DISTRIBUTED FILESYSTEMS

Ori File System

Janeczek, Bergler 5AHITT

Table of Contents

Task Description

Installation

"Ori is a distributed file system built for offline operation and empowers the user with control over synchronization operations and conflict resolution. We provide history through light weight snapshots and allow users to verify the history has not been tampered with. Through the use of replication instances can be resilient and recover damaged data from other nodes." [1]

Installieren Sie Ori und testen Sie die oben beschriebenen Eckpunkte dieses verteilten Dateisystems (DFS). Verwenden Sie dabei auf jeden Fall alle Funktionalitäten der API von Ori um die Einsatzmöglichkeiten auszuschöpfen. Halten Sie sich dabei zuallererst an die Beispiele aus dem Paper im Kapitel 2 [3]. Zeigen Sie mögliche Einsatzgebiete für Backups und Roadwarriors (z.B. Laptopbenutzer möchte Daten mit zwei oder mehreren Servern synchronisieren). Führen Sie auch die mitgelieferten Tests aus und kontrollieren Sie deren Ausgaben (Hilfestellung durch Wiki [2]).

Comparison

Wo gibt es Überschneidungen zu anderen Implementierungen von DFS? Listen Sie diese auf und dokumentieren Sie mögliche Entscheidungsgrundlagen für mindestens zwei unterschiedliche Einsatzgebiete. Verwenden Sie dabei zumindest HDFS [4] und GlusterFS [5] als Gegenspieler zu Ori. Weitere Implementierungen sind möglich aber nicht verpflichtend. Um aussagekräftige Vergleiche anstellen zu können, wäre es von Vorteil die anderen Systeme ebenfalls - zumindest oberflächlich - zu testen.

Info

Gruppengröße: 2 Mitglieder
Gesamtpunkte: 16

Installation und Testdurchlauf von Ori: 2 Punkte

Einsatz/Dokumentation der Ori API (replicate, snapshot, checkout, graft, filelog, list, log, merge, newfs, pull, remote, removefs, show, status, tip, varlink): 8 Punkte

Gegenüberstellungstabelle: 4 Punkte

Einsatz der Gegenspieler: 2 Punkte

References

[1] Ori File System, Stanford Website, online: http://ori.scs.stanford.edu/, visited: 2015-03-02
[2] Ori File System, Bitbucket Wiki, online: https://bitbucket.org/orifs/ori/wiki/Home, visited: 2015-03-02
[3] Ali José Mashtizadeh, Andrea Bittau, Yifeng Frang Huang, David Mazières. Replication, History, and Grafting in the Ori File System. In Proceedings of the 24th Symposium on Operating Systems Principles, November 2013. Paper.
[4] Apache Hadoop FileSystem, http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html, visited: 2015-03-02
[5] GlusterFS, http://www.gluster.org/documentation/howto/HowTo/, visited: 2015-03-02

Requirement Analysis

- **Installation of the DFS known as ORI**
  Installing the Distributed File System and evaluating its functionalities such as:
  - Synchronization operations
  - Conflict resolution
  - Snapshots
  - Replication instances

- **Backups and Roadwarriors**
  Usage of backups and roadwarrior concerning the DFS

- **Comparison to other Distributed File Systems**
  Comparing ORI's functionalities with other DFSes, e.g.: HDFS, GlusterFS

- **Testing as discussed in the Wiki**
  Testing ORI's functionalities

Design

In this document a distributed file system is evaluated. Therefore no design consideration.

Technology Description

"Ori is a distributed file system built for offline operation and empowers the user with control over synchronization operations and conflict resolution. We provide history through light weight snapshots and allow users to verify the history has not been tampered with. Through the use of replication instances can be resilient and recover damaged data from other nodes." [1]

Peer-to-Peer

Ori operates peer-to-peer among your devices and uses existing secure communication channels such as SSH to transfer your data.

Work Offline

In today's world we often are moving around with intermittent network connectivity and we want to access our data when we board a plane or travel to the office.

Secure

Ori can verify the authenticity of your data and ensure it has not been tampered with. Data is transfered over SSH. Device discovery and automatic synchronization uses a shared secret to initiate transfers.

