Relax and Recover (rear) Workshop

Gratien D'haese
IT3 Consultants
Some Basics

- **What is Disaster Recovery?**
  The process by which a business function is restored to the normal, steady state after a disaster

- **What is Business Continuity?**
  The way that a business function will operate after a disaster, until such time as the normal, steady state is restored
Business Continuity

Prevention
Risk Management

Preparedness
Business Impact Analysis

Response
Incident Response

Rehearse, maintain and review

Recovery
Recovery Plan

Rear

IT3 Consultants
Relax and Recover Workshop
What is your Disaster Recovery Plan?
Like any other UNIX Operating System, Linux is vulnerable for disaster to strike

The question really is “What shall I do if a disaster strikes?”

Dependent on:

- Hardware failure (e.g. boot disk lost)
- Lost everything (fire, water, earthquake, theft)
- The answer: “Act immediately (with a disaster recovery plan)”
Why are backups not enough?

- Backups of data are necessary!
- Are not enough in case of losing the complete Operating System (OS)!
- Reinstalling the OS from scratch takes hours
- Restoring the backups a few more hours
- Fine-tuning of configurations takes days
- Even months later issues pop up!
- It is absolute necessary to foresee an inventory of hard- and software
Disaster Recovery Plan (DRP)

- DRP addresses need to recover from an emergency with minimum impact to the enterprise
- Protects enterprise from major services failure
- Minimizes risk to enterprise from delays in providing services
- Guarantees reliability of standby systems by testing and simulation
- Minimizes personnel decision-making required during disaster recovery
DRP: main steps

- Risk Analysis
- What is the budget?
- Develop the DRP according
  - Required time to normal operations
  - Establish priorities
  - Inventorying equipment and software
  - Make checklists and test procedures
- Test the DRP (at least on yearly basis)
KISS Principle

- The best way to prepare for a disaster is to avoid the disaster.
- Therefore, look for any potential problems you can find, and correct them.
  - Implement data mirrors or RAID systems
  - Take backups and test restores!
  - Use System Inventory software (e.g. cfg2html)
  - Select a Disaster Recovery Program which takes care of bare metal recovery
Relax and Recover (rear) as DR solution

- Rear is a tool that implements a **DR work-flow** for Linux
- Basically meaning:
  - Modular framework written in Bash
  - Easy to extend to own needs
  - Easy to deploy (set up and forget)
  - Integration for various Linux technologies
  - Integration with various back-up solutions
  - Attempts to make system recovery **as easy as possible**
- Rear runs **on-line** (no downtime to create a DR image)
Introduction to Relax and Recover (rear)

- Proven solution at large enterprise customers
- Rear established as standard solution for Linux disaster recovery in data centers
- Shipping with Fedora, openSUSE and RHEL 6.8 (and >)
- Integrates with many “commercial” backup software solutions, e.g. TSM, DP, NBU, NSR, ...
- Integrates with OS backup software solutions as well, e.g. GNU tar, rsync, bacula, bareos, ...
- Scales well with large amounts of servers
Rear Features

- Focus on disaster recovery and not backup
- Tight integration with common backup software
- Simple full backup integrated
- Complements backup software
  - Backup software: data storage and retrieval
  - Rear: recover the system layout and make it work
  - Rear: use the backup software to restore data
- Methodology: use the best tool for the job
DR Flow – BACKUP and OUTPUT

**BACKUP**
- Basic OS archive (tar, rsync)

**internal**

**BACKUP**
- Basic OS archive (external backup sw)

**external**

**OUTPUT**
- Rescue boot image
Decide on DR strategy

- Which backup mechanism to use?
  - **Internal backup:** GNU tar, rsync
  - **External backup:** bacula, bareos, commercial backup solution
- Where will the backups reside?
  - NFS share, CIFS share, external USB disk, tape, local spare disk, cloud storage, DVD
  - Remote network and/or storage location
- How shall we boot the rescue image?
  - Via DVD (ISO image), tape (OBDR), network (PXE), USB disk
Disaster Recovery - Media

- **Most important: External storage!**
- Bootable media: CD/DVD, USB key, LAN, tape ...
- Media usually combination boot and backup media:
  - Bootable CD/DVD, USB key with backup data on it
  - LAN boot (PXE) with backup data via CIFS, NFS ...
  - Bootable tapes - HP OBDR (CD emulation)
- Separation between boot media and backup data
  - Boot the system from a (small) USB key, CD/DVD or LAN
  - Recover the system with backup software, tar, rsync ...
Disaster Recovery – How It Works

