A Robot Framework Test Suite for OpenAFS

Michael Meffie, Sine Nomine Associates

June 21, 2019

ANSIBLE

ROBOT
FRAME
WORK/

Tests suites

- OpenAFS unit tests
  - c-based **unit tests**
  - Test Anything Protocol (TAP) style tests
  - component/function unit tests
  - run with `make check`
- Various c-based “test” programs in `/src`
  - for ad-hoc component tests
  - see `xstat_fs_test`
- OpenAFS `functionality-tests`
  - a mix of c, perl, and shell scripts
  - imported Alra project and CMU AFSTools in 2002

OpenAFS Robotest

- An effort to create a new system level test suite
- Based on the popular Robot Framework
- Functional tests (not performance)
- Scale
  - test a trivial one-host cell
  - test many clients and servers
- Test cells can be spun up with Ansible
  - See `ansible-openafs` roles

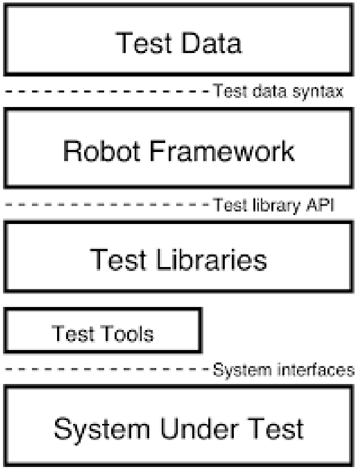
Robot Framework

- `https://robotframework.org`
- General purpose acceptance test automation framework
- Tester oriented
- Well documented and active community
- Originally developed at Nokia
- Open-source and active since 2008
- First Robot Framework conference held in 2019

Robot Framework

- Tester oriented
- Declarative **Keyword** testing methodology
- Supports various testing styles
  - Classic style
  - Data-driven
  - Behavior-driven development (BDD)
- Test report generation (HTML and XML)
- Data driven, declarative test data
- Test execution by hierarchy and tags
- Distributed testing supported via RPC

Robot Framework



Hello World Robot Framework Test

```
*** Test Cases ***
Valid Login
    Open Browser To Login Page
    Input Username    ${USERNAME}
    Input Password    ${PASSWORD}
    Submit Credentials
    Welcome Page Should Be Open
    [Teardown]    Close Browser
```

BDD Style

```
*** Test Cases ***
Valid Login
    Given Login Page is open
    When user "demo" logs in with password "secret"
    Then welcome page should be open

*** Keywords ***
Browser is opened to login page
    Open Browser To Login Page
    ...
User "${username}" logs in with password "${password}"
    Input Username    ${USERNAME}
    ...
```

Test Data

- Tests are defined in **tables**
- Various formats supported
  - plain text `.robot` (most commonly used)
  - TSV format (spreadsheet)
  - reStructedText with embedded plain text
  - early versions supported HTML format
- Robot Framework IDE: `RIDE`

Test Data

- PASS or FAIL, there is no *skip*
- Tags used to categorize tests
- Tests must be *independent*
- Test data are *not* scripts
- Logic and loops belong in the **test libraries**

Standard libraries

- BuiltIn
- Collections
- DateTime
- Dialogs
- OperatingSystem
- Process
- Remote
- Screenshot
- String
- XML

External Libraries

- Android
- AnywhereLibrary (Web testing)
- AppiumLibrary (Adriod/iOS app testing)
- Archive
- AutoItLibrary (Windows GUI testing)
- CncLibrary (CNC milling machines)
- Database
- Debug
- Diff
- Django
- Eclipse
- Faker (generate test data)

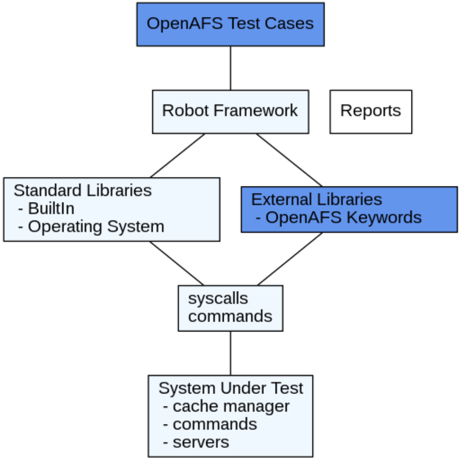
External Libraries . . .

- FTP
- HTTP
- iOS
- ImageHorizon
- JavaFXLibrary
- MongoDB
- Mainframe3270
- MQTT
- NcclientLibrary
- Rammbock (Generic network protocol testing)
- RemoteSwingLibrary
- RESTinstance

External Libraries . . .

