Providing Automated Detection of Problems in Virtualized Servers using Monitor framework

Gunjan Khanna, Saurabh Bagchi
(Purdue University)
Kirk Beaty, Andrzej Kochut, Norman Bobroff and Gautam Kar
(IBM T. J. Watson Research Center)

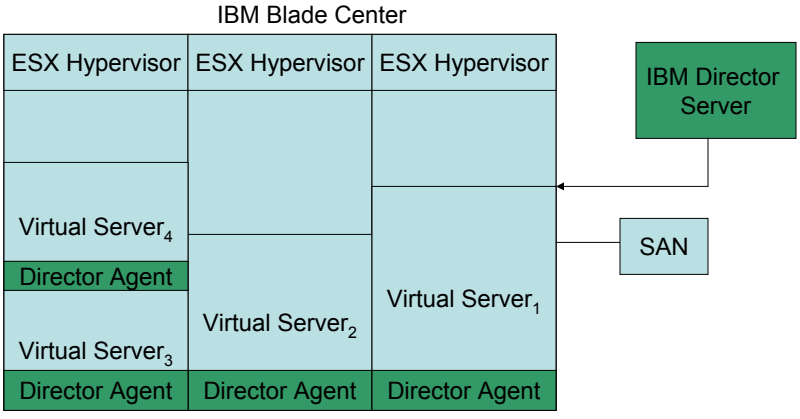
Background

- Detection and diagnosis of problems in IT systems is a challenging task
  - Performance, configuration etc.
- Server virtualization is a widely adopted practice
- Virtualization is a way of providing indirection
  - Server Virtualization aims at converting physical machine(s) into virtual server(s) and deploying on top of a hypervisor
- Provides isolation, homogeneity, and increased resource utilization
  - Prevents server sprawl

Motivation

- Virtualization Technology (hypervisor) and Management systems have been developed independently
  - Both exercise control over the OS
- Dynamic resource allocation via hypervisor further heightens the problem
- Current management systems only concentrate on system resources
  - Centralized and inflexible
  - Most require management agents to be co-located

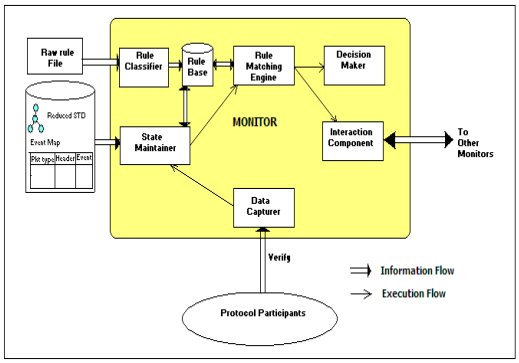
Example Virtual Scenario



Challenges in Addressing Problems in Virtualized Environments

- How to quickly detect the problem arising due to resource contention between co-located virtual machines ?
- How to interpret the system metrics from one virtual machine with the metrics from physical machine ?
  - System Metrics : CPU, Memory, I/O etc.
- How to correlate the application state information with the system metrics ?
- Address error propagation and scalability ?

Monitor(s) for Virtualized Environments



Characteristics

- Observes external interactions between system components
- Uses fast matching against a database of rules to detect problems
- Hierarchical topology allowing for correlation and filtering of information
- Can work with black-box components

Rule Framework

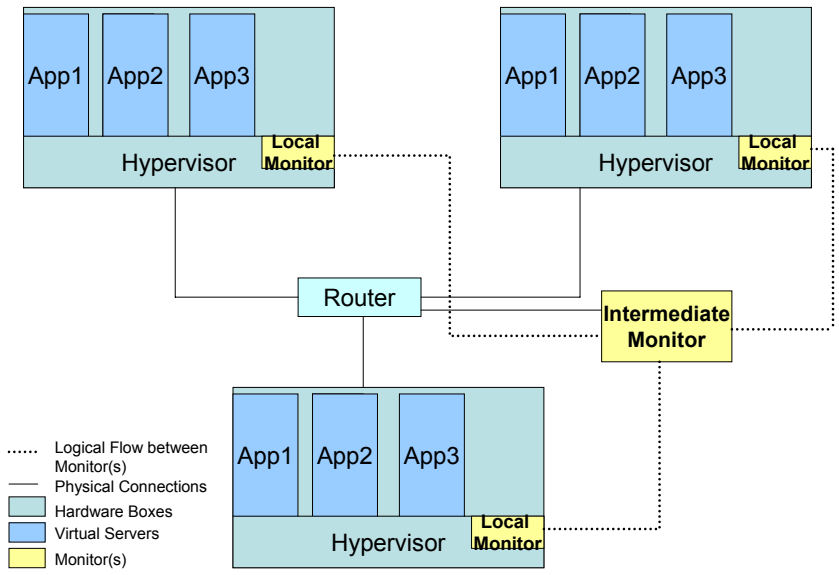
- Combinatorial Rules
  - $S_1 \wedge S_2$
- Temporal Rules
  - 5 different types of temporal rules
  - Expressive enough to form rules for wide range of applications and scenarios
  - Type I:

$$S_p = \text{true for } T \in (t_N, t_N + k) \Rightarrow S_q = \text{true for } T \in (t_I, t_I + b)$$

  - Type IV:

$$If\ S_i = true, and\ \forall t \in (t_0, t_0 + a)\ L \le |V_t| \le U \Rightarrow L' \le |B_q| \le U' \forall t \in (t_n, t_n + b)$$

Using Monitor(s) in Virtualized Server Scenario(s)



Detecting Problems

- Avoid simultaneous Peak Resource Usage for co-located virtual machines
  - S_i is the state of peak resource usage for machine $_i$.
  - $\neg(S_1 \wedge S_2 \wedge S_3)$
- Fault propagation from one physical machine to another
  - WAS sends more messages to a DB2 replica
  - Suitable assertions should be placed
- Correlation of alarms
  - Derived alarms at higher level Monitor(s)

$$\exists t = T \; s.t. \;\; S_i = true \Rightarrow \; A_L \leq A_{LM} \leq A_U \;\; \forall \, t \, \varepsilon \, (T, T + \alpha)$$

Relating the system resources with Application semantics

- Simply looking at system metrics like CPU, memory etc. does not provide a complete view of the system
  - Composite Rules relating the application with system metrics are necessary
- DB2 (Data Base) table contention
  - Response Time $\exists t = T \; s.t. \;\; S_j = true \Rightarrow \; R_{\max} \leq R_{DB} \leq \infty \;\; \forall \, t \, \varepsilon \, (T, T + \delta)$
  - Open Connections

$$\exists t = T \; s.t. \;\; S_i = true \Rightarrow \; L_0 \leq O_c \leq U_0 \;\; \forall \, t \, \varepsilon \, (T, T + \delta)$$

- WAS (Application Server) resource contention
  - C_w and WAS_{CPU} are application and system metric respectively

$$If \;\; C_0 \leq C_W \leq C_1 \;\; \forall \; t \; \varepsilon \; (T, T + \delta)$$
$$\Rightarrow \; L_{CPU} \leq WAS_{CPU} \leq U_{CPU} \;\; \forall \; t \; \varepsilon \; (T, T + \ell)$$

Conclusions

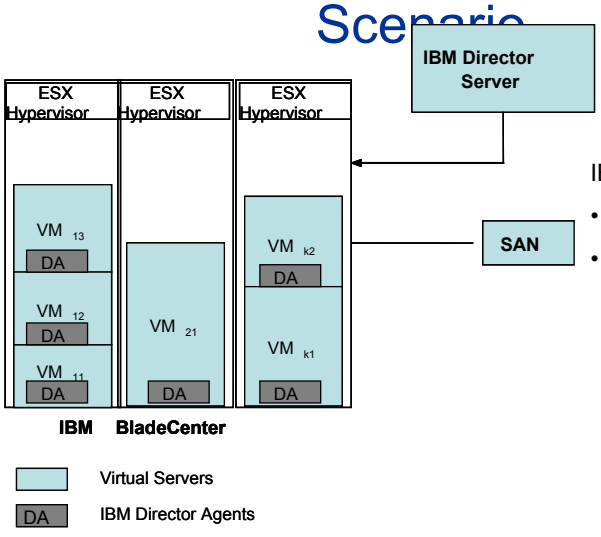
- We show the mismatch between the current management systems and virtualization layer(s)
- We propose a hierarchical management framework for addressing problems faced in a virtualized server system
- Address some of the challenges raised by virtualization
- As a future work we are working on diagnosis of problems in virtualized scenarios

Thank You !

Differences from existing Management framework(s)

Properties	Monitor Framework	Existing Management Framework(s)
Architecture	Hierarchical	Centralized
Adaptability	Re-configurable	Rigid
Internal Access	Non-Intrusive	Require Agents
Applicability	Generic in nature	Sensors for specific applications

Example Virtualized Server Scenario



IBM Director Server

- Management Framework
- Provides discovery of systems and remote management