• Store the disk layout
  • Partitioning, LVM and RAID configuration
  • File systems, file system labels ...
  • Boot loader (GRUB, GRUB2, LILO, UEFI)
• Store the files (tgz, rsync, through backup software ...)
• Create bootable rescue media with system configuration (and backup data)

• **Can be done online**
  • No business interruption
  • 100% compatible with original systems hard- and software
Disaster Recovery – Rescue Media

- Create “rescue linux” from running system
- Optimally compatible “tool box”
- Clone the system environment
  - Linux kernel and modules
  - Device driver configuration
  - Network configuration
  - Basic system software and tools
- Operate entirely in RAM (initrd)
Disaster Recovery – In Action

- Boot system from rescue media
- Restore disk layout
  - Create partitions, RAID configuration and LVM
  - Create file systems (mkfs, mkswap)
  - Configure file systems (labels, mount points)
- Restore the backup data
- Restore the boot loader
- Reboot
- Done!
Relax and Recover – Backup Software

- Supported solutions include:
  - CommVault Galaxy; EMC\(^2\) Networker (Legato)
  - IBM Tivoli Storage Manager
  - Symantec NetBackup; HP Data Protector
  - Bacula, Bareos
  - Duplicity
  - Rsync and other “external” methods
  - GNU tar archive on NAS share – CIFS, NFS, NCP ...
- Very transparent integration
- Can be easily extended to support other backup vendors
**rear dump:**
Dumping out configuration and system information

System definition:

- ARCH = Linux-i386
- OS = GNU/Linux
- OS_VENDOR = Fedora
- OS_VENDOR_ARCH = Fedora/i386
- OS_VENDOR_VERSION = Fedora/12

Configuration tree:

- Linux-i386.conf : OK
- GNU/Linux.conf : OK
- Fedora.conf : missing/empty
- Fedora/i386.conf : missing/empty
- Fedora/12.conf : missing/empty
- site.conf : OK
- local.conf : OK
Usage of rear

- Shell scripts are stored under /usr/share/rear
- Scripts are kept together according work-flows
  - mkrescue (only make rescue image)
  - mkbackup (including make rescue image)
  - mkbackuponly (excluding make rescue image)
  - recover (the actual recovery part)
- Easy to incorporate new scripts, e.g. for information gathering of Hard- and Software, or other goodies
Getting started with rear

• Download it from
  • The official tar-balls
    – https://sourceforge.net/projects/rear/files/rear/1.18/
  • The rear-snapshot rpm's build from GitHub
    – http://download.opensuse.org/repositories/Archiving:/Backup:/Rear:/Snapshot/
  • The official source
    – https://github.com/rear/rear/rear
  • The official repo's (Fedora, RHEL, EPEL and SLES)
    – yum install rear
    – zypper install rear
Installation of rear

- E.g. on Fedora 17
  
  # yum install rear

  Installing:
  rear noarch 1.13.0-1.fc17 fedora 327 k
  Installing for dependencies:
  at i686 3.1.13-7.fc17 fedora 61 k
  bc i686 1.06.95-6.fc17 fedora 106 k
  binutils i686 2.22.52.0.1-5.fc17 fedora 3.6 M
  ed i686 1.5-3.fc17 fedora 72 k
  ethtool i686 2:3.2-2.fc17 fedora 93 k
  genisoimage i686 1.1.11-10.fc17 fedora 338 k

  Install 1 Package (+40 Dependent packages)
  Total download size: 21 M
  Installed size: 65 M
  Is this ok [y/N]: y

- We also need syslinux (and to boot on USB: extlinux)
  
  # yum install syslinux

- Install nfs-utils, cifs-utils, rsync if required

- Do not forget openssh(-clients)
Decide on DR strategy

- Which backup mechanism to use?
  - GNU tar, rsync, bacula, bareos, commercial backup program
- Where will the backups reside?
  - NFS share, CIFS share, external USB disk, tape, local (spare) disk
  - Remote network location
- How shall we start the rescue image
  - Via CDROM (ISO image), tape (OBDR), network (PXE), USB disk
Rear Network Integration

- Disaster recovery as part of network infrastructure
  - Backup software: file level backup storage using LAN or SAN
  - Rear: takes care of the system environment
  - Boot rescue media via PXE or virtual CD image
    - No physical media required
  - Very scalable: automated installation of entire disaster recovery data center
    - Rear distribution via company branded RPM
    - Use scheduler to automate the creation of rescue media
The major “backup types” available are:

- **NETFS**: NFS, CIFS, USB, TAPE, ISO, SSHFS, FILE
- RSYNC: rsync method
- REQUESTRESTORE, EXTERNAL
- BACULA, BAREOS, RBME (open source backup software)
- DP, NBU, TSM, NSR, GALAXY[7], SESAM (commercial backup software)
- DUPLICITY (duplicity and/or duply)
BACKUP and OUTPUT methods

- BACKUP variable defines the “backup” method
  - NETFS, RSYNC, DUPLICITY, ....
- BACKUP_URL variable defines the location where to store the backup archive
- OUTPUT variable defines the “output” method
  - ISO, PXE, OBDR, USB
- OUTPUT_URL variable defines the location where to store the output image (ISO image, pxe configuration, extlinux configuration)
BACKUP type NETFS

- OUTPUT=ISO
- BACKUP=NETFS
- OUTPUT=OBDR
- BACKUP=NETFS
- OUTPUT=USB
- BACKUP=NETFS
- OUTPUT=PXE
- BACKUP=NETFS
**BACKUP=NETFS**

**BACKUP_URL** can be

- File type: `BACKUP_URL=file:///directory/`
- NFS type: `BACKUP_URL=nfs://nfs-server/directory/`
- CIFS type: `BACKUP_URL=cifs://samba/directory/`
- USB type: `BACKUP_URL=usb:///dev/disk/by-label/REAR-000`
- ISO type: `BACKUP_URL=iso://backup`
- Tape type: `BACKUP_URL=tape:///dev/nst0`
Backup Program

- BACKUP=NETFS
- /usr/share/rear/conf/default.conf
  - Default: BACKUP_PROG=tar
  - However, BACKUP_PROG=rsync is possible for local attached storage
  - BACKUP_PROG_COMPRESS_OPTIONS="--gzip"
  - BACKUP_PROG_COMPRESS_SUFFIX=".gz"
  - BACKUP_PROG_EXCLUDE=( '/tmp/*' '/dev/shm/*' )
Rear Backup/Recover tests (NETFS)

Speed in seconds

- Recover
- Backup
- Compressed in MB

Options:
- GZIP
- BZIP2
- XZ
Define your settings in /etc/rear/local.conf (or /etc/rear/site.conf)

```bash
# grep -v -E '(^#|^$)' /etc/rear/local.conf
OUTPUT=ISO
```

Add:
```bash
BACKUP=NETFS
BACKUP_URL=nfs://server/path
```

On NFS server backup => /path/$({hostname})/

- Make sure /path is exported and root can write on it
Case 1: store archive within ISO

- `/etc/rear/site.conf` (or `local.conf`) contains
  - `OUTPUT=ISO`
  - `BACKUP=NETFS`
  - `BACKUP_URL=iso://backup`
  - `#ISO_MAX_SIZE=4500` # physical DVD size
  - `ISO_MAX_SIZE=10000` # an absurd size
  - `#ISO_MAX_SIZE=650` # old physical CD size
  - `TMPDIR=/mnt2/tmp` # root permissions required
  - `OUTPUT_URL=nfs://lnx01/vol/linux_images_dr/rear`
  - `EXCLUDE_MOUNTPOINTS=( $ {EXCLUDE_MOUNTPOINT[@]} /mnt /mnt2 /mnt3 )`
Case 2: Save archive on CIFS share

- Put the following in /etc/rear/site.conf (or local.conf)
  - OUTPUT=ISO
  - BACKUP=NETFS
  - BACKUP_URL=cifs://lnx02/homes/backup/cifs
  - BACKUP_OPTIONS="cred=$CONFIG_DIR/.cifs"
  - The file $CONFIG_DIR/.cifs should contain:
    - username=<your username>
    - password=<your password>
  - Remember: OUTPUT_URL=BACKUP_URL if not specified
Case 3: Save archive on CIFS share (encrypted)

- Put the following in `/etc/rear/site.conf` (or `local.conf`)
  - OUTPUT=ISO
  - BACKUP=NETFS
  - BACKUP_URL=cifs://lnx02/homes/backup/cifs
  - BACKUP_OPTIONS="cred=$CONFIG_DIR/.cifs"
  - BACKUP_PROG_CRYPT_ENABLED=1
  - BACKUP_PROG_CRYPT_KEY="my_Secret_pw"
  - Be careful: `chmod 600 /etc/rear/site.conf`
Case 4: Save archive on NFS (by default not encrypted)

