

Офис LibreOffice LibreOffice  
Электронные книги EBReader EBReader

## Графика

*mup* программы KDE XFCE

Графический редактор Gimp Gimp

Просмотр изображений Gwenview Geeqie

Менеджер изображений digiKam Shotwell

Сканирование изображений XSane XSane

## Мультимедиа

*mup* программы KDE XFCE

Видео-проигрыватель SMPlayer SMPPlayer

Аудио-проигрыватель Amarok Clementine

Запись дисков K3B Xfbutil

Видеомонтаж Kdenlive

## Система

*mup* программы KDE XFCE

Файловый менеджер Dolphin Thunar

Редактор разделов KDE Partition Manager GParted

## Установка с загрузочного CD-диска

- Установка с загрузочного CD-диска
- Запись образа на диск
- growisofs
- К3В
- Опции загрузки
- Разбиение диска

Если ваш компьютер имеет CD/DVD-привод, то вы можете воспользоваться установкой с загрузочного Live CD/DVD. Загрузочный диск - это точный образ установляемой системы, и благодаря ему вы можете оценить работу дистрибутива непосредственно перед установкой на ваш ПК.

Загрузить образ LiveCD можно ddrescue, воспользовавшись BitTorrent клиентом либо загрузив файл напрямую с http://fr.zerkal.

## Запись образа на диск

В зависимости от объёма данных для записи вам может понадобиться CD или DVD.

Записывать загруженный образ нужно в «сыром» режиме. Использование программ growisofs и K3B, входящих в комплект Calculate Linux Desktop, описано ниже.

## growisofs

При использовании growisofs просто выполните:

```
growisofs -Z /dev/cdrom=/usr/calculate/share/linux/cld-10.0-i686.iso
```

В приведенном примере ISO-файл расположен в директории `/usr/calculate/share/linux`.

# Документация Calculate Linux

Различные файловые системы имеют свои преимущества и недостатки. Мы можем оставить здесь лишь рекомендацию на основе личного опыта. В качестве файловой системы для корневого раздела хорошо зарекомендовала себя "ext4", а вот для хранения файлов, благодаря активному использованию кэша, идеальной, на наш взгляд, является "xfs".

## Разбиение диска для Linux-десктона

Настройка [Calculate Linux Desktop](#) для работы в качестве клиента сервера [CDS](#), мы рекомендуем разбить жесткий диск следующим образом:

- ```
/dev/sda1 swap  
/dev/sda2 10Gb Linux (/, ext4)  
/dev/sda3 10Gb Linux (/, ext4)  
/dev/sda4 Extended  
/dev/sda5 Linux (/var/calculate, ext4)
```

Обратите внимание, что свободный раздел монтируется в `/var/calculate`. Таким образом, настройки подключения к серверу сохраняются в отдельном разделе, упрощая переустановку системы. Содержимое раздела `/home` не будет теряться при переустановке, т.к. после входа в домен CDS, директория `/home` располагается в `/var/calculate/home` (монтируется через `bind`). Сохранять данные имеет смысл только для кэширования.

## Переустановка системы

При любом разбиении диска архивы с обновлениями удобно хранить в отдельном разделе - пусть это будет, например, `/home` (если он вынесен на отдельный раздел). Подключить такой диск можно, используя следующую запись в `/etc/fstab`:

```
/home/calculate /usr/calculate/share none bind 0 0
```

## 3. Работа с Portage

### 1. Введение в Portage

### 2. USB-флэши

### 3. Возможности Portage

### 4. Переменные среды

### 5. Файлы и каталоги

### 6. Настройка с помощью переменных

### 7. Сечение ветвей программного обеспечения

### 8. Дополнительные средства Portage

### 9. Отступление от официального дерева

### 10. Использование ebuild

## 4. Утилиты Calculate

### 1. Графический клиент утилит Calculate

### 2. Консольный клиент утилит Calculate

### 3. Сервер-утилит Calculate

### 4. Шаблоны утилит Calculate

### 5. Переменные шаблонов

### 6. Хранение настроек профиля пользователя

### 7. Обновление системы cl-update

### 8. Смена профилей системы cl-update-profile

### 9. Сборка системы

### 10. Calculate API

## 5. Настройка сервера

### 1. Перенос учётных записей пользователей в Calculate Directory Server

### 2. Настройка LDAP сервера

### 3. Использование LDAP сервера для хранения учётных записей

### 4. Настройка Samba сервера

## Программное обеспечение

Состав программного обеспечения подобран с учётом трёх правил:

1. программа должна быть востребована,
2. иметь единый стиль с оконным менеджером,
3. не дублировать функционал.

Вы можете присыпать свои пожелания к изменению программы в [список рассылки](#) calculate-user@ru.

## Calculate Linux Desktop 14.16.2

## Интернет

[mutn](#) программы KDE XFCE

Браузер [Chromium Chromium](#)

Почта [KMail Claws Mail](#)

Jabber/ICQ [Kopete Pidgin](#)

IRC [Konversation HexChat](#)

Twitter [Chocok](#)

RSS-reader [Akregator Liferea](#)

Torrent [KGet Deluge](#)

## Офис

[mutn](#) программы KDE XFCE

```
/dev/sda3 10-20Gb Linux (/)  
/dev/sda4 Extended  
/dev/sda5 Linux (/home)  
... (другие диски и разделы)
```

Если система загружена с раздела "sda2", то при переустановке системы установщик предложит "sda3", и наоборот. Программа установки помнит, с какого раздела производилась установка системы, и в дальнейшем предложит выполнить обновление в предыдущий раздел.

В настольной версии системы "swap"-раздел может отсутствовать вовсе при достаточном количестве оперативной памяти (2 Гб и выше). В этом случае ядро практически не будет "свопить", интенсивнее высвобождая память.

## Настройки

- **зацикленность** - пользовательские данные, размещенные на отдельном разделе, никогда не пострадают при переустановке системы;
- **свободное место** - у вас всегда будет свободное место на диске, т.к. вероятность, что какому-то разделу (например, /boot) не хватит места, сведена к минимуму;
- **обновление системы** - вы сможете выполнять обновление, продолжая работать в системе;
- **надёжность** - вы всегда сможете загрузиться в предыдущую систему, если новая по какой-либо причине будет работать нестабильно.

## Настройки

Для подключения дополнительных разделов к системе используйте файл /etc/fstab.

Для монтирования раздела в директорию /home достаточно в конец файла /etc/fstab вписать строку примерно следующего содержания:

```
/dev/sda5 /home ext4 noatime 0 0
```

В приведенном примере раздел sda5 с файловой системой ext4 монтируется в директорию /home.

Для подключения раздела выполните:

```
mount /home
```

Протисанный таким образом раздел будет автоматически монтироваться во время загрузки системы. При переустановке системы *Calculate* будет переносить точки монтирования из файла /etc/fstab в новую систему.

## Разбиение диска для сервера

Для разбиения диска под *Calculate Directory Server* мы предлагаем использовать следующую схему:

```
/dev/sda1 Swap  
/dev/sda2 20Gb Linux (/, ext4)  
/dev/sda3 20Gb Linux (/, ext4)  
/dev/sda4 Extended  
/dev/sda5 Linux (/var/calculate/server-data, xfs)
```

Размер диска подкачки (swap) следует выбрать произвольно, исходя из объема оперативной памяти и предполагаемой нагрузки. Как правило, рекомендуют использовать в 2 раза больше, чем объем оперативной памяти.

По мере необходимости вы можете расширить ёмкость разделов, подключив дополнительные диски:

```
/dev/sdb1 Linux (/var/calculate/server-data, xfs)  
/dev/sdc1 Linux (/var/calculate/server-data/samba/share, xfs)
```

5. Настройка прав доступа ACL
6. Настройка FTP сервера
7. Настройка логгера сервера
8. Настройка почтового сервера
9. Настройка Proxy сервера
- 10.Настройка DNS сервера
- 11.Настройка DHCP сервера
- 12.Настройка PXE
- 13.Настройка репликации Samba серверов
- 14.Настройка репликации почтовых серверов
- 15.Настройка сервера шлюза
- 16.Настройка Asterisk сервера
- 17.Обзор структуры LDAP сервера
- 18.Управление клиентскими машинами
- 19.Система виртуализации OEMU
- 20.Настройка gitolite
- 21.Резервное копирование
6. **Настройка рабочей станции**
  1. Переход на Linux
  2. Подключение к серверу каталогов
  3. Хранение пользовательских настроек
  4. Установка шаблонов пользовательского окружения
  5. Реализация криптизации NSS для доменной машины
7. **Настройка сети в Calculate**
  1. Настройка сети Gentoo-way
  2. Настройка сети Calculate-way
8. **Настройка оборудования**
  1. Настройка звука
  2. Настройка софтового модема
  3. Настройка TV тюнера AverTV 305/307
  4. Настройка сканера Epson Perfection 1670
  5. Настройка Wake-on-Lan
9. **Справка по основным командам Gentoo/Calculate**
  1. portage - система управления пакетами в Gentoo
  2. eix - набор утилит для поиска ебилдов по дереву Portage и получения информации о них, в том числе работа с локальными настройками, внешними оверлейами, версиями пакетов и др.
  3. layout - утилита для управления оверлейами Gentoo
  4. qrepmic - система управления службами системы, запуском и завершением работы хоста
  5. portage-utils - набор легких и быстрых утилит, написанных на C, для извлечения информации о пакетах в Portage
  6. gentoolkit - набор скриптов для администрирования систем, работающих на Gentoo
  7. gentoolkit-dev - набор скриптов в помощь разработчикам под Gentoo
10. **Руководства**
  1. Описание IRC
  2. Gitраспределённая система управления версиями файлов, правка последнего коммита
  3. Руководство по оформлению программ на Python
  4. Программный Raid
  5. Руководство по Iptables
  6. Перекодировка пр3\_тегов

Содержание этого документа распространяется на условиях [Creative Commons - Attribution / Share Alike](#) лицензии.

# 1. Установка Calculate

1. Об установке Calculate Linux
2. Краткое руководство по установке
3. Установка на жёсткий диск
4. Установка на Flash
5. Аппаратные требования
6. Структура FTP-зеркала
7. Разбиение диска
8. Программное обеспечение
9. Установка с загрузочного CD-диска
10. Чем заняться дальше?

## Об установке Calculate Linux

### Добро пожаловать!

Спасибо за Ваш интерес к Calculate Linux. Мы постоянно работаем над удобством работы с системой и надеемся, что работа с Calculate Linux доставит Вам истинное удовольствие.

Подробнее о проекте Calculate Linux.

### Как организована установка?

Calculate Linux распространяется в виде загрузочного Stage4-образа. Во время загрузки с LiveCD и установки системы на компьютер утилиты Calculate настраивают систему при помощи шаблонов. Таким образом, на LiveCD вы имеете точную копию установленной системы.

Программе установки, запущенной из командной строки, могут быть переданы все опции установки. Существующие пользователи будут перенесены, установка из Stage4 происходит путем распаковки. Тем самым время на установку сокращается до минимума.

Вы легко можете изменить настройки и состав программ, создав изменённый ISO-образ или обновив squashfs-файл на USB Flash. В этом случае установка будет выполняться со всеми внешними изменениями. Для этого воспользуйтесь следующим руководством.

### Какие варианты установки существуют?

Установить систему Calculate Linux вы можете одним из перечисленных способов:

- графическим клиентом `cl-console-gui`;
  - консольным клиентом `cl-console`;
  - напрямую сервером утилит `cl-core`.
- Система может быть установлена из squashfs-образа, если Вы загрузились с liveCD или USB Flash, либо из ISO-файла, размещенного в директории `/var/calculatē/1inux` или `/var/calculatē/remote/1inux`. Во втором случае Вы можете установить любую версию дистрибутива поддерживаемой архитектуры.

Сервер утилит, консольный и графический клиенты входят в состав Calculate Linux, но могут быть установлены из оверлея Calculate в любом Gentoо-совместимом дистрибутиве.

### Появились загруднения?

Если у вас есть вопросы, не стесняйтесь задавать их в IRC-чате, посыпанном Calculate Linux. Канал `#calculatē`ти находится на сервере Freenode, а в качестве IRC клиента Вы можете использовать программы `konversation` или `hexchat`, которые входят в поставку Calculate Linux Desktop.

Вы также можете обратиться к нашему сообществу через спикки рассылки. Найти единомышленников можно в социальной сети ВКонтакте, Facebook или Twitter.

Память 512 Мб

Дисковое пространство 7 Гб

Пространство подкачки как правило, не меньше количества оперативной памяти

## Calculate Linux Desktop XFCE

Центральный процессор i686 или новее (Intel не ниже Pentium Pro, AMD не ниже Athlon)

Память 256 Мб

Дисковое пространство 5 Гб

Пространство подкачки как правило, не меньше количества оперативной памяти

Процессор подкачки как правило, не меньше количества оперативной памяти

## Calculate Linux Scratch

Центральный процессор i686 или новее (Intel не ниже Pentium Pro, AMD не ниже Athlon)

Память 128 Мб

Дисковое пространство 3 Гб

Пространство подкачки как правило, не меньше количества оперативной памяти

Требования к диску и памяти могут возрасти в зависимости от типа выполняемых задач.

## Структура FTP-зеркала

FTP-зеркало с дистрибутивами Calculate Linux выглядит примерно следующим образом:  
`ftp://ftp.calculate-linux.org/calculate/15.12`

где: 'ftp.calculate-linux.org/calculate' - адрес FTP-сервера, 'release' - директория с релизами, '15.12' - версия системы.

Структура файлов в директории выглядит следующим образом:

```
cld-15.12-x86_64.iso  
cld-15.12-x86_64.iso.torrent  
cld-15.12-x86_64.list  
md5sum.txt  
sha1sum.txt
```

где:

- `_cld-15.12-x86_64.iso` - загруженный Live USB образ системы, с возможностью установки;
- `_cld-15.12-x86_64.iso.torrent` - BitTorrent файл - предпочтительный способ для быстрой загрузки системы и дополнений;
- `_cld-15.12-x86_64.list` - список пакетов, входящих в дистрибутив;
- `md5sum.txt` и `sha1sum.txt` - контрольные суммы файлов.

Примечание:

Для 32-битной архитектуры в названиях пакетов будет обозначение `1686` вместо `x86_64`.

## Разбиение диска

### Общая схема

Calculate Linux можно переустановить без каких-либо дополнительных параметров при следующем разбиении диска:

```
/dev/sda1 swap  
/dev/sda2 10-20Gb Linux (/)
```

## Установка системы на Flash консольным клиентом

Установка системы на Flash консольным клиентом ничем не отличается от установки системы на жёсткий диск, за исключением меньшего количества используемых параметров.

Для установки раздела для установки используется ключ `-d`, `- - disk`.

Для указания образа дистрибутива для установки используется ключ `--iso`. Для установки системы не обязательно указывать образ дистрибутива при нахождении его в директориях по умолчанию:

Например:

```
cl-console - -method install - d /dev/sdb1
```

Пример установки системы на Flash с указанием файла образа дистрибутива:

```
cl-console - -method install - d /dev/sdb1 - -iso /var/calculate/linux/cld-20110904-x86_64.iso
```

## Установка системы на Flash с помощью сервера утилит

Установка системы на Flash с помощью сервера утилит ничем не отличается от установки системы на жёсткий диск, за исключением меньшего количества используемых параметров.

Для установки раздела для установки используется ключ `-d`, `- - disk`.

Для указания образа дистрибутива для установки используется ключ `--iso`. Для установки системы не обязательно указывать образ дистрибутива при нахождении его в директориях по умолчанию:

```
/var/calculate/linux /var/calculate/remote/linux.
```

Для установки используется команда `cl-core --method install`:

```
cl-core - -method install - d /dev/sdb1
```

или символическая ссылка на неё `cl-install`, например

```
cl-install - d /dev/sdb1 - -iso /var/calculate/linux/cld-20110904-x86_64.iso
```

## Краткое руководство по установке

### Благодарим за использование Calculate Linux!

Мы постарались сделать для вас максимально удобную для работы систему, используя оригинальный установщик, переработанный интерфейс, шаблоны настроек, утилиты Calculate и Gentoo Portage. Дистрибутив распространяется в виде установочного образа, содержащего лучшее программное обеспечение. Большая часть программ имеет свободную лицензию, позволяющую не только устанавливать и распространять, но и модифицировать исходный код. Используемые сокращенные названия дистрибутивов:

- CLD - Calculate Linux Desktop KDE
- CLDM - Calculate Linux Desktop MATE
- CLDX - Calculate Linux Desktop XFCE
- CLS - Calculate Linux Scratch
- CMC - Calculate Media Center
- CDS - Calculate Directory Server
- CSS - Calculate Scratch Server

Для получения прав администратора системы, запущенной с LiveCD, либо находясь в графическом режиме, выполните `su` в виртуальном терминале, либо перейдите в одну из текстовых консолей нажатием `Ctrl+Alt+F1-F1-8!`. Доступ к рабочему столу CLD, CLDM и CLDX выполняется пользователем `guest` с паролем `guest`.

### Настройка сети

Настройка сети в Calculate Linux осуществляется с помощью сервера утилит Calculate. Как и все действия сервера утилит, настройку сети можно выполнить несколькими способами:

- используя графический клиент;
- используя консольный клиент;
- используя сервер утилит.

Подробнее см. в разделе [Настройка сети](#).

## Аппаратные требования

### Минимальные требования к аппаратному обеспечению

Проверьте, соответствует ли ваше аппаратное обеспечение требованиям выбранной версии системы.

У вас может оказаться компьютер меньше мощности, чем рекомендовано в таблице ниже. При этом установка вполне возможна, однако есть вероятность остаться неудовлетворённым работой системы - поэтому следует учитывать наши рекомендации.

Возможно запускать графическое окружение рабочего стола на старых или дешевых машинах. В этом случае рекомендуется установить менеджер окон, менее требовательный к ресурсам по сравнению с KDE, - например, XFCE.

### Calculate Directory Server

Центральный процессор i686 или новее (Intel не ниже Pentium Pro, AMD не ниже Athlon)

Память 256 Мб

Дисковое пространство 4 Гб

Пространство подкачки как правило, не меньше количества оперативной памяти

### Calculate Linux Desktop KDE

Центральный процессор i686 или новее (Intel не ниже Pentium Pro, AMD не ниже Athlon)

Примечание: для получения прав пользователя `root` в консоли используйте команду `su` либо `sudo`. Если вы никогда не использовали Linux, то вам понадобится немного времени на то, чтобы привыкнуть к другому наименованию разделов. В Linux разделы обозначаются как `sda1`, `sda2`, ... вместо привычных `C:\`, `D:\`, ... Вы также можете воспользоваться уже готовым разделом либо создать его из Windows. Чтобы правильно определить выбранный раздел в Linux, запомните очертность его расположения и размер. Как правило, диску `C:\` соответствует `sda1`, диску `D:\` - `sda2`.

## Варианты установки

- Установить систему Calculate Linux вы можете одним из перечисленных способов:
- графическим клиентом **cl-console-gui**,
- консольным клиентом **cl-console**,
- напрямую сервером утилиты **cl-core**.

Система может быть установлена из squashfs-образа, если Вы загрузились с liveCD или USB Flash, либо из ISO-файла, размещенного в директории `/var/calculate/remote/Linux` или `/var/calculate/Linux` или `/var/calculate/remote/Linux`. Во втором случае Вы можете установить любую версию дистрибутива поддерживаемой архитектуры.

Сервер утилит, консольный и графический клиенты входят в состав Calculate Linux, но могут быть установлены из оверлея Calculate в любом Gentoо-совместимом дистрибутиве. Подробное описание установки смотрите в соответствующих разделах:

- Установка на жёсткий диск
- Установка на Flash

## Первый запуск

Если вы не указали других пользователей, после установки CLD, CLDM и CLDX в системе будут заведены две учётные записи, `root` и `guest`. Доступ к графическому сеансу может получить любой пользователь, кроме `root`. По умолчанию у пользователя `guest` установлен пароль `guest`.

После установки в CLS нет графического приглашения к вводу пароля. Для запуска оконного менеджера выполните:

```
startx
```

Для получения прав пользователя `root` используйте команду `su`. Добавление новых пользователей подробно описано [здесь](#).

По умолчанию вы можете зайти в систему удалённо (по протоколу ssh) только как пользователь `root`. В файле `/etc/ssh/sshd_config` в значение параметра `AllowUsers` можно добавить другие учётные записи. Мы рекомендуем убрать права удалённого доступа к системе для пользователя `root`.

## Обновление

Calculate Linux использует модель обновлений *rolling release*. Вы можете обновлять систему практически неограниченное количество раз, используя утилиту обновления системы `cl-update`.

Для выполнения синхронизации списка пакетов и обновления программ выполните:

```
cl-update
```

Если вы хотите только обновить список пакетов, то выполните:

```
cl-update - sync-only
```

После этого вы можете установить новые программы при помощи менеджера пакетов `emerge`. Краткая справка приведена [здесь](#). Ознакомьтесь также с [Руководством по обновлению системы](#).

## Помощь

Если установка системы вызвала сложности или если вы хотите поделиться своим впечатлением, зайдите на IRC канал `#calculate-ru` (сервер FreeNode) сообщества пользователей Calculate Linux. Для этого достаточно воспользоваться иконкой `Hexchat` на вашем рабочем столе.

Сайт проекта: <http://www.calculate-linux.ru/>  
IRC чат: <http://www.calculate-linux.ru/irc>

Подключите флешку к вашему компьютеру. Для определения имени устройства вашей флешки выполните в консоли с правами `root`:

```
fdisk -l
```

- Например, если ваш флеш-диск определился как устройство `/dev/sdb`, установка системы на него производится командой:

```
cl-install -d /dev/sdb1
```

## Установка системы на Flash графическим клиентом

Для установки системы на Flash с помощью графического клиента утилит `Calculate` выполните одно из действий:

- запустите `cl-console-gui` и в категории **Установка** выберите пункт **Установка на Flash**;
- запустите `cl-console-gui --method install_flash` для открытия окна с установкой системы. Для установки Calculate Linux на Flash необходимо указать лишь небольшое количество параметров.
- Диск для установки. Как правило, определяется автоматически, иначе необходимо выбрать его из выпадающего списка;
- Установочный образ;
- Разметку диска, которая является необязательным параметром.

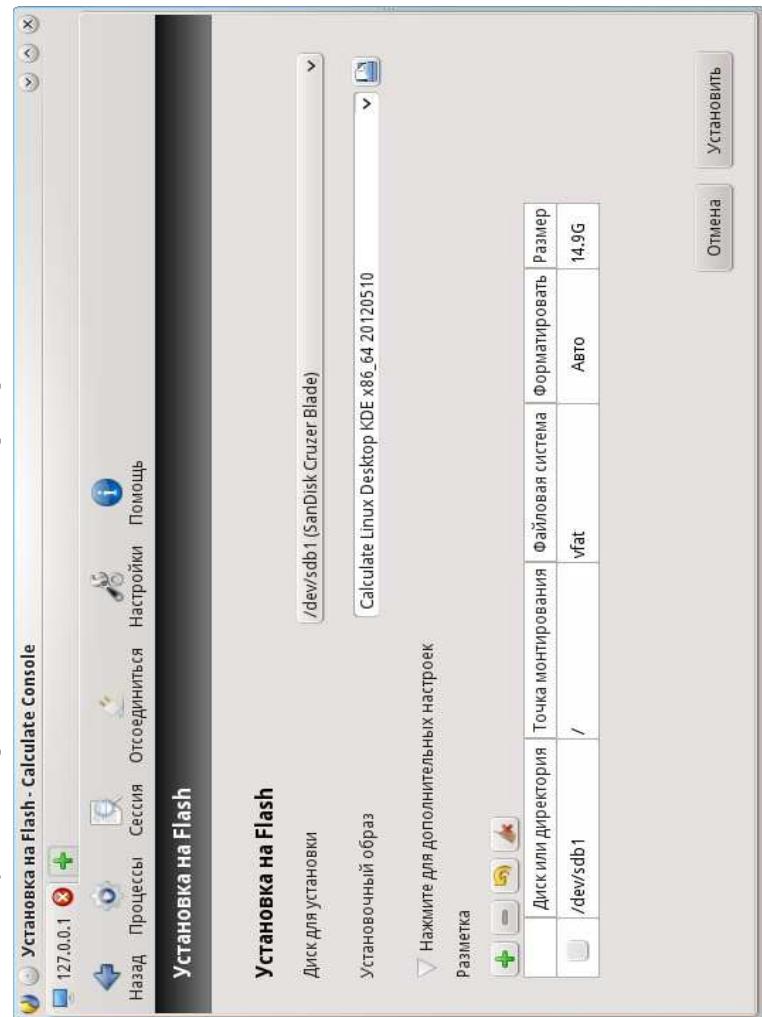


Рис. 1 Установка Calculate Linux на Flash

```
cl-install -netconf openrc -iface eth0:192.168.1.47:24 --hostname  
iiivanov.company.ru --route default:192.168.1.1:eth0:192.168.1.47
```

Данной командой в качестве менеджера сети будет установлен openrc, для сетевого интерфейса eth0 отключится DHCP, установится IP-адрес 192.168.1.47 и маска сети 255.255.255.0. Так же будет задано имя хоста iiivanov.company.ru и маршрутизация для сетевого интерфейса eth0 по умолчанию через 192.168.1.1.

#### Пользователи

Доступные параметры:

- -u USERS, --users USERS - добавить пользователя в установленную систему
- -A USER, --autologin USER - указать пользователя для автозапуска в установленную систему
- @-C [ON/OFF] --crypt-home [ON/OFF] - шифровать пользовательские профили

Примеры:

```
cl-install -u root -u user1
```

В установленной системе будут пользователи root, guest и user1. Автозапуск выполниться не будет.

```
cl-install -u root -u guest -A user1
```

В установленной системе будут пользователи root, guest и user1. Автозапуск будет установлен для пользователя user1.

#### Видео

Доступные параметры:

- --video VIDEODRV - установить видео драйвер
- --composite [ON/OFF] - установить композит
- -X <width>x<height>-установить разрешение Xorg
- -fb <width>x<height>-установить разрешение фреймбуфера
- -grub-terminal TERMINAL - установить grub-терминал (console,gfxterm)

Пример:

```
cl-install --video nouveau -X 1920x1080 -fb 1920x1080-32 -composite
```

В установленной системе будет использованы драйвер nouveau, разрешение Xorg 1920x1080, разрешение фреймбуфера 1920x1080-32 и композит.

#### Помощь

Если установка системы вызвала сложности или вы хотите поделиться своим впечатлением, зайдите на IRC канал #calculate-ru (сервер FreeNode) сообщество пользователей Calculate Linux. Для этого кликните на иконку "Сообщество Calculate Linux" на вашем рабочем столе.

## Установка на Flash

- Установка на Flash
- Установка системы на Flash графическим клиентом
- Установка системы на Flash консольным клиентом
- Установка системы на Flash с помощью сервера утилит

Прежде чем приступить к установке, сделайте резервную копию своих данных на флешке.

## Установка Calculate Linux на жёсткий диск

- Установка Calculate Linux на жёсткий диск
- Графическая установка системы
- Язык и локаль
  - Выбор дистрибутива
  - Распределение места на диске
  - Точки монтирования
  - Сетевые настройки
  - Пользователи
  - Видео
  - Начать установку
  - Установка системы из консоли
- Помощь

### Графическая установка системы

Для установки системы с помощью [графического клиента](#) утилита Calculate выполните одно из действий:

- запустите cl-console-gui и в категории **Установка системы**:
- запустите cl-console-gui --method install для открытия отдельного окна с установкой системы.

Необходимые ярлыки запуска программы можно найти на рабочем столе LiveDVD или в меню.

#### Установка Calculate Linux состоит из следующих шагов:

1. Язык и локаль;
2. Выбор дистрибутива;
3. Распределение места на диске;
4. Точки монтирования;
5. Сетевые настройки;
6. Пользователи;
7. Настройка видео;
8. Установка на жёсткий диск.

Пользователь может перемещаться по шагам с помощью кнопок **Назад** и **Далее**, а также выбрав нужный пункт в левом меню. Во втором случае, если пропущен обязательный шаг или где-то была допущена ошибка, то перед установкой пользователю будет предложено исправить ошибку.

#### Язык и локаль

Первым пунктом установки является выбор языка и часового пояса. Выберите необходимые параметры из выпадающих списков.

Найти единомышленников можно и в социальных сетях:

[ВКонтакте](#), [Facebook](#), [Google+](#), [Одноклассники](#) или [Twitter](#).

Приятной работы!

Команда разработчиков Calculate Linux.

## Установка системы - Calculate Console

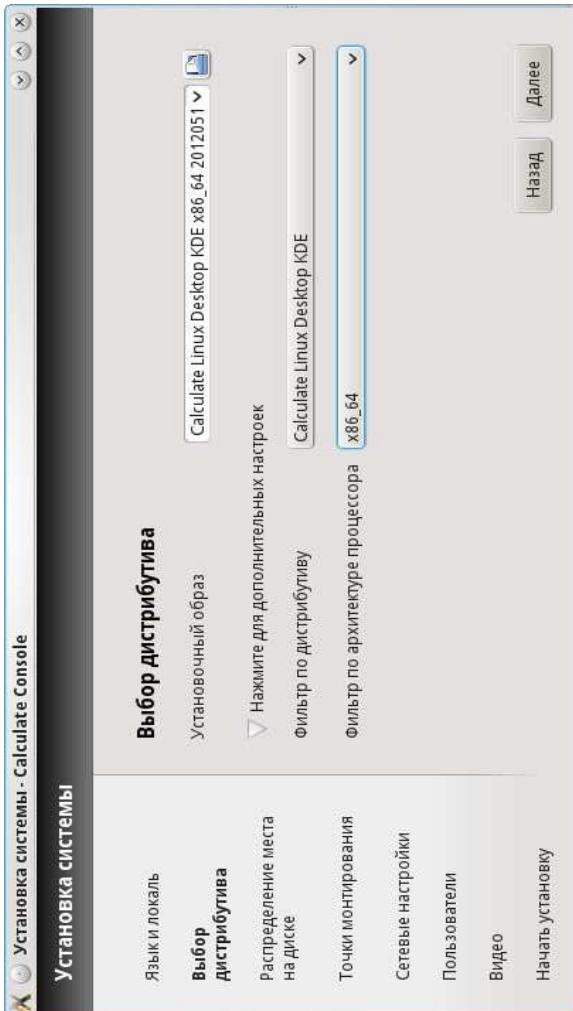


Рис. 1. Настройка языка и локали

## Выбор дистрибутива

По умолчанию для установки будет использоваться дистрибутив, с которого был записан ваш DVD или USB-Flash. Программа установки сканирует директории `/var/calculate/linux` и `/var/calculate/elemente/linux` в случае нахождения там ISO образов с Calculate Linux, отобразит их в списке Установочный образ.

## Установка системы



home - подключить `/var/calculate/home` к `/home`;  
boot - использовать отдельный boot раздел;  
grub - создать bios\_grub раздел;  
uefi - использовать UEFI загрузчик;  
lvm - использовать LVM.

Примеры:

```
cl-install -autopartition on -auto-scheme swap,root,data,home -D /dev/sda
```

Для установки будет использован диск `/dev/sda` с автозаметкой, будут использованы разделы подкачки, `/var/calculate`, дополнительный корневой раздел и подключен `/var/calculate/home` к `/home`.  
`cl-install -autopartition -partition-table dos -root-size 10240`

При установке будет использована автозаметка, таблица разделов для авторазметки установится в значение dos и размер корневого раздела в 10 Гигабайт.

## Точки монтирования

Доступные параметры:

- `-d DISKS, -disk DISKS` - установка точки монтирования;
- `--build [ON/OFF]` - установка для сборки;
- `--uuid [ON/OFF]` - использовать UUID;
- `--type DISKTYPE` - тип устройства для устанавливаемой системы (hdd, flash или usb-hdd);
- `--mbr MBR` - загрузочный диск для устанавливаемой системы;
- `--scheduler SCEDULER` - установить I/O планировщик (deadline, cfq или noop).
- `--uefi [ON/OFF]` - использовать загрузку UEFI

Пример:

```
cl-install -d /dev/sda2:/ext4:on -d /dev/sda3:/var/calculate:reiserfs  
-d /dev/sda1:swap -scheduler cfq
```

Диск `/dev/sda2` (первый параметр после ключа `-d`) будет промонтирован в корень (второй параметр) и будет обязательно оформлен (четвёртый параметр "on") в файловую систему ext4 (третий параметр).

Диск `/dev/sda3` будет промонтирован в `/var/calculate` и отформатирован в файловую систему reiserfs, только если в текущий момент имеет другую файловую систему; если текущая файловая система reiserfs, то форматироваться не будет.

Диск `/dev/sda1` будет промонтирован I/O планировщик cfq.

## Сетевые настройки

Доступные параметры:

- `--netconf NETMANAGER` - выбор менеджера сети (networkmanager или openrc)
- `--iface IFACE_SETTINGS` - установка адреса для сетевого интерфейса
- `--hostname HOSTNAME` - установка короткого или полного имени хоста
- `--ntp NTP` - установка NTP сервера для системы
- `--dns DNS` - установка серверов доменных имен (разделитель - запятая)
- `--domain-search DOMAINS` - установка доменов для поиска (разделитель - запятая)
- `--route NETROUTE` - добавить правило маршрутизации (формат NETWORK:[GATEWAY] [:DEV:[SOURCE]])

Пример:

В этом случае система предложит установить систему с параметрами по умолчанию. Для просмотра всех доступных параметров установки системы используйте команду

```
cl-install -hhelp
```

Все доступные параметры разделены на группы (дублируют шаги в графическом клиенте):

- Язык и локаль
- Выбор дистрибутива
- Распределение места на диске
- Точки монтирования
- Сетевые настройки
- Пользователи
- Видео

#### Языки и локаль

Доступные параметры:

- -l LANG, --lang LANG - установка языка
- -t timezone TIMEZONE - установка часового пояса

Пример:

```
cl-install -l ru_RU -t timezone Europe/MOSCOW
```

#### Выбор дистрибутива

Доступные параметры:

- --iso IMAGE - ISO образ для установки
- -s SYSTEM, --os SYSTEM - выбор операционной системы (CDS,CLDG,CLD,CLDX,CLS,CMC,CSS или Gentoo)
- -march ARCH - выбор архитектуры процессора (auto,i686 или x86\_64)

Пример:

```
cl-install -iso /var/calculate/linux/cld-2011.09.04-x86_64.iso
```

```
cl-install -s CLDX -march i686
```

В первом примере выбран конкретный образ дистрибутива, а во втором сервер утилит будет искать 32-битный образ CLDX среди доступных.

#### Распределение места на диске

Доступные параметры:

- -a autopartition - использовать авторазметку;

- -a auto-scheme AUTOPARTOTS - параметры авторазметки (swap,root,data,home,boot,EFI,grub или lvm);

- -D DEVICE - установить диск для авторазметки;

- -p partition-table TABLE - установить таблицу разделов для авторазметки (dos или gpt);

- -r root-size SIZE - установить размер корневого раздела для авторазметки.

- -s swap-size SIZE - установить размер раздела подкачки для авторазметки.

Параметр --auto-scheme, используемый при авторазметке, может принимать одно или несколько значений из swap,root,data,home,boot,lvm,EFI,grub, где swap - использовать раздел подкачки; root - использовать дополнительный корневой раздел; data - использовать раздел /var/calculate;

Рис. 2 Выбор дистрибутива

Если у вас есть другие дистрибутивы Calculate Linux, вы можете использовать их для установки, указав путь к файлу образа. Среди дополнительных параметров доступны фильтр по дистрибутиву и фильтр по архитектуре процессора.

#### Распределение места на диске

В данной версии программы установки вы не можете изменить существующие разделы. Для этого, в зависимости от того, какой дистрибутив вы используете, можно воспользоваться программой GParted или PartitionManager, либо cfdisk, fdisk или gdisk, работающими из консоли во всех версиях Calculate Linux.

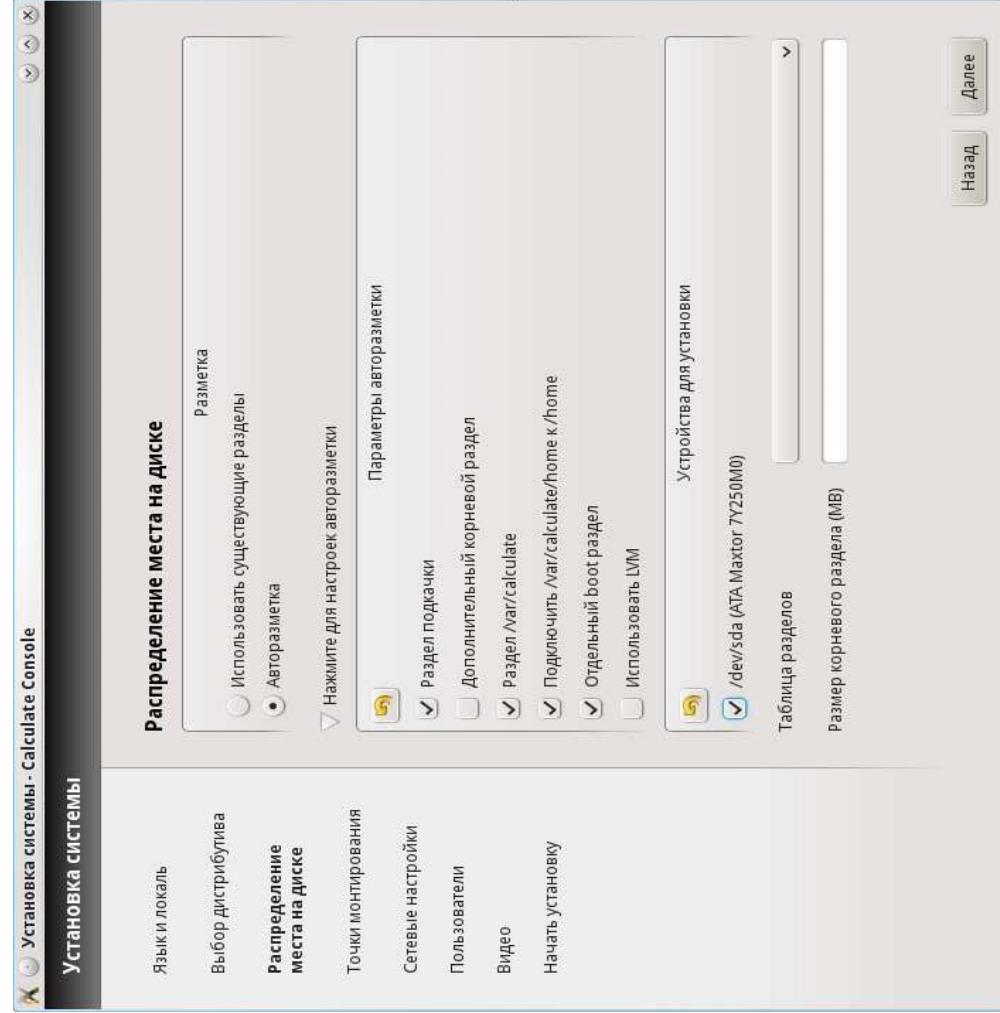


Рис. 3 Распределение места на диске

Если вы готовы использовать весь жесткий диск под систему, воспользуйтесь авторазметкой, либо выберите пункт "использовать существующие разделы" для указания одного или нескольких разделов жесткого диска для установки. При выборе авторазметки вы можете воспользоваться дополнительными параметрами:

- Параметры авторазметки - здесь вы можете указать, на сколько разделов будет разбит ваш жёсткий диск;
- Устройства для установки - выбор устройств для установки;
- Таблица разделов - выбор типа таблицы разделов ("DOS-type Partition Table" или "GUID Partition Table (GPT)");
- Размер корневого раздела - размер корневого раздела в мегабайтах (размеру 20 Гб будет соответствовать значение 20480).

