

Cluster synchronisation with Csync²

Synchronizing configuration files and application images

Clifford Wolf

Csync² - <http://oss.linbit.com/csync2/>

ROCK Linux - <http://www.rocklinux.org/>

SPL - <http://www.clifford.at/>



Introduction

- Synchronous vs. Asynchronous
- Csync² Overview
- Csync² Features

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

Introduction



Synchronous vs. Asynchronous

Introduction

● Synchronous vs. Asynchronous

● Csync² Overview

● Csync² Features

The Csync² Algorithm

Setting up Csync²

Example Configs

URLs and References

- Synchronous synchronization (e.g. DRBD)
 - ◆ Difficult in active-active setups
 - ◆ Even more difficult in clusters with more than two hosts
 - ◆ Needs to be implemented in kernel space
 - ◆ All hosts always have the same data
- Asynchronous synchronization (e.g. Csync²)
 - ◆ Can also be implemented easily for complex environments
 - ◆ Much simpler and thus less error-prone algorithms
 - ◆ It is possible to test changes before deploying them
 - ◆ Hosts can be out of sync temporary
- None of the both is better.
- Use the synchronization method which fits your usage scenario.



Csycnc² Overview

Introduction

● Synchronous vs. Asynchronous

● Csycnc² Overview

● Csycnc² Features

The Csycnc² Algorithm

Setting up Csycnc²

Example Configs

URLs and References

- A Free Software (GPL) Tool for asynchronous synchronization.
- csync2 is a “single shot” command line tool.
- Uses its own network protocol (TCP 30865).
- The daemon can be run via inetd or stand alone.
- Provides a wide range of synchronization features.
- Tested well on Linux and Cygwin.
- Is expected to work well on non-Linux Unices too.
- Uses libsqlite (version 2) for the backend database.
- Might not compile instantaneously with non-gcc compilers.



Csync² Features

Introduction

● Synchronous vs. Asynchronous

● Csync² Overview

● Csync² Features

The Csync² Algorithm

Setting up Csync²

Example Configs

URLs and References

- Conflict detection
 - ◆ When a file has been changed on more than one host, a conflict is detected.
 - ◆ Conflicts can be resolved manually or automatically.
- Replicating file removals
 - ◆ A file removal is detected as such and replicated correctly.
- Complex setups
 - ◆ More than two hosts and multiple synchronization groups
 - ◆ Different base directories on different hosts
- Reacting to updates
 - ◆ Letting csync2 execute arbitrary commands in reaction to file updates.



[Introduction](#)

The Csync² Algorithm

- Scanning
- Updating (1/2)
- Updating (2/2)
- Comparing
- Performance

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

The Csync² Algorithm



Scanning

Introduction

The Csync² Algorithm

● Scanning

● Updating (1/2)

● Updating (2/2)

● Comparing

● Performance

Setting up Csync²

Example Configs

URLs and References

- All files referred by the configuration file are `stat()`ed.
- The file metadata (mtime, size, etc.) are written to the local backend database (sqlite).
- All modifications (addition, removals and changes) are scheduled for syncing.
- This is done by adding the files in question to the `dirty` table in the backend database.
- The scanning code is triggered by `csync2 -c`, `csync2 -x` and when update requests from a peer are received.



Updating (1/2)

[Introduction](#)

[The Csync² Algorithm](#)

● Scanning

● **Updating (1/2)**

● Updating (2/2)

● Comparing

● Performance

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

- The files listed in the `dirty` table are sent to the peers.
- The communication between the peers is handled with Csync²'s own protocol (TCP port 30865).
- Authentication is performed using pre-shared-keys, the ip addresses and the SSL certificates.
- The rsync algorithm is used for actually updating files on the peers.
- Csync² detects if a file didn't change at all (e.g. the file has been `touched`).



