Virtual Machine Monitoring MIB

draft-schoenw-opsawg-vm-mib-01.txt

Michael MacFaden, VMware
Jürgen Schönwälder, Jacobs Univeristy
Tina Tsou, Huawei (USA)
Cathy Zhou, Huawei

Changes since -00

- Specific state change notifications instead of a generic state change notification
- Addition of a 'migrating' state
- Added list of open issues...

vm-mib-02: scaling and caching

- We have to assume up to hundreds of VMs running on a single virtual machine.
- Solution #02-01: Add ..LastChange objects and ...LastStateChange objects to facilitate quick cache validation
- Solution #02-02: Make some tables (which ones?) time filtered

vm-mib-03: virtual CPU type identification

- Solution #03-01: IANA controlled enumeration that provides a CPU classification. The problem will be to provide rules about what constitutes a new CPU type and what not
- Solution #03-02: Use OIDs to identify CPU types. Such a distributed enumeration may not achieve a great deal of interoperability and is likely close to #03-03
- Solution #03-03: Use a string data type and rely on systems to put meaningful information there, perhaps provide guidelines how to structure the CPU type names, e.g. vendor-arch-model(-features)*
- Solution #03-04: Break the string of #03-03 into a set of MIB objects (...CpuVendor, ...CpuArch, ...CpuModel) plus a table of features
- Solution #03-05: Like #03-04 but keep features in a compact string notation

vm-mib-04: physical CPU type identification

- VM migration requires to match physical CPUs and more important also feature sets of physical CPUs.
- Solution #04-01: Extend the ENTITY-MIB with a new MIB module, say an ENTITY-CPU-MIB, providing an entPhyCPUTable, sparsely augmenting the entPhysicalTable for physical entities with entPhysicalClass = cpu. The entPhyCPUTable would contain information about CPU vendor, CPU architecture, CPU mode, CPU features, clock speeds, etc. (see also vm-mib-03).

vm-mib-05: per virtual cpu statistics

- It seems to be useful to provide statistics for each virtual CPU. However, it remains unclear what can be expected to be provided by a typical hypervisor implementation. There are a number of things to consider:
  - a) Reporting the time the virtual CPU has been running (CPU time consumed) seems relatively straight forward.
  - b) Reporting the current state of a virtual CPU requires to first define a suitable state model that is course grained enough to be useful (otherwise CPU state changes far too quickly to yield meaningful results). With a suitable state model, the MIB could provide the time spent in the various CPU states rather than or in addition to the current snapshot state.
  - c) Reporting the affinity mapping of virtual CPUs to physical CPUs. This, of course, requires to have a representation of physical CPUs.

vm-mib-01: storage sizes

- The MIB does not provide storage sizes, assuming this is provided by the hrStorageTable of the HOST-RESOURCES-MIB. However, some well known implementations of the HOST-RESOURCES-MIB only report about file systems used by the host system and not file systems residing in files used by virtual machines.
- Solution #01-01 Provide the storage block sizes as part of the VM-MIB. Provide a pointer to the hrStorageTable on systems that can provide this linkage but allow the pointer to be NULL.