Software requirements

Application name:

"Intelligent incremental backup system for web servers"

Ondrej Makši, Filip Lukáč,

Marek Kukučka, Ivan Martynenko

Table of contents

Introduction **2**

1.1. Purpose of requirements document 2

1.2. Scope of the product 2

1.3. Definitions, acronyms and abbreviations 2

1.4. References 3

1.5. Overview of the remainder of the document 3

General description **3**

2.1. Product perspective 3

2.2. Product functions 4

2.3. User characteristics 5

2.4. General constraints 5

2.5. Assumptions and dependencies 5

Specific requirements **5**

3.1. Functional requirements 5

3.2. Non-functional requirements 9

Introduction

1.1. Purpose of requirements document

This is a formal document created with the standard IEEE/ANSI 830-1998 (Recommended Practice for Software Requirements Specifications). The main purpose of this document is to specify all the requirements for the system, which was created as a project in the subject "Development of Information Systems" in the Faculty of Mathematics Physics and Informatics (FMFI UK). The requirements document is dedicated to all stakeholders. It can also work as an agreement between the developers and the client.

1.2. Scope of the product

The main goal is to create a system that can create backup copies of websites used in the Faculty of Mathematics Physics and Informatics (UK BA) with the ability to store them on a local disk as well as remotely and restore them to their original state when needed.

1.3. Definitions, acronyms and abbreviations

Stakeholder - is anyone who influences or interferes with product development

Full backup - this is the most complete type of backup, which clones all the selected data.

Incremental backup - will only store changes that were made since the previous backup was done.

CLI (command-line interface) - is a text-based user interface used to communicate with the system

Configuration file - A file that defines the parameters, options, settings and preferences applied to computer programs

Client instance - An instance of our application

Local server - A server instance where the client instance is running

Storage server - A server instance where backups are stored

Site - Something we want to back up. e.g. A running website with a database.

1.4. References

- Github project repository
- Subject webpage

1.5. Overview of the remainder of the document

The second chapter describes the product, its features, and users who interact with the product.

The third chapter focuses on the exact specification of the individual functions of the system.

General description

2.1. Product perspective

The application will be a console program which will be distributed as a compiled executable file, but with open-source code so that the user can compile it himself.

The application will be able to backup and restore sites on the local server according to user-configured requirements with the options of a full or incremental backup, and storage server.

2.2. Product functions

The main purpose of the application is to backup the data to the storage server and restore all data (diverse file types) from the storage server.

There will exist two types of backup:

- Full backup - makes a full backup
- Incremental backup - makes a copy only of files/folders that were modified/created/removed since last backup

There will exist two different backup storages:

- local - data will be stored on the local server
- remote - stored on the storage server (storage server is only a remote storage location, commands are not started from there, only from local server)

The application administrator will be able to configure options (per site) within a configuration file. See section 3 for more details. The application could be configured only by a user with root privileges and will run on a linux system.

There will be two folders where configuration will be managed:

- sites-available - contains configurations of all sites on a local server
- sites-enabled - contains links to the configurations in the folder sites - available (only the configurations with a link in this folder are active)

The administrator will be able to change the sites-available folder by hand at any time however the sites-enabled folder can be changed only through service commands within the application. All changes made will take effect the next time the application is started.

2.3. User characteristics

Administrator - A user with root privileges that can modify the config file and run backups on selected sites. This is also the only user that can configure the application and make changes in the configuration file.

2.4. General constraints

For suitable and comfortable usage of the program root (sudo) privileges are needed as well as having the configuration file correctly set up.
There is a restriction of having a minimum of 1 full backup stored.

2.5. Assumptions and dependencies

Software will be developed as a CLI application in Java/C++ programming language.

Application will always ask the user before overwriting any existing data.

Specific requirements

3.1. Functional requirements

3.1.1 The application must be a console program

3.1.2 User interaction with the program will be based on the CLI

3.1.3 The CLI of the application must have the following parameters and features

3.1.3.1 Output the list of all configured sites of all local servers that share storage server with this local server. Each list item will include the date and time of the last backup and the location of the site (user specified id of server provided in config).

Format of each line in output:
site_name , local_server_name, storage_server(IP or hostname), date/time of last backup

3.1.3.2 Output a list of backups for a specified site. This can be one of the sites from the list in 3.1.3.1.

Format of each line in output:
date/time of last backup, type (incremental/full)

3.1.3.3 Config files will be modified independently of the application by user using his or her preferred editor.

3.1.3.4 Be able to enable or disable sites by adding or removing links to the sites-enabled folder using service commands.