## Точки монтирования

При выборе авторазметки на шаге **Распределение места на диске** данный шаг будет **недоступен для редактирования**.

Основным параметром является выбор точек монтирование в виде таблицы "Разметка". Для изменения значения в строке нажмите на неё и в открывшемся окне введите необходимые значения:

- для добавления новой точки монтирования нажмите на кнопку "+" (плоск) над таблицей;
  - для удаления точки монтирования установите флаажки слева от необходимых строк и нажмите на кнопку "-" (минус) над таблицей;
  - для восстановления первоначальных значений ячеек таблицы нажмите на третью кнопку "возврата" (стрелка) над таблицей;
  - для очистки всей таблицы нажмите на четвёртую кнопку "очистки" над таблицей.
- В столбце "Диск или директория" указывается диск (директория) для монтиrovания. Во втором столбце "Точка монтирования" указывается куда этот диск (директория) будет промонтирован. В третьем столбце "Файловая система" указывается, какую файловую систему необходимо использовать для данного диска. В столбце "Форматировать" указывается, следует ли форматировать диск. Если указанная файловая система не совпадает с текущей, то диск будет отформатирован. В последнем столбце "Размер" указан размер выбранного диска (директории). Среди дополнительных параметров можно:
- установить режим сборки;
  - установить использование UUID-дисков;
  - выбрать тип установки (Жёсткий диск, USB Flash или USB жёсткий диск);
  - выбрать загрузочный диск;
  - выбрать I/O планировщик (Deadline, CFQ, No-op).

## Установка системы

- Языки и локаль
- Выбор дистрибутива
- Распределение места на диске
- Точки монтирования
- Сетевые настройки
- Пользователи
- Видео

## Начать установку

▼ Просмотреть параметры

## Язык и локаль

Русский  
Europe/Moscow (+04:00)

**Выбор дистрибутива**

Calculate Linux Desktop KDE x86\_64 20120510

Установочный образ

Фильтр по дистрибутиву  
Фильтр по архитектуре  
процессора  
x86\_64

## Распределение места на диске

Использовать существующие разделы

## Точки монтирования

Назад Установка

Рис. 8 Просмотр параметров и начало установки

## Установка системы из консоли

Для установки системы из консоли используйте команду:  
`cl-install`

cl-install представляет собой символическую ссылку на метод сервера утилиты:  
`cl-core -method install`

Символические ссылки на методы можно создать с помощью команды `cl-core -create-symlink`.



Рис. 4 Выбор точек монтирования

## Сетевые настройки

Для настройки сети необходимо:

- выбрать менеджер сети (NetworkManager или OpenRC);
- настроить адреса для интерфейсов в виде таблицы со столбцами:
  1. использование DHCP;
  2. IP-адрес для сетевого интерфейса;
  3. маска сети;
- ввести имя хоста (короткое или полное);
- сервер доменных имён;
- домены для поиска;
  1. сеть;
  2. шлюз;
- таблицу маршрутизации в виде таблицы со столбцами:

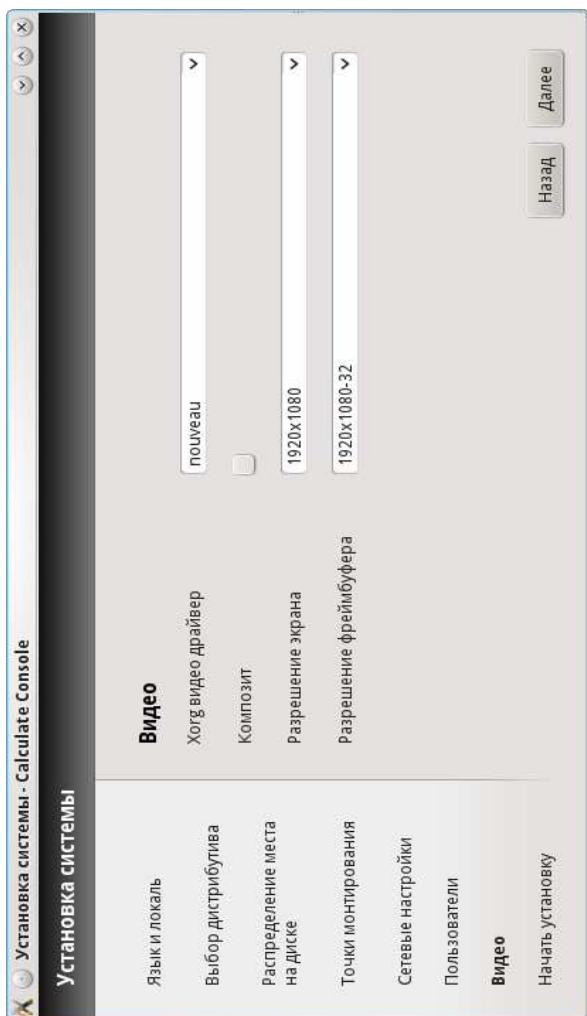


Рис. 7 Настройка видео  
Начать установку

Последний шаг является информационным и отображает значения всех параметров, которые будут использоваться при установке.

Для начала установки нажмите на кнопку "Установка".

### Установка системы - Calculate Console

3. интерфейс;
4. исходный IP.

### Установка системы - Calculate Console

#### Установка системы

##### Сетевые настройки

Менеджер сети

OpenRC

| Адреса | Интерфейс | DHCP | IP адрес  | Макет         | Название              | MAC            |
|--------|-----------|------|-----------|---------------|-----------------------|----------------|
|        | eth0      | Нет  | 10.0.0.81 | 255.255.255.0 | Realtek RTL8111/8168B | 00:FF:D8:39:DF |

##### Точки мониторинга

Пользователи

Видео

##### Настройка установки

##### Пользователи

##### Настройка установки

Домены для поиска

Маршрутизация

|  | Сеть    | Шлюз     | Интерфейс | Исходный IP |
|--|---------|----------|-----------|-------------|
|  | default | 10.0.0.1 | eth0      |             |

Рис. 5 Сетевые настройки

#### Пользователи

Настройка пользователей осуществляется изменением таблицы "Переносимые пользователи" со столбцами "Логин" и "Пароль". Работа с таблицей "Переносимые пользователи" аналогична работе с таблицей "Разметка" на шаге "Точки мониторинга", рассмотренной выше.

Особенностью является ввод пароля для пользователей, который требует повторного ввода и не отображается в явном виде.

Также из выпадающего списка можно выбрать пользователя для автовахода.

Назад

Далее

Рис. 6 Настройка пользователей

#### Видео

Настройка видео состоит из четырёх параметров:

1. Выбор видеодрайвера Xorg из списка (если необходимый драйвер отсутствует в списке, впишите его название);
2. Включение композита;
3. Установка разрешения экрана (если необходимое значение отсутствует в списке, впишите его);
4. Установка разрешения фреймбуфера (если необходимое значение отсутствует в списке, впишите его);

Назад

Далее



В последнем пункте нет ссылки на версию ядра, т.к. установка пакета переписала символическую ссылку /usr/src/linux. Проверить это можно, выполнив:

```
c1>kernel -kver list
* 3.19.0
* 3.18.7-calculate *
```

Если вы не уверены, лучше указать ядро явно:

```
c1>kernel -kver=3.18.7-calculate
```

Обратите внимание, что напротив версии ядра стоит уже не красная, а зелёная звёздочка.  
Пока ядро собирается, посмотрите на полученный шаблон настройки ядра: он будет содержать только внесённые вами изменения - отличия от оригинальной версии настроек ядра.

Пример шаблона после отключения поддержки ReiserFS:

```
cat /var/calculate/templates/kernel/10-calculate-x86_64-3.18
# Calculate format=kernel name=.config
os_install_arch_machine==x86_64&&merge(sys-kernel/calculate-sources )>=3.18
!CONFIG_REISERFS_FS=m
```

Обратите внимание, что повторное выполнение c1>kernel учитывает внесённые вами изменения.  
Чтобы сбросить их, удалите созданный вами шаблон.

## Оптимизация ядра

## Использование патчей

## Для разработчиков

# Интерактивная сборка системы

## Введение

Интерактивная сборка - это новый подход в создании своего собственного загрузочного образа. Вы можете собирать необходимые пакеты, менять настройки и при этом видеть результат своей работы, сразу же тестируя собираемый дистрибутив. При разработке нового метода сборки приступались следующие цели:

- Позволить каждому желающему создавать свой дистрибутив системы в соответствии с его взглядами и потребностями;
- Сделать процесс сборки системы более легким и доступным.

## Как работает Calculate Builder

Использование интерактивного режима сборки доступно во всех дистрибутивах Calculate Linux начиная с версии 9.8. Для использования режима сборки воспользуйтесь режимом загрузки Builder на USB Flash или LiveCD.

Во время загрузки в Builder-режиме файловая система монтируется из трех слоев `aufs2`:

- Первый слой, `calculate`, представляет собой `livecd.squashfs`-образ системы, загружаемый с носителя и примонтированный в режиме "только для чтения". Он берётся за основу будущего дистрибутива.
- Второй слой - `delta` - слой, в котором будут сохраняться все изменения во время сборки нового дистрибутива.
- Третий слой, `workspace`, - рабочий слой, в котором Вы производите все изменения над исходной системой.

## K3B

При использовании K3B выберите *Tools > CD > Burn Image (Инструменты > Компакт-диск > Проделать образ)*. Затем в поле *Image to Burn (Образ для записи)* укажите ISO-файл и нажмите *Start (Запуск)*.

Загрузка с установочного диска может потребовать выполнения определенных действий. Уберите все компакт-диски из приводов, перезагрузите компьютер и войдите в BIOS. В зависимости от BIOS, для этого обычно нужно нажать *DEL*, *F1* или *ESC*. В BIOS измените порядок загрузки так, чтобы обращение к CD-ROM выполнялось до обращения к жёсткому диску. Этот параметр часто задается в разделе *CMS Setup*. Если порядок загрузки невозможно изменить, система просто перезагрузится с жёсткого диска, игнорируя CD-ROM.

Теперь поместите установочный диск в привод CD-ROM и перезагрузитесь. Должно появиться загрузочное приглашение.

## Опции загрузки

Перед началом загрузки с CD вы увидите меню загрузки. Выполните необходимые установки, воспользовавшись клавишами **F1-F5** для перехода в меню.

- **F1 Help - Справка.**
- **F2 Language - Выбор языка. Доступны: русский, украинский, английский, испанский, немецкий, португальский, итальянский,польский и французский.**
- **F3 Keypad - Выбор раскладки клавиатуры. Если вы выбрали язык в предыдущем пункте, то можете этот пункт пропустить.**
- **F4 Timerzone - Часовой пояс. Укажите этот параметр, чтобы время в вашем компьютере совпадало с местным.**
- **F5 Video - Разрешение экрана. В большинстве случаев этот параметр можно не менять.**
- **Tab Edit options - Редактирование строки загрузки ядра. Для опытных пользователей.**

Указанные при загрузке *LiveCD* параметры будут использованы в дальнейшем при установке системы.

## Разбиение диска

В зависимости от выбранного дистрибутива после загрузки компьютера вы можете воспользоваться одной из программ разбиения диска:

CDS, CLS; fdisk, cfdisk  
CLD; KDE Partition Manager, fdisk, cfdisk  
CLDX; Gnome Partition Editor, fdisk, cfdisk

## Чем заняться дальше?

- [Чем заняться дальше?](#)
- [Документация](#)
- [Ресурсы Calculate](#)
- [Calculate в интернете](#)

## Документация

Примите поздравления! У вас теперь появилась работающая система Calculate. И что же делать дальше? Какие у вас появились возможности? На что стоит взглянуть прежде всего? Calculate даёт своим пользователям большой выбор возможностей, а следовательно - и множество документированных (или не очень) свойств.  
Вам обязательно нужно прочитать следующую часть документации, "[Работа с Calculate](#)", в которой рассказывается, как поддерживать программное обеспечение в актуальном состоянии, как дистрибутируются программы. Здесь вы найдёте ответы на часто задаваемые вопросы, узнаете множество полезных команд по управлению системой.

Для управления программным обеспечением Calculate использует [Portage](#). Любой пакет из портежей может быть скомпилирован с учетом ваших требований. Безусловно, было бы не плохо знать, как это всё работает. Вся необходимая документация собрана в разделе "[Работа с Portage](#)". Здесь вы узнаете о USE-флагах, переменных среды, ebuild-файлах.

Узнав основные вещи, вы можете познакомиться с особенностями системы - утилитами Calculate. Вся необходимая документация собрана в разделе "[Утилиты Calculate](#)". Здесь вы научитесь создавать свои шаблоны для настройки системы.

Полный список существующих материалов имеется на [странице документации](#).

## Ресурсы Calculate

Естественно, мы всегда рады видеть вас на [форумах Calculate](#), как и на [русскоязычном IRC-канале Calculate](#).

Кроме того, мы можем предложить вашему вниманию несколько списков рассылки, открытых для всех наших пользователей. Сведения о порядке подписки находятся на той же странице.

## Calculate в интернете

Вы найдете множество мест, где можете поделиться впечатлениями с другими пользователями. Calculate представлен в социальных сетях [ВКонтакте](#), [Facebook](#) и [Twitter](#).

## 2. Работа с Calculate

1. ЧаВО (FAQ)
2. Руководство пользователя
3. Создание учётных записей
4. Шифрование домашних директорий
5. Установка и удаление программ
6. Руководство по обновлению системы
7. Сценарии инициализации
8. Сборка ядра со своей конфигурацией
9. Интерактивная сборка системы
10. Системные утилиты
11. Оптимизация системы
12. Загрузка модулей ядра
13. Ревизии системы
14. Пакеты оверлея Calculate: `calcboot`, `calckernel`, `calculate-sources`, `keyexec`, `pam_keystore`

## Часто задаваемые вопросы (FAQ)

15. Часто задаваемые вопросы (FAQ)
16. Языковые настройки и параметры клавиатуры
17. Переключение языка
18. Как поменять язык установленной системы?
19. Как отключить NumLock при входе в систему?
20. Настройка звуковой карты
21. Создание пользователей
22. Установка программ
23. Настройка маршрутизации пакетов из локальной сети
24. Добавление программ на нижнюю панель KDE

На этой странице собраны рекомендации по эффективной работе в Calculate Linux.

3. Для ядер, отличных от `calculate-sources`, нет шаблонов настроек, поэтому готовый шаблон будет содержать отичия от настроек ядра по умолчанию.
4. С опцией " -s convert " программа возьмёт за основу настройки текущего ядра (из `/boot` или `/proc`), если в директории с исходным кодом ядра нет файла " `.config` ".

Во время выполнения скрипта будет вызвана настройка ядра (вызов `make menuconfig`), сборка и установка. Если не отключена опция `COMITG_BLK_DEV_INITRD`, будет создан `initramfs`.

После завершения не забудьте обновить необходимые модули, выполнив:

```
emerge @module-rebuild
```

## Обновление ядра

Теперь можно перезагрузиться, чтобы проверить работу нового ядра! Во время загрузки обратите внимание, что загружается новое ядро. Если вы ничего не изменили в окне настроек, проблем с загрузкой возникнуть не должно.

Первая строка - это заголовок шаблона. В ней описан формат шаблона, имя настраиваемого файла, выполняется проверка на архитектуру системы, имя и версия ядра.

Из шаблона видно, что он будет работать для всех ядер версии 3.19 и выше.

Для установки 3.19.1 ядра, после установки пакета достаточно будет выполнить:

```
c1-kernel -kver=3.19.1
```

Параметр '`-kver`' можно опустить, если ядро выбрано по умолчанию, когда символическая ссылка `/usr/src/linux` указывает на него. Так будет, если при установке пакета с исходным кодом ядра, вы указали USE-флаг " `symlink`" .Например, предварительно выполнив:

```
echo sys-kernel/vanilla-sources symlink >> /etc/portage/package.use/custom
```

При переходе к более крупной версии ядра, например 3.20.4.0, часто возникает необходимость просмотреть перечень изменений между настройками ядра (`make oldconfig`). Для этого выполните:

```
c1-kernel -kver 4.0.0 -kver-old=3.19.1
```

## Настройка calculate-sources

На примере `vanilla-sources` мы научились устанавливать и собирать различные пакеты ядра. Но как быть, если нужно поставить обновление ядра "на поток" с вашими изменениями настроек и патчами? Нет ничего проще!

1. Сбросьте у ядра USE-флаг "minimal":  

```
echo sys-kernel/calculate-sources -minimal >> /etc/portage/package.use/custom
```
2. Установите исходники ядра без компиляции:  

```
USE="" -vmlinuz" emerge sys-kernel/calculate-sources
```
3. Измените настройки:  

```
c1-kernel
```

- Прозрачная миграция настроек между версиями ядер.
- Интеграция с утилитами Calculate для использования шаблонов настроек во время установки ядра `calcluate-sources`.
- Создание резервных копий настроек.
- Локализация на русский и французский языки.
- Исходный код ядра распаковывается в директорию `/usr/src`. Поменять свободное место можно, выполнив:

```
df -h
```

Обязательно позаботьтесь о наличии резервной копии ядра, с которого всегда можно загрузить систему. Для этого эксперименты удобно проводить с альтернативными пакетами ядер, либо с ядром `calcluate-sources` не установленной версии.

## Сборка ядра

Выберите любое из доступных в портежах ядра. Весь список с описаниями можно посмотреть, выполнив:

```
eix -c sys-kernel/*sources
```

Для примера остановим свой выбор на "ванильном" ядре - оригинальной версии, поддерживаемой Линусом Торvalдсом.

Ядро в портежах отмечено маской, поэтому понадобится сперва её снять:

```
echo sys-kernel/vanilla-sources ~amd64 ~x86 >> /etc/portage/package.keywords/custom
USE="symlink" emerge sys-kernel/vanilla-sources
```

USE-флаг "symlink" следует устанавливать, если вы используете проприетарные пакеты, такие как `nvidia-drivers`, `ati-drivers`, `virtualbox-bin` или `broadcom-sta`. В этом случае после установки ядра следует собрать их модули, выполнив:

```
emerge @module-rebuild
```

Проверьте, что ваше ядро стало доступно:

```
c1-kernel --kver list
* 3.19.0 *
* 3.18.7-calculate
```

Обратите внимание на список. В отличие от `calculate-sources` (и других ядер), ванильное ядро не содержит слова "vanilla". Красная звёздочка слева версии ядра означает, что установленное ядро не содержит полной версии исходного кода. Звёздочка справа отмечает используемое по умолчанию ядро. Оно определяется по символьической ссылке `/usr/src/Linux`.

Для первого запуска вы можете сконвертировать настройки из текущей версии ядра, для этого выполните:

```
c1-kernel -kver 3.19.0 --convert
```

Здесь важно понять особенность работы `c1-kernel` и её отличие от более ранней версии этой программы.

- Скрипт `c1-kernel` работает с конфигурационным файлом ядра, полученным из шаблона.
- По завершении работы программы проанализирует изменения, выполненные пользователем, и создаст новый пользовательский шаблон.

## Языковые настройки и параметры клавиатуры.

### Переключение языка

В русской локализации Calculate Linux переключение языка выполняется клавишей `Caps Lock`, а фиксация верхнего регистра - сочетанием `Shift+CapsLock` (по аналогии с печатной машинкой).

### Как поменять язык установленной системы?

Если во время загрузки LiveCD вы не выбрали свой язык, по умолчанию интерфейс будет английским. Подробности см. в документации утилит `Calculate`.

### Как отключить NumLock при входе в систему?

Выполните с правами рута в консоли:

```
rc-update add numlock default
```

Чтобы вернуть включение NumLock при старте системы, выполните:

```
rc-update del numlock
```

## Настройка звуковой карты

Если у вас не определилась звуковая карта или вы не слышите звука в динамиках, от имени пользователя `root` выполните команду `alsasof`

с помощью которой определяются и настраиваются звуковые карты.

### Создание пользователей

По умолчанию в `Calculate Linux Desktop` существует пользователь `qtest`. Вы можете использовать эту учетную запись для знакомства с системой, а также для предоставления гостевого входа своим знакомым. Для создания новых учетных записей воспользуйтесь следующим руководством.

## Установка программ

При помощи нескольких простых команд вы можете обновлять или устанавливать новое программное обеспечение.

На компьютере хранится локальное дерево портежей со списком всех программ. Перед установкой или обновлением программ стоит обновить его с помощью команды `eix-sync`

Ее нужно выполнять с правами пользователя `root`.

После этого можно установить новый пакет командой:

```
emerge имя_пакета
```

Узнать точное название пакета можно с помощью программы `eix`, выполнив:

```
eix часть_названия
```

## Настройка маршрутизации пакетов из локальной сети

Если вам необходимо настроить Calculate Linux в качестве маршрутизатора для доступа в интернет локальных машин (192.168.0.0/24), выполните следующие команды:

```
sysctl -w net.ipv4.ip_forward=1
```

```
iptables -t nat -I POSTROUTING -s 192.168.0.0/24 -j MASQUERADE
```

```
net.eth0 |
```

Сохранить измененное значение net.ip\_forward можно в файле /etc/sysctl.conf.

## Добавление программ на нижнюю панель KDE

1. Разблокируйте изменение виджетов, кликнув правой кнопкой на рабочем столе;
2. Кликните правой кнопкой на иконке меню слева вверху, выберите пункт "Редактировать меню";
3. Выберите и перетащите иконку приложения в нижнюю панель.

## Руководство пользователя

### Введение

Уважаемый пользователь! Благодарим за выбор операционной системы Calculate Linux!

Надеемся, что работа в новой системе не вызовет сложностей и доставит вам истинное удовольствие.

### Рабочий стол

Рабочий стол Calculate Linux Desktop настроен с учетом комфортной работы пользователя. Каждый элемент рабочего стола (расположение панелей, порядок иконок, управляющие элементы) имеет свое особое место на экране, упрощая работу на компьютере. Большинство приложений предварительно настроены для экономии рабочего времени пользователя.



Выulia, все готово. Теперь, если при загрузке вы выберете вновь созданную запись, то вместо default будет использоваться уровень offline.

### Использование загрузочного уровня (bootlevel)

Использование загрузочного уровня полностью аналогично использованию программного уровня. Единственная разница состоит в том, что вы определяете второй уровень "boot" вместо "default".

## Сборка ядра со своей конфигурацией

- Сборка ядра со своей конфигурацией
- Введение
- cl-kernel
- Сборка ядра
- Обновление ядра
- Настройка calculate-sources
- Оптимизация ядра
- Использование патчей
- Для разработчиков

### Введение

Calculate Linux работает на ядре Linux с длительным сроком поддержки (longterm). Большая часть драйверов вынесена в модули, что позволяет ядру оставаться компактным в размере без потери функционала. Для серверов и десктолов используется различные настройки и патчи. В отличие от других ядер из портажей, пакет sys-kernel/calculate-sources, используемый по умолчанию в дистрибутивах Calculate Linux, компилируется и устанавливается в систему, как и большинство других пакетов, высвобождая место путём удаления за собой большей части исходного кода.

Зачем вам может понадобиться изменять настройки ядра? Оптимизируя ядро, вы можете добиться пророста производительности, поддержки своего железа, высвобождения памяти, снижения энергопотребления, а так же ускорения загрузки системы. Помимо прочего, изучение ядра даёт неплохие знания в понимании устройства работы системы.

Экспериментировать с ядром можно и нужно. Эта статья поможет вам научиться изменять настройки ядра, устанавливать и настраивать различные модификации ядер, использовать свои патчи.

### cl-kernel

Для сборки ядра служит скрипты cl-kernel, входящий в состав пакета sys-apps/calculate-toolkit. Программа написана на Bash и прозрачно интегрирована с системой шаблонов утилит Calculate.

### Особенности

1. Поддержка сборки различных ядер: sys-kernel/calculate-sources, sys-kernel/gentoo-sources, sys-kernel/vanilla-sources и Ap.
2. Поддержка создания ядра, как с использованием initramfs, так и без него.
3. Создание шаблона настроек ядра со всеми внесёнными изменениями.
4. Импорт готовых настроек ядра в шаблон.



Рис. 1 Внешний вид рабочего стола Calculate Linux Desktop 11.0 KDE

### Информационная панель

Вверху рабочего стола находится Информационная панель.



Рис. 2 Информационная панель

### Расположение элементов на панели слева направо:

1. Главное меню - аналог кнопки "Пуск" в ОС Windows.
2. Сетевые диски: Home, Disks и FTP.

## Добавление дополнительных параметров

Если вы хотите ввести в сценарий дополнительные параметры, кроме упомянутых, нужно добавить к переменной `opts` название параметра и создать функцию с названием, соответствующим параметру. Например, для поддержки параметра `restartdelay`. Пример создания дополнительной функции `restartdelay`:

```
opts="$opts"} restartdelay"
{
    stop
    sleep 3 # пауза в 3 секунды перед повторным запуском
    start
}
```

## Переменные для настройки служб

Для поддержки конфигурационного файла в каталоге `/etc/conf.d` ничего дополнительного делать не нужно: при запуске вашего сценария инициализации автоматически включаются следующие файлы (т.е., переменные из них становятся доступны):

- `/etc/conf.d/<ваш сценарий инициализации>`
- `/etc/conf.d/basic`
- `/etc/rc.conf`

Если ваш инициализационный сценарий предоставляет виртуальную зависимость (например, `net`), то также включается файл, соответствующий этой зависимости (например, `/etc/conf.d/net`).

4.e. Изменение поведения уровняй запуска

## Кто от этого выиграет?

Большинству пользователей ноутбуков знакома ситуация: дома вам нужен запуск `net.eth0`, а в дороге, наоборот, запуск `net.eth0` не нужен (так как сеть недоступна). В Calculate можно изменять поведение уровней запуска по своему усмотрению.

Например вы можете создать второй загружаемый уровень запуска «по умолчанию», в котором будут другие сценарии. Затем при загрузке вы сможете выбрать, какой из уровней по умолчанию следует использовать.

## Использование программного уровня (softlevel)

Прежде всего, создайте каталог для своего второго уровня запуска «по умолчанию». Например, создадим уровень запуска `offline`. Пример создания каталога уровня запуска:

```
mkdir /etc/runlevels/offline
```

Добавьте необходимые сценарии инициализации в только что созданный уровень запуска. Например, чтобы получить точную копию уровня `default`, за исключением `net.eth0`:

```
(копирование всех служб с уровня default в уровень offline)
# cd /etc/runlevels/default
# for service in *; do rc-update add $service offline; done
(удаление ненужных сценариев с уровня offline)
# rc-update del net.eth0 offline
(просмотр сценариев, запускаемых на уровне offline
# rc-update show offline
(часть выведенного списка)
    acpid | offline
    domainname | offline
    local | offline
```

3. Переключатель рабочих столов. Для быстрого переключения между рабочими столами.
4. Панель задач. Быстрый доступ к запущенным приложениям текущего рабочего стола.
5. Корзина. Содержит удаленные файлы и директории. Не забывайте иногда чистить корзину.
6. Системный лоток, отображает информацию некоторых приложений. В нем вы всегда найдете такие приложения как буфер обмена, органайзер, утилита для работы с клавиатурой, доступ к USB устройствам и т.д.
7. Часы. Если кликнуть на них мышкой, отобразится программа Календарь.
8. Блокировка доступа и Завершение сеанса. Покидая рабочее место, не забывайте блокировать доступ к компьютеру. Заблокировать доступ к компьютеру можно также по нажатию клавиши "Scroll Lock".

## Панель приложений

Внизу по центру экрана находится Панель приложений.



Рис. 3 Панель приложений

Панель приложений включает все программы, необходимые в повседневной работе. Программы отсортированы по значению слева направо: Интернет-приложения, офис, дополнительные приложения, редактор, мультимедиа-приложения и утилиты.

## Программное обеспечение

Почти всё программное обеспечение Calculate Linux Desktop имеет свободную лицензию, по которой вы можете получать доступ к исходным текстам программ, изменять код, распространять и даже продавать без каких-либо денежных отчислений правообладателям.

## Основные программы

- Для навигации в интернете используется браузер *Chromium*.
- В качестве почтового клиента установлен *Kmail*.
- Текстовый процессор и электронные таблицы представлены офисным пакетом *LibreOffice*. В работе вы можете использовать также простой текстовый редактор *KWrite* (в версии CLD с рабочим окружением KDE).
- Программы, не имеющие аналогов в ОС Linux, могут работать на терминальном Windows-сервере, отображая информацию на экран вашего компьютера.

## Дополнительные программы

Для доступа к остальным программам используйте значок **главного меню**, расположенный слева на верхней панели. Все приложения отсортированы по назначению, возле каждой программы находится пояснение.

- требует сеть (net): виртуальная зависимость, удовлетворяемая, например, /etc/init.d/net .eth0
- использует журнал (logger): виртуальная зависимость, удовлетворяемая, например, /etc/init.d/syslog -ng
- использует службу имен (dns): виртуальная зависимость, удовлетворяемая, например, /etc/init.d/named
- предоставляет почтовый агент (mta): виртуальная зависимость, общая для всех программ --- почтовых серверов

### Порядок запуска

Иногда вам нужна не сама служба, а запуск вашей службы до (или *после*) другой службы, если та присутствует в системе ( обратите внимание на условие: это уже не зависимость) и запускается на том же уровне запуска (отметьте условие: это относится только к службам из одинакового уровня запуска). Такую очередность можно указать, используя значения *before* (.0) или *after* (после).

Например, рассмотрим значения для службы Portmap. Пример функции depend() службы Portmap:

```
depend() {
    need net
    before inetd
    before xinetd
}
```

Также можно использовать знак "\*" , чтобы охватить все службы данного уровня запуска, хотя это не рекомендуется. Пример запуска сценария первым на уровне запуска:

```
depend() {
    before *
}
```

### Стандартные функции

Следом за разделом depend() вам потребуется определить функцию start(). В ней содержатся все команды, необходимые для запуска вашей службы. Рекомендуется применять функции ebegin и end для сообщений пользователю о том, что происходит. Пример функции start():

```
start() {
    ebegin "Запуск - моя_служба"
    start-stop-daemon -start -quiet -exec /path/to/my_service
    end $?
}
```

**Особенности работы**

**Рабочие столы**

Используйте в своей работе несколько рабочих столов! На первом рабочем столе можно выполнять текущие задачи, на втором работать с документами, на третьем просматривать почту, переключаясь между ними мышкой (см. *Переключатель рабочих столов верхней панели*), либо при помощи комбинации клавиш *Ctrl+Shift+F1-F4*.

### Доступ к файлам

Для доступа к файлам используйте иконки *Home*, *Disks* и *FTP* на рабочем столе или на верхней панели текущих задач. Панка *Home* открывает домашний диск пользователя, *Disks* - сетевые диски (общие ресурсы), *FTP* - файлы FTP сервера. Иконки *Disks* и *FTP* будут доступны в случае, если ваш компьютер введен в домен *Calculate Directory Server*.

### Раскладка клавиатуры

Раскладка клавиатуры аналогична раскладке в ОС Windows. Для переключения языка используйте Caps Lock. В время, когда активна не английская раскладка, на клавиатуре будет гореть светодиод Scroll Lock. В системном лотке справа на информационной панели отображается выбранная раскладка. Изменить регистр букв с нижнего на верхний и наоборот можно, нажав комбинацию клавиш Shift+Caps Lock.

Другими функциями, которые можно определить --- stop() и restart(). От вас не требуется определение этих функций! Система инициализации, находящихся в каталоге /etc/init.d. Что касается команды start-stop-daemon, то на случай, если вы используете start-stop-daemon. Пример вызова страницы справки по start-stop-daemon:

```
man start-stop-daemon
```

Синтаксис сценариев инициализации, применяемых в Calculate, основан на оболочке Борна (Bourne Again Shell --- bash), поэтому вы можете свободно использовать внутри своих сценариев bash-совместимые конструкции.

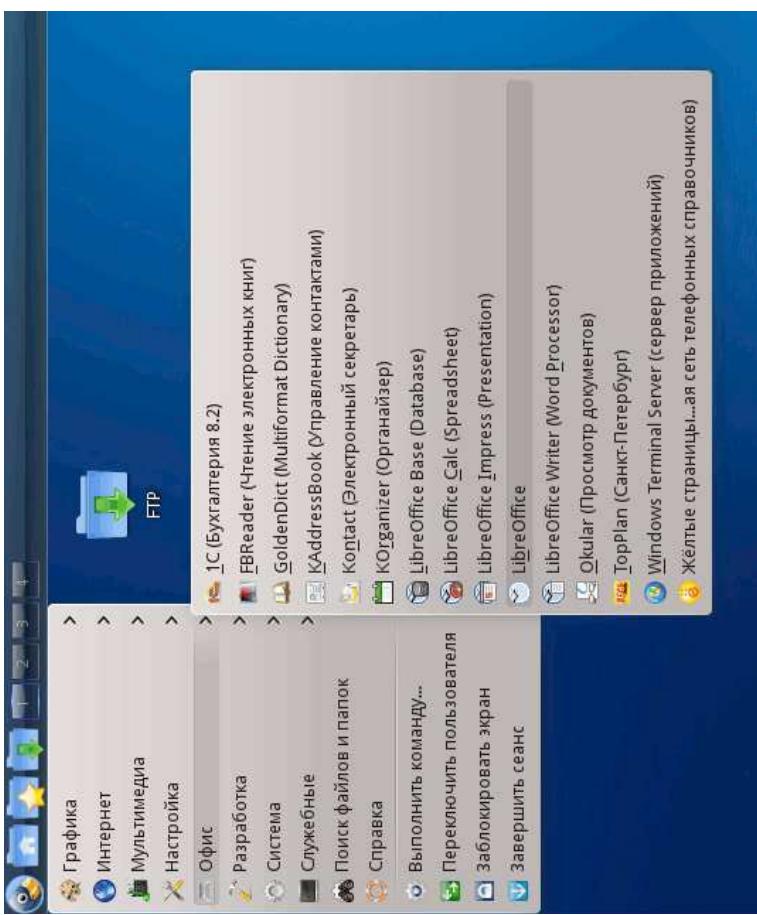


Рис. 4 Главное меню

## APACHE2\_OPTS="-D PHP4"

Такие файлы настроек содержат одни переменные (наподобие `/etc/make.conf`), облегчая настройку служб. Это также позволяет нам давать больше информации о переменных (в комментариях).

## Написание сценариев инициализации

### Мне тоже придется?..

Нет, написание сценариев инициализации обычно не требуется, т.к. Calculate содержит готовые сценарии для всех поддерживаемых служб. Однако, вы можете установить какую-либо службу, не используя систему Portage; в таком случае, вероятно, вам придется создавать сценарий инициализации самостоятельно.

### Структура

Основная структура сценария инициализации показана ниже.

```
#!/sbin/runscript

depend() {
    (информация о зависимостях)

start() {
    (команды, необходимые для запуска службы)
}

stop() {
    (команды, необходимые для остановки службы)
}

restart() {
    (команды, необходимые для перезапуска службы)
}
```

В любом сценарии должна быть определена функция `start()`. Все остальные разделы не обязательны.

### Зависимости

Можно определять два типа зависимостей: `use` (использую) и `need` (нуждаюсь). Как упоминалось ранее, `need`-зависимость более строга, чем `use`-зависимость. Вслед за типом зависимости указывается название службы, от которой существует зависимость, или ссылка на виртуальную (virtual) зависимость.

**Виртуальная зависимость** --- это зависимость от функций, предоставляемых журнала, но таких достаточно много (metalogd, syslogd, sysklogd и т.п.). Поскольку нельзя нажаться в каждой из них (ни в одной виртуальной зависимости вместе этиими службами).

Давайте взглянем на информацию о зависимостях `postfix`:

```
depend() {
    need net
    use logger dns
    provide mta
}
```

Как можно увидеть, `postfix`:



Рис. 5. Меню выбора действий над USB устройством

### Буфер обмена

Calculate Linux Desktop имеет два буфера обмена. Первый работает аналогично OS Windows. Копирование в буфер происходит путем нажатия сочетания клавиш `Ctrl+C` или `Ctrl+Insert`, извлечение – `Ctrl+V` или `Shift+Insert`.

Со вторым буфером обмена работает происходит исклюющеительно мышкой. Во время выделения текста выделенный текст сразу же копируется в буфер обмена. Вставлять этот текст можно так же легко, воспользовавшись третьей клавишей мыши (как правило, колесиком) – просто нажмите на колесико мыши в том месте, где нужно вставить текст. Если у вашей мыши нет колесика или вы работаете с ноутбуком, третью клавишу заменяет одновременное нажатие двух кнопок.

Содержимое двух буферов обмена хранится раздельно. Можно получить доступ к истории буфера обмена, кликнув на крольчке приложения буфера обмена в системном лотке или нажав сочетание клавиш `Ctrl+Alt+U`.

