Some technical information about #rosa-devel

№ 2

19 October 2012 г.

ROSA Planet

Just Open!

Contens

Introduction

Good news everyone! May the successful compilation be with you, programms stay unbugged and kernel won't panic!

So here we are with our second bulletin. We tried to take into account previous experience and expressed wishes. We would like to say «thank you» to all our developers and engineers, who found time in their busy schedule to write and send articles to us.

All your questions, comments, recalls and wishes about interesting themes, please, send to the rosa-point@rosalab.ru.

Desktop

ROSA Popularity Contest — packages popularity research

ROSA has thousands of packages with different apps, shared libraries and utilities. For some of them we provide complete quality support, for others — guarantee only basic functionality.

To help developers of ROSA know what packages are more important for users, we have prepared a special tool — rosa-popularity-contest — which collects statistics about packages used in the system. Once a week the tool sends a report to the server, where the overall statistics is counted.

To take part in this research, it is enough just to install the rosa-popularity-contest package. Nno additional actions are required— the tool is completely automated.

Analys is based on information about the last access to the package's files (atime). You can learn details of work of rosa-popularity-contest here:
http://wiki.rosalab.ru/en/index.php/ROSA_Package_Popularity_Contest

Reports are published here: http://fba.rosalinux.ru/popcon/

Report consist of the following numerical indicators for every package:

Installed	Percent of respondents who installed package
Vote	Percent of respondents who used this package during the last month
Old	Percent of respondentswho didn't use this package
Recent	Percent of respondents who updated this package recently; we can't tell exactly either they use it or not.
No-files	Percent of respondents for whom we haven't got information about this package (for example, package is installed to a file system that is mounted with 'noatime' option)

We have now only a little number of reports from developers, but in future we hope to increase our audience significantly.

Development tools

Comparing repositories of ROSA 2012 Desktop, Mandriva Cooker and Mageia Cauldron

While developing the distribution, the choose of packages is one of the most important tasks. In solving this problem it is useful to perform analysis of other distribution repositories and find out — what do they consider as «important» for their users.

To compare urpmi repositories, we have a really good tool: urpm-repodiff. Using it you can compare repositories of ROSA, Mandriva and Mageia.

The full report about comparison between repositories of ROSA 2012 Desktop, Mandriva Cooker and Mageia Cauldron you can find here. You can separatly see the list of packages which are absent in ROSA but added in related distributions.

The reports are updated daily.

LSB repositories are available in urpmi format

Linux Standart Base (LBS), which is developed by a group of the same name of The Linux Foundation consortium, isn't just a text document, but also provides a set of different software which can simplify developing of distributions and apps which meet requirements of the standard.

Thanks to the efforts of ROSA developers and engineers of The Linux Foundation, repositories with LSB software are available now in urpmi format and can be added in ROSA distributions.

For adding repositories with the latest versions of LSB tools in 32-bit system you should do the following (with root privileges):

urpmi.addmedia lsb http://ftp.linuxfoundation.org/pub/lsb/repositories/rpm/lsb-4.1/repo-ia32/

You can also add repository with snapshot-versions of tools:

urpmi.addmedia lsb-snapshot http://ftp.linuxfoundation.org/pub/lsb/repositories/rpm/lsb-snapshot/repo-ia32/

You should use following commands for 64-bit versions:

urpmi.addmedia lsb64 http://ftp.linuxfoundation.org/pub/lsb/repositories/rpm/lsb-4.1/repo-x86_64/
urpmi.addmedia lsb64-snapshot http://ftp.linuxfoundation.org/pub/lsb/repositories/rpm/lsb-snapshot/repo-x86_64/

After this, the LSB packages will be available for installation. We can divide them into three groups:

- Tools for assessing of portability of apps and checking their complience with the LSB standart. The basic tool is Linux Application Checker (lsb-task-app-testkit package, start command /opt/lsb/app-checker/bin/app-checker-start.pl);

