Some technical information about #rosa-devel

№ 5

ROSA Planet

Just Open!

Contens

INTRODUCTION

The ROSA Planet team is pleased to present a new issue of our bulletin. This is our first issue in the New Year. The year 2013 is going to be really amazing. Take at least the fact that ROSA Planet will change its format. This issue is the last which you receive as a file. In March, we are planning to switch to a Web format which will be incessantly updated.

And not forget about upcoming announcements of new releases and updates of our products! There are numerous ABF improvements coming, as well as ROSA Desktop Fresh fixes, new RELS releases... and a lot of other interesting events, but we won't show our cards prematurely

Meantime, let's look through our current news.

Truly yours,
ROSA Planet team!

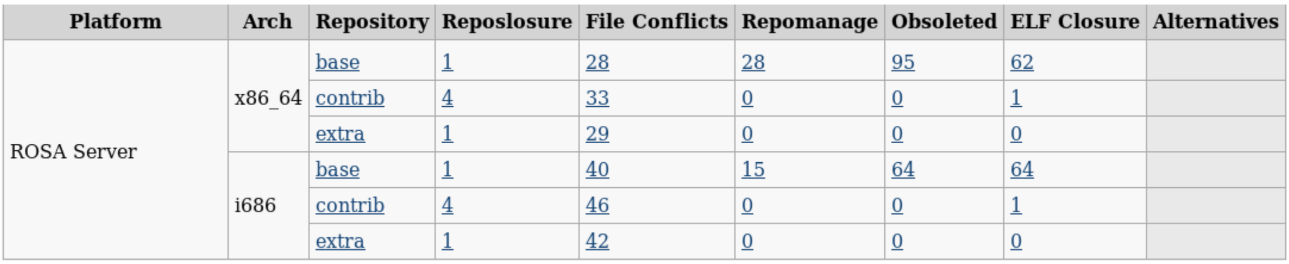
DEVELOPMENT TOOLS

Monitoring of RELS Repositories

As many of you likely know, ROSA repositories are subjected to constant monitoring aimed to detect potential problems in the package base. for a long time, regularly updated results of such monitoring for ROSA Desktop series have been publicly available at http://fba.rosalinux.ru (by the way, we have recently added one more kind of reports - File Conflicts - that reflects packages containing the same files but not explicitly marked as conflicting using the Conflicts tag).

But Desktop is not the only direction of ROSA development; another important member of ROSA OS family is ROSA Enterprise Linux Server (RELS). Currently the same kinds of reports are available for RLES and ROSA Desktop except the Alternatives analysis (which is currently provided for ROSA Desktop only, though we plan to add it for RELS in the near future, as well).

As one can see from the report table, RELS package base is in a really good shape - typical numbers for such reports are dozens or even hundreds of problems, while the number of potentially problematic packages in RELS is close to zero.



Platform	Arch	Repository	Reposlosure	File Conflicts	Repomanage	Obsoleted	ELF Closure	Alternatives
ROSA Server	x86_64	base	1	28	28	95	62	
		contrib	4	33	0	0	1	
		extra	1	29	0	0	0	
	i686	base	1	40	15	64	64	
		contrib	4	46	0	0	1	
		extra	1	42	0	0	0	

ROSA Autotest

After several months of development, the initial version of the automated testing system for ROSA Linux distributions ("ROSA Autotest") is up and running.

For the present, the system checks ROSA Desktop Fresh 2012 releases. It operates as follows:

- Checks daily, whether new ISO images for this distro are available at ABF and downloads these images to a local machine for testing.
- Checks whether the OS can be started in live mode using these ISO images on the virtual machines.
- Checks whether the OS can be installed to the virtual machines from these ISO images.
- Enables the default software sources on the installed systems and performs a software update there.
- Uses Autotest (http://autotest.github.com/) to run a few tests on the installed systems:
    - a clock manipulation test ("hwclock");
    - a stress test for filesystem-related components of the kernel ("dbench");
    - simple tests for IPv6-related software ("ipv6connect");
    - etc.

More tests are to be added to the system in the future.

The results of the automated testing are available at FBA: http://fba.rosalinux.ru/autotest/

ROSA ABF

ABF 2.0: The New Build System

With the new year, a new build subsystem comes to ABF. Initially it was possible to only use it to build packages to personal repositories, but after a testing period the whole ABF was switched to the new core.