- SapGuiLibrary
- Selenium2Screenshots
- Selenium (Browser testing)
- SikuliLibrary (GUI testing)
- SSHLibrary
- SudsLibrary (SOAP web services testing)
- SwingLibrary (Swing GUI)
- TestFX (Java TestFX)
- TFTP
- WhiteLibrary (Windows GUI testing)
- Watir (web testing)

Robot Framework



OpenAFS Library

- Robot Framework keywords for OpenAFS testing
- Pure python module
- Supports Python 2 and Python 3
- BSD License
- `https://github.com/openafs-contrib/robotframework-open`
- Installable with `pip`

```
pip install robotframework-openafslibrary
```

OpenAFS Library Keywords

Types of keywords:

- ACL
- Cache
- Command
- Dump
- Login
- PAG
- Path
- Volume
- Other

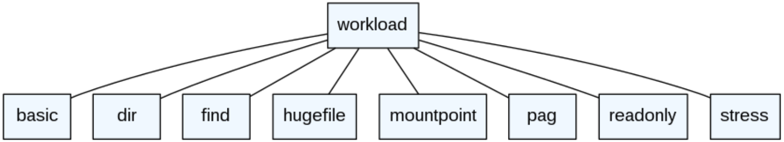
OpenAFS Library Volume Keywords

- `Create Volume` - Create RW and RO, and mount
- `Remove Volume` - Remove RW and Clones, and unmount
- `Release Volume`
- `Volume Should Exist`
- `Volume Should Be Locked`
- `Volume Should Be Unlocked`
- `Get Volume Id`

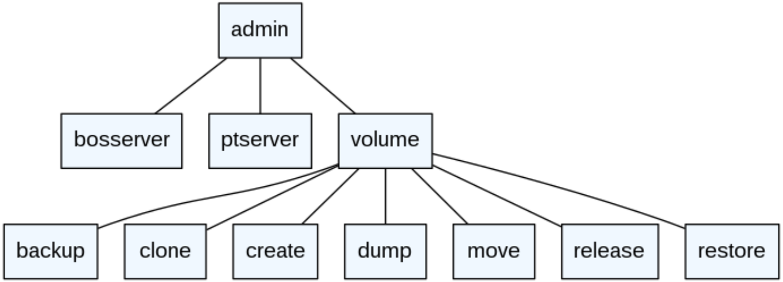
OpenAFS Test Data

- Functional tests for OpenAFS clients and servers
- Requires the OpenAFS Library
- BSD License
- Github: openafs-contrib/openafs-robotest
- `.robot` style test data format (text)

OpenAFS Workload Test Suites



OpenAFS Admin Test Suites



Example Client Test Case

```
*** Test Cases ***
Create a Hard Link within a Directory
    [Setup]  Create File    ${FILE}
    Link Count Should Be    ${FILE}  1
    Link                    ${FILE}  ${LINK}
    Inode Should Be Equal   ${LINK}  ${FILE}
    Link Count Should Be    ${FILE}  2
    Link Count Should Be    ${LINK}  2
    Unlink                  ${LINK}
    Should Not Exist        ${LINK}
    Link Count Should Be    ${FILE}  1
    [Teardown]  Remove File  ${FILE}
```

Example Server Test Case

```
*** Test Cases ***
Release a Volume
    [Setup]     Create Volume  ${v} ${h}  ${p}
    Command Should Succeed    vos addsite ${h} ${p} ${v}
    Command Should Succeed    vos release ${v}
    Volume Should Exist       ${v}.readonly
    Volume Location Matches   ${v}  ${h}  ${p}  ro
    [Teardown]  Remove Volume  ${v}
```

Deployment

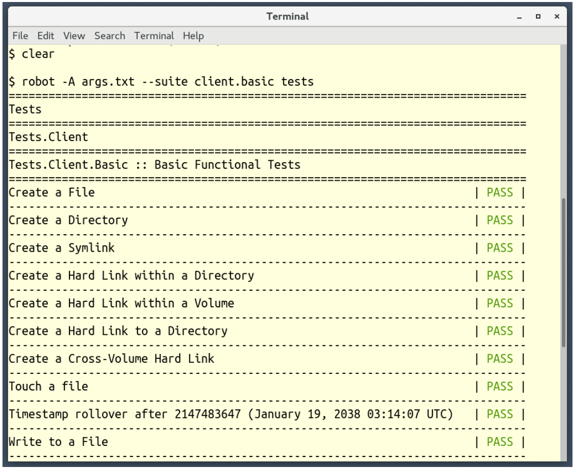
1. Use an existing cell, or create a new cell with Ansible
2. Setup the test control machine on a client
    - Install robotframework and openafslibrary (`pip`)
    - Install the test data (`git clone`)
    - Create RF variable and options files
3. Run tests with RF test runner `robot`