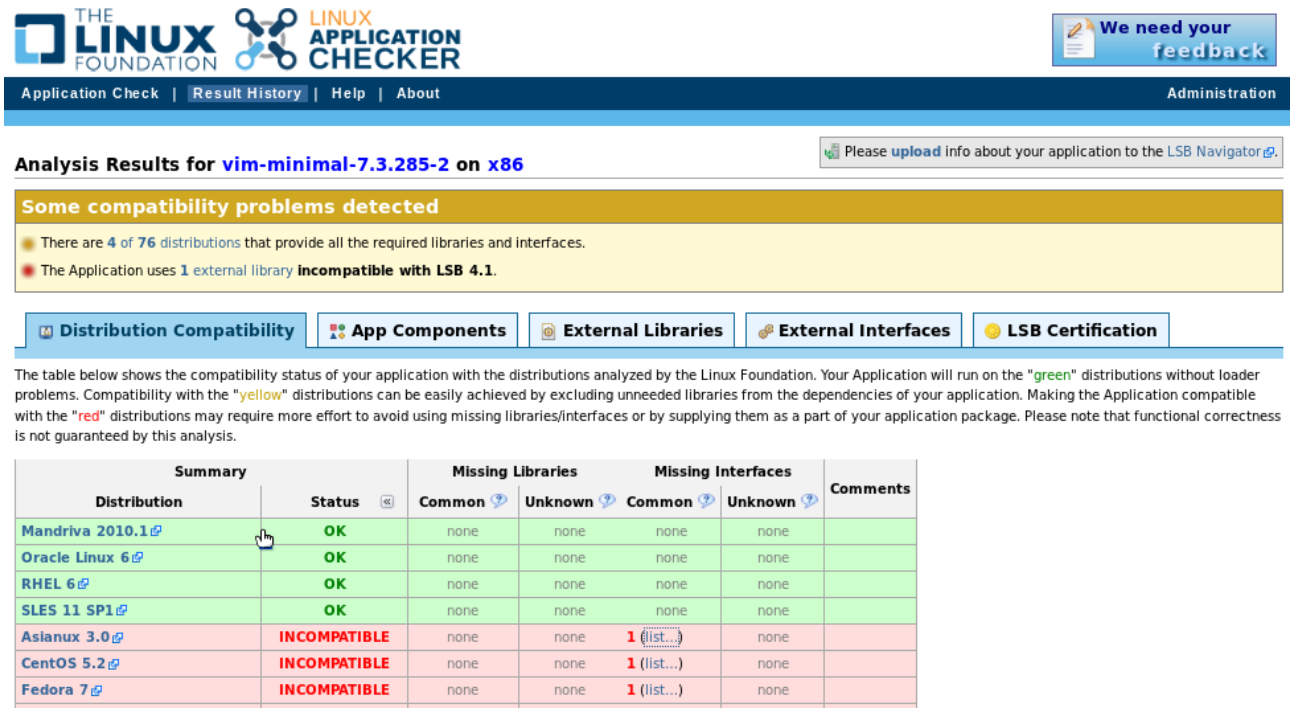
- Compliance tests for checking distributions if they meet the LSB standart and framework for launching tests and analysing their results — Linux Distribution Checker. To receive all tests and the framework, you should install lsb-task-dist-testkit;

- LSB SDK tools for developing portable apps (lsb-task-sdk package).

LSB tools are targeted at developes but can be useful for ordinary users, too. For example, Linux Application Checker allows to check the app's ability to run in the most popular Linux distributions.

Detailed information is available on The Linux foundation site: https://wiki.linuxfundation.org/en/RepositoryInfo

The Linux Application Checker report for the vim-minimal package from ROSA 2012 Marathon. Vim-minimal can be run without recompiling in Mandriva 2010.1, Oracle 6, RHEL 6 и SLES 11 SP1, but not in Asianux 3.0, CentOS 5.2 or Fedora 7.



Urpm-repoclosure — analysis of some repositories and HTML-reports

One of the tools that we use for controling the status of ROSA repositories is urpm-repoclosure which checks the absence of unresolved dependencies.

Initially, urpm-repoclosure was able to check closure of one certain repository or certain set of packages. However, the structure of ROSA repositories is such that only main is self-sufficient repository. While analysing closeness of contrib and non-free, it is important to keep in mind that packages from this repositories can use packages from main. After all, we need to take into consideration updates in the updates branches which can replace packages from release branches.

So far for analysing something besides main/release, we have copied all required packages in one place and then processed them with urpm-repomanage (just to leave only the latest versions). But this approach means that we should work with packages directly and this is resource-intensive activity. At the same time urpm-repoclosure supports faster way of working — using files synthesis.hdlist (-hdlist mode).

Recently we have added in urpm-repoclosure support of working with several synthesis.hdlist files and now it can unite them dropping outdated versions of packages. Moreover, you can specify which additional repositories have updates for analysed files (file with list of corresponding synthesis.hdlist files should be passed to – **update-hdlists**), and which should be used to only satisfy their dependencies (**-dep-hdlists)**. Now you can perform analysis of the contrib repository using only four synthesis.hdlist files (from *main/release*, *main/updates*, *contrib/release* and *contrib/update):*

echo "synthesis.hdlist-contrib.updates" > updates.list

```
echo "synthesis.hdlist-main.release"    > dep.list
echo "synthesis.hdlist-main.updates"    >> dep.list
```

urpm-repoclosure -hdlist synthesis.hdlist-contrib.updates -update-hdlists updates.list -dep-hdlists dep.list

You can similarly check closure of personal repositories in ABF.