## Создание учётных записей

### Пользователи системы

После установки `Calculate Linux Desktop` в системе присутствует пользователь `guest`. Используйте учётную запись пользователя для знакомства с системой.

Пароль гостевого пользователя можно изменить в любой момент с помощью команды `passwd`.

Удаленный доступ к компьютеру (по протоколу `ssh`) разрешен только для пользователя `root`.

Процесс смены паролей при установке системы на несколько компьютеров можно автоматизировать. Для этого установите систему на первый компьютер, измените пароли командой `passwd`, после чего сохраните файл `/etc/shadow` в своем шаблоне.

## Добавление пользователя

Linux несложно спрятывается с контролем над действиями пользователя. Например, можно ограничить доступ к СД/DVD-приводу, звуковой карте, сканеру и даже компьютерным играм. Для наделения пользователя необходимыми полномочиями добавьте его в соответствующие системные группы.

Чтобы не подвергать систему риску, пользователю root запрещено работать в графическом окружении.

Поэтому вам временно придется переключиться в консоль, нажав на клавиатуре Ctrl+Alt+F1. На пропадание ввести логин вредите root, затем пароль. Далее выполните следующие команды:

```
/usr/sbin/useradd -c create-home -g groups  
users,wheel,audio,cdrom,video,cdrw,usd,plugdev,games,lp,scanner,iusr  
<логин>  
/bin/passwd <логин>
```

В качестве логина вы можете использовать любое слово, состоящее из латинских букв и цифр.

При создании пользователя не следует создавать ему домашнюю директорию. В этом случае при первом входе в систему она создается автоматически с учетом настроенного шаблона пользователя. В приведенном примере создается пользователь с правами доступа к нескольким группам:

- users - доступ к менеджеру сети;
- wheel - возможность получить привилегии суперпользователя, используя команду su;
- audio - предоставляет доступ к звуковой карте;
- cdrom - доступ к CD/DVD-приводу;
- video - доступ к TV-тюнеру;
- cdrw - запись на CD/DVD-диски;
- usb - доступ к mp3-плеерам, флеш-накопителям, доступ к USB в VirtualBox;
- plugdev - монтирование USB-устройств;
- games - доступ к играм;
- scanner - доступ к сканеру;
- lp - доступ к принтеру или сканеру, встроенному в МФУ;
- iuscr - доступ к модему.

Для возврата к графическому приглашению ввода пароля выйдите из сеанса, выполнив команду exit (либо нажав Ctrl+D), после чего нажмите Alt+F7.

## Изменение прав доступа

Для того, чтобы добавить созданного пользователя в группу, воспользуйтесь командой gpasswd.

Пример добавления пользователя в группу games:

```
gpasswd -a <логин> games
```

Можно также напрямую редактировать файл /etc/group.

## Перенос пользователя

При установке либо переустановке Calculate Linux Desktop из работающей Linux-системы все пользователи с их правами доступа будут перенесены в новую систему. Потребуется только повторно присвоить им пароли, используя команду passmd, по описанной выше схеме.

Для того, чтобы выяснить зависимости службы, можно использовать аргументы iuse или ineed. С помощью ineed вы увидите те службы, которые действительно необходимы для правильного функционирования интересующей вас службы. С другой стороны, iuse покажет те службы, которые могут использоваться нашей службой, но не обязательны для ее работы. Пример запроса списка всех необходимых служб, от которых зависит Postfix:

```
/etc/init.d/postfix needsme
```

Наконец, можно просмотреть список служб, требующихся для данной, но отсутствующих в системе. Пример запроса списка служб, необходимых Postfix, но отсутствующих:

```
/etc/init.d/postfix broken
```

## Использование rc-update

### Что такое rc-update?

Система инициализации Calculate использует дерево зависимостей для определения служб, которые запускаются в первую очередь. Т. к. это очень утомительное занятие, и мы не хотели, чтобы пользователь занимался этим вручную, были разработаны инструменты, упрощающие управление уровнями запуска и сценариями инициализации.

Используя rc-update, можно включать и исключать сценарии инициализации из уровней запуска. Из update автоматически запускается сценарий depscan.sh, который перестраивает дерево зависимостей.

### Добавление и удаление служб

В процессе установки Calculate вы могли добавлять сценарии инициализации в уровень запуска "default". В тот момент вы, возможно, не имели понятия, что такое "default" и зачем он нужен, но теперь вы это знаете. Сценарии rc-update требуются вторым аргументом, определяющим действие: add (добавить), del (удалить) или show (показать).

Для того, чтобы добавить или удалить сценарий, просто введите rc-update с аргументом add или del, затем название сценария и уровня запуска. Пример удаления Postfix из уровня запуска default:

```
rc-update del postfix default
```

По команде rc-update show выводится список всех доступных сценариев с указанием соответствующих уровней запуска. Пример получения информации о сценариях инициализации:

```
rc-update show
```

## Настройка служб

### Почему нужна дополнительная настройка?

Сценарии инициализации могут быть весьма сложны. Поэтому нежелательно допускать непосредственное редактирование сценария пользователями, т.к. это может привести в систему множество ошибок. Но, с другой стороны, необходимо правильно настроить службу. Например, может понадобиться передать службе дополнительные параметры.

Вторая причина, по которой настройки хранятся отдельно от самого сценария --- это возможность обновления сценария без опасения, что все ваши настройки будут утеряны.

### Каталог /etc/conf.d

В Calculate предусмотрен очень простой способ настройки служб: для каждого сценария, предполагающего настройку, в каталоге /etc/conf.d есть конфигурационный файл. Например, у сценария, запускающего apache2 (под названием /etc/init.d/apache2), есть конфигурационный файл /etc/conf.d/apache2, где могут храниться нужные вам параметры, передаваемые серверу Apache 2 при запуске. Пример переменной, определенной в /etc/conf.d/apache2:

- Шифрование домашних директорий
- Принцип работы
- Настройка профиля на шифрование
- Как восстановить данные

## Шифрование домашних директорий

В строке, определяющей уровень 3, для запуска служб снова используется сценарий `rc` (на этот раз с аргументом `default`). Опять-таки, обратите внимание, что аргумент, передаваемый сценарию `rc`, совпадает с названием подкаталога из `/etc/runlevels`.

По окончании работы `rc`, `init` принимает решение о том, какие виртуальные консоли включить и какие команды выполнить в каждой из них. Пример определения виртуальных консолей:

```
c1:12345:respawn:/sbin/agetty 38400 ttu1 linux
c2:12345:respawn:/sbin/agetty 38400 ttu2 linux
c3:12345:respawn:/sbin/agetty 38400 ttu3 linux
c4:12345:respawn:/sbin/agetty 38400 ttu4 linux
c5:12345:respawn:/sbin/agetty 38400 ttu5 linux
c6:12345:respawn:/sbin/agetty 38400 ttu6 linux
```

## Что такое уровень запуска?

Как вы заметили, `init` применяет нумерацию для определения уровня запуска, который надо использовать. Уровень запуска --- это то состояние, в котором запускается ваша система; он содержит набор сценариев (сценарииев уровня запуска или сценарииев инициализации [`initscript`]), которые следуют выполнять при входе и выходе из определенного уровня запуска.

В Calculate определено семь уровней запуска: три служебных и четыре определяемых пользователем. Служебные называются `sysinit`, `shutdown` и `reboot`. Действия, совершаемые ими, в точности соответствуют их названиям: инициализация системы, выключение системы и ее перезагрузка.

Определенные пользователем уровни --- это те, которым соответствуют подкаталоги в `/etc/runlevels`: `boot`, `default`, `none` и `single`. Уровень `boot` запускает все службы, необходимые системе и используемые всеми остальными уровнями. Остальные уровни отличаются друг от друга запускаемыми службами: `default` используется для повседневной работы, `none` --- для тех случаев, когда не требуется сеть, `a single` --- при необходимости восстановления системы.

## Работа со сценариями инициализации

Сценарии, запускаемые процессом `rc`, называются сценариями инициализации. Каждый сценарий из `/etc/init.d` может запускаться с аргументами `start`, `stop`, `restart`, `pause`, `zap`, `status`, `need`, `use`, `needsme`, `usesme` и `broken`.

Для запуска, остановки или перезапуска службы (и всех, зависящих от нее) следует использовать `start`, `stop` и `restart`. Пример запуска `postfix`:

```
/etc/init.d/postfix start
```

Примечание: Останавливаются или перезапускаются только те службы, которым необходима данная служба. Остальные зависимые службы (те, которые используют службу, но не нуждаются в ней) эта операция не затрагивает.

Если вы хотите остановить службу, но оставить зависимые от нее работоспособными, можно использовать аргумент `pause`. Пример:

```
/etc/init.d/postfix pause
```

Чтобы узнать текущее состояние службы (запущена, остановлена, приостановлена и т.д.), можно использовать аргумент `status`. Пример:

```
/etc/init.d/postfix status
```

Если указано, что служба работает, но вы знаете, что это не так, можно сбросить состояние на `stopped` (остановлена), используя аргумент `stop`. Пример сброса информации о состоянии `postfix`:

```
/etc/init.d/postfix zap
```

Начиная с версии 13.6 в Calculate Linux появилась возможность использовать шифрованные домашние директории. Для организации защиты персональных данных используется `eCryptfs` - файловая система, работающая «поверх» любой другой обычной ФС и прозрачно шифрует/десифрует содержимое файлов. Криптографические метаданные `eCryptfs` хранят в заголовках каждого файла, таким образом, можно без проблем перенести любой файл между различными системами. Всё это реализовано на уровне ядра Linux, обеспечивая хороший уровень производительности по сравнению с FUSE-шифрованием.

## Принцип работы

Домашняя папка пользователя с профилем, настроенным на шифрование, содержит только символьические ссылки на директории с шифрованными данными. Шифрованные данные хранятся в каталоге `/home/ .eCryptfs` и разделены по пользователям, при этом в папке `.Private` хранятся данные, в `.eCryptfs` - информация по подключению шифрования.

При входе в сеанс `/home/ .eCryptfs/имя_пользователя/ .Private` монтируется в домашнюю директорию. Для этого используется пользовательским паролем ключ: `/home/.eCryptfs/имя_пользователя/.eCryptfs/mapped-passphrase`. Дальше происходит синхронизация профиля пользователя и настройка его при необходимости.

При выходе из сеанса домашняя директория отключается от `.Private`.

Данные доменных пользователей также могут быть настроены на шифрование, при этом нужно отметить, что в шифрованном виде эти данные будут находиться на локальной машине, на сервере они находятся в незашифрованном виде. Таким образом обеспечивается безопасность данных, оставленных в пользовательском профиле после выхода из сеанса.

## Настройка профиля на шифрование

Для того, чтобы активировать данную функцию, систему нужно установить с включенным параметром "Шифровать пользовательские профили" (`Crypt user profiles`), либо включить переменную `main.c1.home_crypt_set` в уже установленной системе.

```
cl-core-variables -s main.c1.home_crypt_set=on
```

Шифрование будет настроено только для новых пользователей (тех, у которых нет домашней директории). Включить шифрование для существующего профиля можно при помощи команды `eCryptfs-migrate-home`.

```
ecryptfs-migrate-home -u <имя_пользователя>
```

Команда запросит пароль пользователя.

Для доменных пользователей будет достаточно удалить локальный профиль, когда он не в сеансе, и после этого войти в сеанс.

Стоит отметить, что на настроенный пользовательский профиль значение переменной `cl_home_crypt_set` не влияет, то есть если профиль был настроен на шифрование, а система была установлена без шифрования, профиль будет продолжать работать с шифрованием, и наоборот.

## Как восстановить данные

Если в `/home` остался только каталог `.eCryptfs`, содержащий шифрованные профили, то достаточно создать таких же пользователей с теми же паролями, и при входе в сеанс домашний каталог пользователя будет использовать существующий шифрованный профиль.

## Установка и удаление программ

- Установка и удаление программ
- Обновление дерева портежей
- Поиск программ

- Установка и удаление
- Размаскировка программ
- 1. Проверим доступные версии
- 2. Размаскируем установиваемые пакеты с зависимостями
- 3. Установим программу

## Обновление дерева портежей

Перед установкой программ обновите локальный репозиторий пакетов. Обновлять репозиторий следует не чаще 1 раза в день.

Для обновления достаточно выполнить с правами пользователя тоот команду:

```
c1-update - - sync - only
```

Программа обновляет дерево портежей, оверлей Calculate, а затем синхронизирует свой локальный кэш, используемый при поиске программ.

**Поиск программ**

В программу emerge включен инструмент поиска программ, однако вы можете воспользоваться более быстрым инструментом - программой el\_x.

Пример:

```
elx freeciv  
elx - S game
```

В первом случае поиск производится по названию пакета, во втором - по описанию.

## Установка и удаление

Установка и удаление программ производится при помощи программы [emerge](#). При установке новой программы сначала определяется необходимость в установке дополнительных пакетов (зависимостей), а затем сканируются и компилируются исходные тексты. Помните, что для установки программ вам потребуется наличие интернета и некоторое время для компиляции. Программы, требующие значительного времени для компиляции (например, LibreOffice), распространяются в виде готовых к установке бинарных пакетов; их можно опознать по суффиксу "-bin".

Пример установки игры "Цивилизация" и бинарного LibreOffice:

```
emerge -bk games-strategy/freeciv  
emerge libreoffice-bin
```

Параметры -bk создают локальный архив скомпилированного пакета, а при его наличии программа emerge -C удаляет игру "Цивилизация":

```
emerge -C games-strategy/freeciv
```

Переменные DISTDIR и PKGDIR указывают путь к локальным папкам, в которых сохраняются исходные тексты программ и откомпилированные пакеты. Посмотреть значения этих (и многих других) переменных, используемых emerge, можно с помощью команды

```
emerge --info
```

И, наконец, когда все сценарии выполнены, init подключает терминалы (чаще всего просто виртуальные консоли, которые видны при нажатии ALT+F1, ALT+F2 и т.д.), прикрепляя к каждой консоли специальный процесс под названием getty. Этот процесс впоследствии обеспечивает возможность входа в систему с помощью login.

## Сценарии инициализации

Сейчас процесс init запускает сценарии из каталога /etc/init.d, не в случайном порядке. Более того, запускаются не все сценарии из /etc/init.d, а только те, которые предписано исполнять. Решение о запуске сценария принимается в результате просмотра каталога /etc/runlevels.

Во-первых, init запускает все сценарии из /etc/init.d, на которые есть символьные ссылки из /etc/runlevels/boot. Обычно сценарии запускаются в алфавитном порядке, но в некоторых сценариях имеется информация о зависимостях от других сценариев, указывающая системе на необходимость их предварительного запуска.

Когда все сценарии, указанные в /etc/runlevels/boot, будут выполнены, init переходит к запуску сценарiev, на которые есть символьные ссылки из /etc/runlevels/default. И снова запуск происходит в алфавитном порядке, пока в сценарии не встретится информация о зависимостях; тогда порядок изменяется для обеспечения правильного порядка запуска.

### Как работает init

Конечно, init не принимает решений сам по себе. Ему необходимо конфигурационный файл, где описаны необходимые действия. Этот файл --- /etc/init.tab.

Если вы запомнили последовательность загрузки, описанную чуть ранее, вы вспомните, что первое действие init --- это монтирование всех файловых систем. Это определяется в строке /etc/inittab, приведенной ниже:

```
si::sysinit:/sbin/rc sysinit  
  
Этой строкой процесс init предписывается выполнить /sbin/rc sysinit для инициализации системы.  
Самой инициализацией занимается сценарий /sbin/rc, так что можно сказать, что init делает не слишком много --- он просто делегирует задачу по инициализации системы другому процессу.  
Во-вторых, init выполняет все сценарии, на которые есть символьные ссылки из /etc/runlevels/boot. Это определяется следующей строкой:  
rc:/bootwait:/sbin/rc boot
```

И снова все необходимые действия выполняются сценарием rc. Заметьте, что параметр, переданный гс (boot), совпадает с названием используемого подкаталога в /etc/runlevels.  
Теперь init проверяет свой конфигурационный файл, чтобы определить, какой уровень запуска использовать. Для этого из /etc/inittab считывается строка:

```
id:3:initdefault:
```

В приведенном примере (который подходит для подавляющего большинства пользователей Calculate) номер уровня запуска --- 3. Пример уровня запуска:

```
10:0:wait:/sbin/rc shutdown  
11:S1:wait:/sbin/rc single  
12:2:wait:/sbin/rc nonetwork  
13:3:wait:/sbin/rc default  
14:4:wait:/sbin/rc default  
15:5:wait:/sbin/rc default  
16:6:wait:/sbin/rc reboot
```

## cl-rebuild

## Размаскировка программ

Как правило, портажи содержат несколько версий программы, часть из которых отмечены как нестабильные. Нестабильная версия вовсе не означает, что она нестабильно работает - просто на данный момент эта версия не прошла достаточного тестирования. Вы можете установить как стабильную, так и нестабильную версию программы.

- скорость обновления - потребуется всего 5-7 минут на полное обновление системы.
- надежность - вы всегда сможете загрузиться в предыдущую систему, если новая по какой-либо причине будет работать нестабильно;
- скрытие обновления - потребуется всего 5-7 минут на полное обновление системы.

Состав программного обеспечения ISO образа можно предварительно модифицировать, воспользовавшись руководством по [интегративной сборке системы](#).

### 1. Проверим доступные версии

```
eix firefox
www-client/firefox
Available versions: *10.0.11 17.0.5 17.0.6 17.0.7 ~21.0 ~22.0
...
...
```

### 2. Размаскируем устанавливаемые пакеты с зависимостями

Удостоверьтесь, что директория /etc/portage/package.keywords/ не пустая, иначе выполните:  
touch /etc/portage/package.{keywords,unmask,use}/{custom}

Выполните размаскировку:  
emerge --autounmask-write =www-client/firefox-22.0

Обновите настройки:  
dispatch-conf

Нажмите "U" для подтверждения внесенных изменений.

### 3. Установим программу

```
emerge firefox
```

- Руководство по обновлению системы
- Обновление из пакетов
  - 1. Обновление овердеса и портежей
  - 2. Обновление программ
  - 3. Обновление файла настроек
  - Обновление из ISO образа
  - 1. Обновите установщик
  - 2. Загрузите ISO образ последней Stage-сборки
- Установите новую версию системы

## Руководство по обновлению системы

Для управления пакетами Calculate Linux использует Portage. Вы можете обновить систему двумя способами:

- Обновление системы из пакетов;
- Обновление из ISO образа.

## Обновление из пакетов

Порядок обновления:

1. Установите пакеты, необходимые для обновления (если они не установлены)
2. Установите обновленные пакеты (если они установлены)
3. Обновите конфигурации (если это необходимо)

## Сценарии инициализации

- Сценарии инициализации
- Уровни запуска
- Процесс загрузки системы
- Сценарии инициализации
  - Как работает init?
  - Что такое уровень запуска?
  - Работа со сценариями инициализации
  - Использование rc-update
  - Что такое rc-update?
  - Добавление и удаление служб
  - Настройка служб
  - Почему нужна дополнительная настройка?
  - Каталог /etc/conf.d
  - Написание сценариев инициализации
  - Мне тоже придется...
- Структура
- Зависимости
- Порядок запуска
- Стандартные функции
- Добавление дополнительных параметров
- Переменные для настройки служб
  - Кто от этого выиграет?
  - Использование программного уровня (softlevel)
  - Использование загрузочного уровня (bootlevel)

## Уровни запуска

### Процесс загрузки системы

При загрузке вашей системы по экрану пробегает много текста. Если присмотреться, заметно, что этот текст не меняется от загрузки к загрузке. Последовательность всех этих действий называется последовательностью загрузки и в той или иной степени постоянна.

Во-первых, загрузчик размещает в памяти образ ядра, который вы указали в файле его конфигурации. После этого ядро загружено и запущено, оно инициализирует относящиеся к ядру структуры и задания, и запускает процесс init.

Этот процесс удастся отследить, если все файловые системы (определенные в /etc/fstab) смонтированы и готовы к использованию. Затем он выполняет несколько сценариев, находящихся в каталоге /etc/init.d, которые запускают службы, необходимые для нормального запуска системы.

## 1. Обновление оверлея и портежей

Репозиторий бинарных пакетов постоянно обновляется, поэтому важно перед установкой или обновлением пакетов иметь свежую версию портежей и оверлея calculate.

Выполнить обновление можно одной командой:

```
cl-update --sync-only
```

После запуска последовательно будут выполнены следующие действия:

- обновятся репозитории;
- обновлены шаблоны;
- применены ревизии.

Если у вас подключены сторонние оверлеи, то нужно также указать опцию "--update-other" ("--o"):

```
cl-update --sync-only --update-other
```

## 2. Обновление программ

Каждый дистрибутив Calculate Linux имеет собственный репозиторий бинарных пакетов, оптимизированными под конкретный дистрибутив. По умолчанию обновление производится именно из бинарных пакетов. Изменить способ обновления по умолчанию на обновление из исходных кодов можно добавив в файл /etc/portage/make.conf строку:

```
FEATURES=" gentl pkg"
```

Дистрибутивы Calculate Linux имеют непрерывный цикл обновлений, отличаются составом пакетов, USE-флагами и масками. Внутренние настройки системы находятся в профиле дистрибутива. Список доступных профилей для вашей архитектуры можно просмотреть, выполнив:

```
cl-update-profile list
```

Изменить профиль можно также командой cl-update-profile, указав имя профиля, например:

```
cl-update-profile CLDx
```

Вы можете использовать профиль из стороннего репозитория, для этого воспользуйтесь параметром "--url".

В Calculate Linux 14 появилась утилита для обновления системы cl-update - документацию по ней вы можете прочитать на следующей странице.

Если вам нужно обновить всю систему целиком, включая обновление оверлеев и портежей, то просто используйте команду:

```
cl-update
```

Порядок обновления в общем случае следующий:

1. Синхронизация репозиториев дистрибутива
2. Если репозитории были обновлены, то выполняются действия egencache и eix-update
3. Обновление ревизии и обновление мира
4. Обновление системы
5. Обновление Python/Perl с пересборкой поврежденных пакетов при необходимости
6. Удаление ненужных пакетов
7. Пересборка модулей ядра при необходимости
8. Пересборка проприetaryных пакетов при необходимости
9. Пересборка пакетов для Xorg-сервера, если в этом есть необходимость
10. Выполнение dispatch-conf

## 3. Обновление файлов настроек

По умолчанию, во время обновления системы конфигурационные файлы программ не переписываются, если вы вносите в них изменения. При обнаружении новых изменений вам будет предложено несколько действий: "PageUp"/"PageDown" - перемещаться по файлу, "q" - заменить существующий файл новым, "z" - удалить новый конфигурационный файл, "q" - прервать работу.

Вы также можете использовать автозамену настроек конфигурационных файлов программ при установке значения переменной `cl_auto_update_set` в файле `/etc/calculate/calculate.env`:

```
[main]
cl_autoupdate_set = on
```

В этом случае внимательно относитесь к производимым модификациям файлов, используя для этого шаблоны.

## Обновление из ISO образа

Обновить систему можно путем установки нового образа в свободный системный раздел. Основные настройки, такие как учетные записи пользователей, настройки сети, точки мониторинга, разрешение экрана и прочие, будут перенесены, дополнительные настройки будут выполнены при помощи шаблонов также на этапе установки.  
Если у вас установлен Calculate Directory Server, убедитесь, что директория `/var/calculate` монтируется с однельного раздела жесткого диска. Если это не так, перенесите свои данные и добавьте точку монтирования в `/etc/fstab`.

Порядок обновления:

Откройте консоль с правами пользователя `root` и выполните следующие действия:

### 1. Обновите установщик

Для корректного обновления всегда используйте последнюю доступную версию пакета `calculate-install`. Чтобы обновить программу, выполните:

```
cl-update -s && emerge calculate-utils
```

### 2. Загрузите ISO образ последней Stage-сборки

На [HTTP](#) и [FTP](#) зеркалах в директории `stages` доступны еженедельные сборки дистрибутива.  
Скачайте последний доступный образ:

```
mkdir -p /var/calculate/remote/linux
cd /var/calculate/remote/linux
wget http://mirror.yandex.ru/calculate/CLD/stages/i686/cld-*-*-*-*-
i686.iso
cl-install
```

Подставьте правильный путь к файлу с образом вашего дистрибутива нужной архитектуры.

### 3. Установите новую версию системы

Если вы обновляете Calculate Directory Server, сохраните копию настроек сервисов и базы `LDAP`, выполнив:

```
cl-backup
```

Перезагрузите компьютер. Для восстановления LDAP-базы и настроек сервера выполните:

## Что такое USE-флаги?

### Смысл USE-флагов

Установливая Calculate (или любой другой дистрибутив), вы выбираете те или иные возможности в зависимости от среды, с которой работаете. Установка сервера отличается от установки рабочей станции, а установка игровой станции - от платформы 3D-рендеринга.

Это касается не только того, какие пакеты устанавливаются, но и какие функции определенных пакетов должны поддерживаться. Если вам не нужен OpenGL, то зачем вам его ставить и встраивать поддержку OpenGL в большинство программ? Если вы не собираетесь использовать KDE, зачем собирать пакеты с его поддержкой, если они работают и без этого?

USE флаги - простой способ описания рабочей среды, чтобы помочь пользователям в выборе того, что устанавливать/активировать, а что - нет. Это позволяет пользователю решить, что же ему на самом деле надо, и облегчить работу с Portage - системой управления пакетами.

### Определение USE-флагов

Рассмотрим USE-флаги. USE-флаг - это ключевое слово, включающее сведения о поддержке и зависимостях определенного понятия или функции. При определении какого-либо USE-флага, Portage узнает, что вам нужна поддержка соответствующей функции. Конечно, это также влияет на сведения о зависимостях пакета.

Давайте рассмотрим конкретный пример - ключевое слово kde. Если в вашей переменной USE нет этого слова, то все пакеты, где поддержка KDE является необязательной, собираются без нее. Все пакеты, где зависимость от KDE является обязательной, устанавливаются без установки библиотек KDE (но зависимости). Если же вы определите ключевое слово kde, то эти пакеты будут собираться с поддержкой KDE, а KDE будет установлен в качестве необходимого.

Правильно определяя ключевые слова, вы создаете систему, подогнанную специально для ваших нужд.

### Какие USE-флаги существуют?

Есть два типа USE-флагов: глобальные и локальные.

- Глобальный USE-флаг используется несколькими пакетами и является системным. Это то, что большинство видят в качестве USE-флагов.

Локальный USE-флаг используется единичным пакетом для настройки определенных параметров самого пакета.

Список доступных глобальных USE-флагов можно найти [здесь](#) или локально в `/usr/portage/profiles/use.desc.`

Список локальных USE-флагов находится в вашей системе в `/usr/portage/profiles/use.local.desc.`

## Использование USE-флагов

### Объявление постоянных USE-флагов

Как сказано ранее, все USE-флаги объявляются в переменной USE. Каждый дистрибутив Calculate Linux имеет свой набор USE-флагов, оптимизированный под конкретные задачи. Профиль, на который ориентируется ваша система, указывается символьной ссылкой `/etc/make.profile`. К концу файла будет добавлен порядковый номер сборки. При следующей загрузке будет использован новый образ со всеми изменениями. При последующих сборках старые файлы с образами будут удалены.

В качестве примера можно посмотреть значение USE флагов профиля Calculate Directory Server:

После загрузки все три слоя будут доступны в директории `/mnt/scratch`.

Вы можете запускать программы, менять настройки, создавать файлы - все ваши изменения будут сохраняться в слое `workspace`, не внося изменений в итоговый образ нового дистрибутива.

Интерактивная сборка происходит в директории `/mnt/build`, являющейся результатом объединения двух слоев - `calculate` и `delta`. Вы также можете видеть все происходящие изменения, выполняния в процессе сборки необходимое тестирование собираемых приложений.

### Процесс сборки системы

В пакет `calculate-build` входит утилита `cl-builder`, которая используетя для перехода в интерактивный режим сборки. Выполните `cl-builder` для подготовки системы к сборке. После выполнения команды приглашения в командной строке изменит свой цвет на коричневый (цвет может быть другим в зависимости от типа терминала) и вы окажетесь в `chroot`-окружении /`mnt/build`. Директории `/proc`, `/dev`, `/dev/pts`, `/usr/calculate/share` базовой системы будут примонтированы автоматически, а также перенесён файл `resolv.conf`. Таким образом, сразу после выполнения `cl-builder` можно приступить к изменению системы. Вы можете обновить дерево портфей (команда `cl-uprdate -sync-only`), а также обновлять, устанавливать или удалять программы. Результат установки программы будет отражаться и на загруженной системе. При этом все ваши действия в загруженной системе не затронут `/mnt/build` и останутся только в слое `workspace`. Для избежания конфликтов в работе программ перед установкой пакетов всегда выполняйте команду `cl-builder`.

По завершении сборки выйдите из `chroot`-окружения, набрав в консоли `exit` либо нажав комбинацию клавиш `Ctrl+D`.

### Шаблоны установки

Шаблоны - это конфигурационные файлы, в которых хранятся изменения настроек программ. Шаблоны могут содержать условные блоки, а также внутренние переменные для более гибкой настройки системы.

Шаблоны утилит Calculate хранятся в директории `/usr/share/calculate/templates`. По аналогии с ними вы можете создать свои шаблоны в директории `/var/calculate/templates`.

### Сохранение внесенных изменений

После того как вы закончили работу над изменениями текущего дистрибутива и вышли из `chroot`-окружения, вы можете создать загрузочный образ LiveCD, включающий все внесённые изменения. Для этого воспользуйтесь командой

```
cl-image iso
```

Загрузочный образ будет создан в файле с расширением `.iso` в директории `/var/calculate/1linux`. Если вы загружались с CD либо USB-Flash, то для всех действий может не хватить оперативной памяти компьютера. Чтобы избежать этого, прмонтируйте свободный раздел жесткого диска либо сетевого диска в директорию `/var/calculate/1linux`.

При загрузке с USB Flash вы можете сохранить все изменения в файле `livedcd.squashfs` на вашей флешке. К концу файла будет добавлен порядковый номер номера сборки. При следующей загрузке будет использован новый образ со всеми изменениями. При последующих сборках старые файлы с образами будут удалены.

## Установка системы

Полученный в результате измениений текущей системы ISO образ на 100% совместим с Gentoo и обладает всеми свойствами Calculate Linux. Систему можно загрузить с Live CD, установить на жесткий диск, записать на USB Flash либо переносной USB-HDD. Возможность модификации полученного дистрибутива с помощью загрузки в Builder-режиме сохраняется. Таким образом, вы можете неограниченное число раз менять состав пакетов обычным для Gentoo образом - через обновление дерева портажей.

## Примеры

### Добавление в дистрибутив CLS браузера Opera, используя загрузочный CD

Выполните следующие шаги:

1. загрузитесь с CD в режиме Builder
2. выполните в терминале: `c1-builder`
3. убедившись что цвет курсора изменился, установим браузер командой: `emerge opera`
4. выйдем из сессии, набрав `exit`
5. при необходимости подмонтируем свободный раздел жёсткого диска: `mount /dev/sdaX /var/calculate`
6. сохраним изменения в новом файле с ISO образом: `c1-image iso`

### Установка CLS на флешку и обновление дерева портажей

Для выполнения этой операции на компьютере должно быть установлено не менее 2 Гб оперативной памяти, т.к. на обновление дерева портажей может потребоваться достаточно большое количество памяти.

Выполните следующие шаги:

1. загрузитесь с CD в обычном режиме
2. установите систему на флешку: `c1-install -d /dev/sdX1` (вместо `sdX1` укажите необходимое устройство, например, `sdb1`)
3. перезагрузите компьютер, выбрав загрузку с флешки, и выберите в меню загрузки режим Builder
4. выполните в терминале команду `c1-builder`
5. убедившись, что курсор изменил цвет, обновите дерево портажей, выполнив `c1-update --sync-only`
6. выйдем из сессии, набрав `exit`
7. обновите файл `livedsquashfs`, выполнив: `c1-image squash`
8. перезагрузите компьютер

При недостаточном объеме оперативной памяти следует установить CLS на жесткий диск в режиме Builder, тогда все изменения будут копироваться на жестком диске. Команда `c1-image squash` при этом будет не доступна, а результат работы можно получить в виде готового ISO образа, при помощи команды `c1-image iso`.

## Системные утилиты

- Системные утилиты
- Управление ПО
- Поиск программ
- Установка и удаление программ
- Исправление зависимостей
- Настройки программ
- Управление сервисами
- Содержимое пакета
- Обеспечение безопасности

```
app-text/ghostscript-7.05.3-r1  
ebuild / app-text/ghostscript-7.05.3-r1 зависит от ebuild /  
net-print/cups-1.1.15-r2 )
```

Два или более пакета, которые вы хотите установить, взаимно зависимы, и в результате их установка невозможна. Скорее всего, это ошибка в дереве портажей. Пожалуйста, выждав время, обновите дерево портажей, и попытайтесь снова. Вы можете также проверить, есть ли эта ошибка в [bugzilla](#), и если нет, сообщить о ней.

## Ошибка извлечения

Пример предупреждения Portage об ошибке извлечения:

```
!!! Fetch failed for sys-libs/ncurses-5.4-r5, continuing...  
(...)  
!!! Some fetch errors were encountered. Please see above for details.  
( !!! Ошибка при извлечении sys-libs/ncurses-5.4-r5, продолжение...  
(...)  
!!! При извлечении произошли ошибки. Подробности выше. )
```

Portage не смогла загрузить исходный код данного приложения и попытается продолжить установку других приложений (если запланирована). Эта ошибка может произойти из-за неправильно синхронизированного зеркала, или из-за того, что ebuild указывает на неверное место. Сервер, где находятся исходные коды, также может почему-либо не работать.

Повторите действие через час, чтобы посмотреть, повторится ли эта ошибка.

## Зашита системного профиля

Пример предупреждения Portage о пакете, защищенным профилем:

```
!!! Trying to unmerge package(s) in system profile. 'sys-apps/portage'  
!!! This could be damaging to your system.  
( !!! Попытка удаления пакетов из системного профиля. 'sys-apps/portage'  
!!! Это может повредить вашей системе. )
```

Вы попросили удалить пакет, входящий в состав базовых пакетов вашей системы. Он отмечен в вашем профиле как обязательный, и его не следует удалять из системы.

## USE-флаги

- USE-флаги
- Что такое USE-флаги?
- Смысл USE-флагов
- Определение USE-флагов
- Какие USE-флаги существуют?
- Использование USE-флагов
- Объявление постоянных USE-флагов
- Объявление временных USE-флагов
- Наследование
- Адаптация всей системы под новые USE-флаги
- USE-флаги отдельных пакетов
- Просмотр доступных USE-флагов

- **ключ отсутствует:** пакет еще не тестирулся в вашей архитектуре. Попросите группу портирования в архитектуру проверить пакет, или протестируйте его за них и сообщите о своих изысканиях в bugzilla.
- **package.mask:** обнаружено повреждение пакета, нестабильность или что-то худшее, и пакет заблокирован специально.
- **profile:** пакет считается не предназначенным для вашего профиля. В случае установки приложение может вызвать сбой системы или просто несвоевременно с использованием профилем.

## Отсутствие нужных пакетов

Пример предупреждения об отсутствии пакета:

```
emerge: there are no ebuilds to satisfy ">=sys-devel/gcc-3.4.2-r4".
!!! Problem with ebuild sys-devel/gcc-3.4.2-r2
!!! Possibly a DEPEND/* DEPEND problem.

( emerge: нет сборок, удовлетворяющих ">=sys-devel/gcc-3.4.2-r4".
!!! Проблема с ebuild sys-devel/gcc-3.4.2-r2
!!! Возможна ошибка в DEPEND/* DEPEND. )
```

Приложение, которое вы пытаетесь установить, зависит от другого пакета, недоступного вашей системе. Пожалуйста, проверьте, есть ли такой запрос в [bugzilla](#), а если нет, сообщите об ошибке. Если вы не сменяете ветви, такого не должно происходить, и это - явная ошибка.

## Неоднозначность названия пакета

Пример предупреждения о повторяющихся именах ebuild:

```
!!! The short ebuild name "aterm" is ambiguous. Please specify
!!! one of the following fully-qualified ebuild names instead:
dev-libs/aterm
x11-terms/aterm
```

```
( !!! Короткое название ebuild "aterm" неоднозначно. Пожалуйста,
!!! вместо него укажите одно из полных названий ebuild:
```

```
dev-libs/aterm
x11-terms/aterm )
```

Название приложения, которое вы собираетесь установить, соответствует более чем одному пакету. Требуется также указать название категории. Portage предложит вам возможные варианты.

## Циклические зависимости

Пример предупреждения Portage о циклических зависимостях:

```
!!! Error: circular dependencies:
ebuild / net-print/cups-1.1.15-r2 depends on ebuild /
app-text/ghostscript-7.05.3-r1
ebuild / app-text/ghostscript-7.05.3-r1 depends on ebuild /
net-print/cups-1.1.15-r2

( !!! Ошибка: циклические зависимости:
```

```
ebuild / net-print/cups-1.1.15-r2 зависит от ebuild /
ebuild / net-print/cups-1.1.15-r2 зависит от ebuild /
```

- Полезное
  - Зависимости пакетов
  - Экономия графика
  - Чистка distfiles
  - Дедрагментация дисков
  - Проверка жесткого диска

## Управление ПО

### Поиск программ

Для быстрого поиска программ служит программа `eix` (пакет `app-portage/eix`), имеющая собственный базу данных для ускорения поиска.

Пример:

```
eix mozilla
eix -S browser
```

Отобразить список установленных пакетов можно при помощи команды:

```
qlist -I
```

Отобразить список установленных пакетов с версией:

```
qlist -IV
```

Отобразить список установленных пакетов с версией и USE флагами:

```
qlist -UV
```

Если вы не нашли интересующий вас программы в дереве порткей, вы можете поискать ее в оверлейх. Для поиска воспользуйтесь сайтом <http://gpo.zugaina.org>. Название оверлея будет справа в нижней строке результата поиска.

Прежде чем установить программу, найденную на сайте, подключите оверлей в вашу систему, выполнив:

```
layman -a <оверлей>
```

Список оверлеев можно получить командой:

```
layman -L
```

### Установка и удаление программ

Для установки и удаления программ используйте программу `emerge` (`sys-apps/portage`).

```
emerge kde-base/kgoldrunner
emerge -C mc
```

В приведенном примере будет установлена игра `kgoldrunner` и удалена программа `mc` (Midnight Commander).