Updating (2/2)

[Introduction](#)

[The Csync² Algorithm](#)

- Scanning
- Updating (1/2)
- Updating (2/2)
- Comparing
- Performance

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

- The peer can refuse to update the file (e.g. if a conflict is detected).
- The record in the `dirty` table is removed if the update was successful.
- The peer executes the action handlers, if any are configured for the updated files.
- The updating code is triggered by `csync2 -u` and `csync2 -X`.



Comparing

[Introduction](#)

[The Csync² Algorithm](#)

- Scanning
- Updating (1/2)
- Updating (2/2)
- **Comparing**
- Performance

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

- It also is possible to compare the csync2 databases on two hosts.
- **Csync²** can generate lists of different files and unified diffs.
- This is e.g. useful for resolving conflicts.
- The databases are compared, not the filesystem.
- So it is a good idea to do a full scan on the hosts before comparing them.
- The compare code is triggered by `csync2 -T`.



Performance

Introduction

The **Csync²** Algorithm

- Scanning
- Updating (1/2)
- Updating (2/2)
- Comparing
- Performance

Setting up **Csync²**

Example Configs

URLs and References

- We have excellent experiences with setups with a few hundred thousand files with a few (up to four) hosts as well as a few thousand files with many hosts.
- The actual performance depends a lot on the hardware, filesystems and I/O load of the hosts.
- **Csync²** implements a full mesh synchronisation network. Really huge setups with hundreds of hosts and many changes might be better synchronized using a star or tree synchronisation network.
- The term "good performance" always depends on your requirements. Make tests before deploying **Csync²** in huge environments.
- There is a setup with 3.6 million files on four hosts. This is pretty much the limit of what makes sense to synchronize with **Csync²**.



[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

- Building and Installing
- Configuration
- Running Csync²
- Bootstrapping

[Example Configs](#)

[URLs and References](#)

Setting up Csync²



Building and Installing

Introduction

The **Csync²** Algorithm

Setting up **Csync²**

● Building and Installing

● Configuration

● Running **Csync²**

● Bootstrapping

Example Configs

URLs and References

- **Csync²** can be built with
`./configure && make && make install`
- Beware of distributions with horribly outdated **Csync²** packages!
- The **Csync²** sources are prepared to be built with `rpmbuild` and `debuild`.
- `make cert` creates a self-signed SSL certificate for **Csync²**.
- Records in `/etc/services` and `/etc/inetd.conf` need to be created.
- **Csync²** depends on `libsqlite` (version 2) and `libsync`.



Configuration

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

● Building and Installing

● Configuration

● Running Csync²

● Bootstrapping

[Example Configs](#)

[URLs and References](#)

- The configuration file `/etc/csync2.cfg` needs to be written.
- This configuration file defines so-called synchronization groups.
- The synchronization groups contain:
 - ◆ A list of hosts
 - ◆ A pre-shared key file
 - ◆ File include/exclude patterns
 - ◆ Action handlers (optional)
- The same configuration file can be used on all hosts.
- Some example configurations are discussed later in the presentation.



Running Csync²

Introduction

The Csync² Algorithm

Setting up Csync²

- Building and Installing
- Configuration

● Running Csync²

- Bootstrapping

Example Configs

URLs and References

- Running full scan and update: `csync2 -x`
- .. usually **Csync²** is always executed this way
- Just running a full scan: `csync2 -cr /`
- .. the `-r` makes **Csync²** operate recursively
- Just running a full update: `csync2 -u`
- Showing a unified diff for a file: `csync2 -TT /etc/hosts`
- Printing the help message: `csync2`
- .. or `man csync2` for a more verbose version.



Bootstrapping

Introduction

The **Csync²** Algorithm

Setting up **Csync²**

- Building and Installing
- Configuration
- Running **Csync²**
- Bootstrapping

Example Configs

URLs and References

- In most cases it is ok to simply run a full **Csync²** scan and update cycle on all hosts (`csync2 -x`).
- This will try to sync everything from every host to all peers.
- For large setups this is a waste of resources.
- It is possible to run a full scan without scheduling updates by running `csync2 -cIr /`.
- And it is possible to schedule updates by comparing the hosts using `csync2 -TI`.
- This is much faster.
- But it is not applicable for daily operations because the detection of conflicts and file removals doesn't work this way.