Finally, now urpm-repoclosure can generate reports in HTML format. You don't have to specify any additional options for this: HTML-report is created automaticly along with the text one. Example of HTML reports you can see at daily updated page about ROSA repositories status: http://upstream-tracker.org/repoclosure_logs/.

ROSA ABF

ABF update

We want to present 3 new features in ABF:

1. Email notification about your build tasks;
2. Support for markup of issues and comments;
3. New look of projects list page (https://abf.rosalinux.ru/projects)

Some details:

* Email notifications about your build tasks
Email notification was one of the most requested features. From now on, besides automatically updated monitoring page
and activity feed with RSS you can receive email notifications about your build tasks. There are a lot of configuration options for notifications: you cat set up events which will trigger mail sending and turn them on/off completely in the Notification center
(https://abf.rosalinux.ru/settings/notifiers).

* Support for markup of issues and comments
Comments and task descriptions are very important and good design and decorations for them does matter. Now in ABF we have an easy way to markup
your messages using Markdown - easy and lightweight markup language. We have also prepared a short note about Markdown with most useful methods
of markup (thanks github!). You can find it near any comment or issue.

* New look of projects list page
Until recently, projects list page was not very useful if you are a member of thousands of projects. But now you can restrict the visibility by owners of projects. It provides you with ability to see all projects (private or open) of the required group and fast filter using live search. Go to https://abf.rosalinux.ru/projects and see it in action.

Tips: you can select several parameters at once!

Separate storage for source code archives

A common practice for build systems of Linux distributions is to store all files required to build a package in some version control system. In case of RPM packages such files include archive with program source code, spec-file with build instructions for rpmbuild and different patches. Initially, ABF stored all such files in git repository of a corresponding project. However, it turned out that storing source code archives in Git is ineffective and unreasonable - since these are binary files for which many Git advantages (such as tracking history or merging modifications from different users) are not applicable. In addition, sometimes size of archives is rather huge, so cloning of Git repository with all branches and history information can take significant time.

An alternative way is to download archives from external sources (e.g., directly from developer sites); rpmbuild in ROSA supports such possibility. However, using external sites is not safe and reliable - at the moment when build starts necessary site can become unavailable, download of

archive can take significant time, archive can be modified or moved and so on.

In order to reduce Git load and spepd up work with repositories without involving external storages, we provide possibility to put archives with project source code into special file storage located directly on ABF servers - http://file-store.rosalinux.ru/. In order to use this solution, you need:
- upload the source code archive to http://file-store.rosalinux.ru/. After loading, you will be provided sha1 sum of the the uploaded files.
- at the project folder, create a file .abf.yml with the following content:

```
sources:
    "<file name>": <sha1 sum>
```

(for example: https://abf.rosalinux.ru/import/libreoffice/blob/rosa2012lts/.abf.yml)

Now ABF will get archive with source code from the file storage.

You can check examples of big projects using .abf.yml:
- Libreoffice: https://abf.rosalinux.ru/import/libreoffice/tree/rosa2012.1?
- Flightgear-data: https://abf.rosalinux.ru/warpc/flightgear-data

Future plans are to convert all projects to use .abf.yml and file-store and integrate autoimport and import from src.rpm with file-store. In addition, we will provide public API for file storage and add possibility to work with the storage into ABF console client.

Pull Requests

After a long period of development, we are glad to present a new ABF feature - support for Pull Requests!

* What is Pull Request?

Pull requests let you tell others about changes you've pushed to a git-repository of some ABF project. Once a pull request is sent, interested
parties can review the set of changes, discuss potential modifications, and even push follow-up commits if necessary.

* Why do we need it?

Before pull request, the only model of collaborative development available in ABF was Shared Repository Model

The Shared Repository Model is more prevalent in small teams and organizations collaborating on private projects.
Everyone is granted with push access to a single shared repository and topic branches are used to isolate changes.

And now one more model is available in ABF: Fork & Pull

The Fork & Pull Model allows everyone to fork an existing project and push changes to his personal fork without requiring access
be granted to the parent project. The changes must then be pulled into the parent project by the

project maintainer. This model reduces the amount
of friction for new contributors and is popular with open source projects because it allows people to work independently without upfront coordination.

* How can we use it?