В случае если программа замаскирована, вы можете воспользоваться опцией `--autounmask` для вызова справки по размаскировке пакетов.

Пример установки замаскированной версии пакета:

```
emerge --autounmask =www-client/opera-11.10.2092
```

В конце перечня пакетов, подлежащих установке, вы увидите следующие рекомендации:

```
The following keyword changes are necessary to proceed:  
#required by =www-client/opera-11.10.2092 (argument)  
=www-client/opera-11.10.2092 ~amd64
```

Поместите текст под фразой "The following keyword changes are necessary to proceed:" в файл /etc/portage/package.keywords/custom (комментарии можно отпустить).  
echo "=www-client/opera-11.10.2092 ~amd64" >> /etc/portage/package.keywords/custom

Есть и более простой способ подтверждения размаскировки - использовать dispatch-conf (см. ниже).

## Исправление зависимостей

Когда вы вносите изменения в установленные пакеты, может наступить случай нарушения зависимостей. Чтобы выявить и исправить такие нарушения, мы рекомендуем после обновления или удаления какой-либо программы выполнить команду revdep-rebuild.

Пример:  
revdep-rebuild

## Настройки программ

Во время обновления программы, для предотвращения ошибок, новые файлы настроек создаются с суффиксом .\_c\_f00000\_. Для некоторых сервисов важно после обновления заменять старые настройки новыми. Иногда старые настройки могут привести в нерабочее состояние установочную программу, например, старые скрипты запуска сервисов, расположенные в директории /etc/init.d.

Для своевременной замены конфигурационных файлов используйте программу dispatch-conf.

Пример:  
dispatch-conf

После запуска программы будет показывать отличия новой версии каждого конфигурационного файла от старого. Обратите внимание: если настройки файла были модифицированы Calculate (вы увидите соответствующий комментарий), заменять эти настройки не следует - нажмите клавишу z для пропуска изменений. Для замены файла новым нажмите клавишу u.

## Управление сервисами

Для добавления и удаления скрипта из уровня запуска служит скрипт rc-update.

```
Примеры:  
# Вывести список сервисов  
rc-update show  
# Добавить numlock на уровень запуска default  
rc-update add numlock default  
# Перестать запускать numlock  
rc-update del numlock
```

Показать запущенные сервисы можно командой:  
rc-status - -all

## Содержимое пакета

Получить перечень файлов установленного пакета, а также узнать, какому пакету принадлежит файл в системе, можно при помощи программ qlist и qfile (app-portage/portage-utils).

```
!!! Error: the mail-mta/postfix package conflicts with another package.  
!!! both can't be installed on the same system together.  
Please use 'emerge --pretend' to determine blockers.
```

```
( !!! Ошибка: пакет mail-mta/postfix конфликтует с другим пакетом.  
!!! оба не могут находиться в системе одновременно. Пожалуйста,  
!!! запустите 'emerge --pretend' для выявления блокирующих пакетов. )
```

В файлах ebuild есть специальные поля, сообщающие Portage о зависимостях. Возможны два вида зависимости: зависимость сборки, обозначенная в DEPEND, и зависимость выполнения, обозначенная в RDEPEND. Когда одна из этих зависимостей явно указывает на несовместимость пакета или виртуального пакета, это вызывает блокировку.

Для разблокировки можно отказаться от установки пакета или предварительно удалить конфликтующий пакет. В данном примере можно отказаться от установки postfix или сначала удалить smtp. Также возможно, что два пакета, подлежащие установке, блокируют друг друга. В этом редчайшем случае следует определить, зачем вам устанавливать оба пакета. В большинстве случаев можно обойтись одним.

## Маскировка пакетов

Пример предупреждения о замаскированных пакетах:

```
!!! all ebuilds that could satisfy "bootsplash" have been masked.  
( !!! все сборки, удовлетворяющие "bootsplash", замаскированы.)
```

Пример предупреждения о замаскированных пакетах с указанием причин:

```
!!! possible candidates are:  
- gnome-base/gnome-2.8.0_pre1 (masked by: ~x86 keyword)  
- lm-sensors/lm-sensors-2.8.7 (masked by: -sparc keyword)  
- sys-libs/glibc-2.3.4.20040808 (masked by: -* keyword)  
- dev-util/cvsd-1.0.2 (masked by: missing keyword)  
- media-video/ati-gatos-4.3.0 (masked by: package.mask)  
- sys-libs/glibc-2.3.2-r11 (masked by: profile)
```

( !!! возможные кандидаты:

```
- gnome-base/gnome-2.8.0_pre1 (маскировка: ключ ~x86)  
- lm-sensors/lm-sensors-2.8.7 (маскировка: ключ -sparc)  
- sys-libs/glibc-2.3.4.20040808 (маскировка: ключ -*)  
- dev-util/cvsd-1.0.2 (маскировка: ключ отсутствует)  
- media-video/ati-gatos-4.3.0 (маскировка: package.mask)  
- sys-libs/glibc-2.3.2-r11 (маскировка: profile) )
```

Когда вы собираетесь установить пакет, не предназначенный для вашей системы, выдается ошибка маскировки. Нужно попытаться установить другую программу, существующую для вашей системы, или дождаться, пока пакет станет доступным. Всегда есть причина, по которой пакет замаскирован:

- **ключ ~arch:** пакет недостаточно проверен для помещения в стабильную ветвь. Подождите несколько дней или неделю и повторюйте установку его еще раз.
- **ключ -arch или ключ -\*:** пакет не работает способен в вашей архитектуре. Если вы полагаете, что он работает, сообщите об этом в bugzilla.

## Мегапакеты

У некоторых пакетов в дереве портежей нет содержимого как такого, и они используются для установки набора других пакетов. Например, пакет `kde` полностью устанавливает среду KDE в вашей системе, привлекая различные KDE-пакеты в качестве зависимостей.

Если вы когда-либо захотите удалить из системы такой пакет, запуск `emerge -unmerge` не возымеет должного эффекта, так как пакеты, от которых он зависит, останутся в системе.

В Portage существует возможность удаления остаточных зависимостей, но так как зависимости программы меняются со временем, доступность программного обеспечения, прежде всего требуется полностью обновить всю систему, включая реализацию изменений, произведенных путем модификации USE-флагов. После этого можно запустить `emerge -c` (или `--depclean`), чтобы удалить остаточные зависимости. Когда это сделано, вам потребуется пересобрать приложения, ранее динамически связанные с удаленными пакетами, в которых они теперь не нуждаются.

Со всем этим управляются следующие три команды:

```
emerge -uDNA world  
emerge -ca  
revdep-rebuild
```

## Обеспечение безопасности

У списка файлов пакета `app-portage/portage-utils` есть команда `qlist -ae app-portage/portage-utils`, которая выводит пакет в который входит `qfile`.

Вам также может пригодиться программа `which (sys-apps/which)`, показывающая путь к файлу.

Пример:

```
# which equerry  
/usr/bin/equerry
```

## Когда Portage жалуется...

### Слоты, виртуалы, ветви, архитектуры и профили

Как уже сказано, Portage – чрезвычайно мощная система, поддерживающая множество возможностей, нехватавших других системам управления программами. Чтобы это стало понятно, разберем несколько аспектов Portage, не вникая в подробности.

С помощью Portage разные версии отдельного пакета могут сосуществовать в одной системе. В то время, как другая системы управления стремятся называть пакеты в соответствии с версией (например `freetype` и `freetype2`), в Portage используется технология слотов (*SLOT*), или областей. Пакет присваивается определенный слот своей версии. Пакеты с разными слотами способны сосуществовать в одной системе. Например, у пакета `freetype` есть `ebuild` как со `SLOT="1"`, так и со `SLOT="2"`.

Существуют также пакеты, выполняющие одни и те же функции, но отличающиеся в реализации. Например `metalogd`, `syslogd` и `syslog-ng` являются системными службами журнализирования. Приложения, использующие "системный журнал", не могут зависеть от одной конкретной программы, например от `metalogd`, так как остальные программы ничем не хуже. В Portage предусмотрены виртуальные пакеты: каждая служба журнализирования предоставляет `virtual/syslog`, и в результате в приложениях можно указывать зависимость от `virtual/syslog`.

Программное обеспечение может располагаться в различных ветвях дерева портежей. По умолчанию в системе разрешено только использование стабильных пакетов. Большинство новых программ при поступлении включаются в тестовую ветвь, что указывает на необходимость дополнительного тестирования перед тем, как включить их в стабильные. Хотя в дереве портежей и видны сборочные файлы для таких программ, Portage не станет обновлять их до тех пор, пока они не будут помещены в стабильную ветвь.

Некоторые программы имеют не для всех архитектур. Либо они не работают в определенных архитектурах, либо требуют дополнительного тестирования, или у разработчика нет возможности проверить, работает ли пакет в различных архитектурах.

Каждая установка Gentoo придерживается определенного профиля, который содержит, помимо прочего, список пакетов, необходимых для работы способности системы.

## Блокировка пакетов

Пример предупреждения о заблокированных пакетах (`c-pretend`)

```
[blocks] mail-mta/ssmtp (is blocking mail-mta/postfix-2.2.2-r1)
```

Пример:

```
# список файлов пакета app-portage/portage-utils  
qlist -ae app-portage/portage-utils  
# узнать пакет в который входит qfile
```

Постоянное обновление системы – одно из важнейших мероприятий по обеспечению безопасности.

Можно следить за обновлениями посредством `GLSA-glsas-check` (`app-portage/gentoolkit`).

Просмотреть пакеты, которые необходимо переустановить:

```
glsa-check -p $(glsa-check -t all)
```

Переустановка уязвимых пакетов:

```
glsa -check -f $(glsa-check -t all)
```

Не помешает после этой операции проверить зависимости, см. выше *Исправление зависимостей*.

## Полезное

### Зависимости пакетов

Получить информацию о зависимостях пакетов можно с помощью программы `qdepends` (`app-portage/portage-utils`).

Пример:

```
qdepends -Q python
```

### Экономия графика

Если у Вас дорогой или лимитированный трафик, то можно сэкономить с помощью пакета `getdelta`.

Для его установки, если у вас архитектура `i686`, выполните:

```
ACCEPT_KEYWORDS=-x86 emerge getdelta
```

если `_x86_64`, то:

```
ACCEPT_KEYWORDS=-amd64 emerge getdelta
```

Необходимо добавить в файл `/etc/make.conf` строку:

```
FETCHCOMMAND="/usr/bin/getdelta.sh \$URI \$DIR"
```

Далее все действия не отличаются от обычной установки программ, с той лишь разницей, что качаются дельты исходников, что значительно сокращает объем необходимого на обновление трафика.

```

-rw-r--r-- 1 root root 14202 янв 26 21:36 ChangeLog.bz2
-rw-r--r-- 1 root root 5320 янв 26 21:36 deveoper-guide.txt.bz2
-rw-r--r-- 1 root root 3837 янв 26 21:36 NEWS.bz2
-rw-r--r-- 1 root root 541 янв 26 21:36 README.bz2
-rw-r--r-- 1 root root 703 янв 26 21:36 release-guide.txt.bz2
-rw-r--r-- 1 root root 471 янв 26 21:36 TODO.bz2
-rw-r--r-- 1 root root 2067 янв 26 21:36 user-guide.txt.bz2

```

## Дефрагментация дисков

Современные файловые системы, используемые в Linux, минимизируют фрагментацию дисков, поэтому существует достаточно небольшое количество программ для дефрагментации.

Для файловой системы XFS - пакет `sys-fs/xfsdump` (утилиты для дефрагментации и настройки XFS).

Пример команды для просмотра текущей фрагментации диска:

```
xfs_db -r -c frag /dev/sdax
```

где `X` - номер раздела.

Пример команды для дефрагментирования:

```
xfs_fsr -v /dev/sdax
```

## Проверка жесткого диска

Для проверки жесткого диска на битые секторы используйте утилиту `badblocks`.

```
badblocks -svn -o /sda_log.txt -b 4096 -c 256 -r 2 /dev/sda
```

```

# equery f eselect | less
/usr
/usr/bin
/usr/bin/bashcomp-config
/usr/bin/eselect
...

```

## Удаление пакета

Когда вы захотите удалить пакет из системы, используйте команду `emerge -C` (или `-chipurge`). Это приведет к удалению из системы всех файлов, установленных пакетом, кроме конфигурационных файлов приложения, изменившихся после установки. Сохранение конфигурационных файлов позволяет вернуться к работе с пакетом, если вы когда-нибудь решите снова его установить.

Внимание: Portage не проверяет, зависят ли другие пакеты от удалаемого! Однако вы получите предупреждение, если удаление пакета приведет к неработоспособности системы.

```
emerge -C grimegitc
```

После удаления пакета из системы, остаются пакеты, установленные по зависимостям. Чтобы Portage вывела все когда-то нужные пакеты, которые теперь можно удалить, используйте команду `emerge -C` (или `-depclean`). Мы вернемся к этому ниже.

## Обновление системы

Чтобы система сохранялась в отличной форме (не говоря уже об установке свежайших обновлений, связанных с безопасностью), ее нужно регулярно обновлять. Так как Portage просматривает сборочные файлы только в локальном дереве портхек и оверле, сперва потребуется обновить их. Обновив дерево портхек, вы сможете обновить систему командой `emerge -u world`. В следующем примере мы также пользуемся параметром `-a` (или `-ask`), который поручает Portage вывести список пакетов, которые она собирается обновить, и спросить вас, можно ли продолжать:

```
emerge -ua world
```

Portage будет искать более новые версии установленных приложений. Однако проверяется только версия приложения, явно установленных вами, а не тех, от которых они зависят. Если вы хотите обновить каждый пакет в системе, добавьте аргумент `-D` (или `-deep`):

```
emerge -uDa world
```

Поскольку обновления, относящиеся к безопасности, случаются и в пакетах, которые были установлены по зависимостям, рекомендуется изредка запускать эту команду.

Если вы меняли какие-либо из USE-флагов, возможно, потом вы также захотите добавить параметр `-N` (или `-newuse`). Тогда Portage проверит, требует ли изменение установки новых пакетов или перекомпиляции существующих:

```
emerge -uDNA world
```

## Ускорение загрузки

Calclate Linux использует для загрузки `OpenRC`. Благодаря переходу на бинарные утилиты загрузки и дополнительной оптимизации производительность существенно выросла, система стала загружаться значительно быстрее. В OpenRC предусмотрен режим параллельной загрузки, которая теперь включена по умолчанию в файле `/etc/rc.conf`.

```
rc_parallel="YES"
```

## Оптимизация системы

- Оптимизация системы.
- Ускорение загрузки
- Ускорение запуска приложений
- prelink
- preload
- Повышение производительности
- Уменьшение размера дистрибутива
- Удаление неиспользуемых языков
- Высвобождение дополнительного места

## Ускорение запуска приложений

`Calclate Linux` использует для загрузки `OpenRC`. Благодаря переходу на бинарные утилиты загрузки и дополнительной оптимизации производительность существенно выросла, система стала загружаться значительно быстрее. В OpenRC предусмотрена опция `rc_parallel="YES"`

## Ускорение запуска приложений

### prelink

`prelink` - механизм предварительного связывания пакетов. Выполняется при сборке всех дистрибутивов Calculate Linux. В CLD/CLDX 9.9 `prelink` был добавлен в `etc/conf.d/prelink` опцией:

```
Latest version installed: [ Not Installed ]
Size of files: 59,577 kB
Homepage: http://www.mozilla.com/firefox
Description: Firefox Web Browser
License: | ( MPL-1.1 GPL-2 LGPL-2.1 )
...
```

Существуют вспомогательные средства для ускорения и автоматизации стандартных задач типа поиска по дереву портежей, формирования списка установленных пакетов, принадлежащих какой-либо категории и т.д. Для выполнения быстрого поиска пакета используйте утилита eix. Пример поиска браузера firefox:

```
eix firefox
```

Пример поиска всех пакетов, в описании которых присутствует слово "browsen".

```
eix -S browser
```

## Установка программ

После того, как вы нашли нужное программное обеспечение, его можно легко установить с помощью команды emerge. Вот пример установки пакета gnumetirc:

```
emerge gnumetirc
```

Так как множество приложений зависит друг от друга, любая попытка установить какой-либо пакет программ может повлечь за собой также установку дополнительных пакетов. Не беспокойтесь, Portage справится и с этим. Если вы захотите выяснить, что именно Portage собирается установить вместе с нужным вам пакетом, добавьте параметр -r (или --pretend). Например:

```
emerge -r gnumetirc
```

После команды на установку пакета, Portage загружает из интернета необходимый исходный код (при необходимости), и по умолчанию сохраняет его в каталоге /var/calculate/remote/distfiles. После этого пакет распаковывается, компилируется и устанавливается. Если вы хотите, чтобы Portage только загрузил исходный код без его установки, добавьте к команде emerge параметр -f (или --fetchonly):

```
emerge -f gnumetirc
```

## Обнаружение документации к пакету

Многие пакеты содержат собственную документацию. Иногда USE-флаг doc определяет, следует ли устанавливать документацию к пакету. Проверить наличие USE-флага doc можно командой emerge -U <название пакета>. Пример:

```
emerge -Uv eeselect
```

```
[ebuild R ] app-admin/eeselect-1.2.11 USE="bash-completion -doc" 0 kB
```

USE-флаг doc можно включить или отключить как глобально в файле /etc/portage/make.conf/custom, так и для отдельных пакетов, создав файл в директории /etc/portage/package.use и указать в нём флаг: В главе [USE-флаги](#) этот вопрос описывается более подробно.

Документация от новой установленного пакета обычно находится в подкаталоге каталога /usr/share/doc, соответствующем названию пакета. Кроме того, можно вывести список всех установленных файлов утилой equery. Примеры:

```
# ls -1 /usr/share/doc/eeselect-1.2.11
total 48
-rw-r--r-- 1 root root 296 Янв 26 21:36 AUTHORS.bzz
```

```
PRELINKING="yes"
```

## preload

Начиная с версии 9.9, Calculate Linux Desktop включает утилиту preload. Работа программы становится заметной спустя некоторое время. Старгут в boot-уроне загрузки, демон на протяжении всей работы компьютера анализирует время работы приложений. Впоследствии он уже сам поддерживает необходимые программы и библиотеки в кэш памяти.

Благодаря тому, что preload занимает кэш дисковой системы, это не препятствует обычной работе – более того, ускоряет не только запуск отдельных приложений, но и загрузку системы в целом, создавая подобие параллельной загрузки.

## Повышение производительности

Дистрибутивы Calculate Linux собраны с флагом компиляции, позволяющим запускать систему на любых [i686](#) и [x86\\_64](#)-совместимых процессорах. Вы можете несколько улучшить производительность вновь собранных пакетов, раскомментировав новые значения переменных `CFLAGS` и `CXXFLAGS` в файле /etc/make.conf:

```
CFLAGS="-O2 -march=native -pipe"
CXXFLAGS="$[CFLAGS]"
```

Для повышения производительности системы в целом необходима пересборка всех пакетов.

```
emerge -e system
emerge -e world
```

Будьте внимательны, пересборка пакетов может потребовать определенного опыта, в случае если придется разрешать зависимости во время сборки. При выполнении будут загружены и скомпилированы все пакеты программ, входящие в дистрибутив. На выполнение может потребоваться от 5 часов и более, в зависимости от дистрибутива и производительности компьютера.

## Уменьшение размера дистрибутива

### Удаление неиспользуемых языков

Calculate Linux собирается с поддержкой нескольких языков. Тем не менее после установки системы в переменной LINGUAS файла /etc/make.conf можно оставить только нужный язык или языки.

Пример:

```
LINGUAS="en ru"
```

Если вы готовите систему к установке и планируете в дальнейшем обновлять только из бинарных пакетов собственной сборки, вы можете существенно сократить объем LiveCD-образа, удалив исходники ядра, горячек и, возможно, сам компилятор GCC. Вы можете выиграть в объеме, удалив:

- Portage : 35Mb в образе и 513Mb после установки
- исходники ядра : 76Mb в образе и 367Mb в установленной системе

## Высвобождение дополнительного места

Для удаления из системы неиспользуемых языковых файлов потребуется обновление, которое будет заключаться в пересборке большого количества пакетов. Обратите внимание, что, если вы переопределите данную переменную в /etc/make.conf, все запущенные бинарные пакеты отныне всегда будут компилироваться.

## Загрузка модулей ядра

Большинство пользователей взаимодействует с Portage с помощью команды emerge. Эта глава не призвана заменить страницу справки emerge. Просмотреть все возможные параметры команды emerge можно [здесь](#).

### Формат OpenRC

Для управления автоматической загрузкой модулей ядра в Gentoo используется файл /etc/conf.d/modules. В этом файле можно определить список модулей как для определенной версии ядра, так и для релиза ядра или основной версии ядра. Причем в автозагрузке используется только один параметр, точнее указывающий на версию ядра.

Например, если есть параметры:

```
modules_3="mperf"
modules_3_6="acpi -sriureq"
```

то для ядра 3.6.x будет загружен модуль acpi - sriureq, для всех остальных ядер 3.x будет загружен mperf. Использование только одного параметра затрудняет управление списком загружаемых модулей в зависимости от параметров системы, состава пакетов.

### Система загрузки модулей в Calculate Linux

Для удобного управления загрузкой модулей ядра в /etc/conf.d/modules добавлен код, загружающий модули из конфигурационных файлов в директорию /etc/modules-load.d. Используя шаблоны можно создавать конфигурационные файлы со списками в зависимости от версии ядра, версии определенных пакетов и т.д. Например шаблонами оверлея при установке пакета VirtualBox-modules в автозагрузку будут добавлены модули vboxdrv, vboxnetflt и vboxnetadp в файле /etc/modules-load.d/virtualbox.conf.

### Формат файлов

Конфигурационные файлы описывающих модули находятся в директории /etc/modules-load.d и имеют расширение ".conf". Файлы состоят из списка имен модулей ядра, разделенных переводом строки. Пустые строки и строки начинающиеся с "#" или ";" игнорируются. Списки не поддерживают указания параметров модулей ядра - для указания параметров используйте /etc/modprobe.d.

### Миграция на новый формат

При выполнении обновления eix-sync будет создан файл копии старых настроек /etc/conf.d/modules.old, пользовательские модули будут перенесены в файл /etc/modules-load.d/migrate.conf. В процессе миграции будут пропущены модули управления частотой процессора, т.к. они подгружаются автоматически при изменении параметров частоты.

## Ревизии системы

- Ревизии системы
- Формирование состава пакетов
- Управление world-файллом при помощи шаблонов
- Ревизии системы

### Формирование состава пакетов

По мере установки новых программ формируется системный файл (/var/lib/portage/world), содержащий список установленных пакетов. Файл содержит неполный список пакетов, установленных в системе. В нем отсутствуют пакеты, вычисляемые по зависимостям, например библиотеки. В качестве примера, установив электронные таблицы Gnumeric, выполним:

```
emerge -a gnumeric
```

Latest version available: 3.6.13

## Дерево портежей

### Сборочные файлы ebuild

Говоря о пакетах, мы часто имеем в виду программы, доступные пользователям Calculate через дерево портежей. Дерево портежей - это набор сборочных файлов ebuild, содержащих всю информацию, необходимую Portage для управления программным обеспечением (установки, поиска, извлечения и т.п.) По умолчанию сборочные файлы находятся в /usr/portage и /var/lib/layman/calculate. По второму пути находится оверлей Calculate, который также содержит сборочные ebuild файлы.

Когда Portage выполняет любые действия над пакетами программ, эти действия опираются на сборочные файлы, имеющиеся в системе. Поэтому необходимо регулярно обновлять сборочные файлы, чтобы Portage знал о новых программах, обновлениях, связанных с безопасностью и т.д.

### Обновление дерева портежей

Дерево портежей обычно обновляется с помощью gsups. Обновление выполняется довольно просто, так как запуск gsups обеспечивается командой emerge:

```
emerge --sync
```

Если gsups выполнить невозможно из-за ограниченной межсетевого экрана, дерево портежей все-таки можно обновить из ежедневных «снимков». Для автоматического извлечения и установки в системе новейшего снимка служит утилита emerge-websync:

```
emerge-websync
```

Важно вместе с деревом портежей выполнить обновление оверлея Calculate. Оверлей хранится в Git, его обновление можно выполнить утилитой layman:

```
layman -s calculate
```

Выполнить обновление сразу дерева портежей и оверлея Calculate, а также базы данных программы e10x можно одной командой:

```
e10x-sync
```

## Обслуживание программного обеспечения

### Поиск программ

Для поиска программ в дереве портежей по названию можно использовать встроенные возможности команды emerge. По умолчанию команда emerge --searchdesc (или -S):

```
# emerge --searchdesc pdf
Searching...
[ Results for search key : firefox ]
[ Applications found : 10 ]
* www-client/firefox
```

Например, чтобы найти все пакеты, содержащие "pdf" в названии:

```
emerge --search pdf
```

Для поиска пакетов еще и по тексту описания можно использовать параметр --searchdescs (или -S):

```
# emerge --searchdescs pdf
Searching...
[ Results for search key : firefox ]
[ Applications found : 10 ]
* www-client/firefox
```

## Получение пароля пользователя из службы хранения ядра.

Пароль пользователя, который зашел в систему, может получить только пользователь root.

Свой пароль пользователь получить не может.

Чтобы получить пароль пользователя, необходимо пользователем root выполнить команду:

```
keyctl print $(keyctl request user user_name )
```

где

- user\_name - имя пользователя, который вошел в систему.

Примечание: keyctl входит состав пакета sys-apps/keyutils.

## 3. Работа с Portage

1. Введение в Portage
2. USE-флаги
3. Возможности Portage
4. Переменные среды
5. Файлы и каталоги
6. Настройка с помощью переменных
7. Смешение ветвей программного обеспечения
8. Дополнительные средства Portage
9. Отступление от официального дерева
10. Использование ebuild
11. Введение в Portage
12. Добро пожаловать в Portage
13. Дерево портежей
14. Сборочные файлы ebuild
15. Обновление дерева портежей
16. Обслуживание программного обеспечения
17. Поиск программ
18. Установка программ
19. Обнаружение документации к пакету
20. Удаление пакета
21. Обновление системы
22. Метапакеты
23. Когда Portage жалуется...
24. Слоты, виртуальные, ветви, архитектуры и профили
25. Блокировка пакетов
26. Маскировка пакетов
27. Отсутствие нужных пакетов
28. Неоднозначность названия пакета
29. Циклические зависимости
30. Ошибки извлечения
31. Защита системного профиля

при выполнении пакетный менеджер может предложить установить несколько пакетов - сам app-office/gnumeric, а так же необходимые для работы пакеты gpm -extra/libgsf и x11-libgsf/goffice. В world-файл будет добавлен только пакет app-office/gnumeric, - пакет, который вы указали в качестве параметра emerge.

При удалении Gnumeric:

```
emerge -C gnumeric
```

пакет app-office/gnumeric будет удален из world-файла, при этом все зависимые пакеты останутся в системе.

- Для удаления зависимостей, выполните:

```
emerge -ac
```

При выполнении этой команды сформируется дерево пакетов исходя из списка в world-файле, с включением зависимостей и, в случае наличия в системе установленных и не связанных пакетов, будет предложено их удалить.

### Управление world-файлом при помощи шаблонов

Начиная с 13-й версии Calculate Linux изменилась система управления составом пакетов дистрибутива. В предыдущих версиях системы дерево зависимостей формировалось через так называемые метапакеты, содержащие только необходимые зависимости. Такой подход имел свои плюсы и минусы. В любой момент новая версия одного из мета-пакетов могла включить новую программу или исключить либо принудительно удалить другую. Мета-пакеты также успешно справлялись с возможными блокировками, которые могут возникнуть при формировании дерева зависимостей. Основной недостаток же был в сложности удаления предустановленных программ пользователем.

Сейчас помимо пакетного менеджера, в world-файл могут вносить изменения утилита Calculate. Это позволяет избавиться от мета-пакетов. Во время выполнения синхронизации дерева portejey и overleaf, содержащего шаблоны утилит Calculate командой eix-sync, происходит вызов cl-core, который в свою очередь обрабатывает шаблоны.

При помощи шаблонов формируются версии состава пакетов дистрибутива. К примеру, в случае замены одной программы на другую, вторая будет записана в world-файл вместо первой. Шаблоны позволяют гибко формировать world-файл, учитывая наличие заменяемой программы в системе.

Версию world-файла можно посмотреть в /etc/calculate/init.env, пример:

```
[update]  
world = 22
```

Шаблоны формирования world на данный момент можно посмотреть здесь:

```
/var/lib/layman/calculate/profiles/templates/3.1/6_ac_update_sync/world/
```

Шаблоны делятся на две группы:

- Формирование world-файла с нуля;
- Обновление текущего world: удаление, замена, добавление пакетов;

Шаблоны разделяются на категории, такие как graphics, network, multimedia, и т.д., в каждой из них перечислены необходимые для дистрибутива пакеты.

Шаблоны разделены на версии таким образом, что в каждом из них описаны правила обновления с предыдущей версии, пример:

```
#Удалить geeqie в CLD  
#?os_linux shortname=CLD#  
!media-gfx/geeqie
```

## Добро пожаловать в Portage

Благодаря высокой гибкости и чрезвычайно богатым возможностям Portage можно признать одним из лучших средств управления программным обеспечением из существующих в Linux. Portage написана на Python и Bash.

## #os\_linux\_shortname#

```
# добавить catfish в CLDX
#os_linux_shortname==CLDX#
dev_util/catfish
#os_linux_shortname#
```

```
# заменить chromium на firefox
#?pkg(www-client/chromium)!=#
www-client/firefox
#pkg#
```

Обновить world-файл можно выполнить вручную, при помощи команды:

```
c1-update - - update - rev
```

Заново сформировать world-файл, без учёта установленных программ:

```
c1-update - - rebuild -world
```

Сформировать world-файл исходя из лога установленных программ, без учёта преустановленных программ:

```
regenworld
```

## Ревизии системы

Изменения в формировании файла world также повлияли на систему исправления ошибок/настроек в дистрибутиве. Ранее исправление ошибок происходило при обновлении пакетов и имело следующие недостатки:

- создание новых ревизии пакетов (чаще мета пакетов), для исправления во время полного обновления системы
- переустановка мета пакетов вызывала повторное наложение шаблонов для исправления
- создание ревизий для пакетов, не связанных с этой ошибкой, если исправление нужно внести до выполнения обновлений определенных пакетов
- ошибка не будет исправлена если не будет переустановлен необходимый пакет

Новое исправление ошибок и настроек запускается командой c1-update - - update - rev и не привязана к конкретным пакетам. Команда запускается с обновлением mod1d файла, при выполнении eix-sync.

Все исправления содержатся в шаблонах 6\_ac\_update\_sync/revision. Исправление может быть как скриптом, так и указанием списка пакетов, которые нужно перенастроить. Каждое исправление привязано к конкретной ревизии.

Установленный дистрибутив содержит ревизию системы в файле /etc/calculate/ini.env, поэтому одни и те же исправления не будут применены несколько раз.

Еще одним преимуществом такого подхода является то, что обновление рабочих систем и подготовка образа дистрибутива происходит по одной схеме, что позволяет уменьшить количество ошибок при обновлении системы, не проявляющихся в стейджах.

## calboot

- calboot
- Назначение
- Описание модуля calcmenu.c32
- 1. запоминание значений параметров

## keyexec

### Описание

keyexec предназначен для доступа терминального клиента к удаленному рабочему столу Windows. Программа работает через rdesktop-клиент.

При запуске программы будет передавать в rdesktop введенный пользователем при авторизации пароль, таким образом, использование Windows-приложений в линуксе будет происходить без дополнительных сложностей для пользователя.

За хранение паролей отвечает служба хранения ключей ядра Linux. Более подробно это описано [здесь](#). Для работы программы необходимо наличие модуля pam\_keystore в авторизации PAM, который сохраняет пароль пользователя в ключах ядра.

### Формат запуска

```
keyexec rdesktop[0-9] <параметры>
```

### Примеры

```
Запуск рабочего стола Windows-сервера с IP 192.168.0.1:  
keyexec rdesktop " -p - 192.168.0.1"
```

Запуск рабочего стола Windows-сервера на 4-м рабочем столе, учетная запись берется из домена "calculate":

```
keyexec rdesktop4 " -d calculate -p - 192.168.0.1"
```

Запуск Adobe Photoshop в экране размером 1280x999, с оптимизацией передаваемого трафика:

```
keyexec rdesktop " -s 'C:\Program Files\Adobe Photoshop CS3\photoshop.exe' -d calculate -5 -a 16 -g 1280x999 -p - -T 'Adobe Photoshop CS3' -S standard -ZNDKE 192.168.0.1"
```

## pam\_keystore

### Описание

\_pam\_keystore\_ - pam-модуль предназначенный для хранения логина и пароля пользователя в "службе хранения ключей" ядра Linux.

Необходим для программ keyexec. Используется при монтировании сетевых дисков пользователя при хранении учетных записей на сервере. Модуль входит в состав дистрибутива Calculate Linux Desktop.

### Использование

#### Дистрибутив Gentoo

Для подключения модуля измените файл /etc/pam.d/system-auth следующим образом:

|                          |                               |                             |                           |
|--------------------------|-------------------------------|-----------------------------|---------------------------|
| auth required pam_env.so | auth optional pam_keystore.so | auth sufficient pam_unix.so | auth required pam_deny.so |
|--------------------------|-------------------------------|-----------------------------|---------------------------|

Добавив строку:

|                               |                |
|-------------------------------|----------------|
| auth optional pam_keystore.so | use_first_pass |
|-------------------------------|----------------|

## Пользовательские параметры ядра

Так как создание конфигурационного файла происходит при сборке пакета, то для изменения параметров ядра необходимо создать файл с измененными параметрами, связанный с действием `ac_install_patch`. Например, чтобы включить в ядро файловые системы EXT2, EXT3 и EXT4 можно создать следующий файл:

```
# Calculc config name=openS
install.ac_install_patch=on&merge (sys-kernel/calculate-sources)>=версия
CONFIG_EXT2_FS=y
CONFIG_EXT3_FS=y
CONFIG_EXT4_FS=y
```

## Порядок маскировки ядер

Если ядро находится в hard маске, то это означает что оно еще не готово для полноценного использования: не все необходимые патчи, не готова конфигурация, в ходе эксплуатации выявлены критичные ошибки, ошибки во время сборки. После решения выше перечисленных задач с ядра снимается hard маска, и оно остается в маске по keywords, пока не будет использовано в одном из стейджей.

## Имена ядер

При установке ядра из пакета `calculate-sources` с включенным USE-флагом `vmlinux` происходит компиляция и установка ядра в директорию `/boot`. К именам файлов `vmlinux`, `config`, `initramfs` и `System.map` дописывается окончание, состоящее из версии ядра, архитектуры и сокращенного имени дистрибутива (пример: "-3.9.7-1686-CLD"). В случае наличия таких файлов в директории `/boot`, к старым версиям дописывается окончание ".old".

На ядро и вспомогательные файлы создаются символические ссылки. Для того чтобы не модифицировать загрузчики, имена символьических ссылок не подлежат изменению. Символические ссылки были необходимы для Legacy Grub. Grub 2 сам автоматически определяет используемые файлы для формирования своего конфигурационного файла. В ближайших версиях символьические ссылки и `kernel.old` будут удалены.

При необходимости, сгенерировать настройки Grub можно командой:

```
cl-setup-boot
```

## Удаление старых версий ядра

Старые версии ядра удаляются при удалении соответствующих версий пакета `calculate-sources`. Поэтому, если вы не хотите оставлять их в системе, убедившись, что новое ядро работает normally, удалите ненужные пакеты:

```
emerge -c
```

Если же необходимо удалить только исходный код ядра, оставив собранное ядро, модули и конфигурационный файл - выключите автоматическое удаление в файле `/etc/calculat/ini.env`

```
[update]
remove_kernel1 = off
```

- 2. замещение значений сохраненных параметров
- 3. замещение значений параметров по позиции
- 4. восстановление начальной позиции курсора в меню по параметру
- 5. указание значений по умолчанию для параметров (добавлено в версии 3.86.5)

## Назначение

`calcboot` представляет собой модуль для Syslinux (`calsmenu.c32`), позволяющий запоминать выбор при использовании нескольких загрузочных меню. Также в нем хранятся фоновые изображения для Grub и Syslinux.

## Описание модуля `calsmenu.c32`

`calsmenu.c32` основан на `vesamenu.c32` и реализует как все его возможности, так и дополнительные:

### 1. запоминание значений параметров

При использовании нескольких меню в поле APPEND заносится имя файла меню, к которому нужно перейти при выборе данного пункта. В отличии от стандартного `vesamenu.c32` указанные параметры для меню не сбрасываются.

Пример пункта меню:

```
APPEND isolinux.cfg calculate=be_BY
```

При выборе такого пункта меню будет загружено меню из файла `isolinux.cfg`, а параметр `calculate=be_BY` будет сохранен. В итоге при выборе пункта меню из `isolinux.cfg`, содержащего

```
KERNEL /boot/vmlinuz
APPEND root=/dev/ram0 initrd=/boot/initrd init=/linuxrc looptype=squashfs
unionfs
```

будет запущено ядро с параметрами, указанными в APPEND, плюс `calculate=be_BY`.

### 2. замещение значений сохраненных параметров

Сохраненные параметры не повторяются. То есть если параметр `calculate` уже сохранен, и осуществляется выбор пункта меню, у которого также указан параметр `calculate`, значение `calculate` будет заменено на новое.

### 3. замещение значений параметров по позициям

Каждый параметр может хранить несколько значений, если они разделены запятой. Например, параметр `calculate` будет хранить язык и временную зону (`calculate=ru_RU,Europe/Moscow`). Для реализации этого создаются два различных меню (`lang.cfg`, `timezone.cfg`, пункты меню которых содержат следующие APPEND:

```
lang.cfg
...
APPEND timezone.cfg calculate=ru_RU,
...
...
```

```
timezone.cfg
...
APPEND othermenu.cfg calculate=Europe/Moscow
...
...
```

При выборе таких пунктов меню для `othermenu.cfg` будет сохранен параметр `calculation co` значением `ru_RU_Europe/MOSCOW`.