[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

Example Configs

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- Prefix Sections
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

Example Configs



Minimalistic

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- Prefix Sections
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group mygroup
{
    host host1 host2;

    key /etc/csync2.key_mygroup;

    include /var/sharedfiles;
}
```



Multiple Groups

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

Example Configs

- Minimalistic
- **Multiple Groups**
- Include/Exclude Patterns
- Prefix Sections
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group biggroup {
    host host1 host2 host3 host4;
    key /etc/csync2.key_biggroup;
    include /etc/hosts;
}

group smallgroup12 {
    host host1 host2;
    key /etc/csync2.key_smallgroup12;
    include /etc/passwd /etc/shadow /etc/group;
}

group smallgroup34 {
    host host3 host4;
    key /etc/csync2.key_smallgroup34;
    include /etc/passwd /etc/shadow /etc/group;
}
```



Include/Exclude Patterns

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

Example Configs

- Minimalistic
- Multiple Groups
- **Include/Exclude Patterns**
- Prefix Sections
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group mygroup
{
    host host1 host2;
    key /etc/csync2.key_mygroup;

    # pathname patterns
    # last match is decisive, exclude per default
    include /etc/apache2;
    exclude /etc/apache2/envvars;
    exclude /etc/apache2/ssl;

    # basename patterns
    # last match is decisive, include per default
    exclude *~ *.swp;
    include apache2.conf~;
}
```



Prefix Sections

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

Example Configs

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- **Prefix Sections**
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group mygroup
{
    host host1 host2 host3;
    key /etc/csync2.key_mygroup;
    include %homedir%/bob;
}

prefix homedir
{
    on host1:
        /export/users;
    on *:
        /home;
}
```



Actions

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- Prefix Sections

● Actions

- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group mygroup
{
    host host1 host2 host3;
    key /etc/csync2.key_mygroup;
    include /etc/apache2;

    action
    {
        pattern /etc/apache/httpd.conf;
        pattern /etc/apache/sites-available/*;
        exec "/usr/sbin/apache2ctl graceful";
        logfile "/var/log/csync2_action.log";
        do-local;
    }
}
```



Backups

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- Prefix Sections
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group mygroup
{
    host host1 host2;
    key /etc/csync2.key_mygroup;
    include /var/sharedfiles;

    backup-directory /var/backups/csync2;
    backup-generations 3;
}
```



Sync Networks

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

Example Configs

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- Prefix Sections
- Actions
- Backups
- Sync Networks
- More..

[URLs and References](#)

```
group mygroup
{
    host host1@host1-back host2@host2-back;

    key /etc/csync2.key_mygroup;
    include /var/sharedfiles;
}
```




More..

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

- Minimalistic
- Multiple Groups
- Include/Exclude Patterns
- Prefix Sections
- Actions
- Backups
- Sync Networks
- **More..**

[URLs and References](#)

A full description of the `csync2.cfg` syntax can be found in the **Csync²** manual.



[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

- [Related Projects](#)
- [Credits](#)

URLs and References



Related Projects

[Introduction](#)

[The Csync² Algorithm](#)

[Setting up Csync²](#)

[Example Configs](#)

[URLs and References](#)

● [Related Projects](#)

● [Credits](#)

- **librsync**: A library for rsync-like binary diffing
<http://librsync.sourceforge.net/>
- **sqlite**: A small embeddable SQL database engine
<http://www.sqlite.org/>
- **rsync**: popular file synchronisation tool
<http://rsync.samba.org/>
- **unison**: another file synchronisation tool
<http://www.cis.upenn.edu/~bcpierce/unison/>



Credits

Introduction

The **Csync²** Algorithm

Setting up **Csync²**

Example Configs

URLs and References

● Related Projects

● Credits

- LINBIT Information Technologies GmbH:
<http://www.linbit.com/>

- Clifford Wolf:
<http://www.clifford.at/>

<http://oss.linbit.com/csync2/>