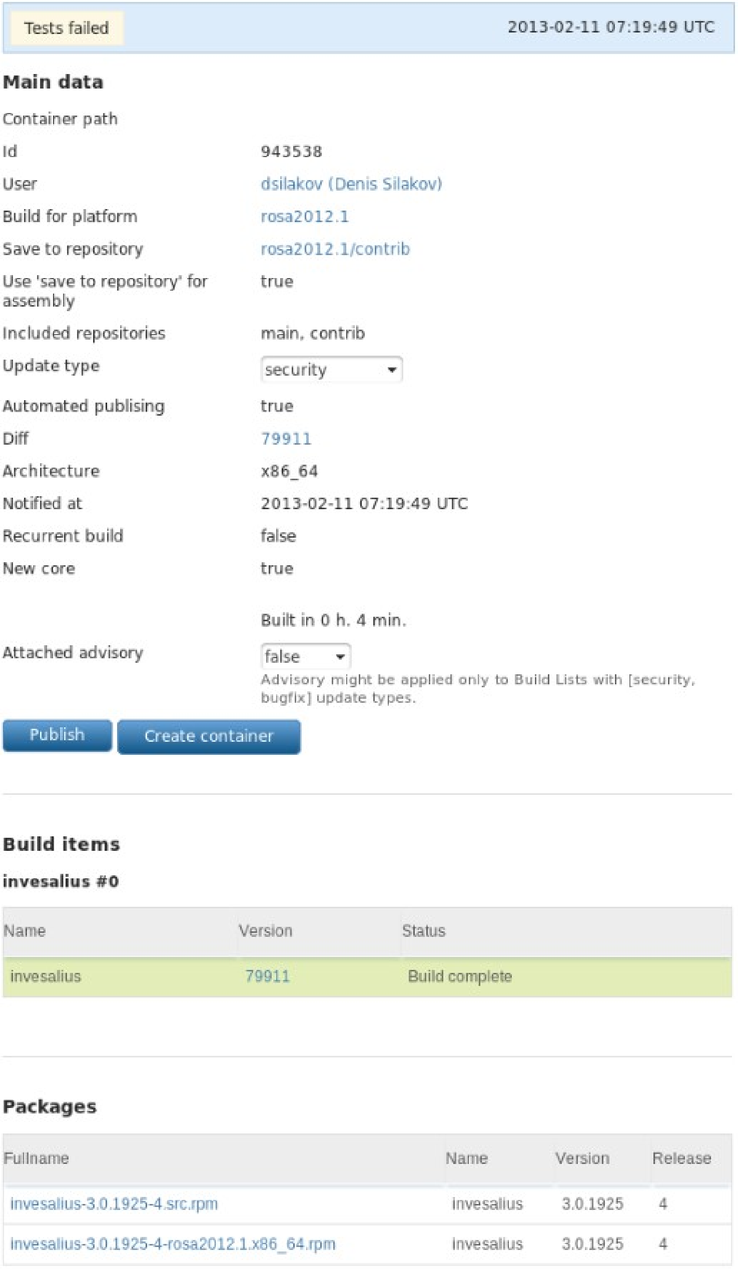
What do you get if using this subsystem?

- improved stability, scalability and security; in particular, we use temporary virtual machines for user tasks;
- ability to cancel already running task (previously you could cancel only pending builds);
- abbreviated auto updated log of all stages of the build process (previously only the package build log was provided) in the web interface. All detailed logs are also available, as before;
- universal assembly client: any distribution will receive as many clients as available;
- excellent opportunities to support different distributions using universal web and assembly client;
- publishing packages as a transaction: in case of failure, the repository will return to its original state. Also added the log of the publishing process;
- complete data on every assembly task: packages and logs will be available after publication (at present such data is dropped after publication);
- use of mock-urpm to build packages for ROSA (previously custom scripts were used that were not available to users for local assembly);
- packager tag is now set automatically;
- checking built package by installing it in the chroot; if this check fails, then the whole build ends with "Test Failed" status and resulting packages are not published, even if you have chosen appropriate flag for the build. However, the packages are not deleted and if you are completely sure that it is ok (e.g., all necessary dependencies have been built in the neighboring project and just haven't been published yet), then you can publish the package;
- strict cleanup of obsolete packages when publishing newer version for all platforms;
- publication of custom i686 packages to i686 and x86_64 for RHEL type platforms at the

same time;

- ability to create a link to custom line in file, for example: https://abf.rosalinux.ru/import/e_modules/blob/master/e_modules.spec#L73;
- ability to rollback package publication for administrators of platforms and repositories;
- ability do delete your own files from file-store (http://abf-doc.rosalinux.ru/v1/file-store/#destroy-file);
- ability **not to use** the target personal repository as a source of packages for assembly;
- collection of packages data (names, versions, releases) for RELS (previously such information was collected only for ROSA).