## 4. Восстановление начальной позиции курсора в меню по параметру

Чтобы запоминалось положение курсора выбора языка, временной зоны и т.д., в файле описания меню используется

`MENUPARAM` параметр `номер`

где "параметр" - название параметра, который хранит выбранное значение, а "номер" - номер позиции в меню. Например, для `timelzone.cfg`, хранящей описание меню с часовыми поясами, используется `MENUPARAM` `calculation 1`

т.е. параметр `calculation`, второе значение. Таким образом, при отображении меню `timelzone.cfg`, если есть сохраненный параметр `calculation` и у него есть значение во второй позиции, то курсор будет указывать на необходимый пункт меню.

## 5. Указание значений по умолчанию для параметров (добавлено в версии

**3.86.5)**

Для указания значений используется

`DEFAULTPARAM` параметры

Например, для указания русского языка по умолчанию:

`DEFAULTPARAM calculate=lang:ru_RU,keyboard:ru_RU`

## calckernel

### Назначение

`calckernel ---` это инструмент, позволяющий автоматизировать компилиацию ядра и создание Ramdisk.

### Ограничие от genkernel

`calckernel` добавляет в Ramdisk ряд возможностей:

- реализация загрузки в режиме интерактивной сборки
- поддержка `aufs` и `unionfs` для LiveCD
- загрузка образа LiveCD в память с autoувеличением размера `tmpfs`
- отмонтирование CD при успешной загрузке образа в память
- использование KMS (Kernel Mode Setting)
- использование `idev` для автоматической загрузки необходимых модулей
- автоматическое определение видео драйвера

- использование параметра ядра `Calcilate` для загрузки определенного видео драйвера

## calculate-sources

Пакет `sys-kernel/calculate-sources` отличается от других пакетов `*sources` из портфеля тем, что позволяет автоматизировать сборку ядра Linux:

- наложение патчей при помощи шаблонов;
- создание конфигурации при помощи шаблонов;
- компиляция ядра и модулей;

## Используемые USE флаги

Для управления порядком сборки используются следующие USE флаги:

- при включенным USE флаге 'umlinit' во время сборки пакета ядро и модули будут скомпилированы и помещены в /boot и /lib соответственно.
- флаг 'minimal' используется для удаления из системы исходного кода собранного ядра, при этом в исходниках остаются только файлы, необходимые для сборки сторонних модулей ядра.

Для сборки ядра и удаления исходного кода `ebuild` использует различные версии `calculate-kernel e-` классов.

## Патчи на ядро

До марта 2013 патчи хранились в архивах `calculate-sources-3.X.tar.bz2`, что приводило к тому, что при обновлении одного из патчей (например при обновлении подверсии ядра) приходилось создавать новые архивы с дополнительными номерами (версии, ревизии и т.д.) и удалять `ebuild`. В связи с изменением способа наложения патчей во время сборки пакета, появилась возможность перенести патчи в шаблоны утилит `Calculate`, получив дополнительные преимущества:

- разделять патчи на десктопные и серверные;
- использовать разные патчи для разных версий ядра `Calclulate-sources` использует следующие основные патчи:
  - `aufs3` - файловая система, производящая каскадно-объединённое монтирование других файловых систем, используемая для запуска LiveCD;
  - `fbcondecor` - декорации в консоли и дополнительные, используемые в десктопных системах:
  - `TuxOnIce` - используется для ускорения "засыпания" и "пробуждения" компьютера;
  - `BFQ` - I/O планировщик, созданный как замена CFQ (и основан на его коде), основная мысль -- более честное разделение I/O между процессами.

## Пользовательские патчи

Выполнение патчей происходит при событии компиляция пакета с действием `ac_install_patch` относительно пути `S ("$ {WORkdir} / ${P}")`, фильтрация по пакетам и версиям осуществляется через функцию `merge()`, формат шаблонов обычно `diff` (представляющие обычные патчи). Таким образом в простейшем варианте необходимо файл патча со следующим заголовком:

```
# Calculate format=diff install.ac_install_patch==on&&merge(sys-  
kernel/calculate-sources)>=версия  
...  
содержимое патча  
...
```

## Конфигурация ядра

До изменения системы наложения патчей конфигурация ядра хранилась в оверлее `profiles/kernel` в виде готового конфигурационного файла как для сервера, так и для десктопа. В связи с изменением метода наложения патчей на пакет изменился и способ создания конфигурационного файла, что дало ряд преимуществ:

- нет разделения настроек на десктопные и серверные ядра (серверное ядро получается из конфигурации ядра, выполнением изменений по шаблону);
- параметры ядра, которые необходимо изменить при наличии патчей вынесены в отдельные шаблоны;
- возможность менять параметры ядра, патчи и маскировку не затрагивая `ebuild`-файлы

```
... USE="acl amd64 bash-completion berkdb bittorrent bzip2 cli cracklib  
crypt cups cxx dri exif foomaticdb fortran gif gpm iconv imap  
ip6 jpeg jpeg2k ldap logrotate maildir mmx modules mudflap multilib  
ncurses nfs nls pth ptl pthonly opengl pam pcrc perl png ppd python radius  
readline samba session sse sse2 ssl sysfs tcpcd tiff truetype unicode  
userlocales xorg zlib" ...
```

Как видите, эта переменная уже содержит достаточно много ключевых слов.

Для изменения значения по умолчанию, нужно добавлять или удалять ключевые слова из переменной USE. Это делается глобально, определив переменной USE в [/etc/make.conf](#). В эту переменную можно добавить нужные вам USE-флаги, или удалить ненужные. Для удаления флага, его надо указывать со знаком минус в виде приставки (<->).

Например, чтобы убрать поддержку KDE и QT, но добавить поддержку ldap, можно определить в [/etc/make.conf](#) переменную USE следующего вида:

```
USE="-kde -qt ldap"
```

Дистрибутивы Calculate Linux по умолчанию используют бинарный профиль, игнорирующий Вашу изменения USE флагов. Чтобы использовать изменения, во время установки или обновления пакетов используйте флаг -N (или --newuse). Пример:

```
emerge -uN world
```

## Объявление USE-флагов для отдельных пакетов

Иногда нужно определить некоторые USE-флаги только для одного или нескольких пакетов, не трогая системных настроек. Для этого необходимо создать файл в каталоге [/etc/portage/package.use](#) и отредактировать его значение. Имя файла может быть любым, удобным для вас.

Например, вам не нужна глобальная поддержка berkdb, но она необходима в mysql. Пример файла [/etc/portage/package.use/mysql](#):

```
dev-db/mysql berkdb
```

Естественно, можно в явном виде отключить USE-флаги для определенного пакета. Например, если вам не нужна поддержка java в PHP. Пример файла [/etc/portage/package.use/php](#):

```
dev-php/php -java
```

## Объявление временных USE-флагов

Иногда необходимо установить какой-то USE-флаг только на один раз. Вместо того, чтобы дважды редактировать [/etc/make.conf](#) (сначала добавить изменение USE, а потом удалить), можно просто объявить USE как переменную среды. Помните, что при переустановке или обновлении приложения (явном или в составе обновления системы) ваши изменения будут утеряны!

Например, уберем java из значения USE на время установки firefox:

```
USE="-java" emerge firefox
```

## Наследование

Конечно же, существует определенная последовательность формирования значения USE. Вы же не хотите объявлять USE="-java" только для того, чтобы узнать, что java все еще включена из-за значения с более высоким приоритетом. После установки значения USE в порядке приоритета (от меньшего к большему) такова:

- значение USE по умолчанию, объявленное пользователем в [/etc/make.conf](#)
- значение, определенное пользователем в файлах [/etc/portage/package.use/](#)
- значение, указанное пользователем в файлах [/etc/portage/package.use/](#)

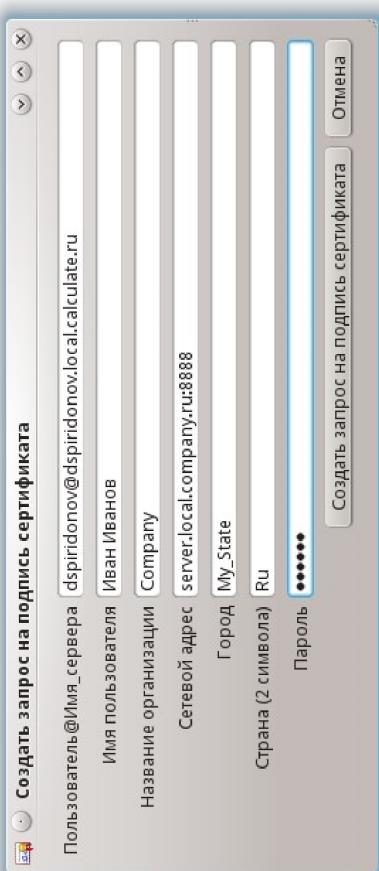


Рис. 3 Создания запроса на подпись сертификата

Большая часть данных, кроме названия организации, города и пароля, будет заполнена автоматически. Не меняйте автоматически подставленные значения, если не уверены, что значения верны. Используйте поле "Пароль", если необходимо дополнительное шифрование закрытого ключа. Введённый пароль будет запрашиваться для установки соединения с сервером утилит.

После создания, запрос автоматически отправится на сервер утилит и появится окно с номером запроса.



Рис. 4 Информация о номере запроса

Подписать запрос можно на сервере утилит через клиента или с помощью команды:

```
cl-core --sign-client n
```

Где n - номер запроса.

После подписания сертификата на сервере, его необходимо забрать нажав на кнопке "Получить сертификат" в окне сертификата. Если запрос ещё не подписан или отвернут, то появится соответствующее сообщение.

## Режимы запуска

Графический клиент может быть запущен в одном из двух режимов:

- запуск в обычном режиме командой

```
cl-console-gui
```

- запуск отдельного метода с помощью ключа -method

```
cl-console-gui -method methodname
```

- значение, определенное пользователем в переменной среды

Чтобы узнать, какие же настройки USE в конечном счете видит Portage, запустите `emerge --info` . Эта команда выводит значения всех переменных (включая USE), используемых Portage:

```
emerge --info
```

## Адаптация всей системы под новые USE-флаги

Если вы изменили свои USE-флаги и хотите обновить всю систему в соответствии с новым значением USE, запустите emerge с параметром --newuse:

```
Primer пересборки всей системы:
```

```
emerge -uDN world
```

Теперь запустите функцию Portage depclean, чтобы удалить условные зависимости, присутствующие в "старой" системе, но больше не нужные при новом составе USE-флагов.

Предупреждение: Запуск emerge -c (или --depclean) является опасной операцией, которую следует использовать с осторожностью. Дважды проверьте список «ненужных» пакетов и убедитесь, что не удаляются нужные пакеты. В следующем примере мы добавляем ключ -a, чтобы depclean потребовал подтверждения перед удалением.

Удаление ненужных пакетов:

```
emerge -ac
```

Когда depclean закончит свою работу, запустите revdep-rebuild, чтобы пересобрать программы, динамически связанные с библиотеками, входящими в потенциально удаленные пакеты:

```
revdep-rebuild
```

После выполнения всех этих действий, ваша система будет полностью использовать новые значения USE-флагов.

## USE-флаги отдельных пакетов

### Просмотр доступных USE-флагов

Возьмем, к примеру, firefox - какие USE-флаги он может использовать? Чтобы это выяснить, запустим emerge с параметрами -r (или --pretend) и -v (или --verbose). Пример:

```
# emerge -pv firefox
```

These are the packages that would be merged, in order:

calculating dependencies... done!

```
[ebuild N ] www-client/firefox-3.6.13 USE="alsa bindist custom-optimization dbus ipc libnotify gnome java startup-notification -system-sqlite -wifi" LINGUAS="bg de en es fr it pl pt_BR ru uk -af -ar -as -be -bn -bn_BD -bn_IN -ca -cs -cy -da -el -en_GB -en_US -eo -es_AR -es_CL -es_ES -es_MX -et -eu -fa -fi -fy_NL -ga -ga_IE -gl -gu -IN -he -hi -hi_IN -hr -hu -id -is -ja -ka -kk -kn -ko -lt -lv -mk -ml -mr -nb -nb_NO -nl -nn -nn_NO -oc -or -pa -pa_IN -pt -pt_PT -ro -si -sk -sl -sq -sr -sv -sv_SE -ta_LK -te -th -tr -vi -zh_CN -zh_TW" 0 kB
```

emerge -e не единственное средство для решения этой задачи. Существует программа eqnucy, специально предназначенная для вывода информации о пакетах. Для просмотра USE-флагов какого-нибудь пакета запустим eqnucy с аргументом uses. Пусть это будет такая группка:

```
# eqnucy uses app-office/gnumeric-1.10.13 -a
[ Searching for packages matching app-office/gnumeric-1.10.13... ]
```

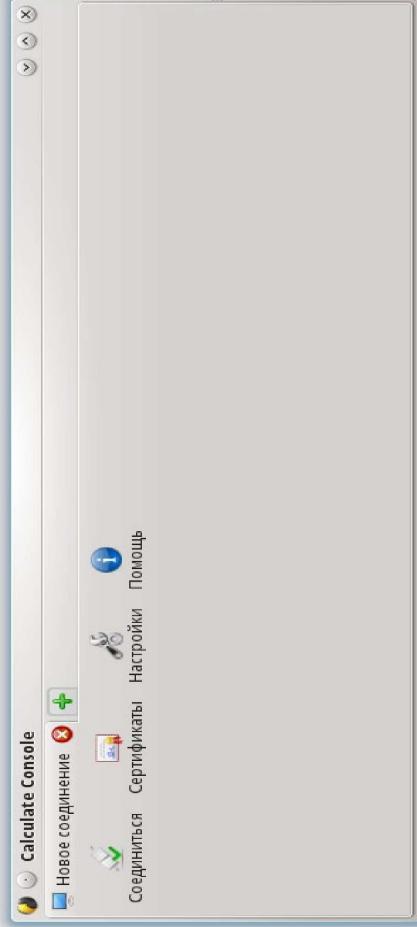


Рис. 1 Внешний вид окна с l-console gui

## Создание сертификата

Для работы с сервером утилит необходим подписаный им сертификат. Для генерации запроса на подписание сертификата от сервера утилит используйте пункт "Сертификаты" в меню.



Рис. 2 Запрос на подпись сертификата

В поле Хост введите адрес хоста сервера утилит, в поле Порт номер порта, который прослушивает сервер (по умолчанию 8888). Используйте кнопку "Отправить запрос" для создания и отправления запроса на сервер, утилит и кнопку "Получить сертификат" для получения подписанныго сертификата с сервера утилит. Для создания запроса и секретного ключа необходимо ввести некоторые данные в окне "Создание запроса на подпись сертификата".

## 4. Утилиты Calculate

1. Графический клиент утилит Calculate
2. Консольный клиент утилит Calculate
3. Сервер утилит Calculate
4. Шаблону утилит Calculate
5. Переменные шаблонов
6. Хранение настроек профиля пользователя
7. Обновление системы cl-update
8. Смена профилей системы cl-update-profile
9. Сборка системы
10. Calculate API

## Графический клиент утилит Calculate

11. Графический клиент утилит Calculate
12. Введение
13. Создание сертификата
14. Режимы запуска
15. Соединение и работа с сервером утилит
16. Панель меню
17. Возврат в Главное меню
18. Запущенные процессы
19. Информация о сессии
20. Настройка программы
21. Помощь
22. Выполнение методов на сервере утилит
23. Безопасность и разграничение прав
24. Запросы
25. Сертификаты
26. Права групп

### Введение

Клиент утилит Calculate служит для сетевого доступа по протоколу [https](https://) к функциям сервера утилит Calculate 3. Установить клиент можно при помощи пакета `SYS-apps/calculate-console-gui`.

Для запуска клиента выполните команду:

```
cl-console-gui
```

```
[ Colour Code : set unset ]  
[ Legend : Left column (U) - USE flags from make.conf  
[           : Right column (I) - USE flags packages was installed with ]  
[ Found these USE variables for app-office/gnumeric-1.10.13 ]  
U I  
- gnome : Adds GNOME support  
+ perl : Adds support/bindings for the Perl language  
++ python : Adds support/bindings for the Python language
```

## Возможности Portage

- Возможности Portage
- Возможности Portage
- Распределенная компиляция
- Использование distcc
- Установка distcc
- Подключение поддержки Portage
- Кэширование компиляции
- О средствах ccache
- Установка ccache
- Подключение поддержки Portage
- Использование ccache для компиляции Си не в Portage
- Поддержка двоичных пакетов
- Создание готовых (заранее собранных) пакетов
- Установка двоичных пакетов

## Возможности Portage

В Portage есть несколько дополнительных возможностей. Многие из этих возможностей полагаются на определенные программы, повышающие производительность, надежность, безопасность и т.п.

Для включения и выключения определенных возможностей Portage нужно редактировать в файле `/etc/make.conf` переменную FEATURES, в которой перечислены ключевые слова, разделенные пробелами, обозначающие различные возможности. Иногда для использования соответствующих возможностей потребуется установка дополнительных утилит.

Здесь перечислены не все возможности, поддерживаемые Portage. Полный перечень представлен на странице справки [/etc/make.conf](#).

Чтобы узнать, какие возможности включены по умолчанию, запустите `emerge --info` и найдите переменную FEATURES (или отфильтруйте ее с помощью grep):

```
emerge --info | grep FEATURES
```

## Распределенная компиляция

### Использование distcc

distcc - программа, распределяющая компиляцию по нескольким, не обязательно одинаковым, машинам в сети. Клиент distcc посылает всю необходимую информацию на доступные серверы distcc (на которых выполняется distccd), чтобы они могли компилировать для клиента части исходного кода. Чистый выигрыш - более быстрая компиляция.

## Установка distcc

Distcc поставляется с графическим монитором (средством контроля), позволяющим отслеживать задачи, которые ваш компьютер отсылает на компиляцию. Если вы используете Gnome, тогда добавьте "gpmote" к переменной USE. А если вы не пользуетесь Gnome, но при этом хотите пользоваться монитором, добавьте "gtk" к переменной USE.

Установка distcc:  
emerge distcc

## Подключение поддержки Portage

Добавьте distcc к переменной FEATURES в файле /etc/make.conf. Затем отредактируйте переменную MAKEOPTS, как вам нравится. Известная рекомендация - указывать директиву "-jX", где X - число центральных процессоров, на которых работает distccd (включая текущий компьютер) плюс один; у вас могут получиться лучшие результаты и с другими значениями.

Теперь запустите distcc-config и введите список доступных серверов distcc. Для простоты примера, предположим, что доступные серверы DistCC - 192.168.1.102 (текущий компьютер), 192.168.1.103 и 192.168.1.104 (два «удаленных» компьютера):

```
distcc-config --set-hosts "192.168.1.102 192.168.1.103 192.168.1.104"
```

Не забудьте также запустить демон distccd:

```
rc-update add distccd default  
/etc/init.d/distccd start
```

## Выполнение действий по настройке

В некоторых приложениях содержатся инструкции по дальнейшей настройке установленного пакета.

Эти инструкции могут потребовать участия пользователя, и, следовательно, не выполняться автоматически. Для запуска шагов настройки, указанных в необязательной функции config() сборочного файла, используйте команду config программы ebuild. Пример настройки пакета:

## Сборка пакета (RPM)

Вы можете попросить Portage создать двоичный пакет или даже RPM из вашего сборочного файла, восторг зевавшихся командами package и rpm, соответственно. Эти команды несколько различаются:

- команда package во многом похожа на merge, выполняя все необходимые шаги (извлечение, распаковку, компиляцию, установку) перед созданием пакета
- команда rpm собирает пакет RPM из файлов созданных после запуска окончания функции install программы ebuild

Пример создания пакетов:

```
(#(создание двоичного пакета, совместимого с Portage)  
ebuild путь/к/файлу-ebuild package  
(#(создание пакета RPM)  
ebuild путь/к/файлу-ebuild rpm
```

Созданный RPM, однако, не будет содержать информацию о зависимостях из сборочного файла ebuild.

## Дополнительная информация

За дополнительными сведениями о системе Portage, программе ebuild и сценариях ebuild обращайтесь к следующим страницам справки man:

```
man portage (сама система Portage)  
man emerge (команда emerge)  
man ebuild (команда ebuild)  
man 5 ebuild (синтаксис файлов ebuild)
```

Кроме того, дополнительные сведения, относящиеся к разработке, находятся в настольной книге разработчика (англ.).

- выполняется функция pkg\_postinst(), если она определена

Запустите функцию rpmmerge программы ebuild, чтобы выполнить этот этап:  
ebuild путь/к/файлу-ebuild qmerge

## Очистка временного каталога

Наконец, можно очистить временный каталог, используя команду clean программы ebuild:  
ebuild путь/к/файлу-ebuild clean

## Дополнительные возможности Ebuild

### Запуск всех команд установки

С помощью функции merge программы ebuild, можно запустить команды извлечения, распаковки, компиляции, установки и помещения за один раз:  
ebuild путь/к/файлу-ebuild merge

## Кэширование компиляции

### O средстве ssache

ssache - это быстрый кеш компилятора. Когда вы компилируете программу, он кэширует промежуточные результаты так, что всякий раз, когда вы перекомпилируете ту же самую программу, время компиляции значительно сокращается. В типичных случаях общее время компиляции может сократиться в 5-10 раз. Более подробно про ssache можно узнать на [сайте проекта](#).

### Установка ssache

Для установки ssache, выполните emerge ssache:  
emerge ssache

## Подключение поддержки Portage

Откройте /etc/make.conf и добавьте ssache к переменной FEATURES. Затем добавьте новую переменную по имени CCACHE\_SIZE (размер кеша), и установите её равной "2G":  
CCACHE\_SIZE="2G"

Для проверки работоспособности ssache, запросите статистику ssache:  
ssache -s

Из-за того, что Portage использует другой домашний каталог ssache, вам также потребуется установить переменную CCACHE\_DIR:  
CCACHE\_DIR="/var/tmp/ccache"

Если контрольная сумма md5 сборочного файла не совпадает с той, что указана в файле `Manifest`, или же один из загруженных файлов не совпадает с описанием в файле `files/digest<пакет>`, вы получите сообщение об ошибке, похожее на такое:

```
!!! File is corrupt or incomplete. (Digests do not match)
>>> our recorded digest: db20421ce35e8e54346e3ef19e60e4ee
>>> your file's digest: f10392b7c0bbcb463ad69642606a7d6

(! ! ! файл поврежден или усечен. (Контрольные суммы не совпадают) )
```

На следующей строке указывается проблемный файл.

Если вы абсолютно уверены, что загруженные исходные коды и сам сборочный файл ebuild именно те, что вам нужны, можете пересоздать файлы `Manifest` и `digest -<пакет>`, используя функцию `digest ebuild` путь/к/файлу -ebuild digest

### Компиляция исходных кодов

Чтобы разархивировать исходные коды в `/var/tmp/portage` (или любой другой каталог, указанный в `/etc/make.conf`), запустите функцию `unpack` программы `ebuild`. Пример распаковки исходных кодов:

```
ebuild путь/к/файлу -ebuild unpack
```

Эта команда выполнит функцию `src_unpack()` программы `ebuild` (которая по умолчанию просто выполняет распаковку, если функция `src_unpack()` не определена). Все необходимые заплатки накладываются также на этом этапе.

### Создание временного места

Следующий шаг в процессе установки -- компиляция исходных кодов. Для этого выполняется функция `src_compile()` вашего сборочного файла. Если нужно, заодно выполняется конфигурация. Пример компиляции исходных кодов:

```
ebuild путь/к/файлу -ebuild compile
```

Если вы хотите изменить инструкции компиляции, советуем отредактировать функцию `src_compile()`. Однако, вы можете также обмануть Portage, заставив ее поверить, что программа `ebuild` уже завершила компиляцию. Запустите нужные команды самостоятельно и создайте пустой файл `.compiled` в рабочем каталоге:

```
touch .compiled
```

### Установка файлов в рабочую файловую систему

Следующий шаг -- установка всех необходимых файлов во временный каталог. В него помещаются все файлы, подлежащие включению в рабочую файловую систему. Вы можете выполнить этот этап, запустив функцию установки программы `ebuild`, которая исполняет функцию `src_install()` сборочного файла:

```
ebuild путь/к/файлу -ebuild install
```

### Помещение файлов в рабочую файловую систему

Последний этап -- перенос всех файлов в рабочую файловую систему и их регистрация в системе Portage. В `ebuild` этот этап называется "cpmerge", и включает следующие действия:

- выполняется функция `pkg_preinst()`, если она определена
- все файлы копируются в рабочую файловую систему Portage
- файлы регистрируются в системе Portage

Домашний каталог `ccache` по умолчанию - `/var/tmp/ccache`; изменить это назначение можно, определив переменную `CCACHE_DIR` в `/etc/make.conf`.

Однако, при запуске `ccache` используется каталог по умолчанию, `$_[HOME]'/ccache`, вот почему при запросе статистики (Portage) `ccache` требует определять переменную `CCACHE_DIR`.

### Использование ccache для компиляции Си не в Portage

Если вы хотите использовать `ccache` для компиляций не в Portage, добавьте `/usr/lib/ccache/bin` в начало вашей переменной PATH (перед `/usr/bin`). Это можно сделать, отредактировав `/etc/env.d/0basics`, который является первым файлом среды, где определяется переменная PATH:  
`PATH="/usr/lib/ccache/bin:/opt/bin"`

### Поддержка двоичных пакетов

#### Создание готовых (заранее собранных) пакетов

Portage поддерживает установку заранее собранных готовых пакетов. Несмотря на то, что для большинства дистрибутивов Calculate Linux есть поддержка заранее собранных пакетов, их количество ограничено составом пакетов дистрибутива. Вам также может понадобиться подготовить пакеты с необходимыми вам [USE-флагами](#).

Чтобы создать двоичный пакет, можно использовать `quicrkpg`, если пакет уже установлен в вашей системе, или `emerge` с параметрами `-buildrpkgs` или `-buildrpkgsonly`.

Если вы хотите, чтобы Portage создавал двоичные пакеты из каждого пакета, который вы будете устанавливать, в `/etc/make.conf` добавьте `buildrpkgs` к переменной FEATURES.

#### Установка двоичных пакетов

Дистрибутивы Calculate Linux Desktop (KDE и XFCE) и Calculate Directory Server используют бинарное хранилище обновлений содержащее пакеты, входящие в образ дистрибутива. Каждый из перечисленных дистрибутивов имеет как обычный профиль, так и бинарный, используемый по умолчанию. В бинарном профиле доступна только одна стабильная версия для каждого двоичного пакета.

Обратите внимание, перед установкой двоичного пакета из хранилища нужно выполнить обновление оверлея Calculate `layman -s calculate`.

Помимо маскировки, профиль устанавливает переменной FEATURES значение `getbinpkgs`, отдавая приоритет установке бинарных пакетов.

Путь к хранилищу указан в переменной `PORTAGE_BINHOST` в профиле дистрибутива. Для примера, в файле `/etc/make.conf` приведены пути к альтернативным хранилищам.

Вы можете устанавливать двоичные пакеты используя и обычный профиль. Для этого указывайте в команде `emerge` параметр `-g` (или `--getbinpkgs`) вместе с параметром `-k` или `-usebinpkgs`. Первый указывает emerge загрузить двоичный пакет с сервера, определенного раньше, а второй сообщает emerge, что до загрузки исходных кодов и их компиляции сначала нужно попытаться установить этот двоичный пакет. Например, чтобы установить `gnutls` из двоичных пакетов:

```
emerge -kg gnutls
```

Подробную информацию о параметрах установки двоичных пакетов можно найти на странице [справки emerge](#).

### Переменные среды

- Переменные среды
- Переменные среды
- Что это такое?

- Важные примеры
- Глобальное определение переменных
- Каталог /etc/env.d
- Сценарий env-update
- Локальное определение переменных
- Пользовательские переменные
- Сессионные переменные

## Переменные среды

### Что это такое?

Переменная среды --- это именованный объект, который содержит информацию, используемую одним или несколькими приложениями. Многие пользователи (особенно новички в Linux) находят этот подход несколько странным или неуправляемым. Но это впечатление ошибочно: используя переменные среды, можно очень легко изменить настройку разнообразных программ.

## Важные примеры

В следующей таблице описывается ряд переменных, используемых в системе Linux. Примеры их значений приведены далее. **Переменная Описание**

PATH В этой переменной содержится список каталогов, разделенных двоеточиями, в которых система ищет исполняемые файлы. Если вы вводите имя исполняемого файла например ls, ls-упрощение или emerge, который не находится ни в одной из перечисленных здесь каталогов, этот файл не запустится (если, конечно, вы не указали полный путь, например /bin/ls).

ROOTPATH У этой переменной такое же значение, что и у PATH, но в ней перечисляются только те каталоги, которые нужно просматривать при вводе команды пользователем с правами root.

LDPATH В этой переменной содержится список каталогов, разделенных двоеточиями, в которых динамический компоновщик ищет библиотеки.

MANPATH В этой переменной содержится список каталогов, разделенных двоеточиями, в которых команда man ищет страницы справки.

INFODIR В этой переменной содержится список каталогов, разделенных двоеточиями, в которых команда info ищет info-страницы.

PAGER В этой переменной содержится путь к программе, позволяющей просмотривать содержимое файлов, например less или more.

EDITOR В этой переменной содержится путь к программе, используемой для изменения файлов, например vi или nano.

KDEDIRS В этой переменной содержится список каталогов, разделенных двоеточиями, в которых находится ресурсы KDE.

CLASSPATH В этой переменной содержится список каталогов, разделенных двоеточиями, в которых находятся классы Java.

CONFIG\_PROTECT В этой переменной содержится список каталогов, защищаемых Portage при обновлении, разделенных пробелами.

CONFIG\_PROTECT\_MASK В этой переменной содержится список каталогов, исключаемых из защиты Portage при обновлении, разделенных пробелами.

Ниже представлен пример определения всех этих переменных:

```
PATH="/bin:/usr/bin:/usr/local/bin:/opt/bin:/usr/games/bin"
ROOTPATH="sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin"
LDPATH="/lib:/usr/lib:/usr/local/lib:/usr/lib/gcc-lib/i686-pc-linux-gnu/3.2.3"
MANPATH="/usr/share/man:/usr/local/share/man"
```

## Программы, поддерживаемые не Portage

### Использование Portage с пакетами самостоятельной сборки

- Иногда вам может потребоваться сконфигурировать, установить и поддерживать программное обеспечение самостоятельно, без автоматизации со стороны Portage, несмотря на то, что оно поддерживается Portage. Наиболее известные случаи --- это исходные коды ядра и драйверы от NVIDIA. Вы можете настроить Portage так, чтобы системе стало известно, что определенные пакеты установлены вручную. Этот процесс называется **внедрение**, и поддерживается Portage посредством файла /etc/portage/profile/package.provided.

Например, если вы захотите сообщить Portage, что пакет vanilla-sources-2.6.37.2 установлен вручную, нужно добавить следующую строку в /etc/portage/profile/package.provided:

```
sys-kernel/vanilla-sources-2.6.37.2
```

## Использование ebuild

- Использование ebuild
- emerge и ebuild
- Ручная установка программ
- Извлечение исходных кодов и проверка контрольных сумм
- Распаковка исходных кодов
- Компиляция исходных кодов
- Установка файлов во временное место
- Помещение файлов в рабочую файловую систему
- Очистка временного каталога
- Дополнительные возможности Ebuild
- Запуск всех команд установки
- Выполнение действий по настройке
- Сборка пакета (RPM)
- Дополнительная информация

## emerge и ebuild

Программа ebuild --- это низкоуровневый интерфейс системы Portage. С ее помощью можно выполнять определенные действия над заданными сбоями ebuild. Например, вы можете самостоятельно выполнить отдельные этапы установки.

Программа ebuild предназначена в основном для разработчиков, поэтому более подробная информация находится в [настольной книге разработчика](#) (англ.). Однако, мы расскажем, какие экземпляры ebuild вызываются системой Portage на разных этапах установки, и как выполнить пост-конфигурационные шаги, которые допускаются некоторыми пакетами.

## Ручная установка программ

### Извлечение исходных кодов и проверка контрольных сумм

Каждый раз, когда вы вызываете ebuild для какого-то ebuild-файла, проверяется совпадение контрольной суммы всех задействованных файлов с указанной в файлах Manifest или files/digest -<имя>-<версия>. Проверка выполняется после загрузки исходных кодов.

Чтобы загрузить исходные коды с помощью ebuild, запустите:

```
ebuild путь/к/файлу-ebuild fetch
```

- Определение оверлейного каталога портажей
- Работа с несколькими оверлейными каталогами
- Программы, поддерживаемые не Portage
- Использование Portage с пакетами самостоятельной сборки

## Использование собственного дерева Portage

### Использование пакета/категории

Вы можете выбирочно обновлять определенные категории/пакеты, игнорируя обновление других категорий/пакетов. Это достигается путем исключения таких категорий/пакетов из списка на этапе выполнения emerge --sync.

Bam потребуется определить имя файла, содержащего шаблоны исключаемых пакетов, в переменной RSYNC\_EXCLUDEFROM в своем файле /etc/make.conf:

```
RSYNC_EXCLUDEFROM=/etc/portage/rsync_excludes
games-*/*
```

Заметьте, однако, что это может привести к проблемам с зависимостями, так как новые разрешенные пакеты могут зависеть от других новых, но исключенных из обновления пакетов.

### Добавление неофициального сборочного файла ebuild

#### Определение оверлейного каталога портажей

Вы можете указать Portage использовать сборочные файлы, не входящие в официальное дерево Portage. Создайте новый каталог (к примеру, /usr/local/portage), в котором будут находиться файлы ebuild сторонних разработчиков. Используйте в точности такую же структуру каталогов, как и в официальном дереве портажей!

Затем определите переменную PORTDIR\_OVERLAY в /etc/make.conf, чтобы она указывала на ранее созданный каталог. Теперь при использовании Portage, эти сборочные файлы будут рассматриваться как часть системы, и не будут удаляться/перезаписываться при последующих запусках emerge --sync.

### Работа с несколькими оверлейными каталогами

Для продвинутых пользователей, ведущих разработку в нескольких оверлейных каталогах, тестирующих пакеты перед включением в основное дерево портажей или просто желающих использовать неофициальные сборочные файлы ebuild из разных источников, в пакете app-portage/layoutan есть утилита layoutan, которая поможет поддерживать ваши оверлейные репозитории в актуальном состоянии.

Настройте layoutan, а затем добавьте интересующие вас оверлеи командой layoutan -a название\оверлея\_Предположим, что у вас есть два дополнительных репозитория с названиями java (для сборочных файлов разработок ведущихся на java) и elatarrs (для внутренних приложений, разработанных на вашем предпринятии). Вы можете обновить эти репозитории следующей командой:

```
layoutan -S
```

```
INFODIR="/usr/share/info:/usr/local/share/info"
PAGER="/usr/bin/less"
EDITOR="/usr/bin/vim"
KDEDIRS="/usr"
CLASSPATH="/opt/blackdown-jre-1.4.1/lib/rt.jar:."
CONFIG_PROTECT="/usr/X11R6/lib/X11/xkb /opt/tomcat/conf \
/usr/share/texmf/tex/generic/config/ \
/usr/share/texmf/tex/plateX/config/ /usr/share/config"
CONFIG_PROTECT_MASK="/etc/gconf"
```

## Глобальное определение переменных

### Каталог /etc/env.d

Для того, чтобы определить эти переменные централизованно, в Calculate существует каталог /etc/env.d. В нём находится ряд файлов, например, 00basic, 05gcc и так далее, в которых определяются переменные, необходимые программам, указанным в названии файлов.

Например, при установке gcc ebuild создает файл /etc/env.d/05gcc, содержащий следующие определения переменных:

```
PATH="/usr/1686-pc-linux-gnu/gcc-bin/3.2"
ROOTPATH="/usr/1686-pc-linux-gnu/gcc-bin/3.2"
MANPATH="/usr/share/gcc-data/i686-pc-linux-gnu/3.2/man"
INFOPATH="/usr/share/gcc-data/i686-pc-linux-gnu/3.2/info"
CC="gcc"
CXX="g++"
LDPATH="/usr/lib/gcc-lib/i686-pc-linux-gnu/3.2.3"
```

В других дистрибутивах вам предлагаются изменять или добавлять определения переменных среды в /etc/profile или где-нибудь еще. Calculate, с другой стороны, облегчает вам поддержку и управление переменными среды, избавляя от необходимости уделять внимание многочисленным файлам, содержащим определения переменных.

Например, когда обновляется gcc, также без малейшего участия пользователя обновляется и /etc/env.d/05gcc.

От этого выигрывает не только Portage, но и вы, пользователь. Иногда от вас может потребоваться глобальная установка какой-нибудь переменной. Возьмем, к примеру, переменную http\_proxy. Вместо того, чтобы возиться с /etc/profile, теперь можно просто создать файл (/etc/env.d/99local) и добавить нужные определения туда:

```
httpd proxy="proxy.server.com:8080"
```

Используя один и тот же файл для всех своих переменных, вы можете быстро увидеть все определенные вами переменные вместе.

### Сценарий env-update

Переменная PATH определяется в нескольких файлах в /etc/env.d. Нет, нет это не ошибка: при запуске env-update различные определения объединяются перед обновлением переменных среды, позволяя пакетам (или пользователю) добавлять собственные значения переменных, не влияя на уже существующие.

Сценарий env-update объединяет значения переменных из файлов, находящихся в /etc/env.d, в алфавитном порядке. Имена файлов должны начинаться с двух десятичных цифр. Порядок обновления, используемый env-update:

Нет никакого смысла в подробном описании интерактивного обновления. Для полноты изложения, мы перечислим возможные команды, которые можно использовать при интерактивном слиянии двух файлов. Вас встречают две строки (одна исходная, вторая измененная) и запрос, в ответ на который можно ввести одну из следующих команд:

```
ed:          редактировать и использовать оба варианта, каждый пометить
заголовком
eb:          редактировать и использовать оба варианта
el:          редактировать и использовать левый вариант
er:          редактировать и использовать правый вариант
e:           редактировать новую версию
l:           использовать левую версию
r:           использовать правую версию
s:           молча включить общие строки
v:           включить общие строки, сообщив подробности
q:           выход
```

Завершив обновление важных файлов конфигурации, вы можете автоматически обновить оставшиеся файлы конфигурации. etc-чредите выйдет, если не найдет других файлов, подлежащих обновлению.