- Put the following in /etc/rear/site.conf (or local.conf)
  - OUTPUT=ISO
  - BACKUP=NETFS
  - BACKUP_URL=nfs://lnx02/exports
- If remote NFS is a NAS filer it might be useful to add
  - BACKUP_OPTIONS="nfsvers=3,nolock"
- Enable encryption of archive:
  - BACKUP_PROG_CRYPT_ENABLED=1
  - BACKUP_PROG_CRYPT_KEY="my_Secret_pw"
Case 5: Save archive via SSHFS method

- Put the following in /etc/rear/site.conf (or local.conf)
  - OUTPUT=ISO
  - BACKUP=NETFS
  - BACKUP_URL=sshfs://gd@lnx02/home/gd/backup/sshfs
- FUSE-Filesystem to access remote filesystems via SSH
- Define in /home/gd/.ssh/config an entry:
  - HOST lnx02
    - Port=<22> or <another port>
    - ServerAliveInterval 15
Case 6: usage of incremental backup

- Put the following in /etc/rear/site.conf (or local.conf)
  - BACKUP=NETFS
  - BACKUP_TYPE=incremental
  - FULLBACKUPDAY="Mon"
  - BACKUP_URL=nfs://lnx02/exports
Case 7: RSYNC as backup method

- Put the following in /etc/rear/site.conf (or local.conf)
  - OUTPUT=ISO
  - BACKUP=RSYNC
- Using the rsync+ssh protocol method (transfer encrypted)
  - BACKUP_URL=rsync://gd@lnx02/home/gd/backup/rsync
- Or, by using rsync protocol method (transfer encrypted)
  - BACKUP_URL=rsync://gd@lnx02::/backup
- Make sure you protect server Lnx02 as all files under /home/gd/backup are stored unencrypted
Case 8: Use DUPLICITY as backup method

- Put the following in /etc/rear/site.conf (or local.conf)
  - OUTPUT=ISO
  - BACKUP=DUPLICITY
  - #BACKUP_PROG=duply (auto-detected)
  - TMPDIR=/var/tmp (to define a location with more space)
- GnuPG is a requirement
- Using Duply is supported
  - DUPLY_PROFILE="ubuntu-15-04-backup"
• View system configuration:

    # rear dump
    Relax and Recover 1.13.0 / $Date$
    Dumping out configuration and system information
    This is a 'Linux-x86_64' system, compatible with 'Linux-i386'.
    System definition:

          ARCH = Linux-i386
          OS = GNU/Linux

          OS_MASTER_VENDOR =
          OS_MASTER_VERSION =
          OS_MASTER_VENDOR_ARCH =
          OS_MASTER_VENDOR_VERSION =
          OS_MASTER_VENDOR_VERSION_ARCH =
          OS_VENDOR = Fedora
          OS_VERSION = 16
          OS_VENDOR_ARCH = Fedora/i386
          OS_VENDOR_VERSION = Fedora/16
Usage: rear [-dDsSvV] [-r KERNEL] COMMAND [ARGS...]

Available options:

- **-d** debug mode; log debug messages
- **-D** debugscript mode; log every function call
- **-r KERNEL** kernel version to use; current: '2.6.42.3-2.fc15.i686.PAE'
- **-s** simulation mode; show what scripts rear would include
- **-S** step-by-step mode; acknowledge each script individually
- **-v** verbose mode; show more output
- **-V** version information
Rear help

• Usage: rear [-dDsSvV] [-r KERNEL] COMMAND [--ARGS...]

• List of commands:
  - checklayout check if the disk layout has changed
  - format format and label media for use with rear
  - mkbackup create rescue media and backup system
  - mkbackuponly backup system without creating rescue media
  - mkrescue create rescue media only
  - recover recover the system; only valid during rescue
  - savelayout save the disk layout of the system
  - shell start a bash within rear; development tool
Disaster Recovery in Practice

- Gather system information
- Store the disk layout
  - Partitioning, LVM and RAID configuration
  - File systems, file system labels ...
  - Boot loader (GRUB(2), LILO, ELILO)
- Make a system backup (OS and user data)
- Create boot-able rescue media with system configuration (and optional with backup data)
- All steps are done “online”
Disaster Recovery: rescue media

- Create “rescue linux” from running system
- Optimally compatible “tool box”
- Clone the system environment
  - Linux kernel and modules
  - Device driver configuration
  - Network configuration
  - Basic system software and tools
- Operate entirely in RAM (initrd)
Rear mkrescue

- Will create an ISO image stored as
  - `/var/lib/rear/output/rear-$\text{hostname}.iso`
  - On NFS server as `/path/$\text{hostname}/rear-\$	ext{hostname}.iso`
- Inspect file `/var/lib/rear/layout/disklayout.conf`
- Try to boot from the ISO image into the RESCUE system
  - Use 'dmesg' to check if devices were found
- Create rescue image with backup archive
- Do not forget to browse through the /var/log/rear/rear-$ (hostname).log file for errors
Recovery Process in detail