Please note that the new build core doesn't automatically create containers for built but not published projects. This solution was implemented to save the time, since in many cases containers are not necessary and you can download resulting packages directly from the build result page. However, you can still create container and use it as a repository by clicking on appropriate button in the build result page.



Tests failed 2013-02-11 07:19:49 UTC

Main data

Container path	
Id	943538
User	dsilakov (Denis Silakov)
Build for platform	rosa2012.1
Save to repository	rosa2012.1/contrib
Use 'save to repository' for assembly	true
Included repositories	main, contrib
Update type	security
Automated publising	true
Diff	79911
Architecture	x86_64
Notified at	2013-02-11 07:19:49 UTC
Recurrent build	false
New core	true
	Built in 0 h. 4 min.
Attached advisory	false Advisory might be applied only to Build Lists with [security, bugfix] update types.

Build items

invesalius #0

Name	Version	Status
invesalius	79911	Build complete

Packages

Fullname	Name	Version	Release
invesalius-3.0.1925-4.src.rpm	invesalius	3.0.1925	4
invesalius-3.0.1925-4-rosa2012.1.x86_64.rpm	invesalius	3.0.1925	4

New ABF features - the build has succeeded, but the installation of the resulting packaged into chroot failed. Nevertheless, the package was not dropped (it can be downloaded together with srpm by links at the bottom of the page), it can be published to the target repository or one can create a container for it.

What should you expect in the near future?

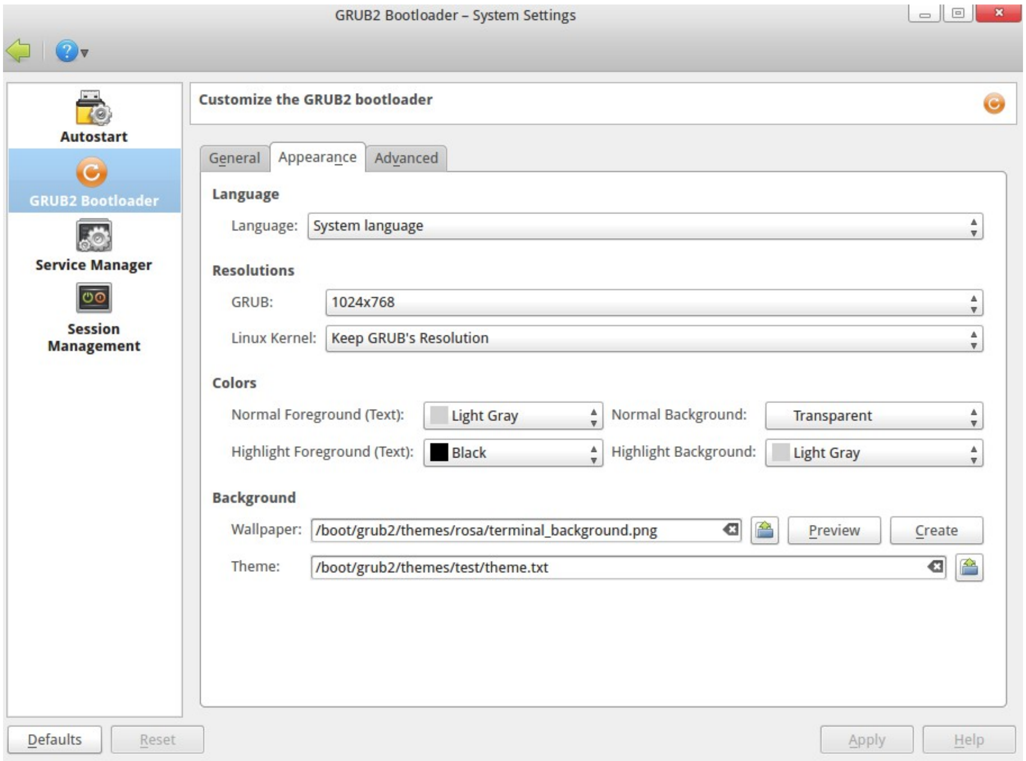
- short log about reasons of assembly failure;
- connecting other personal platforms when building in your own personal platform;
- ability to cache chroot for packaging (this allows to accelerate assembly by cost of some compromising safety and risk to loose possibility to build a package 'from scratch');
- personal repositories will sign packages using signatures generated by ABF in (the same way as packages for the main platforms are signed now);
- obsolete packages deleted after publication of newer ones will be available for download for some time; in particular, this will allow to build ISO images during intensive work with package base;
- ability to use ABF as continuous integration (CI) server;
- registration without invitation!

