A Quick Tour of the QEMU Monitor Protocol

Red Hat
Luiz Capitulino
August 2010
Outline

1. Brief Introduction

2. Key Design Decisions

3. Issues and Challenges
Section 1

Brief Introduction
The QEMU Monitor Protocol (QMP)

- A protocol for applications to talk with QEMU
- Features:
  - Lightweight, text-based, easy to parse syntax (JSON)
  - Asynchronous messages support (ie. events)
  - Capabilities Negotiation
- Main developers: Luiz Capitulino and Markus Armbruster (with help from others, of course)
Status

- Merged in 0.13
- Functional (has issues, though)
- Almost forty commands and eleven events
- Libvirt and kvm-autotest support it
- The current interface is **NOT** stable yet
Section 2

Key Design Decisions
The data format: JSON

- JavaScript Object Notation (RFC 4627)
- Language-independent, lightweight, easy to read, easy to parse
- Data types:
  - Primitives: strings, numbers, booleans, and null
  - Structured:
    - arrays: ["love", 10, true, null]
    - objects: {"french": "C'est la vie"}
QMP example: ejecting a medium (success)

-> {
   "execute": "eject",
   "arguments": {
     "device": "ide1-cd0"
   }
}

<- {
   "return": {}
}
QMP example: ejecting a medium (failure)

-> {
    "execute": "eject",
    "arguments": {
        "device": "foobar"
    }
}

<- {
    "error": {
        "class": "DeviceNotFound",
        "desc": "Device 'foobar' not found",
        "data": {
            "device": "foobar"
        }
    }
}
QMP example: asynchronous message

<- {
  "event": "BLOCK_IO_ERROR",
  "data": {
    "device": "ide0-hd1",
    "operation": "write",
    "action": "stop"
  },
  "timestamp": {
    "seconds": 1265044230,
    "microseconds": 450486
  }
}
The old monitor
The old monitor

User formatted data
monitor_printf()

Human Monitor

Monitor commands (string)

Function call

Monitor Handler
The new monitor: introducing objects
Error reporting

1. qerror_report() call

qerror_report(QERRDEVICE_INITFAILED, "e1000");

2. Error macro

#define QERRDEVICE_INITFAILED
"{ 'class': 'DeviceInitFailed', 'data': { 'device': '%s' } }
"

3. Error table entry

{
  .error_fmt = QERRDEVICE_INITFAILED,
  .desc = "Device '%(device)' could not be initialized",
}
Error reporting

1. qerror_report() call

qerror_report(QERR_DEVICE_INIT_FAILED, "e1000");

2. Error macro

#define QERR_DEVICE_INIT_FAILED "{
    'class': 'DeviceInitFailed',
    'data': {
        'device': %s
    }
}"  

3. Error table entry

{"error_fmt": QERR_DEVICE_INIT_FAILED,  
"desc": "Device '%(device)' could not be initialized"}
Section 3
Issues and Challenges
Main issues

- Bad stuff from the human monitor leaked into QMP
- Existing code dictated the interface
- Code is ugly and needs cleanup
- Tried to do way too much at once
- Error reporting and asynchronous handlers
Error reporting: summary

- Zillions of errors
- One error message per error
- The error object is global in the monitor
- The appropriate solution is still open to debate
Error reporting: summary

- Zillions of errors
- One error message per error
- The error object is global in the monitor
- The appropriate solution is still open to debate
Asynchronous commands

- Some commands are slow or depend on the guest response (e.g. migrate, savevm, device_del, balloon, etc)
- Possible solution: just delay the response object
- Two possible protocol changes:
  - Mark specific asynchronous handlers as so
  - Add a new keyword, eg. "execute_async", and commands must obey both "execute" and "execute_async"
- Error reporting has to be fixed first
Asynchronous commands

- Some commands are slow or depend on the guest response (eg. migrate, savevm, device_del, balloon, etc)
- Possible solution: just delay the response object
- Two possible protocol changes:
  - Mark specific asynchronous handlers as so
  - Add a new keyword, eg. ”execute_async”, and commands must obey both ”execute” and ”execute_async”
- Error reporting has to be fixed first
Challenges

- Define a realistic set of goals
- Improved development process
- Specification review
- We need to ship something useful and stable in 0.14
Challenges

- Define a realistic set of goals
- Improved development process
- Specification review
- We need to ship something useful and stable in 0.14
Thanks for listening!

Luiz Capitulino <lcapitulino@redhat.com>
http://www.linux-kvm.org/page/MonitorProtocol