- Boot system from rescue media
- Restore disk layout
  - Create partitions, RAID configuration and LVM
  - Create file systems (mkfs, mkswap)
  - Configure file systems (labels, mount points)
- Restore the backup data
- Restore the boot loader
- Inspect & Reboot
Recover with rear

- Boot rescue image and select 'recover'

![Recover Screen](image_url)

*Press [Tab] to edit, [F2] for help, [F1] for version info*
Cloning with rear

- Start the recover process: `rear -v recover`
Get your hands dirty?

• We hope you want to dig deeper into rear!

• Getting started:
  
  • Use: `rear -s mkbackup`
    to see the flow of the scripts it will execute
  
  • Depends on BACKUP method, architecture and OS version/brand
  
  • Be careful: `rear -s recover`
    follows a different flow (seems logically, but you must understand the difference)
Where is the code?

- Main script is /usr/sbin/rear
- All the other scripts live under /usr/share/rear
- Documentation is at /usr/share/doc/rear-X.y.z
- **Good news! It's all written in Bash**
Where to put a script?

- mkbackup method: /usr/share/rear/...
  - conf/ - configuration files (/etc/rear/*.conf read last)
  - prep/ - preparation work; checking the environment
  - layout/save/ - save the disk layout /var/lib/rear/layout
  - rescue/ - modules, network, storage,...
  - build/ - populate the initial ramdisk for our rescue image
  - pack/ - create the initrd and copy kernel
  - output/ - create the ISO image and copy to OUTPUT_URL
  - backup/ - make the backup archive to BACKUP_URL
rear -s mkbackup

Relax-and-Recover 1.15 / Git
Using log file: /var/log/secure
Simulation mode activated, Relax-and-Recover base
Source conf/Linux-1386.conf
Source conf/GNU/Linux.conf
Source prep/default/00_remove_workflow_conf.sh
Source prep/default/02_translate_url.sh
Source prep/default/03_translate_tape.sh
Source prep/default/04_check_output_scheme.sh
Source prep/NETFS/default/05_check_NETFS_requirements.sh
Source prep/default/05_check_keep_old_output_copy_var.sh
Source prep/NETFS/default/07_set_backup_archive.sh
Source prep/NETFS/default/09_check_encrypted_backup.sh
Source prep/NETFS/default/15_save_rsync_version.sh
Source prep/GNU/Linux/20_include_agetty.sh
Source prep/NETFS/GNU/Linux/20_selinux_in_use.sh
Source prep/GNU/Linux/21_include_dhclient.sh
Source prep/GNU/Linux/22_include_lvm_tools.sh
Source prep/GNU/Linux/23_include_md_tools.sh
Source prep/GNU/Linux/28_include_systemd.sh
Source prep/GNU/Linux/28_include_vmware_tools.sh
Source prep/GNU/Linux/29_include_drbd.sh
Source prep/GNU/Linux/30_check_backup_and_output_url.sh
Source prep/ISO/default/30_check_iso_dir.sh
Source prep/GNU/Linux/30_include_grub_tools.sh
Source prep/default/31_include_uefi_tools.sh
Source prep/ISO/default/32_check_cdrom_size.sh
Source prep/ISO/GNU/Linux/32_verify_mksisos.sh
Source prep/ISO/Linux-1386/33_find_isolinux.sh
Source prep/NETFS/default/40_automatic_exclude_recreate.sh
Source layout/save/GNU/Linux/10_create_layout_file.sh
Source layout/save/GNU/Linux/20_partition_layout.sh
Source layout/save/GNU/Linux/21 RAID_layout.sh
Source layout/save/GNU/Linux/22_lvm_layout.sh
Source layout/save/GNU/Linux/23_filesystem_layout.sh
Source layout/save/GNU/Linux/24_swaps_layout.sh
Source layout/save/GNU/Linux/25_drbd_layout.sh
Source layout/save/GNU/Linux/26_crypt_layout.sh
Source layout/save/GNU/Linux/27_hpraid_layout.sh
Source layout/save/GNU/Linux/28_multipath_layout.sh
Source layout/save/default/30_list_dependencies.sh
Source layout/save/GNU/Linux/30_save_diskbyid_mappings.sh
Source layout/save/default/31_include_exclude.sh
Source layout/save/default/32_autoexclude.sh
Source layout/save/default/33_remove_exclusions.sh
Source layout/save/default/34_generate_mountpoint_device.sh
Source layout/save/GNU/Linux/35_copy_drbdtab.sh
Source layout/save/GNU/Linux/50_extract_vgcfg.sh
Source layout/save/GNU/Linux/51_current_disk_usage.sh
Source layout/save/default/60_snapshot_files.sh
Source rescue/default/01_merge_skeletons.sh
Source rescue/default/10_hostname.sh
Source rescue/default/20/etc_issue.sh
Source rescue/GNU/Linux/23_storage_and_network_modules.sh
Source rescue/GNU/Linux/24_kernel_modules.sh
Source rescue/GNU/Linux/25_udev.sh
Source rescue/GNU/Linux/26_collect_initrd_modules.sh
Source rescue/GNU/Linux/26_storage_drivers.sh
Source rescue/GNU/Linux/30_dns.sh
Source rescue/GNU/Linux/31_network_devices.sh
Source rescue/GNU/Linux/35_routing.sh
Source rescue/GNU/Linux/39_check_usb_modules.sh
Source rescue/GNU/Linux/40_use_serial_console.sh
Source rescue/GNU/Linux/41_use_xen_console.sh
Source rescue/default/43_prepare_timesync.sh
Source rescue/default/50_ssh.sh
Source rescue/NETFS/default/60_store_NETFS_variables.sh
Source rescue/default/85_save_sysfs_uefi_vars.sh
Source rescue/default/90_clone_users_and_groups.sh
Source rescue/default/91_copy_logfile.sh
Source rescue/GNU/Linux/95_cfg2html.sh
Source rescue/GNU/Linux/96_collect_MC_serviceguard_infos.sh
Where to put a script? (2)