Stay with us:)

UPSTREAM

GRUB2 manager

Our developers have created the update for the GRUB2 manager (kcm-grub2). New option for selecting bootloader's language has been added. It is useful in cases when user needs to set language of the bootloader not equal to system's language.



Language

Language: System language

Resolutions

GRUB: 1024x768

Linux Kernel: Keep GRUB's Resolution

Colors

Normal Foreground (Text): Light Gray

Normal Background: Transparent

Highlight Foreground (Text): Black

Highlight Background: Light Gray

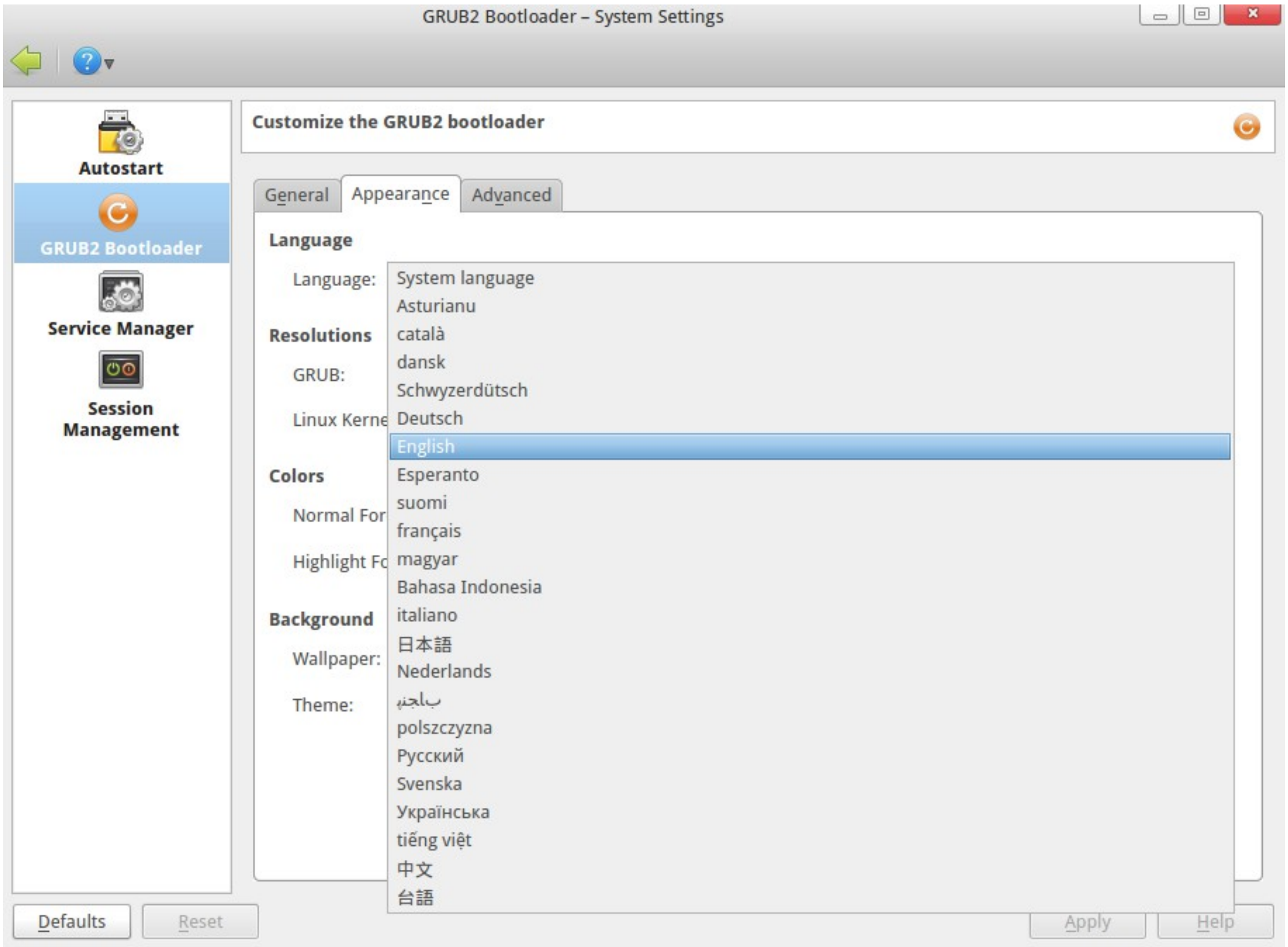
Background

Wallpaper: /boot/grub2/themes/rosa/terminal_background.png

Theme: /boot/grub2/themes/test/theme.txt

"System language" can be selected and then behavior of the GRUB2 manager as well as behavior of the script for renewing bootloader's configuration (update-grub2) will be usual: GRUB2 language will be the same as system's language.