### dispatch-conf

С помощью dispatch-conf можно обновлять файлы конфигурации, сохранив при этом историю изменений. dispatch-conf хранит различия между файлами конфигурации в виде запаток или в системе управления версиями RCS.

Как и с etc-чредите, вы можете попросить сохранить файл конфигурации как есть, использовать новый файл конфигурации, редактировать текущий или обединить изменения интерактивно. Однако, у dispatch-conf также есть приятные дополнительные возможности:

- автоматическое обновление файлов, в которых обновились только комментарии
  - автоматическое обновление файлов, которые отличаются только количеством пробелов
- Убедитесь, что вы сначала определились с /etc/dispatch-conf.conf и создали каталог, прописанный в archive-dir.
- За дополнительными сведениями обращайтесь к странице справки dispatch-conf  
man dispatch-conf

### quickpkg

С quickpkg вы можете создавать архивы из пакетов, уже установленных в системе. Эти архивы можно использовать в качестве двоичных пакетов. Запуск quickpkg прост: только укажите имена пакетов, которые нужно заархивировать.

Например, чтобы поместить в архив curl, arts и procs:

```
quickpkg curl arts procs
```

Двоичные пакеты будут храниться в \$PKGDIR/All (по умолчанию -- /var/calculate/remote/packages). Символьные ссылки, указывающие на эти пакеты, поменяются в \$PKGDIR/<категория>.

## Отступление от официального дерева

- Отступление от официального дерева
- Использование собственного дерева Portage
- Исключение пакета/категории
- Добавление неофициального сборочного файла ebuild

## Файлы и каталоги

```
00basic      99kde-env    99local
+-----+-----+-----+
PATH=/bin:/usr/bin:/usr/kde/3.2/bin:/usr/local/bin"
```

Объединение выполняется не всегда, а только для следующих переменных: KDEDIRS, PATH, CLASSPATH, LDPATH, MANPATH, INFODIR, INFOPATH, ROOTPATH, CONFIG\_PROTECT, CONFIG\_PROTECT\_MASK, PRELINK\_PATH и PRELINK\_MASK. Для всех остальных переменных используется значение, определенное в последнем из файлов (по алфавиту в каталоге /etc/env.d).

При запуске скрипта env-update создаются все переменные среды, и помещаются в /etc/profile (используемый файлом /etc/profile). Кроме того, на основе значения LDPATH создается /etc/ld.so.conf. После этого запускается ldconfig, чтобы вновь создать файла /etc/ld.so.cache, используемый динамическим компоновщиком.

Если вы хотите, чтобы результат работы env-update вступил в силу немедленно, для обновления среды выполните следующую команду:

```
env-update && source /etc/profile
```

Примечание: Эта команда обновляет переменные только в текущем терминале, в новых консолях и их потомках. То есть, если вы работаете в X11, потребуется или набрать source /etc/profile в каждом открываемом терминале, или перезапустить X, чтобы все новые терминалы обращались к новым переменным. Если вы используете диспетчер входа в систему, ставьте пользователем с правами root и наберите /etc/init.d/xdm restart. Если нет, вам придется выйти и снова войти в систему, чтобы X порождала потомков, использующих новые значения переменных.

## Локальное определение переменных

### Пользовательские переменные

Далеко не всегда нужно определять переменные глобально. Например, вам может понадобиться добавить /home/lu\_user/bin в текущий рабочий каталог (где вы находитесь) к переменной PATH, но при этом не нужно, чтобы это добавление появилось и в переменной PATH во всех остальных пользователей. Если вы хотите определить переменную среды локально, используйте ~/.bashrc или ~/.bash\_profile. Пример расширения PATH в ~/.bashrc для локальных нужд:

```
(двоеточие без последующего указания каталога означает текущий рабочий каталог)
PATH="$PATH":~/my_user/bin:"
```

Обновление вашей переменной PATH произойдет, когда вы выйдете и снова войдете в систему.

### Сеансовые переменные

Иногда нужны еще более жесткие ограничения. Вам может потребоваться возможность запуска исполняемых файлов из специально созданного временного каталога без указания полного пути к ним, и без изменения файла ~/.bashrc ради нескольких минут.

В этом случае можно просто определить переменную PATH для текущего сеанса командой export. Переменный будет присвоено временное значение до тех пор, пока вы не завершите сеанс. Пример определения сеансовой переменной среды:

```
export PATH=$PATH:$tmp/usr/bin"
```

Если, к примеру, вы не хотите, чтобы Portage устанавливала исходные коды ядра новее, чем calculate-sources-2.6.36.3, добавьте такую строку в местоположение package.mask:

- Директивы настройки
- Конфигурация, определяемая профилем
- Изменение файлов Portage и размещения каталогов
- Хранение файлов
- Древо Portage
- Двайчные пакеты
- Исходные коды
- Файлы RPM
- База данных Portage
- Кэш Portage
- Сборка программного обеспечения
- Временные файлы Portage
- Каталог сборки
- Размещение «живой файловой системы»
- Ведение журнала
- Журнал Ebuild

## Дополнительные средства Portage

### etc-update

etc-update --- это утилита, предназначенная для обновления в системе файлов .cfg00000\_<имя>. Она обеспечивает интерактивную настройку установки и может также автоматически устанавливать триивальные изменения. Файлы создаются .cfg00000\_<имя> Portage, когда нужно заменить файл в каталоге, защищенным переменной CONFIG\_PROTECT.

Выполнить etc-update довольно просто:

### etc-update

После выполнения триивальных обновлений, вы увидите запрос со списком защищенных файлов, ожидающих обновления. Внизу вам предложат следующие варианты:

```
Please select a file to edit by entering the corresponding number.
(-1 to exit) (-3 to auto merge all remaining files)
(-5 to auto-merge AND not use 'mv -i'):

(Пожалуйста, выберите файл для правки, введя соответствующее число.
(-1 - выход) (-3 - автоустановка всех оставшихся файлов)
(-5 для автоустановки БЕЗ использования 'mv -i') : )
```

При вводе -1, etc-update выходит, прекращая последующие изменения. Если вы введете -3 или -5, все перечисленные файлы конфигурации заменяются более новыми версиями. Следовательно, очень важно сначала отобрать файлы, которые не следует автоматически обновлять. Для этого надо только вводить номер, указанный слева от файлов.

Например, выбираем файл конфигурации /etc/pear.conf:

```
Beginning of differences between /etc/pear.conf and
/etc/.cfg00000_pear.conf
```

```
...
End of differences between /etc/pear.conf and /etc/.cfg00000_pear.conf
1) Replace original with update
2) Delete update, keeping original as is
3) Interactively merge original with update
4) Show differences again
```

Теперь можно увидеть различия между двумя файлами. Если вы считаете, что обновленный файл конфигурации можно использовать без проблем, введите 1. Если вы считаете, что обновленный файл конфигурации не нужен, или не содержит нового или полезную информацию, введите 2. Если вы хотите обновить текущий файл в интерактивном режиме, введите 3.

## Файлы Portage

### Директивы настройки

Настройки Portage по умолчанию хранятся в /etc/make.conf. Когда вы откроете этот файл, вы увидите, что все настройки представляют собой переменные. Что означает каждая из переменных, описано ниже.

Так как многие директивы отличаются в зависимости от используемой архитектуры, к Portage прилагается настройки по умолчанию, которые входят в ваш профиль. На ваш профиль указывает символьическая ссылка /etc/make.profile. Настройка Portage выполняется с помощью файлов make.defaults вашего профилия и всех родительских профилей. Более подробно о профиях и каталоге /etc/make.profile мы расскажем позже.

Если вы планируете вносить изменения в конфигурационные переменные, не изменяйте /etc/make.conf или make.conf. Вместо этого пользуйтесь файлом /etc/make.conf, который имеет приоритет перед вышеуказанными файлами. Вы также обнаружите файл /etc/make.conf.example. Как понятно из его названия, это просто пример - Portage не использует этот файл.

Переменные Portage также можно устанавливать как переменные среды, но мы не рекомендуем этого делать.

### Конфигурация, определяемая профилем

Мы уже встречались с каталогом /etc/make.profile. На самом деле это не каталог; а символьическая ссылка на профиль, по умолчанию на тот, что содержится в /usr/portage/profiles, однако вы можете создавать свои собственные профили где угодно и ссылаться на них. Профиль, указанный ссылкой, является профилем, к которому принадлежит ваша система.

В профиле содержатся сведения для Portage, специфичные для архитектуры, такие как список пакетов, пранадлежащих соответствующей системе, список неработоспособных (или замаскированных) пакетов, и т.д.

## Конфигурация, задаваемая пользователем

Если вам необходимо изменить поведение Portage относительно установки программного обеспечения, вам потребуется отредактировать файлы, находящиеся в `/etc/portage`. Мы настоятельно рекомендуем вам пользоваться файлами из `/etc/portage`, не следует настраивать поведение Portage через переменные среды.

Внутри `/etc/portage` доступны следующие пути:

- директория `package.mask`, в которой можно создать файлы со списком пакетов, для которых Portage никогда не следует устанавливать;
- директория `package.keywords`, в которой можно создать файлы со списком пакетов, для которых вы хотите иметь возможность установки, даже если разработчики Gentoo отговаривают вас от этого;
- директория `package.keywords`, в которой можно создать файлы с перечислением пакетов, которые должны быть доступны для установки, несмотря на то, что они не подходят для вашей системы или архитектуры (пока);
- директория `package.use`, в которой можно создать файлы, где перечислены значения USE-флагов, которые необходимо указывать для конкретных пакетов, а не для всей системы.

Дополнительные сведения о каталоге `/etc/portage`, а также список всех файлов, которые там можно создавать, находятся на справочной странице Portage, см. man portage.

## Изменение файлов Portage и размещение каталогов

Ранее упомянутые конфигурационные файлы нельзя хранить где угодно - Portage всегда ищет свои настроенные файлы в строго определенных местах. Однако Portage также использует множество каталогов для других целей: каталог для сборки, место для хранения исходных кодов, место для дерева Portage, и т.д.

Для этих целей существуют хорошо известные каталоги по умолчанию, положение которых можно изменить на свой вкус, внеся изменения в `/etc/make.conf`. Оставшаяся часть этой главы посвящена описанию того, какие специальные места Portage использует для своих целей, и как изменить их расположение в файловой системе.

Этот документ не претендует на статус справочника. Если вам необходим полный объем информации, покажуйста, обратитесь к странницам справки по Portage и `make.conf` (man portage и man make.conf).

## Хранение файлов

### Дерево Portage

Дерево Portage размещается, по умолчанию, в `/usr/portage`. Это определяется значением переменной PORTDIR. Когда вы храните дерево Portage где-либо в другом месте (изменив эту переменную), не забывайте соответственно изменить символьическую ссылку `/etc/make.profile`. Если вы измените переменную PORTDIR, вам может потребоваться изменить и следующие переменные: PKGDIR, DISTDIR, RPMDIR, так как они не замещают изменений PORTDIR. Это связано с особенностями их обработки Portage.

### Двойочные пакеты

Несмотря на то, что Portage по умолчанию не использует прекомпилированное программное обеспечение, для него предусмотрена очень молчальная поддержка. Если вы укажете Portage работать с прекомпилированными пакетами, они будут разыскиваться в `/var/calculate/remote/packages`.

### Исходные коды

Исходные коды приложений хранятся в `/var/calculate/remote/distfiles` по умолчанию. Это определяется переменной DISTDIR.

Однако будьте готовы к тому, что могут возникнуть проблемы со стабильностью, неудовлетворительной поддержкой пакетов (например неправильные/отсутствующие зависимости), слишком частыми обновлениями (а в результате -- частыми сборками) или невозможностью сбрать пакет. Если вы не знаете, как работает система и как разрешать возникающие проблемы, мы рекомендуем не отходить от стабильной и отестированной ветви.

К примеру, для выбора тестовой ветви на архитектуре x86, отредактируйте `/etc/make.conf` и укажите в нем:

```
ACCEPT_KEYWORDS="-~x86"
```

Если вы запустите обновление системы, то увидите, что многие пакеты нуждаются в обновлении. Обратите внимание, что после перехода на тестовую ветвь и обновления системы, как правило, нет простого пути назад к стабильной официальной ветви (конечно, кроме использования резервной копии).

## Одновременное использование стабильной и тестовой ветвей

### Местоположение package.keywords

Вы можете указать, чтобы Portage использовала тестовую ветвь только для определенных пакетов, а для остальной системы --- стабильную ветвь. Для этого добавьте категорию и имя пакета, для которого вы желаете использовать тестовую ветвь. Создайте файл с любым именем в директории `/etc/portage/package.keywords/` или отредактируйте `/etc/portage/package.keywords/custom`. Например, для использования тестовой ветви для `gnumeric`:

```
app-office/gnumeric ~x86
```

### Тестирование определенных версий

Если вы желаете использовать конкретную версию ПО из тестовой ветви, но не хотите, чтобы Portage использовала тестовую ветвь для последующих версий этого ПО, можно указать в местоположении `package.keywords` номер необходимой версии. В этом случае вы обязаны использовать оператор `=`. Также можно указать диапазон версий, используя операторы

В любом случае, добавляя информацию о версии, вы должны использовать один из этих операторов. Если вы не указываете версию, эти операторы использовать нельзя.

В следующем примере мы просим Portage разрешить установку `gnumeric-1.2.13`:

```
=app-office/gnumeric-1.2.13 ~x86
```

## Использование заблокированных пакетов

### Расположение package.unmask

Если использование пакета было заблокировано, но вы желаете его использовать несмотря на причины блокировки, добавьте для него точно такую же строку, создав файл внутри каталога `/etc/portage/package.unmask`.

Например, если `-net-mail/hotmail-0.8` заблокирован, то разблокировать его можно, прописав в `package.unmask` точно такую же строку:

```
=net-mail/hotmail-0.8
```

### Местоположение package.mask

Если вы не хотите, чтобы Portage использовала какое-то конкретное ПО или конкретные версии ПО, вы можете его самостоятельно заблокировать, добавив соответствующую запись, создав файл внутри каталога `/etc/portage/package.mask`.

## Настройка Gentoo

### Файлы RPM

Несмотря на то, что Portage не может использовать RPM-файлы, есть возможность их создания командой `ebuild` (см. [Использование ebuild](#)). По умолчанию Portage хранит RPM файлы в каталоге `/usr/portage/rpm`, как определяется переменной `RPMDIR`.

### Возможности Portage

Вы можете включить отдельные функции Portage с помощью переменной `FEATURES`. Возможности Portage рассматриваются в предыдущих главах, например [Возможности Portage](#).

### Поведение Portage

#### Распределение ресурсов

Вы можете включить отдельные функции Portage с помощью переменной `FEATURES`. Дополнительная информация о ветвях Gentoo находится в следующей главе.

### Сборка программного обеспечения

#### База данных Portage

Portage хранит состояние вашей системы, какие пакеты установлены, какие файлы относятся к определенным пакетам и т. п.) в `/var/db/pkg`. Не изменяйте эти файлы вручную! Это может разрушить знание вашей системы Portage.

#### Кэш Portage

Кэш Portage (включая сведения о времени изменения, виртуальные пакеты, информацию дерева зависимостей и т. д.) хранится в `/var/cache/edb`. Это место действительно является кэшем: вы можете его очистить в любой момент, когда не запущены приложения, связанные с Portage.

Более подробно о значениях `lsc` написано в странице справки:

#### man lsc

#### Настройки вывода

Переменная `NOCOLOR` (по умолчанию "false") определяет, следует ли Portage отключить цветовую раскраску своих сообщений.

### Сменение ветвей программного обеспечения

- Сменение ветвей программного обеспечения
- Использование одной ветви
- Стабильная ветвь
- Тестовая ветвь
- Одновременное использование стабильной и тестовой ветвей
- Местоположение `package.keywords`
- Тестируование определенных версий
- Использование заблокированных пакетов
- Расположение `package.mask`
- Местоположение `package.pinmask`

### Использование одной ветви

#### Стабильная ветвь

Переменная `ACCEPT_KEYWORDS` определяет, какую из ветвей использовать в вашей системе. По умолчанию используется стабильная ветвь для вашей архитектуры, например `x86`.

#### Тестовая ветвь

Если вы желаете использовать наиболее свежее ПО, подумайте над использованием тестовой ветви. Чтобы Portage начала использовать тестовую ветвь, добавьте «~» перед назначением тестовой ветви.

Тестовая ветвь полностью соответствует своему названию: для тестирования. Если пакет находится в стадии тестирования, это означает, что разработчики считают, что пакет работоспособен, но тщательно он не протестирован. Вы можете оказаться первым, кто столкнется с какой-либо ошибкой. В этом случае вы можете создать [отчет об ошибке](#), чтобы разработчики узнали о ней.

По умолчанию Portage хранит временные файлы в `/var/tmp/portage`. За это отвечает переменная `PORTEAGE_TMPDIR`.

Если вы изменили переменную `PORTEAGE_TMPDIR`, вам может потребоваться изменить и переменную `BUILD_PREFIX`, так как она не замечает изменений `PORTEAGE_TMPDIR`. Это связано с особенностями ее обработки Portage.

#### Каталог сборки

Portage создает специфичные каталоги сборки для каждого пакета внутри `/var/tmp/portage`. Это расположение задается переменной `BUILD_PREFIX`.

#### Размещение «живой файловой системы»

По умолчанию Portage устанавливает все файлы в текущую файловую систему (`/`), но это можно изменить, установив переменную окружения `ROOT`. Это может оказаться полезным при построении новых образов системы.

#### Ведение журнала

#### Журнал Ebuild

Portage может создавать отдельные файлы журнала для каждого файла `ebuild`, но только тогда, когда переменная `PORT_LOGDIR` указывает на место, доступное для записи для Portage (пользователя `portage`). По умолчанию эта переменная не установлена.

### Настройка с помощью переменных

- Настройка с помощью переменных
- Настройка Portage
- Параметры сборки
- Параметры конфигурирования и компиляции
- Параметры установки
- Защита конфигурационных файлов
- Места, защищаемые Portage
- Исключение каталогов
- Параметры скачивания
- Расположение сервера

- Команды для извлечения
- Настройки rsync
- Настройка Gentoo
- Выбор ветви
- Возможности Portage
- Поведение Portage
- Распределение ресурсов
- Настройки вывода

## Настройка Portage

Как отмечалось ранее, Portage настраивается с помощью множества переменных, которые задаются в файле [/etc/make.conf](#).

## Параметры сборки

### Параметры конфигурирования и компиляции

Когда Portage собирает приложения, компилятору и сценарию конфигурации передается значения следующих переменных:

- CFLAGS и CXXFLAGS определяют желаемые флаги компилятора для С и С++
- CHOST определяет информацию об используемой платформе для сценария конфигурации приложения
- MAKEOPTS передается команде make и обычно применяется для установки степени распараллеливания компиляции. Более подробная информация о параметрах команды make находится на странице справки по make.

Переменная USE также используется при конфигурировании и компиляции, но о ней уже много и подробно говорилось в предыдущих главах.

### Параметры установки

Когда Portage устанавливает (merge) новую версию программного продукта, файлы более старых версий удаляются из системы. Portage дает пользователю 5-ти секундную задержку перед стиранием старых версий. Эти 5 секунд задаются переменной CLEAN\_DELAY.

## Защита конфигурационных файлов

### Места, защищаемые Portage

Portage записывает файлы, предоставляемые новой версией программы, поверх старых, если только эти файлы не расположены в защищенном месте. Защищенные каталоги определяются переменной CONFIG\_PROTECT. Обычно, это места расположения файлов конфигурации. Каталоги в списке разделяются пробелами.

Файл, который должен быть записан в такой защищенный каталог, переименовывается, а пользователь получает предупреждение о наличии новой версии (обычно) файла конфигурации.

Узнать текущее значение CONFIG\_PROTECT можно из сообщений emerge -info:

```
emerge --info | grep 'CONFIG_PROTECT='
```

Более подробная информация о защите конфигурационных файлов, осуществляемой системой Portage, доступна по команде emerge:

```
emerge --help config
```

## Изключение каталогов

- Чтобы снять защиту с отдельенных подкаталогов защищенного каталога, можно использовать переменную CONFIG\_PROTECT\_MASK.

## Параметры скачивания

### Расположение сервера

Если запрошенная информация или данные отсутствуют в вашей системе, Portage обращается за ними в интернет. Расположение серверов для различных каналов получчения информации задается следующими переменными:

- GENTOO\_MIRRORS определяет список адресов серверов, содержащих исходный код (distfiles)
- PORTAGE\_BINHOST указывает расположение определенного сервера, содержащего двоичные пакеты (prebuilt packages) для вашей системы

Третья переменная содержит расположение сервера rsync, который используется при обновлении вашего дерева портажей:

- SYNC указывает сервер, с которого Portage извлекает дерево портажей
- Переменные GENTOO\_MIRRORS и SYNC можно установить автоматически программой mirrorselect. Перед тем, как использовать, ее нужно установить, выполнив emerge mirrorselect. За дополнительной информацией обращайтесь к оперативной справке mirrorselect -h help

Если вы вынуждены использовать прокси-сервер, для его указания можно использовать переменные HTTP\_PROXY, FTP\_PROXY и RSYNC\_PROXY.

### Команды для извлечения

Когда Portage требуется извлечь исходный код, по умолчанию используется wget. Вы можете это изменить с помощью переменной FETCHCOMMAND.

Portage может возобновлять скачивание частично загруженного исходного кода. По умолчанию используется wget, но это можно переопределить переменной RESUMECOMMAND.

Удостоверьтесь, что ваши команды FETCHCOMMAND и RESUMECOMMAND сохраняют исходный код в нужном месте. Внутри этих переменных следует использовать \\$[URI] и \\$[DIR], для указания расположения исходных кодов и distfiles, соответственно.

Также существует возможность определить индивидуальные настройки для различных протоколов, используя FETCHCOMMAND\_HTTP, FETCHCOMMAND\_FTP, RESUMECOMMAND\_HTTP, RESUMECOMMAND\_FTP и т.п.

### Настройки rsync

Вы не можете заменить команду rsync, которую Portage использует для обновления дерева портажей, но можно установить несколько переменных, определяющих ее поведение:

- RSYNC\_EXCLUDEFROM указывает на файл, где перечислены пакеты и/или категории, которые rsync должна игнорировать во время обновления.
- RSYNC\_RETRIES определяет, сколько раз rsync должна пытаться соединиться с зеркалом, на которое указывает переменная SYNC. По умолчанию равна 3.
- RSYNC\_TIMEOUT определяет количество секунд, в течение которых rsync соединение может бездействовать, перед тем как rsync соединит его превышшим время ожидания. По умолчанию равна 180, но если вы используете соединение по модему или у вас медленный компьютер, возможно, следует установить значение этой переменной равным 300 или большим.

```
#-disk(/boot/grub) -#
```

получаем значение 0,1  
**elog(pkg)** - получить отметку времени (timestamp) установки указанного пакета. Если пакет не указан, возвращается время последнего установленного пакета. Информация извлекается из `emerge .log`.  
Функция используется для определения необходимости настройки профиля пользователя при входе его в сессию.

где:  
`pkg` - полное название пакета с категорией

**env(service,var\_name)** - чтение значения записанной в переменной шаблона сервиса. Информация получается путем обработки файлов хранения значений переменных шаблонов. где:

- `service` - название сервиса.
- `_var_name_` - переменная сервиса.
- разделитель - '`:`' точка Примеры:
- прочитаем доменное имя для соединения с jabber сервисом

```
#-info(jabber,sr_jabber_host) -#
```

получаем значение `jabber.calculate.ru`

**exists(path,opt)** - проверка существования файла или директории.

Если файл или директория существует, выводит '1', иначе ''

`path` - путь в файловой системе.

`opt` - необходимая опция. Возможные значения "Opt":

- `root` - путь к файлу не будет содержать `chroot`-пути. (Какой путь указан, такой будет в действительности - вне зависимости от переменных `C1_chroot_path` и `C1_root_path`)

Пример. Проверим существование директории `/etc`:

```
#-exists(/etc) -#
```

Результатом будет '1'.

**grep(file,regex)** -\*(добавлена в 3.3.1) получить значение из файла `file`, соответствующее регулярному выражению `gex`. Если в регулярном выражении используется группировка значений, функция возвращает содержимое первой группы.

Начиная с версии 3.3.1.2 преобразовывается \xFF, где FF двузначный шестнадцатеричный код символа.

Пример. Получить значение `nofsck` из `dracut.conf`.

```
#-grep(/etc/dracut.conf,nofscks="(.*?") ) -#
```

Специальные символы регулярного выражения (РВ):

"`"`" - соответствует любому символу, исключая новую строку. При включённом режиме **dotall** соответствует любому символу.

"`\`" - соответствует началу файла. При включённом режиме **multiline** соответствует началу строки.

"`$`" - соответствует концу файла. При включённом режиме **multiline** соответствует концу строки.

"`*`" - соответствует 0 и более повторения предшествующего РВ. РВ будет соответствовать максимально возможной части текста.

"`*?`" - "не жадная" версия использования предыдущего символа. РВ будет соответствовать минимально возможной части текста.

"`+`" - соответствует 1 и более повторения предшествующего РВ. РВ будет соответствовать максимально возможной части текста.

"`+?`" - "не жадная" версия использования предыдущего символа. РВ будет соответствовать минимально

## Соединение и работа с сервером утилит

Для соединения с сервером утилит используйте пункт "Соединиться" в меню.

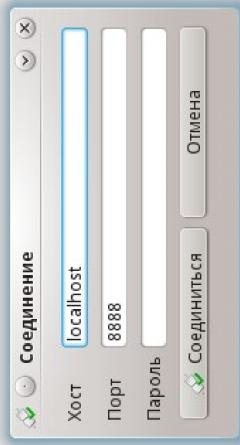


Рис. 5 Установка соединения с сервером утилит

В поле `Хост` введите адрес хоста сервера утилит, в поле `Порт` номер порта, который прослушивает сервер утилит, в поле `Пароль` введите пароль, если он используется для дополнительного шифрования закрытого ключа. Затем нажмите на кнопку `Соединиться`.

В случае успешного соединения, в главном окне появятся доступные на данном сервере утилит методы (см. раздел "Выполнение методов на сервере утилит").

## Панель меню

После установления соединения с сервером утилит верхнее меню приобретает вид, показанный на предыдущем изображении и состоит из кнопок: "Назад" (кроме главного меню), "Процессы", "Сессия", "Отсоединиться", "Настройки" и "Помощь".

## Возврат в Главное меню

Для возврата в главное меню служит кнопка "Назад" в главном меню. При отображении главного меню кнопка не отображается.

## Запущенные процессы

Кнопка "Процессы" в панели меню покажет информацию о запущенных процессах. К работающему процессу можно подключиться, прервать его выполнение или просмотреть результат работы.

- если у функции нет аргумента то сравнивается значение `cl_merge_pkg` и имя директории в которой находится шаблон с функцией `merge()`, в случае совпадения результат - "1" иначе "0"

При выполнении шаблонов для модификации исходного кода пакетов (`ac_install_patch`) функция в случае положительного результата возвращает номер версии вместо 1.

Пример.

Заголовок шаблона1:

```
# Calculate merge(sys-auth/nss_ldap) !=
```

Заголовок шаблона2:

```
# Calculate merge(kde-libs/kdm) !=
```

1. Наложим шаблон для программы `sys-auth/nss_ldap`: Установим значение переменной `cl_merge_pkg` равным `sys-auth/nss_ldap`. Будет применен только шаблон1
2. Наложим все шаблоны. Значение переменной `cl_merge_pkg` по умолчанию - ". Будут применены шаблон1 и шаблон2.

Если параметром для `merge` указан (или определен по пути шаблона) не существующий пакет, то функция возвращает пустое значение. `case(type,var)` - вывод значения переменной шаблона с изменением регистра символов.  
где:

- `type` - тип изменения регистра: `upper` - верхний регистр, `lower` - нижний регистр, `capitalize` - первая буква в верхнем регистре.
- `var` - название переменной шаблона.

Пример. Выведем название хоста в верхнем регистре:  
`#-case(upper,os_net_hostname) -#`

`disk(mount_point,name)` - выводит значение параметра жесткого диска при инсталляции системы. Значение функция получает из переменной (`os_install_disk' + name`, если такой не существует, то `'os_disk'_+ name`), для поиска нужного значения используется переменная `os_install_disk_mount` (точка монтируания при инсталляции). где:

- `_mount_point` - точка монтирования при инсталляции.
- `name` - последний элемент переменных начинавшихся на `os_install_disk/_os_disk`  
(`os_instll_disk'/'os_disk'_+ name`).

значения переменных

```
os_install_disk_mount = ['swap', '/', '', '/var/calculate']
os_disk_grub [0,0, 0,1, 0,2, 0,3, 0,4']
```

функция

```
disk('/var/calculate.grub')
```

- в переменной `os_install_disk_mount` находим `/var/calculate`, получаем индекс 4
- в переменной `os_disk_grub` по индексу 4 получаем значение '0,4'  
результат функции `disk('/var/calculate.grub') 0,4'`
- функция
- в переменной `os_install_disk_mount /boot` не найден, индекс отсутствует
- если индекс отсутствует, в переменной `os_disk_grub` получаем индекс 1
- в переменной `os_disk_grub` по индексу 1 получаем значение 0,1  
результат функции `disk('/boot.grub') 0,1'`

Пример.

Выведем диск для загрузчика grub:



Рис. 6 Просмотр информации о запущенных процессах

## Информация о сессии

Кнопка "Сессия" в панели меню отобразит окно с информацией о текущей сессии с возможностью очистить на сервере кэш процессов, принадлежащих данной сессии.

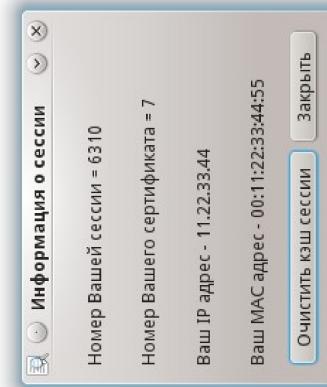


Рис. 7 Информация о текущей сессии Кнопка "Отсоединиться" в панели меню отобразит диалоговое окно закрытия сессии и тремя действиями:

- "Да" - отсоединиться и закрыть сессию.
- "Нет" - Отсоединиться, но не закрывать сессию.
- "Отмена" - не отсоединяться.

Условные блоки - выделенный текст шаблона, подставляемый в случае соответствия *регулярного выражения*.

- **переменные** - метод проверки значений выражений. В качестве значений могут применяться следующие типы данных:
  - числа - целые и дробные числа, в качестве разделителя дробной части выступает точка "."
  - строки - буквы, цифры, специальные символы.
  - **номера версий** - числа и одна или две точки, разделяющих номер версии (например, 8.5.1).

Методом проверки выступает арифметическая операция `>`, Арифметические операции могут объединяться условием "И" (`&`) и "ИЛИ" (`|`). Приоритет в данном случае будет отдаваться условию "И".

Условный блок должен начинаться с метки "#*переменная1==значение1|...|#"* и заканчивается "#*переменная1#*", записанными в начале строки.

```
Пример условного блока файла /etc/make.conf:
#OS_arch_march==i686&&OS_linux_shortname==CLD#
CFLAGS="-O2 -march=i686 -pipe"
HOST="i686-pc-linux-gnu"
#OS_arch_march#
```

В приведенном примере сравниваются переменные `setup_march` и `setup_sys_shortname` со строковыми значениями "i686" и "CLD" соответственно. В случае, если значения обоих переменных совпадают, текст блока будет подставлен в результатирующий файл.

#### Функции

Для формирования сложных файлов, требующих вычисления во время обработки, служат функции. Подобно переменным, функции вставляются в текст шаблона при помощи конструкции "#-функция( ) #-".

Функции, использующие в аргументах путь к файлу (`path`), могут использовать в качестве домашней директории пользователя `~`.

#### Доступные функции:

**belong** - **(не используется начиная с 3.1.1)** - назначение такое же как у `merge`, за исключением, что она работает только с именем пакета, не учитывая категорию.

**merge([category/pkg\_name])** - проверка имени устанавливаемого пакета, если выполняется установка пакета. В остальных случаях функция будет возвращать положительный результат.

`pkg_name` - категория пакета.  
`pkg_name` - имя пакета.

Если у функции нет аргумента, то именем пакета становится имя шаблона (для директорий - имя директории, категорий - родительская директория). При чем если в имени категории или имени пакета есть версия и/или число для сортировки (20-mc,40-xfce-4.6), то они отбрасываются.

Примеры:

```
sys-apps/15-portage-2.2/.calculate_directory,category=sys-fs,pkg_name=udev
sys-fs/udev,(portage файл),category=sys-fs,pkg_name=udev Результат зависит от значения
переменной шаблона cl_merge_pkg (в версиях ниже 3.1.1 cl_belong_pkg)
  • значение cl_merge_pkg "-"
    • результат '1', независимо от того, есть или нет аргумент у функции
  • значение cl_merge_pkg 'имя_пакета'
    • если совпадают значения cl_merge_pkg и аргумента функции, результат - '1' иначе "
```

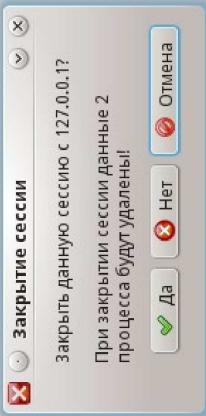


Рис. 8 Закрытие сессии

#### Настройка программы

Кнопка "Настройки" в панели меню служит для вызова окна настроек Calculate Console и состоит из вкладок "Настройки интерфейса" и "Прочие настройки". Вкладка "Настройки интерфейса" включает поля:

- "Расширенный просмотр" - устанавливает режим просмотра результатов работы процесса:
- "расширенный или сокращённый".
- "Высота изображений" - устанавливает фиксированную высоту изображений (0 - убрать изображения).
- "Количество строк в таблице" - устанавливает максимальное количество строк в таблицах, находящихся в результатах работы процесса и отображающихся списками (например, список запросов на подпись сертификата).



Рис. 9 Настройка интерфейса. Вкладка "Прочие настройки" включает поля:

- "Выбор языка" - устанавливает язык интерфейса.
- "Директория с сертификатами" - устанавливает директорию, где находятся сертификаты клиента (при хранении их не в стандартной директории или на внешнем носителе).

- **path=путь** - путь к директории, в которой будет находиться конфигурационный файл
- **name=имя** - имя конфигурационного файла
- **mirror** - выполнять объединение только в случае существования конфигурационного файла. Если конфигурационный файл задан параметром "link" и он не существует, файл назначения удаляется.
- **protected (добавлено в 3.1.1)** - защищить файл при удалении пакета, которому он принадлежит.
- **symbolic** - создать символическую ссылку на файл, указанный параметром "link".
- **autoupdate** - при установке любого пакета запускается специальный скрипт с-1-update-config, который применяет шаблоны для установленываемого пакета. Конфигурационные файлы устанавливаются пакета защищены. Иными словами, при установке пакета, если такой же файл существует в системе и изменен, он не будет заменен. Замена конфигурационных файлов установленных пакетов производится командой **dispatch-conf**. Если в заголовке шаблона есть параметр **autoupdate** то после применения шаблона конфигурационный файл будет скопирован в систему. Если параметр отсутствует то для переноса измененного конфигурационного файла необходимо использовать **dispatch-conf**.
- **run (добавлено в 3.1.1)** - оболочка для выполнения конфигурационного файла, полученного из шаблона. Сценарий выполняется сразу же после обработки этого шаблона.  
Пример: "run=/bin/bash"
- **exec** - оболочка для выполнения конфигурационного файла, полученного из шаблона. Сценарий добавляется в очередь, которая выполняется после обработки всех неисполненных шаблонов. Пример: "exec=/bin/bash" (начиная с версии 3.1.1 выполняемый сценарий не сохраняется на диск)
- **merge** - выполняет в конце настройку перечисленных пакетов.  
Пример: "merge=opencs,grub" (начиная с версии 3.1.1 пакеты необходимо указывать вместе с категорией merge=sys-apps/opencs,sys-boot/grub)
- **postmerge** (добавлено в 3.1.10) - выполнить в конце настройки перечисленных пакетов. В отличие от merge пакеты будут настроены после pkg\_postinst() во время сборки пакета.  
Пример: "postmerge=sys-apps/opencs,sys-boot/grub"
- **env** - (добавлено в 3.1) использовать в шаблоне переменные указанного модуля. Если модуль в системе отсутствует - шаблон будет пропущен.  
Пример: "env=install" Права доступа
- **chmod=XXX** - права доступа к конечному файлу (пример: "644"). По умолчанию права соответствуют ортогональному файлу. Если его нет, права соответствуют файлу шаблона.
- **chown=user:group** - владелец и/или группа конечного файла (пример: "root:root"). Условия
- **переменная[>]**

### Метки

Вы можете настроить систему исходя из аппаратных требований компьютера, сетевых установок и прочих условий. Для этого в файле шаблона вы можете вместо постоянных значений устанавливать метки переменных.

Пример файла /etc/conf.d/hostname:  
HOSTNAME="# - os\_net\_hostname -!"

Для использования переменных из другого модуля перед переменной, через точку указывается модуль (**добавлено в 3.1**):  
Пример:  
#-install.os\_install\_root\_dev -#

**Условные блоки**  
Файл шаблона может содержать условные блоки.

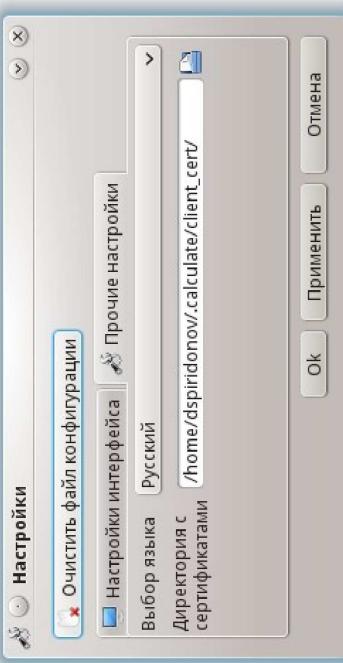


Рис. 10 Прочие настройки программы

### Помощь

Кнопка "Помощь" в панели меню отобразит подменю из пунктов:

- "О программе" - выведет окно с информацией о Calculate Console
- "Справка" - откроет веб-страницу с этим руководством.
- "Сообщить об ошибке" - отобразит окно для отправки сообщения об ошибке разработчикам:

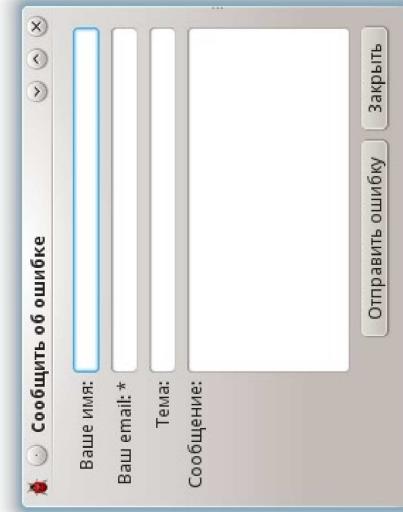


Рис. 11 Окно отправки сообщения об ошибке

## Выполнение методов на сервере утилит

После соединения с сервером утилит в Главном окне появятся доступные для данного сертификата методы:

## Управляющие элементы

Помимо настроек сервисов, записанных в оригинальном формате программы, файлы шаблонов содержат служебные записи, которые условно можно разделить на несколько типов.