Installation with Ansible

Ansible role `openafs_robotest`:

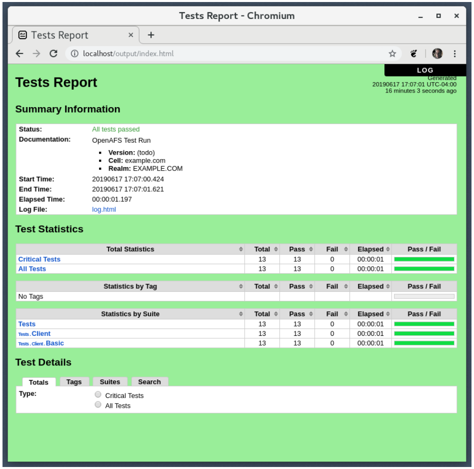
- Installs Robot Framework and the OpenAFS Library with `pip`
- Installs the OpenAFS Robotest test cases with `git`
- Creates RF variable files to match the test cell (paths, cell name, etc)

Running the tests


```
$ clear

$ robot -A args.txt --suite client.basic tests
==============================================================================
Tests
==============================================================================
Tests.Client
==============================================================================
Tests.Client.Basic :: Basic Functional Tests
==============================================================================
Create a File                                                         | PASS |
------------------------------------------------------------------------------
Create a Directory                                                    | PASS |
------------------------------------------------------------------------------
Create a Symlink                                                      | PASS |
------------------------------------------------------------------------------
Create a Hard Link within a Directory                                 | PASS |
------------------------------------------------------------------------------
Create a Hard Link within a Volume                                    | PASS |
------------------------------------------------------------------------------
Create a Hard Link to a Directory                                     | PASS |
------------------------------------------------------------------------------
Create a Cross-Volume Hard Link                                       | PASS |
------------------------------------------------------------------------------
Touch a file                                                          | PASS |
------------------------------------------------------------------------------
Timestamp rollover after 2147483647 (January 19, 2038 03:14:07 UTC)   | PASS |
------------------------------------------------------------------------------
Write to a File                                                       | PASS |
------------------------------------------------------------------------------
```

Output report



Generated
20190617 17:07:01 UTC-04:00
16 minutes 3 seconds ago

Tests Report

Summary Information

Status:	All tests passed
Documentation:	OpenAFS Test Run ▪ **Version:** (todo) ▪ **Cell:** example.com ▪ **Realm:** EXAMPLE.COM
Start Time:	20190617 17:07:00.424
End Time:	20190617 17:07:01.621
Elapsed Time:	00:00:01.197
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Elapsed	Pass / Fail
Critical Tests	13	13	0	00:00:01	🟩
All Tests	13	13	0	00:00:01	🟩

Statistics by Tag	Total	Pass	Fail	Elapsed	Pass / Fail
No Tags					

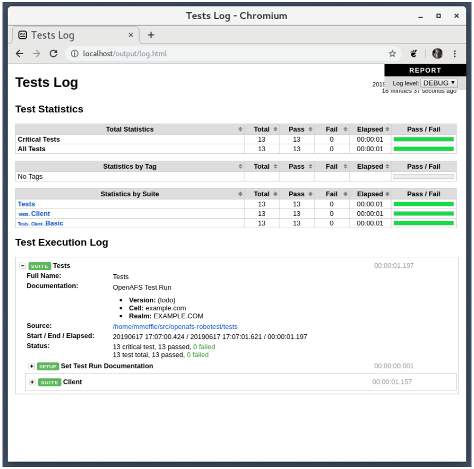
Statistics by Suite	Total	Pass	Fail	Elapsed	Pass / Fail
Tests	13	13	0	00:00:01	🟩
Tests . Client	13	13	0	00:00:01	🟩
Tests . Client . Basic	13	13	0	00:00:01	🟩

Test Details

Totals | Tags | Suites | Search

Type: ○ Critical Tests ○ All Tests

Output log



Tests Log

2019
16 minutes 37 seconds ago

Log level: DEBUG

Test Statistics

Total Statistics	Total	Pass	Fail	Elapsed	Pass / Fail
Critical Tests	13	13	0	00:00:01	🟩
All Tests	13	13	0	00:00:01	🟩