Moreover, specific language can be selected, e.g. "English" or "Русский". Then as result of configuration update (by saving new configuration using GRUB2 manager or calling script update-grub2) we will see GRUB2 menu in selected language.



The patch, which adds option to choose bootloader's language, was sent to the upstream and will be embedded in next version of the program.

ROSA tools in upstream

While creating ROSA, we not only develop and adapt different packages for our distribution, but also design new tools for other developers.

One of such tools is API Sanity Checker, which is aimed to automatically generate tests for C++/C libraries. for its work, the tool requires only header files with declarations of library functions (and all necessary data types). Using this information, API Sanity Checker generates tests that calls every function from the library with proper arguments. Usually these automatically generated tests are used as a template for developing more complex sets (with enumeration of different values of parameters, their combinations, etc).

The tool is absolutely free (the source code can be found here) and can be used by everybody. For example, not so far API Sanity Checker has been integrated in the development cycle of the GammaLib library. As a result of the tool's work, 11 errors were found and fixed in that library (https://cta-jenkins.irap.omp.eu/job/gammalib-sanity/5/changes, https://cta-jenkins.irap.omp.eu/job/gammalib-sanity/6/changes).

We recommend all upstream developers of different C/C++ libraries to follow this successful example. The resources to create test set is minimal, but number of errors found can be really great.

ADMIN

Command-not-found

You can often see a massage "bash: foo: command not found. And you surely want to know why. For example, necessary package is not installed or there is just a misprint. Probably many users become confused after this:

```
$ rpmbuild
bash: rpmbuild: command not found
$ sudo urpmi rpmbuild
No package named rpmbuild
The following packages contain rpmbuild: java-rpmbuild,    rpmbuildupdate
You should use "-a" to use all of them
```

For such cases “command-not-found” tool has been created for ROSA! There are similar tools in other distributions, but there was no such tool for ROSA/Mandriva up to now. But from now on, all you need is to install command-not-found package and open new terminal. Try to type something weird:

```
$ foo
 No command 'foo' found, did you mean:
  Command 'fio' from package 'fio' (contrib)
  Command 'fop' from package 'fop' (main, installed)
  Command 'for' from package 'execline' (contrib)
  Command 'zoo' from package 'zoo' (restricted)
```

Perfect! I just wanted to call “zoo”, I made a mistake. (By the way, notice that “fop” package is already installed, but now we don’t need it)

```
$ zoo
 Command 'zoo' can be found in:
    package 'zoo' (restricted)
 You can install it by typing:
    urpmi zoo
Do you want to install it? (y/N)
```

All you need is to type “y”. You don’t want to receive an offer to install a package? Set environment variable “COMMAND_NOT_FOUND_TURN_OFF_INSTALL_PROMPT=1”, and there will be no stupid questions.

It should be mentioned that when you run the program without the help of TTY, it won’t execute any check, it will just write “command not found” like bash itself. Also whatever command-not-found writes, it will exit with code 127, as bash does in such cases.

One more command-not-found feature is analysis of installed packages. If you type

```
$ ifconfig
 Command 'ifconfig' can be found in:
    package 'net-tools' (main, installed)
File /sbin/ifconfig exists! Check your PATH variable, or call it using an
absolute path
```

Also “cnf” utility presents in command-not-found. It allows doing everything that is described above (in fact, it is executed every time bash fails to run command). In other words, "cnf foo" will give you the same output, as when you type “foo” in console. You can use “cnf” to learn from which package an installed program came.

Probably you have already installed command-not-found. Did you notice that there was another one package installed - command-not-found-data. This package contains data base (JSON format file) from which information is taken while cnf is working. As repositories are always changing, it is necessary to update information of this base from time to time. That is why this package is rebuilt with actual data once a week and comes to you with other updates.

We hope that your work with console will become more pleasant :)

AUTHORS

Anton Chernyshov

Petr Fomichev

Anton Kirilenko

Eugene A. Shatokhin

Vladimir Sharshov

Denis Silakov

Vladimir Testov

Konstantin Kochereskin

All your questions, comments, recalls and wishes about interesting themes,

please, send to the rosa-point@rosalab.ru.