### Переменная

Переменная - текстовый элемент в шаблоне, имеющий имя, который заменяется в соответствующем конфигурационном файле значением. Переменная имеет имя, значение, область действия.

- имя - латинские буквы и цифры
- значение - текст для замены (создается в программе)
- область действия - существует глобально для всех шаблонов или локально для одного

Переменные подразделяются на переменные шаблонов и переменные функций.

У переменных шаблонов глобальная область действия, то есть любая переменная шаблона, доступная в программе, может быть использована в любом шаблоне. Значение переменной нельзя изменить в шаблоне.

Переменные функций могут быть созданы в шаблоне, также можно изменить значение переменной в шаблоне. Область действия переменной функции - текущий шаблон.

Для передачи значений переменных функций из текущего шаблона в другой шаблон используется стек переменных функций шаблонов (LIFO), в который при помощи функции шаблонов `push()` записывается значение из текущего шаблона, а функцией шаблона `pop()` получаем значение в другом шаблоне.

### Заголовок

#### Стек переменных функций шаблонов

Стек (LIFO - "последним пришел, первым вышел") для хранения значений переменных функций.

Доступен глобально для всех шаблонов, для работы используются функции шаблонов, `push()` - запись, `pop()` - чтение. Запись и чтение возможны как в одном шаблоне, так и в разных.

### Формат

Заголовок - управляющая запись шаблона, определяющая методы переноса шаблона в систему. Заголовок шаблона записывается первой строкой файла и имеет следующий вид:

```
# Calculate параметр1=значение1 параметр2=[параметр3=значение3 ...]
```

Содержимое заголовка может быть разбито на строки. В этом случае в конце каждой строки заголовка, кроме последней, должен стоять знак "\" (обратная косая черта).

Если заголовок отсутствует, настройки файла шаблона определяются исходя из принятых значений по умолчанию. **Допустимые параметры:**

- **format=[...]** - формат файла шаблона (см. форматы файлов). По умолчанию формат файла шаблона определяется как "raw", или "bin" для файлов, содержащих двоичные данные.
- **comment=[#]** - обозначение начала строкового комментария (пример: "#"). **Объединение** способ объединения устанавливается в соответствии с форматом файла. Если `append=skip` - шаблон пропускается. Если `append=clear`, в случае шаблона файла конфигурационный файл будет очищен - длина файла 0, а в случае шаблона директории все файлы и директории внутри конфигурационной директории будут удалены.
- **force** - удалять существующие файлы перед записью конфигурационного файла. Правило действует по умолчанию, если указан параметр "symbolic".
- **link=путь** - путь к конфигурационному файлу, с которым объединяется файл шаблона. По умолчанию путь совпадает.

Пример: "link=/etc/conf.d/net.example"

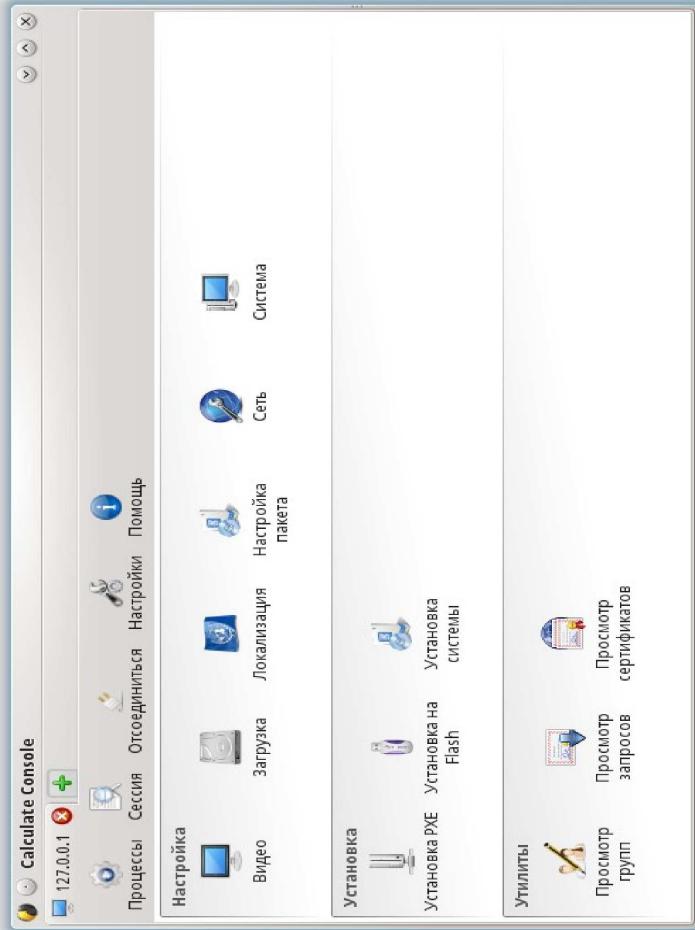


Рис. 12 Главное окно с доступными методами

Все методы разделены по категориям. При открытии метода открывается окно с необходимыми данными. Методы могут состоять из одного шага:

- Поддерживаются форматы XML
- Отличия от формата XMLConf
- Объединение элементов XML
- Примеры использования
  - Формат 'patch'
  - Особенности.
  - Описание.
  - Формат 'diff'
  - Формат 'kernel'
  - Формат 'dconf'
  - Формат 'json'

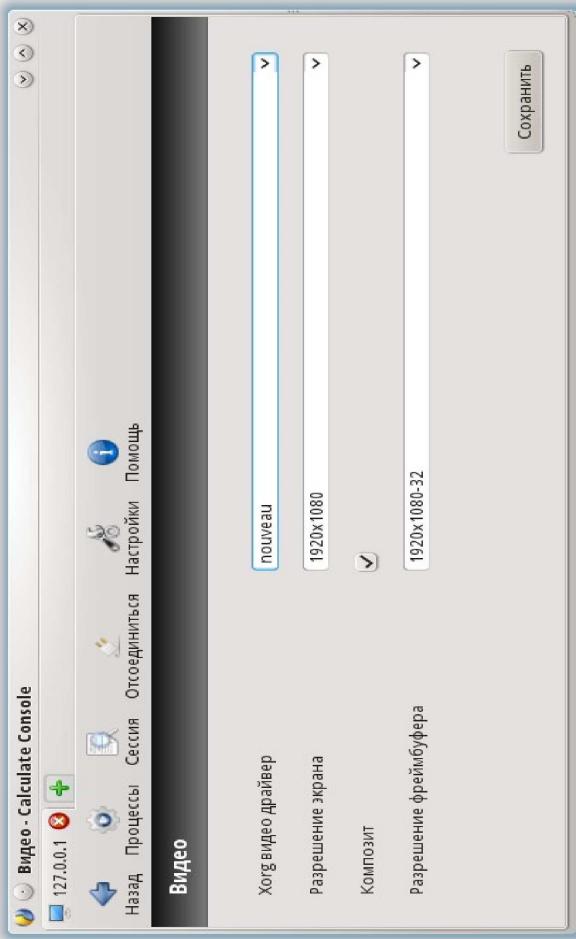


Рис. 13 Метод настройки видеокарты

Или из нескольких шагов:

## Введение

Традиционно в ОС Linux настройки приложений хранятся в текстовых файлах, как правило в директории /etc, реже в /var. Форматы таких конфигурационных файлов различаются: от простых "переменная=значение", до более сложных С-подобных конструкций, либо составленных в XML. Простой на первый взгляд подход обеспечивает исключительную стабильность в работе приложения, так как за сохранность настроек отвечает исключительно файловая система.

Различные дистрибутивы Linux, как правило, предлагают свои программы настройки приложений. К сожалению, такой подход, имея несопротивимое преимущество в удобстве, имеет также ряд недостатков:
 

- пользователь привязывается к определенной программе настройки (дистрибутиву);
- количество настроек, как правило, ограничено интерфейсом программы;
- прямое редактирование настроек становится затруднительным, т.к. программа переписывает файл при внесении изменений.

## Особенности работы с шаблонами

В утилите Calculate 2 и 3 подход к шаблонам был существенно переработан и обладает рядом отличий от первой версии:

- Основной метод переноса шаблонов - объединение с конфигурационными файлами системы. При этом поддерживается все популярные форматы файлов.
- Объединение шаблона с конфигурационным файлом производится посредством конвертации в XML формат. При этом формат файла шаблона может отличаться от конфигурационного файла.
- Файл шаблона может содержать заголовок, описывающий методы объединения.
- Имена встроенных переменных переименованы согласно типу.

## Шаблоны установки

Программа Calculate заменяет прямое редактирование файлов настройки на создание шаблонов.

## Форматы файлов

Согласно методу хранения данных, файлы шаблонов могут иметь один из перечисленных форматов:

- apache, kde, bind, postfix, proftpd, samba, procmail, ldap, dovecot, xmldap, xmldovecot, xmldproftpd, xmlds, xmldsconf, xmldgconf, xmldgconf\_tree, compiz, plasma, squid, dhcp, openrc - форматы файлов настроек распространенных приложений
- bin - двоичный формат файлов
- raw - сырой текст
- patch - шаблон для применения регулярных выражений (использует специальный вид объединения patch).

Для формата kde не обрабатываются параметры '-' и '+' для элементов внутри области. Для применения параметров в openrc-формате нечувствительны к регистру букв.

## Стандартные методы сервера утилит

Все методы сервера утилит разделены на категории. В версии сервера утилит 3.2.0 существуют следующие стандартные категории: Клиент, Настройка, Обновить, Рабочий стол, Установка и Утилиты.

**Клиент**  
Категория Клиент включает в себя методы для изменения поведения системы(локальная/доменная) и для смены паролей пользователей.

**Настройка**  
Категория Настройка включает в себя методы для настройки параметров системы и пакетов.

**Обновить**  
Категория Обновить включает в себя методы для обновления системы, настроек и смены профиля.

**Рабочий стол**  
Категория Рабочий стол включает в себя методы для принудительного выхода пользователяй из сеанса и настройки профилей пользователей.

**Установка**  
Категория Установка включает в себя методы для установки системы.

**Утилиты**  
Категория Утилиты включает в себя методы для работы с сертификатами, запросами и группами прав сертификаторов.

## Шаблоны Calculate

- Шаблоны Calculate
- Введение
- Особенности работы с шаблонами
- Шаблоны установки
- Форматы файлов
- Управляющие элементы
- Способы объединения
- Расположение
- Правила именования файлов
- Правила именования директорий
- Права доступа
- Символические ссылки
- Схема объединения
- Правила объединения
- Правила объединения, действующие по умолчанию
- Изменение правил объединения
- Формат XmlDocument
- Области
- Переменные
- Списки
- Разделённые списки
- Комментарии
- Управляющие элементы
- Формат XML

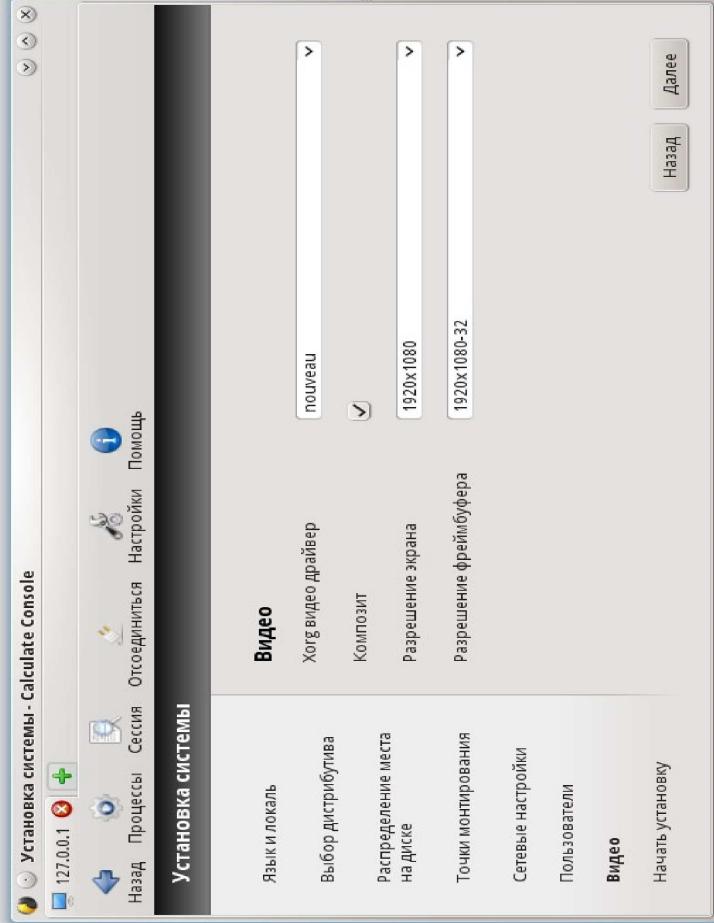


Рис. 14 Настойка видеокарты, как один из шагов метода установки системы  
По шагам можно перемещаться, используя кнопки "Назад" и "Далее" или выбрав нужный шаг в списке слева. Начать выполнение метода можно только из последнего шага.  
При выполнении метода открывается окно с результатами работы процесса. Во время выполнения, процесс можно прервать, нажав на кнопку "Прервать процесс".  
По завершении работы процесса можно вернуться назад с помощью кнопки "Назад" в верхнем меню или закрыть процесс, нажав на кнопку "Закрыть". В этом случае данные о его работе будут удалены на сервере утилит и не будут отображаться в окне "Просмотр информации о запущенных процессах".

## Безопасность и разграничение прав

Как уже упоминалось, для взаимодействия графического клиента с сервером утилит Calculate используется протокол "https", который позволяет зашифровывать передаваемые данные, используя сертификаты и ключи. Процесс получения сертификата описан ранее в разделе "Создание сертификата". В данном разделе речь пойдёт о подписании запросов и разграничении прав доступа по сертификатам.

## Запросы

Запрос на подпись сертификата создаётся на стороне клиента после создания открытого и секретного ключей, и передаётся серверу утилит для подписания сертификата.  
В графическом клиенте для работы с запросами на сервере утилит используется метод "Простмотр запросов" в категории утилит.

**Просмотр запросов - Calculate Console - Выполнение**

| Id | Имя пользователя | IP          | MAC               | Дата                  | Расположение            | Группа      |
|----|------------------|-------------|-------------------|-----------------------|-------------------------|-------------|
| 2  | Иван Иванов      | 11.22.33.44 | 00:11:22:33:44:55 | 2012-06-27 10:52:0... | ivanov.company.com:8888 | Не подписан |
| 3  | Иван Иванов      | 11.22.33.44 | 00:11:22:33:44:55 | 2012-06-27 10:55:0... | ivanov.company.com:8888 | develop     |
| 4  | Иван Иванов      | 11.22.33.44 | 00:11:22:33:44:55 | 2012-06-27 10:57:0... | ivanov.company.com:8888 | Не подписан |
| 5  | Иван Иванов      | 11.22.33.44 | 00:11:22:33:44:55 | 2012-06-27 10:58:0... | ivanov.company.com:8888 | Не подписан |
| 6  | Иван Иванов      | 11.22.33.44 | 00:11:22:33:44:55 | 2012-06-27 10:59:0... | ivanov.company.com:8888 | Не подписан |

**Просмотр запросов**

Закрыть

Рис. 15 Просмотр запросов

Для выбора запроса необходимо нажать на строку таблицы, после чего откроется окно с информацией о запросе, где можно изменить номер запроса (при этом данные о запросе не обновляются) и группу будущего сертификата.

Выбранный запрос можно подписать "Подтвердить" или отказать в подписании "Удалить".

Подписанный запрос, сертификат которого не забран клиентом, будет отображаться в таблице "Просмотр запросов" с называнием группы в столбце "Группа".

Для запуска метода используйте команду  
**cl-core -method МЕТОД**

например:

```
cl-core -method install -iso /path_to_image/cld-x86_64.iso -d
/dev/sda1:swap -d /dev/sda2:/ext4:on
```

Для просмотра справки действий используйте команду  
**cl-core -method МЕТОД -help**

например:

```
cl-core -method install -help
```

Ключ **-f**, --force устанавливает режим, при котором пользователю не задаются вопросы и не отображаются предварительные параметры (brief).  
 Ключ **--progress** отключает отображение прогрессбаров (текущего прогресса выполнения задачи).  
 Ссылки на методы

При установке системы для всех действий на сервере утилит создаются символические ссылки вида **cl-method**, где **method** - название метода, которое можно получить с помощью **cl-core -list-methods**. Например, для метода **setup\_network** ссылкой будет команда **cl-setup-network** (символы "\_" заменяются на "")).

Для создания отсутствующих ссылок и удаления ссылок на отсутствующие действия сервера утилит используйте команду  
**cl-core --create-symlink**

Выполнить команду **cl-core** и все символические ссылки на неё может только пользователь root. Все символические ссылки работают по принципу вызова метода сервера утилит - только на локальной машине без использования сертификатов и шифрования.

## Другие действия

Проверить конфигурацию (наличие сертификата и секретного ключа, их соответствие друг другу, а также действителен ли сертификат) можно с помощью ключа **--check**

Указать путь к файлу журнала событий (логов) можно с помощью ключа **-log-path**, например:  
**cl-core -log-path /var/log/calculate/my\_logs/**

Файл логов по умолчанию - **/var/log/calculate/logging\_cl\_core.out**.

Указание пути к PID файлу осуществляется с помощью ключа **-pid-file PIDFILE** Дополнительно есть две опции для отображения прогресса выполнения в GUI:

- **--gui-progress** - отображает индикатор прогресса в GUI
- **--gui-warning** - отображает предупреждения в конце

Передача пароля со стандартного входа осуществляется с помощью ключа **-P**, например:  
**cat pass | cl-core -P -method install -u test -f**

Опция **-P** должна использоваться вместе с опцией **-f**, т.к. при перенаправлении потоков ввода/вывода использование интерактивного режима невозможно.

Подписание осуществляется корневым сертификатом.

Если необходимо отвергнуть запрос сервера утилит, используйте ключ --del-server-req с указанием номера запроса, например

```
cl-core - -del-server-req 4
```

Для отзыва подписанного сертификата сервера утилит (добавление в список отзыва) используйте ключ --revoke-cert с указанием номера сертификата, например

```
cl-core - -revoke-cert 4
```

Для удаления списка отзывов сертификатов, используйте команду

```
cl-core - -revoke-cert rm
```

#### Изменение прав сертификатов

Права для групп сертификатов по умолчанию хранятся в файле /var/calculate/server/conf/group\_right.conf в следующем виде: group right1,[right2,right3...]], например

```
manager install, get-sessions, request
user get-sessions, request,view_cert
```

Для изменения прав конкретного сертификата используется файл /var/calculate/server/conf/right.conf, куда необходимо вписать права и номера сертификатов, например

```
install 1 2 -3
```

для сертификатов с номерами 1 и 2 добавить право на действие install и удалить его для сертификата с номером 3.

Права для конкретного сертификата имеют приоритет перед правами для группы сертификата.

Для изменения прав конкретного сертификата клиента используетсяключи --right-add и -right-del совместно с ключом --cert, например

```
cl-core -c 6 --right-del install_pxe,install
cl-core -c 7 --right-add install_pxe,install,configure_video
```

В примере для сертификата с номером 6 устанавливается запрет на методы, требующие права install\_pxe и install, а для сертификата с номером 7 устанавливается разрешение на методы install\_pxe, install и configure\_video.

#### Локальный запуск процессов

Запуск методов с помощью ключа --method

Все действия на сервере утилит можно запускать как через клиентов (cl-console-gui, cl-console), с использованием шифрования по сертификатам и позволяющим выполнять действия на удаленных серверах утилит, так и с помощью самих серверов утилит, выполняющих действия напрямую и только на локальных серверах утилит.

Для просмотра всех доступных действий на сервере утилит используйте команду

```
# cl-core - -list-methods
install - Установка системы
setup_boot - Загрузка
core_setup - Настройка пакета
```

The screenshot shows the 'Calculate Console' application interface. In the top left, there's a toolbar with icons for file operations. The main title bar says 'Подробности запроса - Calculate Console'. Below the title bar, there's a header with '127.0.0.1' and a red 'X' button. The main content area is titled 'Подробности запроса' (Request Details). It displays a list of fields for a request entry:

| Идентификатор запроса | 3                          |
|-----------------------|----------------------------|
| Имя владельца запроса | Иван Иванов                |
| IP-адрес              | 11.22.33.44                |
| MAC-адрес             | 00:12:23:34:45:55          |
| Дата запроса          | 2012-06-27 10:55:03.230280 |
| Расположение          | ivanov@company.com:8888    |
| Группа сертификата    | develop                    |

At the bottom right of the details window, there are three buttons: 'Назад' (Back), 'Подтвердить' (Confirm), and 'Удалить' (Delete).

Рис. 16 Просмотр запросов

#### Сертификаты

Сертификат подтверждает соответствие между открытым ключом и информацией, идентифицирующей владельца ключа. Кроме того, в утилитах Calculate сертификаты используются для разграничения прав доступа с помощью указания в них группы прав.

При генерации сертификата для него устанавливается группа прав. Права записываются в одно из полей сертификата и не подлежат изменению.

#### Права групп

Права групп служат для разграничения прав на сервере утилит и указываются в сертификате клиента. Каждой группе соответствует список прав, к которым данная группа имеет доступ.

Права для групп сертификатов по умолчанию хранятся в файле /var/calculate/server/conf/group\_right.conf в следующем виде: group right1[,right2[,right3...]], например

```
manager install, get-sessions, request
user get-sessions, request,view_cert
```

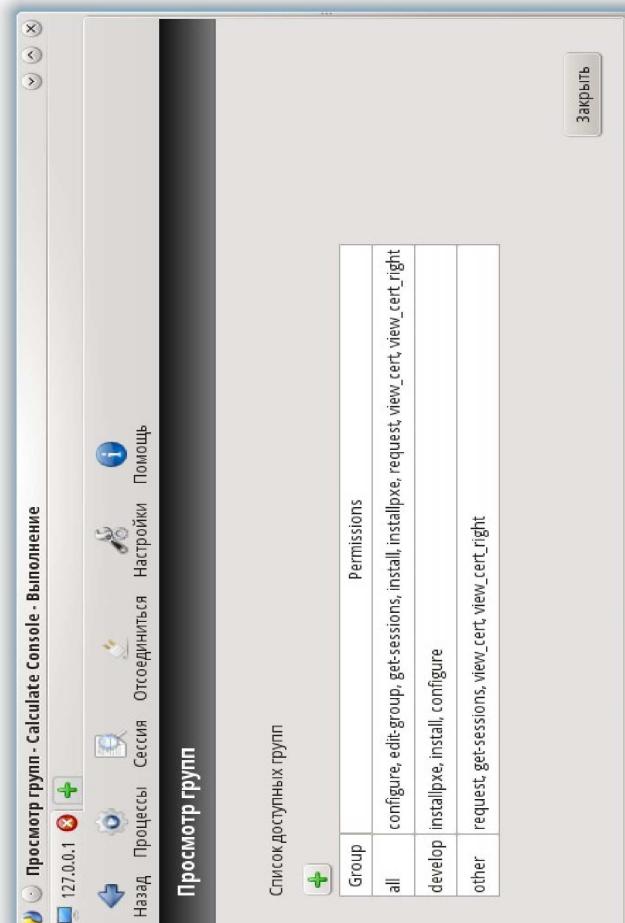
Для изменения прав конкретного сертификата используется файл `/var/calculatc/server/conf/right.conf`, куда необходимо вписать права и номера сертификатов.

Пример:  
`install 1 2 -3`

Для смены прав на метод "install" и удалить его для сертификата с номером 3.

Права для конкретного сертификата имеют приоритет перед правами для группы сертификата.

Для управления правами групп на сервере утилиту в клиенте используются метод "Просмотр групп" в категории утилиты.



Для просмотра клиентских подписанных сертификатов используйте ключ `--cert` с указанием номера сертификата или слова "all" для просмотра списка сертификатов, например

```
cl-core -cert all  
cl-core -cert 2
```

С помощью ключа `--dump` можно просмотреть сразу все сертификаты

```
cl-core -cert all -dump
```

или содержимое файла сертификата, например

```
cl-core -cert 4 -dump
```

Для просмотра запросов и сертификатов серверов утилит используйте ключ `--server-cert` с указанием номера или "all", например

```
cl-core -server-cert all  
cl-core -server-cert 2
```

Подписание запросов и отзыв сертификатов клиента

Для подписаня клиентского запроса на подпись сертификата используйте ключ `--sign-client` с указанием номера запроса, например

```
cl-core -sign-client 4
```

После этого введите группу прав для нового сертификата (изменить её в дальнейшем невозможно).

Подписание осуществляется серверным сертификатом.

Если необходимо отвергнуть клиентский запрос, используйте ключ `-del-client-req` с указанием номера запроса, например

```
cl-core -del-client-req 4
```

Для удаления уже подписанного сертификата клиента совместно с ключом `-cert` и указанием номера сертификата используйте ключ `-remove`, например

```
cl-core -cert 4 -remove
```

Также создать сертификат пользователя с правами группы "all" можно с помощью команды

```
cl-core --bootstrap username
```

например, с помощью команды

```
cl-core --bootstrap ivanov
```

для пользователя ivanov будет создан сертификат с правами группы "all" и добавлен в доверенные сертификат сервера утилиты.

Для удаления всех сертификатов, запросов и конфигурационных файлов на сервере совместно с ключом `--bootstrap` используйте ключ `--remove-certs`, например:

```
cl-core -bootstrap ivanov -remove-certs
```

## Консольный клиент утилит Calculate

- Консольный клиент утилит Calculate
- Введение
- Создание сертификата
- Выполнение методов на сервере утилит

Подписание запросов и отзыва сертификатов сервера утилит

Для подписание запроса на подпись сертификата от другого сервера утилиты ключ `--sign-server` с указанием номера запроса, например

```
cl-core -sign-server 4
```

## Подписание сертификата у другого сервера утилит

Для генерации секретного ключа и запроса, а так же для отправки запроса на сервер утилит используйте команду

```
c1-core -gen-cert-by HOST -port PORT
```

, где HOST - сетевой адрес сервера утилит, PORT - порт, который прослушивает сервер утилит (по умолчанию 8888)

Например,

```
c1-core -gen-cert-by 192.168.0.123 -port 4567
```

После подписания сертификата на сервере утилит необходимо забрать его с помощью команды

```
c1-core -get-cert-from ROOT_HOST -port PORT
```

, где ROOT\_HOST - сетевой адрес сервера утилит, PORT - порт, который прослушивает сервер утилит (по умолчанию 8888)

Например,

```
c1-core -get-cert-from 192.168.0.123 -port 4567
```

## Введение

Клиент утилит Calculate Console служит для сетевого доступа по протоколу [https](https://) к функциям сервера утилит Calculate 3 ([calculate-core](#)). Установить клиент можно при помощи пакета sys-apps/calculate-console.

## Создание сертификата

Для работы с сервером утилит необходимы секретный ключ и сертификат, поданный сервером утилит.

Для генерации секретного ключа и создания запроса на подпись сертификата, используйте ключ `-gen-cert-by <хост>` и ключ `-port <порт>` для указания номера порта (по умолчанию порт 8888), например:

```
c1-console -gen-cert-by 192.168.0.56 -port 8888
```

Если по адресу 192.168.0.56 порт 8888 прослушивает сервер утилит, то пользователю будет предложено создать запрос на подпись сертификата.

Первым действием является установка пароля для дополнительного шифрования секретного ключа. Введённый пароль будет запрашиваться для установки соединения с сервером утилит. Чтобы не зашифровывать секретный ключ, оставьте поле пароль пустым.

Также необходимо ввести некоторые данные, необходимые для создания сертификата. В квадратных скобках будут показаны значения по умолчанию, например:

```
Имя хоста [iivanov@iivanov.company.ru]:
```

```
Имя пользователя [Иван Иванов]:
```

## Запуск сервера утилит

После генерации сертификата сервер утилит можно запустить с помощью команды

```
c1-core -gen-root-cert
```

Для использования созданного самодиддисного корневого сертификата как сертификат сервера утилит используйте команду

```
c1-core -use-root-as-server
```

Для запуска в режиме отладки (debug), используйте ключ `-debug`

## Действия с сервером утилит

### Операции с запросами и сертификатами

#### Просмотр запросов и сертификатов

Многие действия с сервером утилит можно выполнять непосредственно на сервере, без использования клиентов. Для этого необходимы права суперпользователя - root.

Для просмотра клиентских запросов используйте ключ `--show-request` с указанием номера запроса или слова "all" для просмотра списка запросов, например

```
c1-core -show-request all
```

```
c1-core -show-request 2
```

## Выполнение методов на сервере утилит

Для просмотра доступных методов на сервере утилит используйте команду

```
cl-console -host <хост> -port <порт>
```

По умолчанию клиент соединяется с локальным сервером утилит и 8888 портом, например:

```
cl-console  
cl-console -host 192.168.0.56 -port 9999
```

Первая команда выведет все доступные методы на локальном сервере утилит, прослушивающем порт 8888, вторая - на сервере утилит, находящемуся по адресу 192.168.0.56:9999.

Для запуска метода используйте ключ `-method <метод>`, например:

```
cl-console -method install -iso /path_to_image/cld-x86_64.iso -d  
/dev/sda1:swap -d /dev/sda2:/ext4:on
```

Для просмотра справки метода используйте ключ `-h, -help`:

```
cl-console -method install -help
```

Ключ `-f, --force` устанавливает режим, при котором пользователю не задаются вопросы и не отображаются предварительные параметры.

Ключ `-no-progress` отключает отображение прогрессбаров (текущего прогресса выполнения задачи).

## Действия с клиентом

### Работа с сессией

Для просмотра информации о сессии и сертификате используйте ключ `--session-info`:

```
cl-console -session-info
```

```
cl-console -session-list
```

Для просмотра списка активных сессий на сервере утилит используйте ключ `--session-list`:

```
cl-console -session-list
```

```
cl-console -session-num-info 5474
```

выведет информацию о сессии с номером 5474.

Для очистки кэша своей сессии используйте ключ `-session-clean`:

```
cl-console -session-clean
```

Для просмотра списка запущенных процессов текущей сессии, используйте ключ `-list-pid`:

```
cl-console -list-pid
```

Каждый запущенный метод на сервере утилит является процессом, результат работы которого можно просмотреть.

Для более подробного вывода совместно с ключом `-list-pid` используйте ключ `-d, -dump`:

```
cl-console -list-pid -dump
```

Для просмотра результатов работы процесса, используйте ключ `-pid-result <pid>`, например

```
cl-console -pid-result 1234
```

выведет результат работы процесса с номером 1234.

Если необходимо прервать работающий процесс, используйте ключ `--pid-kill <pid>`, например

```
cl-core -pid-kill 1234
```

сообщит серверу утилит, что необходимо завершить процесс с номером 1234.

## cl-console

Если пользователем используется пароль для дополнительного шифрования секретного ключа, то консольным клиентом используется демон `cl-consoleed`.

`cl-consoleed` предназначен для кратковременного хранения введенных паролей, чтобы пользователю не было необходимо при каждом соединении вводить пароль. Пароль к секретному ключу сервера утилит выдаётся только при совпадении ряда данных.

Для остановки `cl-consoleed` используйте команду

```
cl-console -stop-consoleed
```

В этом случае при следующем запуске `cl-console` заново потребует пароль и запустит `cl-consoleed`.

При необходимости обновить список отзыва сертификатов, используйте ключ `--update-crl`

## Сервер утилит Calculate

- Сервер утилит Calculate
- Введение
- Начало работы, создание сертификата
- Подписание сертификата у другого сервера утилит
- Создание самоподписанного сертификата
- Запуск сервера утилит
- Действия с сервером утилит
- Операции с запросами и сертификатами
- Локальный запуск процессов
- Другие действия
- Стандартные методы сервера утилит

## Введение

Сервер утилит Calculate Core служит для выполнения методов утилит, таких как установка, настройка системы и т.д. и для осуществления сетевого доступа клиентам (`cl-console` и `cl-console-gui`) по протоколу [https](https://) к функциям утилит Calculate. Calculate Core входит в состав утилит Calculate 3. Установить сервер утилит можно при помощи пакета `sys-apps/calculate-core`.

## Начало работы, создание сертификата

Для запуска сервера утилит необходимо создать сертификат одним из двух способов:

- создать запрос на подписание сертификата (далее запрос) и подписать сертификат у другого сервера;

• создать самоподписьной корневой сертификат и использовать его в качестве сертификата сервера.

Для запуска сервера утилит необходим серверный сертификат. Корневой самоподписной сертификат необходим для подписания сертификатов другим серверам утилит.

## Изменение значения переменной в командной строке

Для изменения значений переменных используются различные параметры команд. Например

- для изменения `cl_autopartition.root_size` в команде `cl-install` используется опция `-root-size`
- для изменения `os_install_x11_video_drv` в команде `cl-install` используется опция `-video`

## Изменение значения через конфигурационный файл

Вы можете предопределить значения переменных в файле `calculate.env`. Пути к файлам находятся в переменной `c1_env_path`:

```
main.c1_env_path | w1 | |
/etc/calculate/calculate.env,		
/var/calculate/calculate.env,		
/var/calculate/remote/calculate.env		
/var/calculate/remote/remote/calculate.env		
```

Приоритет распределяется от первой к последней записи. Т.е. переменная, измененная в файле `/var/calculate/remote/calculate.env`, перепишет другое значение.

Пример содержимого файла `calculate.env`:

```
[install]
os_install_ntp = ntp0.zenon.net
```

Обратите внимание: переменная находится в секции `[install]`, т.е. это модуль `calculate-install`.

Для изменения значений переменных в `calculate.env` файлах можно воспользоваться утилитой `c1-core-variables` с параметром `-sset`. Преимущество этого способа заключается в том, что перед записью значения переменной выполняется проверка на допустимость.

Изменение значения переменной `install.c1_autoLogin`

```
c1-core-variables -sset install.c1_autoLogin=guest
```

Изменение значения переменной `install.cl_autoLogin` в `/var/calculate/calculate.env`

```
c1-core-variables --set install.cl_autoLogin=guest:local
```

Удаление значения `install.c1_autoLogin` из `calculate.env`

```
c1-core-variables --set install.c1_autoLogin
```

## Использование переменных

Переменные - основа шаблонов. Их значения настраивают систему в зависимости от текущего состояния оборудования и определяют логику работы. Здесь мы рассмотрим все случаи использования переменных.

## Вставка значений в шаблон

Для вставки значения переменной в шаблон используйте конструкцию `#-__имя_переменной__-` # или расширенную конструкцию `#-__модуль_.имя_переменной__-` #.

Пример настройки Xorg-сервера:

```
Modes "#-1 install.os_install_x11_resolution-#"
```

возможной части текста.

"?" - соответствует 0 или 1 повторению предшествующего РВ. РВ будет соответствовать максимально возможной части текста.

"??" - "не жадная" версия использования предыдущего символа. РВ будет соответствовать минимально возможной части текста.

"{m,n}" - соответствует от m до n повторений предыдущей конструкции. РВ будет соответствовать

"{m,n}?" - "не жадная" версия использования предыдущей конструкции. РВ будет соответствовать

минимально возможной части текста.

"\\" - экранирует специальные символы из указанного набора.

"[\^]" - соответствует любому символу из указанного набора.

"[A\B]" - создать РВ соответствующее А и В.

"(...)" - группировка РВ с сохранением групп (где возвращают содержимое первой группы)

"(?ims)" - устанавливает режимы РВ: регистрационный, multiline, dotall.

"(?:...)" - группировка РВ без сохранения групп

"?(=...)" - проверка текста после РВ на совпадение.

"?(!...)" - проверка текста перед РВ на совпадение (это РВ должно быть фиксированной длины).

"?(<=...)" - проверка текста перед РВ на не совпадение (это РВ должно быть фиксированной длины).

"?(<!=...)" - проверка текста перед РВ на совпадение

"\W" - соответствует любому символу из набора [a-zA-Z0-9\_]

"\W" - соответствует любому символу из набора [a-zA-Z0-9\_]

"\S" - соответствует любому пробельному символу

"\S" - соответствует любому символу из набора [^\a-zA-Z0-9\_]

"d" - соответствует любой десятичной цифре [0-9]

"D" - соответствует любой символу, не являющемуся десятичной цифрой [^0-9]

groups(group1,group2,...groupN) - проверка входления пользователя в группы group1,group2,...groupN

Если пользователь входит хотя бы в одну из групп выводит '1', иначе "0".

group1 .. groupN - названия групп.

Пример. Проверим входит ли пользователь в группу wheel:

```
#-groups(wheel)-#
```

Если пользователь входит в группу wheel, результатом будет '1'.

init(var, value, opt) - запись и чтение переменной из конфигурационного файла пользователя (`~/calculate.ini`). env.

Начиная с версии 2.2.20-4: если функция выполняется для настройки системы, то конфигурационный файл будет находиться в `/etc/calculatate/init.env`. где:

- var - имя переменной функции. Имя должно начинаться с буквами и может содержать латинские буквы и цифры, а также точку. Точка служит разделителем для раздела и имени переменной для размещения в конфигурационном файле.
- value - значение переменной функции, значение присваивается переменной функции и записывается в конфигурационный файл (функция при этом возвращает пустое значение). При отсутствии второго аргумента переменная считывается из конфигурационного файла; при отсутствии названия переменной в конфигурационном файле в переменную записывается пустое значение.