Instant Access

Instantly mount remote file systems and start working while you synchronize data in the background.

Effort Estimation

Janeczek:

DATE	PHASE	TASK	ESTIMATION	ACTUAL	COMMENT
03/18/2015	Installation	The Installation of the ORI FS	0:30:00	0:20:00	
03/19/2015	Documentation	Writing the documentation for DEZSYS07	1:00:00	2:00:00	
03/18/2015	Implementation	Creating the Filesystem + Sync	2:00:00	1:30:00	
03/19/2015	Comparison	OriFS vs HDFS vs GlusterFS	2:00:00	1:00:00	
		SUM	5:30:00	4:50:00	

Bergler:

DATE	PHASE	TASK	ESTIMATION	ACTUAL	COMMENT
03/18/2015	Installation	The Installation of the ORI FS	0:30:00	0:25:00	
03/19/2015	Documentation	Writing parts of the documentation	1:00:00	0:40:00	
03.19.2015	Tests	Running the Tests	1:00:00	0:40:00	
03/19/2015	Comparison	OriFS vs HDFS vs GlusterFS	2:00:00	1:00:00	
		SUM	2:30:00	1:45:00	

Task Execution

Installation

Ori Distributed File System is available for several OSes, for example FreeBSD, Homebrew, Archlinux, Ubuntu, etc.

In the „Getting Ori“ section of their homepage, the installation on an Ubuntu OS is described as follows:

```
apt-get install scons build-essential pkg-config

apt-get install libboost-dev uuid-dev libfuse-dev libevent-dev libssl-dev

apt-get install libedit-dev

git clone https://bitbucket.org/orifs/ori-orisyncng.git

cd ori-orisyncng

scons

vim SConstruct

//change PREFIX to /usr/local/ instead of /usr/local/bin

sudo scons install
```

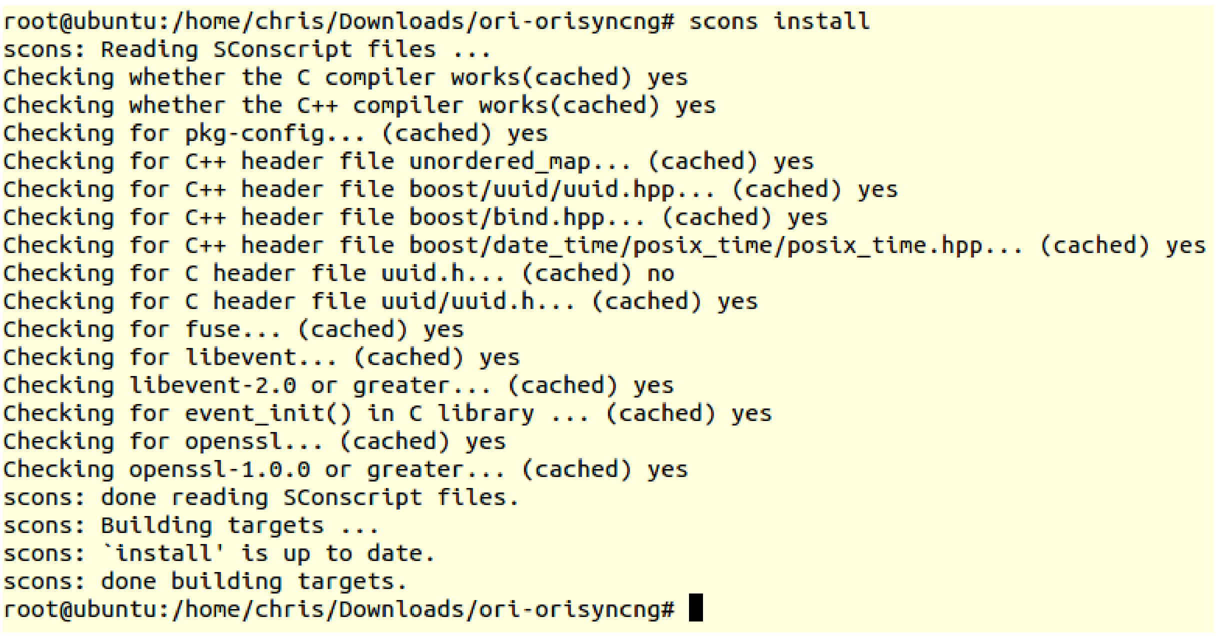
The first command installs scons as well as several packages needed for the ori file system.