Statistics by Tag	Total	Pass	Fail	Elapsed	Pass / Fail
No Tags					

Statistics by Suite	Total	Pass	Fail	Elapsed	Pass / Fail
Tests	13	13	0	00:00:01	🟩
Tests . Client	13	13	0	00:00:01	🟩
Tests . Client . Basic	13	13	0	00:00:01	🟩

Test Execution Log

SUITE **Tests** 00:00:01.197

Full Name: Tests

Documentation: OpenAFS Test Run

- **Version:** (todo)
- **Cell:** example.com
- **Realm:** EXAMPLE.COM

Source: /home/mmeffie/src/openafs-robotest/tests

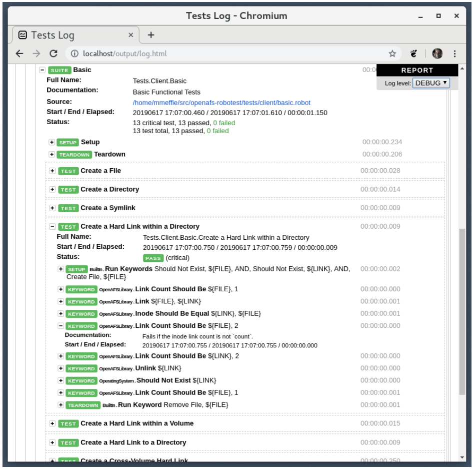
Start / End / Elapsed: 20190617 17:07:00.424 / 20190617 17:07:01.621 / 00:00:01.197

Status: 13 critical test, 13 passed, 0 failed
13 test total, 13 passed, 0 failed

SETUP **Set Test Run Documentation** 00:00:00.001

SUITE **Client** 00:00:01.157

Log details



```
Log level: DEBUG

− SUITE Basic                                                          00:0
  Full Name:            Tests.Client.Basic
  Documentation:        Basic Functional Tests
  Source:               /home/mmeffie/src/openafs-robotest/tests/client/basic.robot
  Start / End / Elapsed: 20190617 17:07:00.460 / 20190617 17:07:01.610 / 00:00:01.150
  Status:               13 critical test, 13 passed, 0 failed
                        13 test total, 13 passed, 0 failed
  + SETUP Setup                                                        00:00:00.234
  + TEARDOWN Teardown                                                  00:00:00.206
  + TEST Create a File                                                 00:00:00.028
  + TEST Create a Directory                                            00:00:00.014
  + TEST Create a Symlink                                              00:00:00.009
  − TEST Create a Hard Link within a Directory                         00:00:00.009
    Full Name:            Tests.Client.Basic.Create a Hard Link within a Directory
    Start / End / Elapsed: 20190617 17:07:00.750 / 20190617 17:07:00.759 / 00:00:00.009
    Status:               PASS (critical)
    + SETUP BuiltIn.Run Keywords Should Not Exist, ${FILE}, AND, Should Not Exist, ${LINK}, AND,   00:00:00.002
      Create File, ${FILE}
    + KEYWORD OpenAFSLibrary.Link Count Should Be ${FILE}, 1           00:00:00.000
    + KEYWORD OpenAFSLibrary.Link ${FILE}, ${LINK}                     00:00:00.001
    + KEYWORD OpenAFSLibrary.Inode Should Be Equal ${LINK}, ${FILE}    00:00:00.001
    − KEYWORD OpenAFSLibrary.Link Count Should Be ${FILE}, 2           00:00:00.000
      Documentation:        Fails if the inode link count is not `count`.
      Start / End / Elapsed: 20190617 17:07:00.755 / 20190617 17:07:00.755 / 00:00:00.000
    + KEYWORD OpenAFSLibrary.Link Count Should Be ${LINK}, 2           00:00:00.000
    + KEYWORD OpenAFSLibrary.Unlink ${LINK}                            00:00:00.000
    + KEYWORD OperatingSystem.Should Not Exist ${LINK}                 00:00:00.000
    + KEYWORD OpenAFSLibrary.Link Count Should Be ${FILE}, 1           00:00:00.001
    + TEARDOWN BuiltIn.Run Keyword Remove File, ${FILE}                00:00:00.001
  + TEST Create a Hard Link within a Volume                            00:00:00.015
  + TEST Create a Hard Link to a Directory                             00:00:00.009
  + TEST Create a Cross-Volume Hard Link                               00:00:00.250
```

Future

- **More** tests
- **Better** tests
- Convert `.robot` files to the new *pipe* (|) style format (?)
- Nightly multi-host cell test on the buildbot.openafs.org
  - currently: a single host test on each linux-rc build
- More and better OpenAFS Library keywords (as needed)
- Distributed testing with Robot Framework RPC

Thank you

Questions?