- recover method: /usr/share/rear/...
  - conf/ - read the configuration file + /etc/rear/*.conf
  - setup/ - user defined scripts to run before recover
  - verify/ - to check if a recover is possible at all
  - layout/prepare – recreate the disk layout
  - restore/ - restore the archive from BACKUP_URL
  - finalize/ - do some dirty tricks for disks, grub,...
  - wrapup/ - copy the recover log to /mnt/local/root/
rear -s recover

Relax-and-Recover 1.15 / Git
Using log file: /var/log/rear/rear-fedora19.log
Simulation mode activated, Relax-and-Recover base directory: /usr/share/rear
Source conf/Linux-i386.conf
Source conf/GNU/Linux.conf
Source setup/default/01_pre_recovery_script.sh
Source verify/default/02_cciss_scsi_engage.sh
Source verify/default/02_translate_url.sh
Source verify/default/03_translate_tape.sh
Source verify/default/04_validate_variables.sh
Source verify/NETFS/default/05_check_NETFS_requirements.sh
Source verify/GNU/Linux/05_sane_recovery_check.sh
Source verify/NETFS/default/07_set_backup_archive.sh
Source verify/NETFS/default/08_start_required_daemons.sh
Source verify/NETFS/default/09_set_readonly_options.sh
Source verify/NETFS/default/10_mount_NETFS_path.sh
Source verify/GNU/Linux/23_storage_and_network_modules.sh
Source verify/GNU/Linux/26_recovery_storage_drivers.sh
Source verify/NETFS/default/55_check_backup_archive.sh
Source verify/NETFS/default/60_check_encryption_key.sh
Source layout/prepare/default/01_prepare_files.sh
Source layout/prepare/GNU/Linux/10_include_partition_code.sh
Source layout/prepare/GNU/Linux/11_include_lvm_code.sh
Source layout/prepare/GNU/Linux/12_include_raid_code.sh
Source layout/prepare/GNU/Linux/13_include_filesystem_code.sh
Source layout/prepare/GNU/Linux/14_include_swap_code.sh
Source layout/prepare/GNU/Linux/15_include_drbd_code.sh
Source layout/prepare/GNU/Linux/16_include_luks_code.sh
Source layout/prepare/GNU/Linux/17_include_hpraid_code.sh
Source layout/prepare/default/20_recreate_hpraid.sh
Source layout/prepare/GNU/Linux/21_load_multipath.sh
Source layout/prepare/default/25_compare_disks.sh
Source layout/prepare/default/30_map_disks.sh
Source layout/prepare/default/31_remove_exclusions.sh
Source layout/prepare/default/32_apply_mappings.sh
Source layout/prepare/default/40_autoresize_disks.sh
Source layout/prepare/default/50_confirm_layout.sh
Source layout/prepare/default/52_exclude_components.sh
Source layout/prepare/default/54_generate_device_code.sh
Source layout/prepare/default/55_finalize_script.sh
Source layout/prepare/default/60_show_unprocessed.sh
Source layout/prepare/default/61_exclude_from_restore.sh
Source layout/recreate/default/10_ask_confirmation.sh
Source layout/recreate/default/20_run_script.sh
Source layout/recreate/default/25_verify_mount.sh
Source restore/Fedora/05_copy_dev_files.sh
Source restore/NETFS/default/38_prepare_multiple_isos.sh
Source restore/NETFS/default/40_restore_backup.sh
Source restore/NETFS/default/50_selinux_autorelabel.sh
Source restore/NETFS/Linux-i386/51_selinux_fixfiles_exclude_dirs.sh
Source restore/default/90_create_missing_directories.sh
Source restore/NETFS/default/98_umount_NETFS_dir.sh
Source finalize/default/01_prepare_checks.sh
Source finalize/default/10_populate_dev.sh
Source finalize/GNU/Linux/15_migrate_disk_devices_layout.sh
Source finalize/GNU/Linux/15_migrate_uuid_tags.sh
Source finalize/GNU/Linux/16_rename_diskbyid.sh
Source finalize/Fedora/1386/17_rebuild_initramfs.sh
Source finalize/Linux-i386/21_install_grub.sh
Source finalize/Linux-i386/22_install_grub2.sh
Source finalize/Linux-i386/23_run_efibootmgr.sh
Source finalize/GNU/Linux/30_create_mac_mapping.sh
Source finalize/GNU/Linux/41_migrate_udev_rules.sh
Source finalize/GNU/Linux/42_migrate_network_configuration_files.sh
Source finalize/default/88_check_for_mount_by_id.sh
Source finalize/default/89_finish_checks.sh
Source finalize/default/90_remount_sync.sh
Source wrapup/default/50_post_recovery_script.sh
Source wrapup/default/98_good_bye.sh
Source wrapup/default/99_copy_logfile.sh
Cfg2html: hard- and software details