The next step, is to clone the ori-orisyncng repository from bitbucket using git.

Before we are able to install ori using scons, we have to make some changes to the SConstruct file.

After the above change, the scons command can be executed to install ori.

If everything went as expected you should get the following message:



```
root@ubuntu:/home/chris/Downloads/ori-orisyncng# scons install
scons: Reading SConscript files ...
Checking whether the C compiler works(cached) yes
Checking whether the C++ compiler works(cached) yes
Checking for pkg-config... (cached) yes
Checking for C++ header file unordered_map... (cached) yes
Checking for C++ header file boost/uuid/uuid.hpp... (cached) yes
Checking for C++ header file boost/bind.hpp... (cached) yes
Checking for C++ header file boost/date_time/posix_time/posix_time.hpp... (cached) yes
Checking for C header file uuid.h... (cached) no
Checking for C header file uuid/uuid.h... (cached) yes
Checking for fuse... (cached) yes
Checking for libevent... (cached) yes
Checking libevent-2.0 or greater... (cached) yes
Checking for event_init() in C library ... (cached) yes
Checking for openssl... (cached) yes
Checking openssl-1.0.0 or greater... (cached) yes
scons: done reading SConscript files.
scons: Building targets ...
scons: `install' is up to date.
scons: done building targets.
root@ubuntu:/home/chris/Downloads/ori-orisyncng# 
```

Creating a new File System

For creating a new File System the following commands are required:

Initialize the ori file service:

```
orisync init
```

Create a new file system:

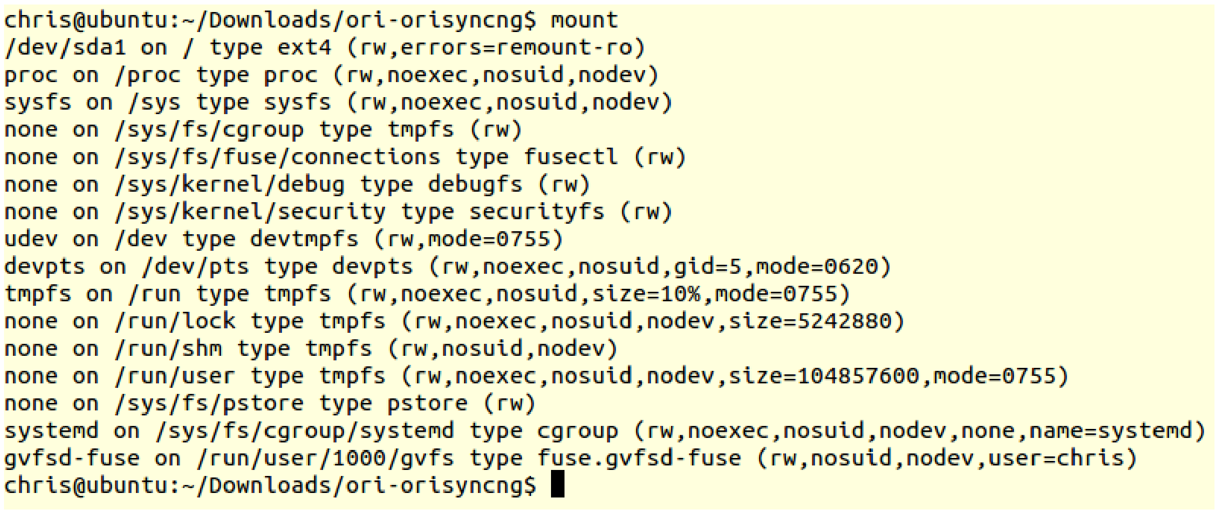
```
ori newfs CBRepo

orisync add /home/chris/.ori/CBRepo.ori
```

