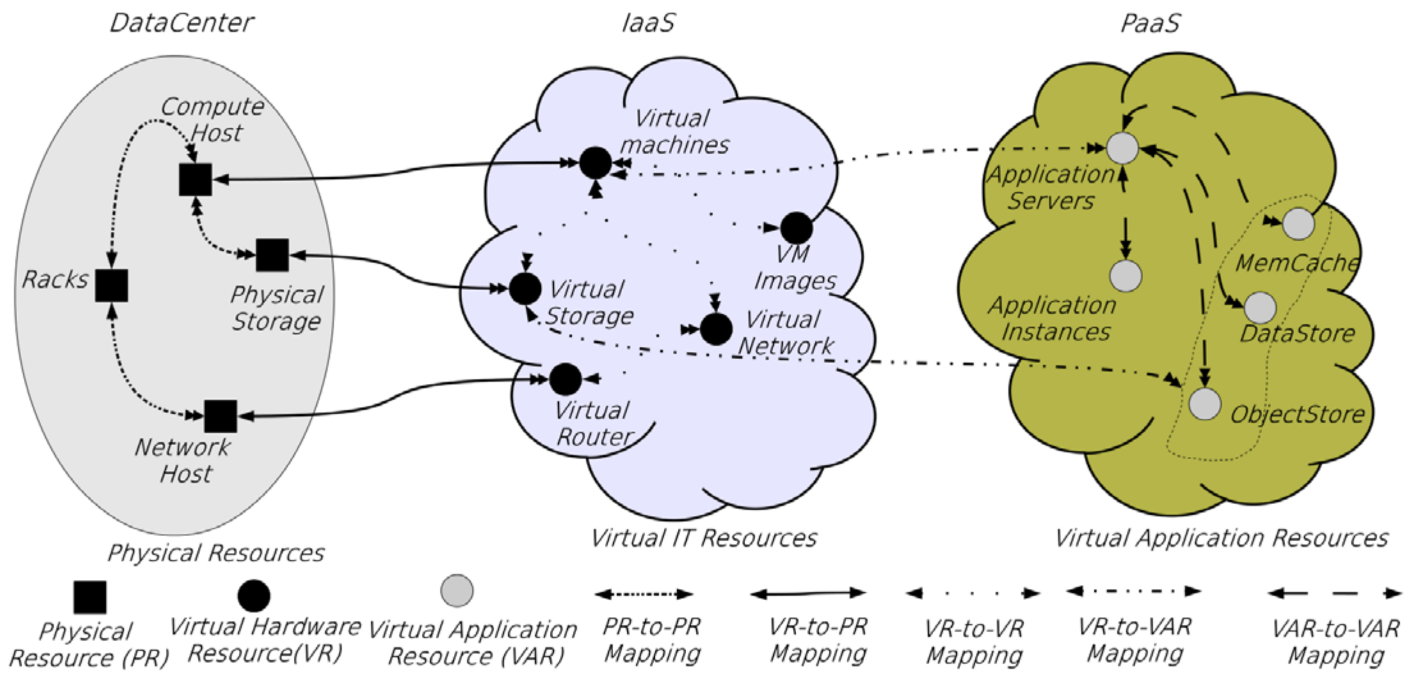
% of Total VMs that Require Migration

UTSA Engineering

I·C·S The Institute for Cyber Security

Ongoing/Future Directions

- Constraints that span further levels of abstractions
  - o PaaS and SaaS



UTSA Engineering

I·C·S The Institute for Cyber Security

Ongoing/Future Directions (continued)

- Constraints involving other virtual resources
  - Storage, Network, etc.
- Managing conflict changes over time
- Incremental conflict specification
- Attribute computation to inform conflict specification

Summary

- A conflict specification framework for resources in IaaS
  - Conflict-free partitioning is NP-Complete
- Prototyped and experimented in OpenStack

Thank you!

UTSA Engineering

I·C·S
The Institute for Cyber Security