- When **cfg2html** is installed and in local.conf “USE_CFG2HTML=y” has been set

```
# rear mkrescue
Relax & Recover Version 1.7.24 / 2009-12-09
The preparation phase OK
Physical devices that will be recovered: /dev/sda
Collecting general system information (cfg2html) OK
Creating root FS layout OK
Copy files and directories OK
Copy program files & libraries OK
Copy kernel modules OK
Create initramfs OK
Making ISO image OK
Wrote ISO Image /tmp/ReaR.iso (17M)
The cleanup phase OK
Finished in 488 seconds.
```

- Kernel Interface table
- list of all sockets
- dig hostname
- /etc/hosts
- IP forward
- iptables list chains
- iptables rules
- hosts.allow
- hosts.deny
- /etc/xinetd.d/ section
- DNS & Names
- Email Aliases
- NFSD and BIOD utilization
- XNTP Time Protocol Daemon
- ntp.conf
- FTP Login Shells
- host.conf
- Simple Network Management Protocol (SNMP)
- SNMP Trapdaemon config
- sshd config
- ssh config

- Kernel, Modules and Libraries
  - GRUB Boot Manager
  - Files in /boot
  - Loaded Kernel Modules
  - Available Modules Trees
  - Modules for the ramdisk
  - System boot
  - Kernel commandline
  - libc Version (getconf)
  - libc6 Version
  - libc6 Version (RPM)
  - Run-time link bindings
Example script: sysreqs.sh

- A simple script to save basic system requirements – sysreqs.sh
  - OS version; rear version
  - CPU, memory
  - Disk space requirements
  - IP addresses in use; routes
- Copy sysreqs.sh to a flow, e.g. rescue is a good choice
  - `# cp /tmp/sysreqs.sh /
    /usr/share/rear/rescue/GNU/Linux/96_sysreqs.sh`
Test the script