List the existing file systems:

```
orisync

orisync list
```

The created file system has got to be mounted:

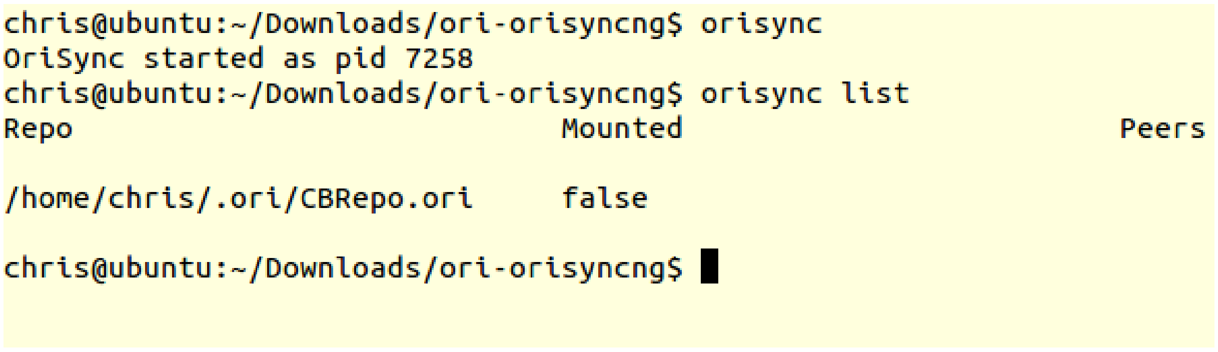
```
mkdir ori/CBRepo

orifs /home/chris/ori/CBRepo

mount
```



```
chris@ubuntu:~/Downloads/ori-orisyncng$ mount
/dev/sda1 on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
none on /sys/fs/cgroup type tmpfs (rw)
none on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
udev on /dev type devtmpfs (rw,mode=0755)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
tmpfs on /run type tmpfs (rw,noexec,nosuid,size=10%,mode=0755)
none on /run/lock type tmpfs (rw,noexec,nosuid,nodev,size=5242880)
none on /run/shm type tmpfs (rw,nosuid,nodev)
none on /run/user type tmpfs (rw,noexec,nosuid,nodev,size=104857600,mode=0755)
none on /sys/fs/pstore type pstore (rw)
systemd on /sys/fs/cgroup/systemd type cgroup (rw,noexec,nosuid,nodev,none,name=systemd)
gvfsd-fuse on /run/user/1000/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,user=chris)
chris@ubuntu:~/Downloads/ori-orisyncng$
```

Check if the file system is mounted by calling the following command again:

```
orisync list
```


```
chris@ubuntu:~/Downloads/ori-orisyncng$ orisync
OriSync started as pid 7258
chris@ubuntu:~/Downloads/ori-orisyncng$ orisync list
Repo                            Mounted                         Peers

/home/chris/.ori/CBRepo.ori     false

chris@ubuntu:~/Downloads/ori-orisyncng$ 
```

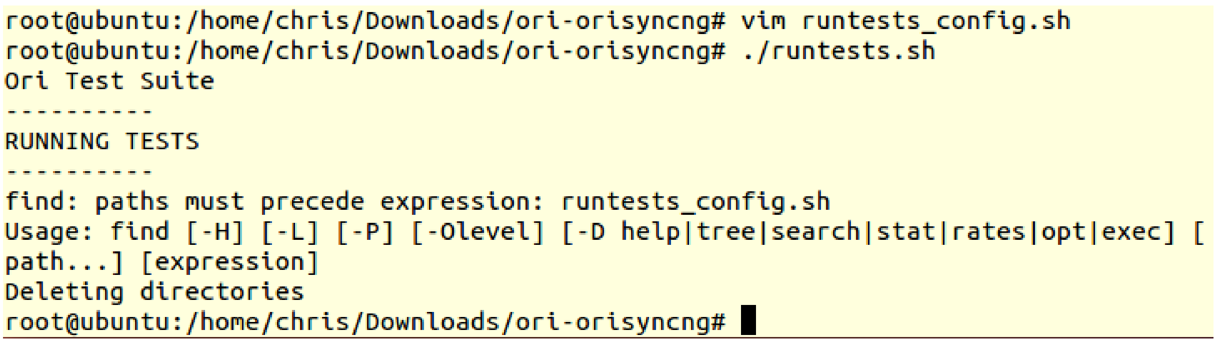
Sadly, the file system is not mounted and I have not found a solution for this yet.