3.1.3.5 Be able to manually run a full or incremental backup of one site only from the local machine at any time, the site is specified by its site name.

3.1.3.6 Be able to restore a specific selected backup

3.1.4 The application must be configurable using configuration files

3.1.5 The configuration file must have the following parameters

3.1.5.1 Id (site name)

3.1.5.2 Id (local server name)

3.1.5.3 List of included paths

These files will be included in a backup.
Glob patterns are accepted.

3.1.5.4 List of excluded files

These files will be excluded from the backup.
Glob patterns are accepted.

3.1.5.5 Pre_backup script - optional, if anything is needed to prepare the website for backup it should be included in this script which will be run before each backup. For example, stopping the instance or dumping the db.
If the script returns nonzero exit code, the backup will not proceed.

3.1.5.6 Post_backup script - optional, counterpart of 3.1.5.5 - if anything is needed to be executed after the backup, for example starting the instance, or the database, or cleaning up the db dump, it should be included in this script which will be run after each backup.

3.1.5.7 Pre_restore script - same as 3.1.5.5 but run before each restore.

3.1.5.8 Post_restore script - same as 3.1.5.6 but run after each restore.

3.1.5.9 List of full backup periods that are kept on storage server

These periods are specified as the age of the backup in days. The number of periods (intervals) depends on the user requirements and can be variable. Optional req: the granularity can be more precise than in days.

Example:

- Keep one 180 days old
- Keep one 30 days old
- Keep one 7 days old

That means to keep one backup A that is at least *n* days old and backup B that is *n* days younger than backup A for each specified period. Backup B becomes backup A when its age will reach *n* days.

3.1.5.10 incremental backup period in days, for example 1 means add one incremental backup per day

3.1.5.11 backup time of day - specifies at what time in day to perform incremental backup (interpreted as not before this time, the next time the application is started from cron), example: 04:10.

3.1.5.12 remote storage address - ip address or hostname, user name (it is expected that scp/sftp will not ask for password, because ssh keys are present in the .ssh folder of the backup user)

3.1.5.13 switch: 0/1 - keep backup also on local server (if backups are kept on local server, then all incremental backups must be there plus exactly one full backup, otherwise no local backups are kept after copying to remote server)

3.1.5.14 local storage location - directory

3.1.5.15 remote storage location - directory

3.1.5.16 e-mail of the admin, which should be notified if backup runs into a problem.

3.1.6 The application can get started automatically and periodically by cron according to cron´s configuration. The administrator is responsible for cron configuration, it is not managed by this application.

3.1.7 When the application is started by cron it must make sure that it will read configuration files, start scripts and create backups correctly according to its parameters and the sites-enabled folder.

3.1.8 The application will be run with root rights

3.1.9 Backups will always be saved to a local drive first and copied to a remote server, they are kept locally only if required by a setting in the config file.

3.1.10 It must be possible to restore backups to their original location or to a different location specified by the user.

3.1.11 It must be possible to restore a specific requested file(s) (specified using full relative path(s) listed in a separate text file given in the command line) from a specified backup. Glob patterns are accepted.

3.1.12 The application will have a general log file where all backup and restore actions will be logged. This log will contain general information related to the action performed. This log will not contain information on the files affected by this action. This log will be stored only locally.

3.1.13 The application will create additional log files for each backup and restore performed with detailed information on each affected file. This log will be stored only locally.

3.1.14 The application when started from cron runs without producing any output on its standard output.

3.1.15 When restoring from a backup, the application prints status information in terms of percentage of files restored.

3.1.16 Restoring from a backup also restores the original file permissions and owners.

3.1.17 If restoring runs into some problem (no space, non-existent user, etc), it pauses, and asks the user whether it should attempt to repeat the same action again. If started with --noprompt, it will never ask the user, only report in the log file, but continues with the remaining work.

3.1.18 When backup runs into a problem: it continues with the remaining work as if no problem occurred, it reports the problem in the logfile, and it notifies the admin by email.

3.2. Non-functional requirements

3.2.1 The application must be simple to use

3.2.2 The application should run smoothly and efficiently, without any significant time loss

3.2.3 The application must be reasonably safe from malicious attacks

3.2.4 The application must be able to check the authorization and authentication of its users

3.2.5 The application should not save more backups than necessary

3.2.6 The application has to be developed primarily for the linux platform