- # rear -s mkrescue | grep sysreqs
  Source rescue/GNU/Linux/96_sysreqs.sh
- # rear -v mkrescue
- # cat /var/lib/rear/sysreqs/Minimal_System_Requirements.txt
2010-03-12 13:09:07 Using 'blkid' for vol_id
2010-03-12 13:09:07 Relax & Recover Version 1.7.24 / 2009-12-09
2010-03-12 13:09:07 Combining configuration files
2010-03-12 13:09:07 Skipping /etc/rear/os.conf (file not found or empty)
2010-03-12 13:09:07 Skipping /etc/rear/mkrescue.conf (file not found or empty)
2010-03-12 13:09:08 Including conf/Linux-i386.conf
2010-03-12 13:09:08 Including conf/GNU/Linux.conf
2010-03-12 13:09:08 Skipping /usr/share/rear/conf/Fedora.conf (file not found or empty)
2010-03-12 13:09:08 Skipping /usr/share/rear/conf/Fedora/i386.conf (file not found or empty)
2010-03-12 13:09:08 Skipping /usr/share/rear/conf/Fedora/12.conf (file not found or empty)
2010-03-12 13:09:08 Skipping /usr/share/rear/conf/Fedora/12/i386.conf (file not found or empty)
2010-03-12 13:09:08 Including /etc/rear/site.conf
2010-03-12 13:09:08 Including /etc/rear/local.conf
2010-03-12 13:09:08 Creating build area '/tmp/rear.10018'
2010-03-12 13:09:08 Running mkrescue workflow
2010-03-12 13:09:08 Running 'prep' stage
2010-03-12 13:09:08 Including prep/default/01_progress_start.sh
2010-03-12 13:09:08 Including prep/GNU/Linux/28_include_vmware_tools.sh
2010-03-12 13:09:08 Including prep/ISO/default/30_check_iso_dir.sh
2010-03-12 13:09:08 Including prep/ISO/default/32_check_cdrom_size.sh
2010-03-12 13:09:08 ISO Directory '/tmp' [/dev/mapper/VolGroup-lv_root] has 3087 MB free space
2010-03-12 13:09:08 Including prep/ISO/GNU/Linux/32_verify_mksvos.sh
2010-03-12 13:09:08 Using '/usr/bin/mksvos' to create ISO images
2010-03-12 13:09:08 Including prep/ISO/Linux-i386/33_find_isolinux.sh
2010-03-12 13:09:18 Including prep/default/99_progress_stop.sh
2010-03-12 13:09:18 Finished running 'prep' stage in 10 seconds

Done with: Ending Paddrack Block(s) 150
Max brk space used 0
8427 extents written (16 MB)
2010-03-12 13:10:35 Including output/default/95_email_result_files.sh
2010-03-12 13:10:35 Finished running 'output' stage in 1 seconds
2010-03-12 13:10:35 Running 'cleanup' stage
2010-03-12 13:10:35 Including cleanup/default/01_progress_start.sh
2010-03-12 13:10:35 Including cleanup/default/99_progress_stop.sh
2010-03-12 13:10:35 Finished running 'cleanup' stage in 0 seconds
2010-03-12 13:10:35 Finished running mkrescue workflow
2010-03-12 13:10:35 Removing build area '/tmp/rear.10018'
2010-03-12 13:10:35 End of program reached
Relax-and-Recover Status

- Stable software
  - i386 and x86_64 are well tested
  - ia64 and ppc, ppc64, ppc64le less tested
- Released as RPM, TAR, DEB
- Rear ships with
  - SUSE Linux Enterprise HA extension 11 SPx
  - OpenSUSE and Fedora
- Support available (community and/or commercial)
- Open for patch submissions by rear community
Relax-and-Recover is a setup-and-forget *Linux bare metal disaster recovery* solution. It is easy to set up and requires no maintenance so there is no excuse for not using it.

Learn more about Relax-and-Recover from the selected usage scenarios below:

**Home user**
- recover from a broken hard disk using a [bootable USB stick](#)
- recover a broken system from your [bootloader](#)

**Enterprise user**
- collect small ISO images on a [central server](#)
- integrate with your [backup solution](#)
- integrate with your [monitoring solution](#)

Or watch a 4-minute complete backup and restore demo. Real time, no cheating!
https://github.com/rear/rear/issues
What is missing?

- Most customers miss a central component for ReaR that
  - Gathers information about rear
  - Stores rear boot images
  - Initiates Disaster Recovery
  - Makes rear information available for 3rd party

- Disaster Recovery Linux Manager (DRLM)
  - http://drlm.org/
  - Open Source software from brainupdaters.net
Relax and Recover (rear)

Great Tool for your Disaster Recovery Team
Contacts

Web-site: http://relax-and-recover.org/
GitHub: https://github.com/rear/rear
Mailing list: rear-users@lists.relax-and-recover.org

Rear Maintainer - Gratien D'haese - gratien.dhaese@it3.be
Rear Maintainer - Schlomo Schapiro - schlomo@schapiro.org
Rear Developer – Johannes Meixner – jsmeix@suse.com
Rear Developer – Jeroen Hoekx - jeroen.hoekx@hamok.be
Rear Developer – Dag Wieers - dag@wieers.com