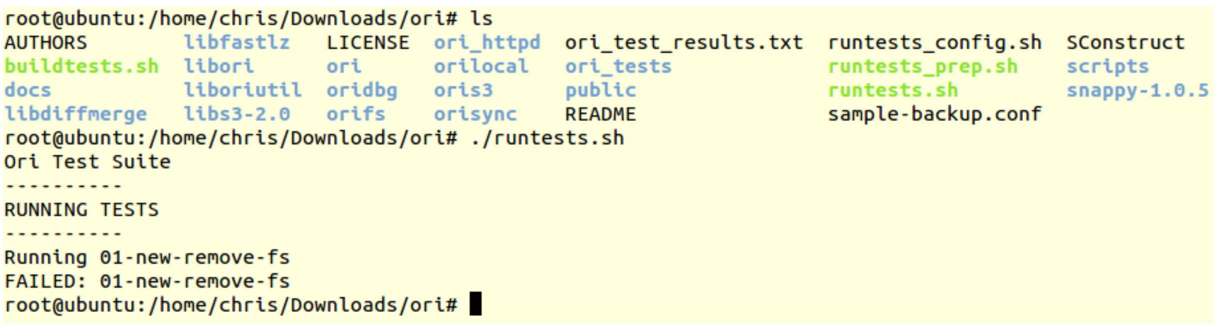
Running the Tests

To run the test suite you will need to configure an SSH public key to access your local machine without a password. You will also want to save the following into `runtests_config.sh`.

```
# Required for Mac OS X and FreeBSD only (comment out on Linux machines)
export UMOUNT="umount"

# Not updated to new CLI
HTTP_CLONE="no"
HTTP_PULL="no"
MERGE="no"
MOUNT_WRITE="no"
MOUNT_WRITE_PYTHON_MT="no"
```

Once configured you can run `runtests.sh`. On an error you may have to cleanup the tempdir and test repositories on your system before rerunning. The logs will be available inside the tempdir if an error occurred.


```
root@ubuntu:/home/chris/Downloads/ori-orisyncng# vim runtests_config.sh
root@ubuntu:/home/chris/Downloads/ori-orisyncng# ./runtests.sh
Ori Test Suite
----------
RUNNING TESTS
----------
find: paths must precede expression: runtests_config.sh
Usage: find [-H] [-L] [-P] [-Olevel] [-D help|tree|search|stat|rates|opt|exec] [
path...] [expression]
Deleting directories
root@ubuntu:/home/chris/Downloads/ori-orisyncng# 
```


```
root@ubuntu:/home/chris/Downloads/ori# ls
AUTHORS         libfastlz    LICENSE  ori_httpd  ori_test_results.txt  runtests_config.sh  SConstruct
buildtests.sh   libori       ori      orilocal   ori_tests             runtests_prep.sh    scripts
docs            liboriutil   oridbg   oris3      public                runtests.sh         snappy-1.0.5
libdiffmerge    libs3-2.0    orifs    orisync    README                sample-backup.conf
root@ubuntu:/home/chris/Downloads/ori# ./runtests.sh
Ori Test Suite
----------
RUNNING TESTS
----------
Running 01-new-remove-fs
FAILED: 01-new-remove-fs
root@ubuntu:/home/chris/Downloads/ori# 
```

Comparison to other DFS

Apache Hadoop FileSystem

The Features of Apache's HDFS[4]:

- A list of features that Apache's DFS has to offer:
- Single Node Setup, Cluster Setup
- Web Interface (NameNode and DataNode each run an internal web server)
- Shell Commands (e.g.: DFSAdmin Command), Shell Commands to directly interact with HDFS
- Secondary NameNode (NameNode stores modifications to the file system as a logfile)
- Checkpoint Node (persistance of the namespace established)
- Backup Node (provides backup nodes to restore files)
- Import Checkpoint (if files are lost, they can be restored by importing checkpoints)
- Balancer (Placement of Data across multiple DataNodes)
- Rack Awareness (Hadoop Clusters are arranged in racks and network traffic between different notes)
- Safemode (File system state is loaded from the fsimage and the edits log file)
- Fsck (command used to check for various inconsistencies)
- Fetchdt (fetch Delegation Token and store it in a file on the local system)
- Recovery Mode (interactive prompts in the command line to recover your data)
- Upgrade and Rollback
- File Permissions and Security
- Scalability
- Big community

GlusterFS

The Features of the GlusterFS[14]:

- **Block Device Translator**

  Block Device translator

- **client_t**

  Refactor server_connection_t in xlator/protocol/server to a new client_t in libglusterfs.

- **Disperse**

  A new layout translator that uses an information dispersal algorithm (IDA) to fragment the file contents and disperse them amongst a set of bricks with a configurable level of redundancy

- **Duplicate Request Cache**

  DRC for GlusterNFS to provide resilience to rpc restransmissions of non-idempotent operations.

- **Elastic Brick**

  Currently in concept.

- **Event History**

  Maintain an in-memory history of events happening

- **Event Hooks**

  Volume lifecycle extensions.

- **Inotify**

  Inotify is a File system event monitoring tool which is is essential for many types of programs ranging from file managers to security tools.

- **Multi Tenancy**
  Concept stage
- **NFSACL**

  ACLv3 support for NFS.

- **Operating Version**

  “operating-version” support in *glusterd* is required to ensure that different versions of gluster binary bits interact with each other without problems.

- **Puppet Module Puppet Module (WIP)**

  Build a puppet module for installing/adding node to a Gluster cluster.

- **Server Quorum**

  This feature, when enabled, kills the bricks in the volume that do not meet the quorum because of network splits/outages.

- **Snapshot**

  Crash consistent snapshot of glusterfs volumes

- **Split Network**

  For many users, having a separate network exclusively for servers is highly desirable for both performance reasons (segregating administrative traffic and/or second-hop NFS traffic from ongoing user I/O) and security reasons (limiting administrative access to the private network).

- **Write Once Read Many times option**

  “worm” support in *gluster volume set <volname> worm on* is required to enable volumes to be converted a worm type.

List of References

[1] Ori File System, Stanford Website, online: http://ori.scs.stanford.edu/, visited: 2015-03-16
[2] Ori File System, Bitbucket Wiki, online: https://bitbucket.org/orifs/ori/wiki/Home, visited: 2015-03-16
[3] Ali José Mashtizadeh, Andrea Bittau, Yifeng Frang Huang, David Mazières. Replication, History, and Grafting in the Ori File System. In Proceedings of the 24th Symposium on Operating Systems Principles, November 2013. Paper.
[4] Apache Hadoop FileSystem, http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html, visited: 2015-03-16
[5] GlusterFS, http://www.gluster.org/documentation/howto/HowTo/, visited: 2015-03-16

[6] https://mailman.stanford.edu/pipermail/orifs-devel/2014-October/000075.html, visited: 2015-03-9

[7] https://mailman.stanford.edu/pipermail/orifs-devel/2015-January/000116.html, visited: 2015-03-19
[8] https://bitbucket.org/orifs/ori-orisyncng/src/04a8f3e31f01ff18ec2bc72482efed4732c58a57/orisync/?at=master, visited: 2015-03-19
[9] http://ori.scs.stanford.edu/#source, visited: 2015-03-19
[10] http://ori.scs.stanford.edu/building.html, visited: 2015-03-19
[11] https://github.com/orifs/ori, visited: 2015-03-19
[12] https://news.ycombinator.com/item?id=7072492, visited: 2015-03-19

[13] http://sigops.org/sosp/sosp13/papers/p151-mashtizadeh.pdf, visited: 2015-03-19

[14] http://www.gluster.org/documentation/architecture/Features_Overview/, visited: 2015-03-19