Fork a project, make the planned changes, build if needed add pull request to the parent project, discuss the changes with other participants,
fix the code, if required. No need to request access to the parent project or discuss every patch by mail. All members of the project and interested users
can see the changes and can help with coding or testing or simply vote or express criticism / appreciation.

* We develop a project in ABF. What about us?

You can use PullRequest not only to interact between different projects. PullRequest can be used to sync two branches of the same project!
Perfect solution for code reviews. Try it!

Frontend Brother of ABF

ABF is a powerful building system and project's development. Inter alia, we create both server and desktop versions of ROSA using ABF. However, while building distribution, some tasks arise that can not be solved with ABF by virtue of its universality and aiming to the wide range of platforms. For example, we are interested in closure of the dependences for ROSA repositories, comparing the number of packages with other distributions and etc.

We have created a special site http://fba.rosalinux.ru for collecting and publishing such statistics. FBA monitors the repositories of ROSA which are located in ABF and analyzes them with urpm-tools and Upstream Tracker utilities.

Today the following kinds of reports are available:

- closure of dependences;
- comparison of package's versions with related distributions and upstream;
- list of outdated packages;
- reports of ROSA Package Popularity about the most popular packages;
- reports of Upstream Tracker about backward compatibility of different versions of shared libraries.

First version of ABF console client is available

For a long time the only way to interact with ABF was to use its Web interface. But despite rich functionality and nice look, it's not always convenient to use Web. For example, if you want to just start builds for several projects, you have to visit a page of every of that projects,
click "New build" on that pages, select target platform, etc.

For such purposes a console client would be helpful that could be launched from scripts. Thanks to

publishing of ABF API, it became possible to create such a client and its development is already in progress.
The first version is actually ready to use; it provides basic functions to work with ABF projects.

You can download abf-console-client from here:
(32 bit)
http://abf.rosalinux.ru/downloads/akirilenko_personal/repository/rosa2012lts/i586/main/release/
(64 bit)
http://abf.rosalinux.ru/downloads/akirilenko_personal/repository/rosa2012lts/x86_64/main/release/

You can get a list of available actions by invoking "abf -h". For every available command, you can get detailed information by running "abf help <command_name>" (e.g., "abf help build").

In the near future we are planning to significantly enhance functionality of the client and prepare detailed documentation concerning its usage.

2Safe

A simple client for 2Safe. API review

For two years the «2Safe» cloud storage is working in test mode and just a small range of people know what our service can do. Let's consider its functions in order.

API itself is just a number of service's functions which can be used for full work with 2Safe. For example, with them you can provide different checks, register and delete accounts, manage files, etc. You can see full list of functions here.

So now let's try to write a simple console client. This example also can be useful for automating the most frequently performed operations. We will write in Perl language and use perl-lib2safe library. It is availble in ROSA contrib repository.

Installing library:
```
$ sudo urpmi perl-lib2safe
```

In library some names of functions wer changed a bit, but they won't mislead you. For example: list_dir -> ls, copy_file -> cp and etc.

Reading manual:

```
$ man lib2safe
```

Open your favorite text editor:

```
$ vi test2safe-client.pl
```

And write your program:

```
#!/usr/bin/perl

use lib2safe;

my $psync = new lib2safe (URL => 'https://api.2safe.com/');
```
//By the way, we can write it without any parameters. https://api.2safe.com/ is the default URL.

```
my $mylogin = $psync->login(login=>'mylogin', password=>'****');
```
//Here we are authorizing as user that was registered earlier.

```
// Cheching authorize status
die('Error Authorize') unless
  ($$mylogin{response}{success} && $$mylogin{response}{success} eq 'true');

my $ls = $psync->ls();
```
//if we will leave it without parameters, it will display content of the root folder

```
my $new_dir = $psync->mkdir(dir_id => $$ls{response}{root}{id}, dir_name =>
'new_dir');
```
// here we created new folder named «new_dir»

```
my $testfile = $psync->put(dir_id => $$new_dir{response}{dir_id}, file => "test2safe-
client.pl", name => '2safe-client.pl');
```
//here we are uploading into «new_dir» our script with different name

We should note that the example contents only a few functions. About 40 of them are implemented in the library, here is a list of sections:

- User functions;
- File system functions;
- Lock functions;
- Sharing and file's publication functions;
- Versioning functions

Also I'd like to note that the official site is using the same list of functions. Developers don't use any other calls that differ from listed.

Authors

Alexander Burmashev

Vladimir Sharshov

Denis Koryavov

Denis Silakov

Sergey Sokolov

Anton Chernyshov

Konstantin Kochereskin