Making KVM autotest useful for KVM developers

Lucas Meneghel Rodrigues
`lmr@redhat.com`

KVM Forum 2011
August 16th, 2011

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]

[1] `http://en.wikipedia.org/wiki/Usability`

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]
- It's a non objective measure of the following attributes:

[1] http://en.wikipedia.org/wiki/Usability

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]
- It's a non objective measure of the following attributes:
  - Learnability: How easy is it for users to accomplish basic tasks?

[1] `http://en.wikipedia.org/wiki/Usability`

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]
- It's a non objective measure of the following attributes:
    - Learnability: How easy is it for users to accomplish basic tasks?
    - Efficiency: How quickly a user can perform tasks?

[1] http://en.wikipedia.org/wiki/Usability

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]
- It's a non objective measure of the following attributes:
    - Learnability: How easy is it for users to accomplish basic tasks?
    - Efficiency: How quickly a user can perform tasks?
    - Memorability: How easily a user can re-establish proficiency?

[1] `http://en.wikipedia.org/wiki/Usability`

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]
- It's a non objective measure of the following attributes:
    - Learnability: How easy is it for users to accomplish basic tasks?
    - Efficiency: How quickly a user can perform tasks?
    - Memorability: How easily a user can re-establish proficiency?
    - Errors: How many errors do users make? How severe are these errors? How easily can they recover from the errors?

[1] `http://en.wikipedia.org/wiki/Usability`

Usability

- In layman terms, it is the perceived ease of use of a human made design[1]
- It's a non objective measure of the following attributes:
  - Learnability: How easy is it for users to accomplish basic tasks?
  - Efficiency: How quickly a user can perform tasks?
  - Memorability: How easily a user can re-establish proficiency?
  - Errors: How many errors do users make? How severe are these errors? How easily can they recover from the errors?
  - Satisfaction: How pleasant is it to use the design?

[1] http://en.wikipedia.org/wiki/Usability

What KVM autotest offers

- Knowledge of guest OS install (lots of them)

What KVM autotest offers

- Knowledge of guest OS install (lots of them)
- Automation of cumbersome procedures (ex win virtio drivers)

What KVM autotest offers

- Knowledge of guest OS install (lots of them)
- Automation of cumbersome procedures (ex win virtio drivers)
- Reproduction of hard to hit issues

What KVM autotest offers

- Knowledge of guest OS install (lots of them)
- Automation of cumbersome procedures (ex win virtio drivers)
- Reproduction of hard to hit issues
- Automatic collection of commands executed, serial console

What KVM autotest offers

- Knowledge of guest OS install (lots of them)
- Automation of cumbersome procedures (ex win virtio drivers)
- Reproduction of hard to hit issues
- Automatic collection of commands executed, serial console
- Catch of guest kernel crashes (linux platform)

What KVM autotest offers

- Knowledge of guest OS install (lots of them)
- Automation of cumbersome procedures (ex win virtio drivers)
- Reproduction of hard to hit issues
- Automatic collection of commands executed, serial console
- Catch of guest kernel crashes (linux platform)
- Test grid might even automate host install

What KVM autotest offers

- Knowledge of guest OS install (lots of them)
- Automation of cumbersome procedures (ex win virtio drivers)
- Reproduction of hard to hit issues
- Automatic collection of commands executed, serial console
- Catch of guest kernel crashes (linux platform)
- Test grid might even automate host install
- Good right? But it's worth nothing if people can't use it

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase

KVM R&D

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix

KVM R&D

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix
- Pull source code and build, or install packages and run tests

KVM R&D

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix
- Pull source code and build, or install packages and run tests
- Is dedicated to write, build and manage this infra

KVM R&D

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix
- Pull source code and build, or install packages and run tests
- Is dedicated to write, build and manage this infra

KVM R&D

- Hacking on kernel, qemu-kvm, new features, bug fixing

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix
- Pull source code and build, or install packages and run tests
- Is dedicated to write, build and manage this infra

KVM R&D

- Hacking on kernel, qemu-kvm, new features, bug fixing
- Wants to know if his changes did break functionality

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix
- Pull source code and build, or install packages and run tests
- Is dedicated to write, build and manage this infra

KVM R&D

- Hacking on kernel, qemu-kvm, new features, bug fixing
- Wants to know if his changes did break functionality
- Has enough work chasing bugs on the tool he's contributing to

Clash of use cases - Personas

KVM QA

- Needs to provide automated and regular testing of codebase
- Without human intervention, large test matrix
- Pull source code and build, or install packages and run tests
- Is dedicated to write, build and manage this infra

KVM R&D

- Hacking on kernel, qemu-kvm, new features, bug fixing
- Wants to know if his changes did break functionality
- Has enough work chasing bugs on the tool he's contributing to
- On a perfect world, a single command line would be the goal

Everybody hates the cartesian config

- Test config format to define a matrix of tests to be executed

Everybody hates the cartesian config

- Test config format to define a matrix of tests to be executed
- It has its merits, makes sense for the QA case

Everybody hates the cartesian config

- Test config format to define a matrix of tests to be executed
- It has its merits, makes sense for the QA case
- However, it's showing up to be cumbersome for R&D case

Everybody hates the cartesian config

- Test config format to define a matrix of tests to be executed
- It has its merits, makes sense for the QA case
- However, it's showing up to be cumbersome for R&D case
- We will re-think the R&D use case, proposing 'a back to autotest basics' approach, that we want to discuss on some BoF session

Initiatives to improve KVM autotest usability

- Sane defaults

Initiatives to improve KVM autotest usability

- Sane defaults
- Minimizing points of failure

Initiatives to improve KVM autotest usability

- Sane defaults
- Minimizing points of failure
- Improve reporting

Initiatives to improve KVM autotest usability

- Sane defaults
- Minimizing points of failure
- Improve reporting
- Rigorous review and automated checks to keep code good quality

Initiatives to improve KVM autotest usability

- Sane defaults
- Minimizing points of failure
- Improve reporting
- Rigorous review and automated checks to keep code good quality
- Documentation. Tips of the week

Initiatives to improve KVM autotest usability

- Sane defaults
- Minimizing points of failure
- Improve reporting
- Rigorous review and automated checks to keep code good quality
- Documentation. Tips of the week
- **Conversation.** Lots of it

Sane defaults

- Use pre-installed qemu-kvm

Sane defaults

- Use pre-installed qemu-kvm
- Install a single, widely available guest OS

Sane defaults

- Use pre-installed qemu-kvm
- Install a single, widely available guest OS
- Boot it and shut it down

Sane defaults

- Use pre-installed qemu-kvm
- Install a single, widely available guest OS
- Boot it and shut it down
- We can improve the sample config comments

Minimizing points of failure

- Graceful degradation of optional functionality

Minimizing points of failure

- Graceful degradation of optional functionality
    - `kvm_stat` profiler

Minimizing points of failure

- Graceful degradation of optional functionality
  - `kvm_stat` profiler
  - VM screenshot conversion

Minimizing points of failure

- Graceful degradation of optional functionality
  - `kvm_stat` profiler
  - VM screenshot conversion
- Minimizing obligatory dependencies

Improve reporting

- Do not depend on web interface for reports

Improve reporting

- Do not depend on web interface for reports
  - Autotest upstream now has html reports per job

Improve reporting

- Do not depend on web interface for reports
  - Autotest upstream now has html reports per job
  - Uses some fancy javascript to spice up things

Documentation

- Tips of the week series - How tos

Documentation

- Tips of the week series - How tos
  - Week 1 - Installing virtio drivers

Documentation

- Tips of the week series - How tos
  - Week 1 - Installing virtio drivers
  - Week 2 - Running qemu-kvm unittests

Documentation

- Tips of the week series - How tos
  - Week 1 - Installing virtio drivers
  - Week 2 - Running qemu-kvm unittests
  - Week 3 - Running tests with existing guest

Documentation

- Tips of the week series - How tos
  - Week 1 - Installing virtio drivers
  - Week 2 - Running qemu-kvm unittests
  - Week 3 - Running tests with existing guest
  - Week 4 - Writing your own tests

Documentation

- Tips of the week series - How tos
  - Week 1 - Installing virtio drivers
  - Week 2 - Running qemu-kvm unittests
  - Week 3 - Running tests with existing guest
  - Week 4 - Writing your own tests
  - Week 5 - You name it!

Further work

- In the end, our goal would be to have

Further work

- In the end, our goal would be to have
- `make autotest` target

Further work

- In the end, our goal would be to have
- `make autotest` target
- Would probably call a text menu helping user to configure some few questions, or assume some defaults. But that needs more thinking.

Further work

- In the end, our goal would be to have
- `make autotest` target
- Would probably call a text menu helping user to configure some few questions, or assume some defaults. But that needs more thinking.
- Sane way to entirely bypass the cartesian config system, for the single developer use case.

How to contribute

- `http://autotest.kernel.org/wiki/KVMAutotest`

How to contribute

- `http://autotest.kernel.org/wiki/KVMAutotest`
- Go through the docs, let us know what your opinions are

How to contribute

- `http://autotest.kernel.org/wiki/KVMAutotest`
- Go through the docs, let us know what your opinions are
- Git clone the autotest repo, play with it

How to contribute

- http://autotest.kernel.org/wiki/KVMAutotest
- Go through the docs, let us know what your opinions are
- Git clone the autotest repo, play with it
- The autotest team is waiting to implement your suggestions

Contact

- cleber@redhat.com
- lmr@redhat.com
- Autotest mailing list (autotest@test.kernel.org)
- KVM mailing list (kvm@vger.kernel.org)