Начиная с версии 2.2.20-4, если аргумент пустая строка без кавычек, то переменная удаляется.

Если аргумент - пустая строка в кавычках, то значение переменной в ini-файле очищается.

- opt - опция преобразования значения переменной, необязательный параметр; возможные значения url, purl, unicode. При использовании этого аргумента второй аргумент функции должен быть пустым.

Примеры:

1. Создадим переменную "test" и присвоим ей значение 15, запишем в конфигурационный файл:

```
#-ini(test,15) -#  
  
2. Считаем значение ранее записанной в конфигурационный файл переменной test, заменим функцию  
значением считанной переменной, в нашем случае "15":  
 #-ini(test) #
```

3. Использование функции ini с тремя аргументами. Предположим, что в файле  
~/calculate.ini.enr существует секция

```
[test]  
param = Тестовый параметр
```

```
тогда функция  
 #-ini(test.param,,unicode) -#
```

```
вернет  
\u0422\u0435\u0441\u0442\u043e\u0432\u044b\u0439  
\u043f\u0430\u0440\u0430\u0435\u0442\u0442\u0440
```

а функция  
 #-ini(test.param,,url) -#

```
вернет  
%d%a2%d0%b5%d1%81%d1%82%d0%be%d0%b2%d1%8b%d0%b9%d0%bf  
%d0%b0%d1%80%d0%b0%d0%bc%d0%b5%d1%82%d1%80
```

Опция pur1 отличается от url тем, что преобразует '/' в код '%2F'

```
4. помещение сервиса sshd в автозапуск с проверкой и отметкой в ini.env  
# Calculate path=/etc/runlevels/default name=sshd link=/etc/init.d/sshd  
symbolic ini(runlevels.openssh)!=on&&ini(funlevels.openssh, on)==
```

livenemu(mode) - (добавлена в версии 3.4) - функция специального назначения для формирования  
мультизагрузочного меню для Flash. mode - режим работы функции, и может быть следующим:

- submenu - возвращает список систем в следующем формате:

```
идентификатор системы;  
полное название в загрузочном меню;  
путь до ядра;  
параметры для загрузки;  
путь до initrd;  
параметры загрузки splash;
```

```
Пример:  
cl-2;  
Calculate Linux Desktop Xfce;  
/boot/vmlinuz-2;  
root=live:LABEL=CALCULATE;  
/boot/initrd-2;  
splash;
```

- xorg - возвращает список систем, в которых установлен xorg-server. Например cl-1 cl-2 cl-

```
| main.os_locale_locale | rs | ru_RU.UTF-8  
| main.os_net_allow | rs | 10.0.0.0/24  
| main.os_net_ip | rs | 10.0.0.84  
| main.os_x11_video_drv | rs | radeon
```

1. значение для условного выражения участвует в условных выражениях;

Пример:

```
| main.hr_video | rs | ati  
| install.os_install_linux_system | rs | desktop  
| install.os_install_locale_language | rs | ru
```

1. информация отображает текущие настройки системы;

Пример:

```
| install.os_net_interfaces_info | rs | enp1s0 (10.0.0.84)  
| main.hr_video_name | rs | Advanced Radeon HD  
7540D
```

1. массив значений используется другими переменными или функциями.

Пример:

```
| install.cl_migrate_user | w1 | guest  
| install.os_device_data | rt | Advanced Radeon HD  
7540D  
/dev/sda, dos, hdd, 0, ATA OCZ-VERTEX3, 60022480896 |  
desktop.cl_desktop_online_data | rt | guest,0,6
```

Переменные разбиты на типы условно. Одно и то же значение может соответствовать нескольким  
типам.

## Изменение значений переменных

При обращении к переменной ее значение определяется программой на основании настроек системы, а  
также значений других переменных. Значения некоторых переменных можно изменить. Такие  
переменные отмечены первой буквой в поле "Режим". Остальные переменные, доступные только для  
чтения, отмечены первой буквой г. Пример:

```
| main.cl_ver | rs | 3.1.7  
| install.os_install_x11_composite | wb | on
```

Изменить значение переменной можно либо из командной строки параметрами утилиты, либо сохранив  
значение в конфигурационном файле. Если переменная меняется в обоих местах, приоритет отдается  
командной строке.

- video - возвращает список систем, в которых присутствуют proprietарные драйверы. Например

c1 -2 c1 -3

Функция используется в шаблонах 6\_ac\_builder\_iso/0\_bootmenu/desktop.config, а также для формирования gfxboot.cfg.

- ac\_client\_domain==on (если машина настроена на работу в домене)

- ac\_client\_undomain==on (если машина настроена как локальная)

#### Обновление портежей

Выполняется после обновления портежей и оверлея командой eix-sync

- ac\_update\_sync==on

#### Модификация исходного кода пакета

Выполняется во время сборки пакета emerge <название\_пакета> перед его компиляцией

- ac\_install\_patch==on

#### Подготовка squash-образа (Calculate утилиты 2.2)

Выполняется перед упаковкой в Squash-образ cl-image iso или cl-image squash.

- ac\_builder\_squash==up

#### Подготовка ISO-образа (Calculate утилиты 2.2)

Выполняется перед созданием ISO-образа командой cl-image iso.

- ac\_builder\_iso==up

#### Подготовка к сборке дистрибутива (Calculate утилиты 2.2)

Выполняется во время подготовки к сборке дистрибутива, после распаковки Stage-образа, командой cl-assemble.

- ac\_assemble\_prepare==up

#### Настройка системы во время сборки (Calculate утилиты 2.2)

Выполняется во время подготовки к сборке дистрибутива, после добавления оверлея calculate, командой cl-assemble.

- ac\_assemble\_setup==up

#### Обновление системы во время сборки (Calculate утилиты 2.2)

Выполняется перед сборкой пакетов в собираемой системе, командой cl-make -u или cl-make -U.

- ac\_assemble\_prepare==up

- ac\_assemble\_setup==up

#### Произвольное событие

Выполняется при вызове команды cl-core-custom имя\_события

- ac\_core\_custom==имя\_события

## Значения переменных

Переменные могут содержать несколько типов значений:

1. значение для подстановки используется для подстановки значения в шаблон;

Пример:

```
Значение переменной строки
os_install_linux_shortname = CLDX
#- in (os_install_linux_shortname, CLDX, CLD, CLDG) -#
```

- video - возвращает список систем, в которых присутствуют proprietарные драйверы. Например

c1 -2 c1 -3

Функция используется в шаблонах 6\_ac\_builder\_iso/0\_bootmenu/desktop.config, а также для формирования gfxboot.cfg.

- server(service,option,var) - чтение значения параметра сервиса. Информация получается путем обработки файла /var/calculcate/remote/server.env, где:

- service - название сервиса.

- option - опция сервиса.

Разделитель service и option: ' ' - точка.

В случае использования var - имени переменной функции, полученное значение опции сервиса будет записана в переменную функции var.  
Если var не используется, получченное значение опции сервиса будет вставлено в текст конфигурационного файла. Примеры:

- прочитаем значение порта для соединения с jabber-сервисом

```
#-server(jabber,port) -#
```

получаем значение 5223

1. прочитаем значение порта для соединения с jabber-сервисом и поместим в переменную функции jabber\_port

```
#-server(jabber,port,jabber_port) -#
```

Значение переменной функции jabber\_port - 5223. list(var,index) - вывод значения по индексу из где:

- var - название переменной функции или переменной шаблона (значение переменной должно быть списком).
- index - индекс для поиска значения в переменной функции или переменной шаблона (первый элемент имеет индекс 0, и т.д.).

Пример.

Значение переменной os\_disk\_dev = [/dev/sda1,'/dev/sda2','/dev/sda3','/dev/sda4','/dev/sda5'] #list(os\_disk\_dev,1) -#

Метка функции будет заменена на '/dev/sda2'.

Если запрошен элемент, отсутствующий в списке, наложение шаблонов прерывается с ошибкой.

*Начиная с версии 2.2.17 - возвращает пустую строку*

**in(var,value1,value2...)** - проверяет есть ли среди значений переменной var, хотя бы одно значение из value. Если значение найдено, то возвращает "1", иначе пустую строку. Переменная может быть как списком, так и строкой.

- var - название переменной

- valueX - список значений для сравнения

Пример:

```
Значение переменной строки
os_install_linux_shortname = CLDX
#- in (os_install_linux_shortname, CLDX, CLD, CLDG) -#
```

## Пример 2:

Значение переменной список  
os\_linux\_pklist = CLDX\_base  
#-in (os\_linux\_pkglst, CLDX, CLD, CLDG) -#

Вернет 1, так как есть общее значение - CLDX.

Метка функции будет заменена на CLDX.

**kernel(kernel\_opt)** - (добавлена в версии 3.3.1) получить значение параметра конфигурации ядра.  
Возвращает "y", "n" или пустую строку, у - опция включена в ядро, n - модуль ядра, пустая строка -  
опция выключена.

Пример 1. Включен ли ext4:

```
kernel1(ext4_fs)!=
```

Пример 1. reiserfs - модуль ядра, а ext4 - включен в ядро  
kernel(reiserfs\_fs)==m&&kernel(ext4\_fs)==y

**load(arg,path,opt)** - отображение информации из файла.  
где:  
arg - тип содержимого файла;  
path - путь к файлу;

опт - необязательная опция Возможные значения "opt":  
root - путь к файлу не будет содержать chroot-пути. (Какой путь указан, такой и будет в  
действительности, вне зависимости от переменных \$1\_chroot\_path и \$1\_root\_path.)

Возможные значения "arg":

- ver - содержитимое файла номер версии
- num - содержитимое файла число
- char - содержитимое файла строка
- entry - результат, содержитимое файла без закомментированных (комментарии # или ; ) и пустых  
строк. Возможные значения "path":
- path - абсолютный или относительный путь к файлу.

Пример. Выведем содержимое /proc/cruiinfo на место объявления функции:

```
#-load(char,/proc/cruiinfo)-#
```

**module(name\_space)** - (удалена в 3.1) получение переменных шаблонов пакета или выполнение  
методов пакета, используя его арп-модуль или выполнение определенного метода для всех пакетов, у  
которых есть арп-модуль,  
где:

- \_name\_space\_ - пространство имен арп-модуля.

Пространство имен для получения переменных состоит из элементов разделенных точкой.  
Первый элемент пространства имен - имя пакета. Имя пакета это название установленного пакета,  
имеющего арп-модуль (тире '-' в названии должно быть заменено на нижнее подчеркивание '\_').  
Первый элемент пространства имен для всех пакетов - 'all' (зарезервированное название).  
Пример первого элемента для пакета calculate\_idap:

```
calculate_idap
```

## Установка системы на USB-Flash

Выполняется при установке системы, если в качестве носителя используется USB-Flash cl-install.

- ac\_install\_flash=on

## Установка системы для загрузки по сети (PXE)

Выполняется при установке системы на сервере для загрузки по сети cl-install --pxe.

- ac\_install\_px=on
- ac\_install\_assemble

Выполняется во время установки программы emerge <название\_пакета> в builder или через calculate\_assemble

- ac\_install\_merge=on Выполняется во время удаления программы emerge <название\_пакета> в builder или через calcuate-assembly
- ac\_install\_unmerge=on Выполняется во время установки программы emerge <название\_пакета> в рабочей системе
- ac\_client\_merge=on
- ac\_install\_live=on
- ac\_desktop\_merge=on
- ac\_client\_domain=on (если машина настроена на работу в домене)
- ac\_client\_undoain=on (если машина настроена как локальная) Выполняется во время  
удаления программы emerge -C <название\_пакета> в рабочей системе
- ac\_install\_unmerge=on
- ac\_install\_merge=on
- ac\_install\_live=on
- ac\_desktop\_merge=on
- ac\_client\_merge=on
- ac\_client\_domain=on (если машина настроена на работу в домене)
- ac\_client\_undoain=on (если машина настроена как локальная)

## Настройка профиля пользователя

Выполняется при запуске cl-desktop <логин\_пользователя> во время входа в сеанс.

- ac\_desktop\_profile=up Выполняется во время установки программы в рабочей системе  
emerge <название\_пакета>
- ac\_desktop\_profile=on

## Вход компьютера в домен

Выполняется при вводе компьютера в домен командой cl-client .  
• ac\_client\_domain=on

- ac\_client\_merge=on

## Вывод компьютера из домена

Выполняется при выводе компьютера из домена командой cl-client -r.

- ac\_client\_undoain=on
- ac\_client\_merge=on

## calculate-install

- ac\_install\_configure - выполнение шаблонов настройки компонентов системы
- ac\_install\_disk - настройки системы для загрузки с жесткого диска
- ac\_install\_flash - настройка системы для загрузки с USB-Flash
- ac\_install\_live - настройка пакета (динамические параметры)
- ac\_install\_merge - настройка пакета (статические параметры)
- ac\_install\_patch - модификация исходного кода пакета
- ac\_install\_pxe - настройка PXE загрузки
- ac\_install\_unmerge - настройка системы при удалении пакета

### calculate-desktop

- ac\_desktop\_profile - настройка пакета в профиле пользователя.
- ac\_desktop\_merge - настройка пакета в рабочей системе

### calculate-client

- ac\_client\_merge - настройка пакета в рабочей системе
- ac\_client\_domain - настройка пакета в домене
- ac\_client\_undomain - настройка машины для локальной работы

### calculate-core

- ac\_custom\_name - выполнение произвольного действия calculate-builder (Утилиты Calculate 2.2)
- ac\_builder\_iso - настройка ISO-образа
- ac\_builder\_squash - настройка Squash-образа calculate-assemble (Утилиты Calculate 2.2)
- ac\_assemble\_prepare - первичная настройка собираемой системы
- ac\_assemble\_setup - настройка системы во время сборки

## События

В зависимости от событий (например, установка пакета), утилиты выставляют значения переменных действий и накладывают шаблоны.

Ниже перечислены события, а также значения переменных действий, записанные через знак двойного равенства, принятый в условных выражениях шаблонов:

### Настройка системы

Выполняется во время первой загрузки системы, загрузке с LiveCD, USB-Flash или USB-HDD, при выполнении команды cl-setup-system -live.

- ac\_client\_domain=on (если машина настроена на работу в домене)
- ac\_client\_undomain=on (если машина настроена как локальная)
- ac\_client\_merge=on
- ac\_install\_live=on
- ac\_desktop\_merge=on

### Установка системы на жесткий диск

Выполняется при установке системы командой cl-install.

- ac\_install\_merge=on
- ac\_install\_live=on
- ac\_install\_disk=on

Второй элемент пространства имен - название метода api или 'var' - зарезервированное название для пространства имен переменных шаблонов пакета.

Пример первого и второго элементов для пакета calculate-ldap для получения доступа к переменным:

```
calculate_ldap.var
```

Пример первого и второго элементов для пакета calculate-ldap для получения результата выполнения метода is\_setup () api-модуля (проверка, настроен ли пакет).

```
calculate_ldap.is_setup
```

Третий элемент пространства - в большинстве случаев это пространство имен для получения значения переменной шаблона.

Пример пространства имен для получения значения переменной шаблонов cl\_name пакета calculate-ldap:

```
calculate_ldap.var.cl_name
```

Примеры использования функции:

Получим значение переменной шаблонов cl\_name (название пакета) пакета calculate-ldap.

```
#-module(calculate_ldap.var.cl_name)-#
```

Получаем значение calculate-ldap

Проверим, настроен ли пакет calculate-ldap для работы:

```
#-module(calcluate_ldap.is_setup)-#
```

если пакет настроен, получаем значение

1

Проверим, настроены ли для работы все установленные пакеты, имеющие арм-модуль:

```
#-module(all.is_setup)-#
```

Если все пакеты настроены, получаем значение

1

pkgr (category/package) - вывод версии установленного пакета, если в системе установлено несколько версий этого пакета, ты вызовешься версия максимального слота.  
где:

- category - категория пакета
- package - название пакета

• Начиная с версии утилит 3.1.0\_beta2 добавилась возможность указывать слот пакета через двоеточие: pkgr (Kde-base/kdelibs:4)

- Начиная с версии утилит 3.3 добавлена возможность проверять наличие USE флагов у пакета.

Проверяемые USE флаги указываются в квадратных скобках, через запятую. Указание "-" перед USE флагом означает, что флаг у пакета должен быть выключен.

В качестве совместимого режима можно указывать только название пакета; в этом случае скорость обработки шаблона будет ниже.

Пример. Выведем версию установленного пакета (например 4.2.4):

```

#- pkg(kde-base/kdelibs) -#
Значит, в системе установлен пакет kdelibs 4.2.4.

Пример использования USE. Вывести версию установленного пакета, если он собран с включенным
desktop и выключенным client флагами.

#- pkg(sys-apps/calculate-utils[desktop,-client]) -#

```

**print (message)** - вывод на экран информационного сообщения во время выполнения шаблона  
**warning (message)** - вывод на экран предупреждающего сообщения во время выполнения шаблона  
**error (message)** - вывод на экран сообщения об ошибке во время выполнения шаблона

**push(var, value)** - помещение значения переменной в стек переменных функций шаблонов. где:

- *var* - имя переменной функции. Имя должно начинаться с буквами и может содержать латинские буквы и цифры. Переменная создается на время обработки конфигурационного файла.
- *value* - значение переменной функции, значение присваивается переменной функции и заносится в стек переменных функций шаблонов. При отсутствии второго аргумента, в стек переменных функций шаблонов будет записано значение переменной. Примеры:

- Создадим переменную "test" и присвоим ей значение 15, запишем в стек переменных функций шаблонов.

```
#- push(test,15) -#
```

1. Запишем в стек переменных функций шаблонов значение ранее созданной переменной **test**:

```
#- push(test) -#
```

**pop(var)** - извлечение значения из стека переменных функций шаблонов и присвоение его переменной.

- *var* - имя переменной функции. Имя должно начинаться с буквами и может содержать латинские буквы и цифры. Переменная создается на время обработки конфигурационного файла. Примеры:
- Создадим переменную "test" и присвоим ей значение 15, запишем в стек переменных функций шаблонов.

```
#- push(test,15) -#
```

1. Получим значение из стека переменных функций шаблонов и присвоим его переменной **test2**:

```
#- pop(test2) -#
```

Получить значение из стека переменных функций шаблонов возможно как в текущем шаблоне, так и в любом другом.

После получения значения оно удаляется из стека. **replace(old, new, var)** - замена в значении переменной **var** текста **old** на **new**.  
где:

- *old* - текст, каждое вхождение которого в значении переменной будет заменено на текст *new*
- *new* - текст, на который в значении переменной будет заменен текст *old*
- *var* - название переменной функции или переменной шаблона

Текст **old** и **new** должен быть заключен в двойные или одинарные кавычки.

В тексте в двойных кавычках обрабатываются управляющие символы ('', \, \n, \t, \\).  
Начиная с версии 3.2.3-r3 в двойных кавычках также обрабатывается \xFF, где FF двоичный шестнадцатеричный код символа.  
В одинарных кавычках текст не обрабатывается.

Здесь видно, что утилита, отобразила переменные в тех пакетах. Колонка "Переменная" отображает модуль переменной и имя переменной, разделенных точкой. "Режим" состоит из двух букв:

- доступ
  - "r" только для чтения
  - "w" поддерживает изменение
- тип переменной
  - "S" строка
  - "B" логическая переменная (on/off)
  - "I" список
  - "T" таблица
  - "C" строка с выбором

"расположение" указано для тех переменных, чьи значения изменяются в calculate.env файлах. Есть три места в которых можно изменять значения по умолчанию для **writable** переменных: system (etc/calculate/calculate.env), local (/var/calculate/calculate.env), remote (/var/calculate/remote/calculate.env). "Значение" отображает содержимое переменной. Содержимое переменных не точное, так как при различных командах содержимое может изменяться (например изменение значений при указании опций команды). С помощью параметра - **-filter** можно задать фильтрацию выводимых переменных:

- userset - переменные установленные в calculate.env файлах
- writable - переменные доступные для изменения
- system,local,remote - переменные указанные в конкретном calculate.env файле
- часть имени переменной

## Типы переменных

Для удобства в именах переменных используется обозначение типа переменной, часто используется название пакета и может использовать тип возвращаемого значения.

Всего существует семь типов переменных:

1. ac - переменные выполняемых действий
2. cl - общие настройки утилит
3. hr - настройки оборудования
4. ld - атрибуты LDAP
5. os - операционная система
6. sr - настройки сервисов
7. ug - информация о пользователе

Для примера: переменная **cl\_install\_autoupdate\_set** относится к общим настройкам утилит, принадлежащих пакету **calculate-install** и содержит одно из двух значений - on либо off.

## Переменные выполнимых действий

Пакеты утилит содержат шаблоны настройки, сгруппированные по определенным действиям. Например, настройка профиля пользователя, настройка пакета при установке и т.д. Для того, чтобы определить, какие шаблоны следует накладывать, пакеты утилит содержат специальные переменные выполняемых действий.

```
{"google": {"services": {"signin": {"LSID": "", "SID": ""}}},  
"intl": 5}
```

## Переменные шаблонов

- Переменные шаблонов
- Введение
- Типы переменных
- Переменные выполняемых действий
- События
- Значения переменных
- Изменение значений переменных
- Изменение значения переменной в командной строке
- Использование переменных
- Вставка значений в шаблон
- Передача значения функции
- Условия в заголовках шаблонов
- Условные выражения

### Введение

Утилиты Calculate содержат переменные, которые могут быть использованы в [шаблонах](#). Каждый пакет утилит добавляет свой набор переменных. Несмотря на то что переменные принадлежат разным пакетам, их имена уникальны. Для просмотра переменных и их значений используется утилита `c1-core-variables -show`:

```
# c1-core-variables -show  
* Список переменных  
+--  
+--  
+--  
| Режим | Расположение | Значение  
+--  
+--  
+--  
+--  
+--  
+--  
+--  
| main.ac_custom_name | WC | |'  
| main.c1_action | rs | |'  
| main.c1_update_set | wb | system | off  
| install.ac_install_configure | rs | | off  
| install.ac_install_disk | rs | | off  
| install.ac_install_flash | rs | | off  
| update.ac_update_sync | rs | |'
```

### Пример.

```
значение переменной шаблона _ur_signature_  
Компания «КалкулЭйт»\nРоссия, Санкт-Петербург, пл. Стажек,  
4\nhttp://www.calculate.ru\n+7 812 3363632\n+7 495 7727678
```

```
при выполнении функции  
#- replace(' \n ', "\n", _ur_signature) #-
```

- В точку вставки будет вставлен следующий текст (символ "\n" будет заменен на перевод строки)
- Компания «КалкулЭйт»,  
Россия, Санкт-Петербург, пл. Стажек, 4  
http://www.calculate.ru  
+7 812 3363632  
+7 495 7727678

- Изменение приведенный пример позволяет преобразовать пробел в значении `hr_board_model` в символ подчеркивания.

```
... |main.hr_board_model | rs | | P8H77-V LE |  
...  
#?replace("\x20", " ", main.hr_board_model)==P8H77-V LE#  
...  
#replace#
```

Ниже приведенный пример позволяет преобразовать пробел в значении `hr_board_model` в символ подчеркивания.

- `rnd(type,len)` - вывод случайной комбинации символов.
- `type` - используемые символы, возможные значения: num - числа, pas - буквы и цифры, uid - числа и буквы от a-f. (uid добавлен начиная с версии 2.2.17).
- `len` - количество символов (число).

Пример. Выведем три 3 случайных числа (например 372):

```
#- rnd(num,3) #-
```

`sum(var,sum_print,sum_out)` - вычисляемые значения смешений. Функция разрабатывалась для настройки [Plasma](#) (KDE), где:

- `var` - имя переменной функции "sum". Имя должно начинаться с буквы и может содержать латинские буквы и цифры. Переменная создается на время обработки конфигурационного файла.
- `_sum_print` - арифметическое выражение, результат которого будет отображен на месте объявления функции. При отсутствии аргумента значение выводиться не будет. Поддерживается операции сложения "+" и вычитания "-", деления "/" и умножения "\*". В арифметическом выражении могут участвовать другие переменные функции, а также переменные шаблона.
- `_sum_out` - арифметическое выражение, результат которого будет присвоен переменной "var" (первому аргументу функции). При отсутствии третьего аргумента переменной "var" будет присвоено вычисленное значение второго аргумента. Примеры:
- Создадим переменную "clock" и присвоим ей значение 15, не выводя результат.

```
#- sum(clock,15) #-
```

1. Создадим переменную "bt", присвоим ей значение переменной "clock" и выведем значение.

```
1. Создадим переменную "bt", присвоим ей значение переменной "clock" и выведем значение.
```

1. Разместим кнопку на панели шириной "35" пикселов, оставив отступы в "4" пикселя по краям:

```
#- sum(bt, bt+2, bt+35+2) -#
```

- wallpaper(resolution, wallpapers\_path)** - (добавлена в 3.2.3) функция выбирает наиболее подходящее по указанному разрешению из папки, где:
- **resolution** - требуемое разрешение (текущее разрешение можно получить из переменных `os_X11_resolution`, `os_install_X11_resolution`)
  - **wallpapers\_path** - путь, где находится изображение в разных разрешениях. Формат файлов: `разрешение.расширение`. Например: 1024x768.jpg

Пример: Папка `/usr/share/wallpapers/1` содержит следующие изображения:

- 1024x768.jpg
- 1280x1024.jpg
- 1680x1050.jpg
- 1920x1080.jpg

```
#- wallpaper(1280x720, /usr/share/wallpapers/1) -#
```

вернет 1920x1080.jpg как наиболее подходящую по пропорциям.

Для получения текущего разрешения можно использовать выше указанные переменные

```
#-wallpaper(#-install.os_install_X11_resolution -#  
#, /usr/share/wallpapers/1) -#
```

## Способы объединения

Существуют несколько способов объединения шаблона установки с исходным файлом системы:

- **join** - основной способ объединения - методом слияния двух файлов. Подробно описан в "схеме объединения".
- **before** - шаблон переписывается в начало оригинального файла
- **after** - шаблон переписывается в конец оригинального файла
- **replace** - шаблон переписывается заменяя оригиналный файл
- **delete** - объединение не происходит, оригиналный файл удаляется
- **remove** - вместо объединения происходит удаление оригинального файла Изменение шаблона перед объединением, для способов объединения "before", "after", "replace":

  - если существуют управляющие элементы, то они обрабатываются
  - если существует заголовок, то он обрабатывается
  - если существует параметр `format` в заголовке то происходит обработка управляющих символов: "+", "-", "!"

Правила по умолчанию

По умолчанию способ объединения устанавливается в соответствии с форматом файла.

Для форматов файлов основных приложений "sanba", "bind" и т.п. по умолчанию действует объединение "join", для формата "raw" и "bin" - объединение "replace". Для пустого файла по умолчанию действует правило объединения "delete" - конечный файл удаляется из системы.

Правила объединения могут быть изменены установкой параметра "append" заголовка файла шаблона.

- Для разместим кнопку на панели шириной "35" пикселов, оставив отступы в "4" пикселя по краям:

```
#- sidebar=false
```

Для применения шаблонов формата `dconf`, утилиты используют пакет `gnome-base/dconf`.

## Формат 'json'

Добавлен, начиная с версии 3.4.1

Формат `json` используется для модификации настроек в формате `json` (например `Preferences` пакета `www-client/chromium`).

Пример `json`

```
{"google": {"services": {"signin": {"LSID": "", "SID": ""}},  
"intl": {"accept_languages": "ru-RU,ru,en-US,en", "status": 1}}
```

При объединении элементов в конфигурационном файле меняются только конкретные значения.

Для изменения правил объединения действующих по умолчанию, в начале имени элемента в файле шаблона добавляется управляющие символы:

- "-" - ветка заменяется
- "!" - ветка удаляется

Пример объединения

```
# Calculate format=json  
{"intl": {"accept_languages": "en"}}
```

```
{"google": {"services": {"signin": {"LSID": "", "SID": ""}},  
"intl": {"accept_languages": "en"}}
```

Пример удаления

```
# Calculate format=json  
{"!intl": ""}
```

```
{"google": {"services": {"signin": {"LSID": "", "SID": ""}},  
"-intl": {"accept_languages": "en"}}
```

Пример замещения

```
# Calculate format=json  
{"-intl": {"accept_languages": "en"}}
```

```
{"google": {"services": {"signin": {"LSID": "", "SID": ""}},  
"intl": {"accept_languages": "en"}}
```

Ветка настроек также будет замещена если в шаблоне содержится значение.

Пример замещения

```
# Calculate format=json  
{"intl": 5}
```

## Расположение

### Правила именования файлов

Этот шаблон заменит в конфигурационном файле *TEXT&DATA* на \_TEXT\_CONFIG\_.

**Формат 'diff'**

Добавлен начиная с версии 3.1.4

Формат *diff* используется для наложения diff патчей на исходный код пакета. Сам файл в системе не создается, а полученное содержимое обрабатывается относительным каталогом указанного в параметре *path*.

Пример:

```
# Calculate format=diff
--- panel-plugin/xkb-cairo.c 2012-07-17 16:23:24.997030066 +0400
+++ panel-plugin/xkb-cairo.c 2012-07-17 16:47:34.107054590 +0400
@@ -27,7 +27,7 @@
#include "xkb-util.h"
#include "xfc4-xkb-plugin.h"

#define XKB_PREFERRED_FONT "Courier New, Courier 10 Pitch, Monospace Bold %d"
#define XKB_PREFERRED_FONT "Droid Sans, Courier New, Courier 10 Pitch,
Monospace Bold %d"
```

### Формат 'kernel'

Добавлен начиная с версии 3.3.0.16

Формат *kernel* используется для объединения конфигурационных файлов ядра без сохранения комментариев.

Пример:

```
# Calculate format=kernel
CONFIG_XFS_FS=m
CONFIG_RELSERFSS_FS=y
# CONFIG_EXT3_FS is not set
!CONFIG_EXT3_FS_POSIX_ACL=
!CONFIG_EXT3_FS_SECURITY=
!CONFIG_EXT3_FS_XATTR=
```

### Формат 'dconf'

Добавлен, начиная с версии 3.3.2.9

Формат *dconf* используется для модификации настроек пользователя, хранящихся в реестре *dconf*. Формат *dconf* поддерживает параметр заголовка *dconfPath* для возможности изменять базовый путь настроек.

Пример:

```
# Calculate format=dconf dconf=/org/gnome/eog/
[view]
background-color='#000000'
use-background-color=true
statusbar=true
toolbar=true
```

[ui]

### Правила именования директорий

Имена файлов шаблона могут содержать условные операторы, определяющие правила, при выполнении которых файл шаблона будет перенесен в систему. Подобно ссылке на страницу сайта условия отделяются от имени знаком вопроса (?), после которого могут идти условные операторы, подробно описанные в условных блоках (см. выше).

Арифметические операции могут объединяться условием И (&) или ИЛИ (?). Приоритет в данном случае будет отдаваться условию И.

Пример:

```
grub.conf?os_linux_shortname=CDS?os_linux_shortname==CLD
```

В приведенном примере файл шаблона выполнит настройки загрузчика как для систем Calculate Linux Desktop, так и для Calculate Directory Server.

При наличии условных операторов в заголовке файла шаблона для переноса файла шаблона в систему должны выполняться оба условных выражения.

### Правила именования директорий

Правила именования директорий схожи с правилами именования файлов. При выполнении условий, заданных условными операторами, директория будет перенесена в систему, в противном случае директория вместе с содержимым будет пропущена.

### Права доступа

Права доступа конфигурационного файла после объединения с файлом шаблона устанавливаются следующим приоритетом:

- при наличии конфигурационного файла в системе права будут сохранены неизменными;
- в случае отсутствия оригинального конфигурационного файла, права будут выставлены как 644 для файла и 755 для директории;
- в случае установки значения переменной "chmod" или "chown" заголовка файла шаблона, права на конфигурационный файл будут изменены согласно установленному значению.

### Символические ссылки

Для создания символьических ссылок используйте параметры "link + symbolic" заголовка файла шаблона. Если помимо заголовка файл шаблона будет содержать тело шаблона, оригиналный файл будет модифицирован согласно правилам объединения.

Calculate не будет переносить так называемые "битые ссылки" - ссылки, не ведущие ни на какой файл (директорию), если явно не указан параметр *force*.

### Схема объединения

Объединение - изменение настроек оригинального файла настройки в соответствии с настройками файла шаблона.

В процессе объединения все записи оригинального файла и файла шаблона разбиваются на элементы: области, переменные, списки, разделенные списки, комментарии, управляющие элементы (см. ниже). Файл шаблона должен быть составлен с применением синтаксиса оригинального файла. Расположение элементов оригинального файла при объединении сохраняется. Во время объединения комментарии из файла шаблона не переносятся.

### Правила объединения

Над элементами могут происходить операции Объединение, Замена, Удаление:

## Объединение

- Отсутствующие в оригинальном файле элементы дописываются в конец области. При этом в случае наличия перед вставляемым элементом перевод строки добавляется перед последним элементом разделённого списка конфигурационного файла.
- В случае объединения разделённого списка, отсутствующие элементы добавляются следом за вставляемым элементом.

Замена

- Значение элементов заменяется на новое. При этом форматирование переносится из файла шаблона.

Удаление

- Элемент удаляется вместе с переводом строки, стоящим перед элементом.

## Правила объединения, действующие по умолчанию

Правила объединения действуют на элементы с одним именем, расположенные в **одиной области шаблона**. При нахождении различий, по умолчанию действуют **следующие правила**:

- **Области** - содержите двух областей обединяется (+).
- **Переменные** - значение переменных заменяется (-).
- **Списки** - содержимое списков заменяется (-).

• **Разделенные списки** - аналогично правилу обединения переменных - значения списков заменяются (-).

## Изменение правил объединения

Для изменения правил объединения действующих по умолчанию, в начале имени **элемента** в файле шаблона добавляется управляющие символы:

- "+" - обединить элементы (для областей и списков); после обединения остаются только уникальные элементы
- "\_" - значение элемента заменяется
- "!" - элемент удаляется

В CXmlConf описание файла шаблона эти правила определяются тэгом "!" .

## Формат CXmlConf

CXmlConf - универсальный формат описания конфигурационных файлов. Служит для **выборочного изменения настроек** большинства распространенных типов конфигурационных файлов ОС Linux/Unix. XML файл описания настроек разбивает конфигурационный файл на логические структуры - элементы, пригодные для последующего объединения. После объединения файл может быть преобразован в первоозданный вид за некоторыми исключениями (см. Схема объединения).

Описания элементов вкладываются в конструкцию:

```
<cxmlconf>
<head>
  <ver>версия формата</ver>
  <format>формат конфигурационного файла</format>
</head>
<body>
  [<area>...</area>... <field>..</field>]
</body>
</cxmlconf>
```

## Примеры использования

```
<gconf>      </dir>
              </dir>
            </gconf>

# Calculate format=xm_xml_xfce
<?xml version="1.0" encoding="UTF-8"?>
<channel name="xfce4-session" version="1.0">
  <property name="general" type="empty" action="drop">
    <property name="FailsafeSessionName" type="empty" value="" />
    <property name="SessionName" type="string" value="Default" />
    <property name="SaveOnExit" type="bool" value="true" />
  </property>
</channel>
```

## Формат 'patch'

Формат **patch** используется для обработки конфигурационных файлов при помощи регулярных выражений языка программирования Python.

### Особенности.

Формат **patch** использует обработку конфигурационного файла на основании шаблона (тип объединения **patch**); при этом не происходит объединения шаблона и конфигурационного файла.

Обработка исходного файла происходит целиком, а не построчно, поэтому символы '\n' и '\$' означают соответственно начало и конец файла. Символ ' ' означает любой символ исключая перевод строки.

Начиная с версии calculate - lib - 3.1.7 - r5 в параметре **patch** добавлены опции **multiline** и **dotall**. При включенном **multiline** '\n' и '\$' обозначают начало и конец строки. При включенном **dotall** '.' включает в себя перевод строки.

### Описание.

Шаблон формата **patch**:

```
# Calculate format=patch
<reg>регулярное выражение python 1</reg>
<text>текст 1 для замены регулярного выражения</text>
<reg>регулярное выражение python 2</reg>
<text>текст 2 для замены регулярного выражения</text>
...
...
```

В содержимом тегов **reg** и **text** символы "&" , "

Пример:

```
# Calculate format=patch
<reg>TEXT&amp;DATA</reg>
```

Пример элемента "items":

```
<items>
  <item name="xfce4-mixer-plugin" id="9"/>
  <item name="clock" id="10"/>
  <item name="separor" id="52"/>
  <item name="actions" id="12"/>
</items>
```

Формат xml\_gconf

Формат состоит из элементов "entry".

Если у элемента "entry" есть атрибут "type" (список) или атрибут type="string" то этот элемент и внутренние элементы будут заменены на элементы из шаблона, в ином случае элементы будут объединены.

Атрибут элемента "entry" - "mtime" при изменении элемента "entry" показывает время, когда произошло изменение в секундах.

Пример шаблона xml\_gconf:

```
# Calculate format=xml_gconf
<?xml version="1.0"?>
<gconf>
  <entry name="rgba_order" mtime="1235158855" type="string">
    <stringvalue>rgb</stringvalue>
  </entry>
  <entry name="dpi" mtime="1235162438" type="float" value="86">
    </entry>
  <entry name="hinting" mtime="1235266915" type="string">
    <stringvalue>full</stringvalue>
  </entry>
  <entry name="antialiasing" mtime="1235266915" type="string">
    <stringvalue>rgba</stringvalue>
  </entry>
</gconf>
```

Формат состоит из элементов "dir" и "entry" и их атрибутов.

```
# Calculate format=xml_gconf_tree
<?xml version="1.0"?>
<gconf>
  <dir name="desktop">
    <dir name="gnome">
      <entry name="percent_manager">
        <dir name="volume_manager">
          <entry name="percent_used" mtime="1285580987" schema="/schemas/desktop/gnome/volume_manager/percent_used"/>
        </dir>
      </entry>
    </dir>
  </dir>
</gconf>
```

Где:

- ver - передает номер версии разметки
- format - формат конфигурационного файла (определяется по распространенным программам).

Все элементы (см. ниже) помещаются внутри элемента .

## Области

Области конфигурационных файлов разграничивают **пространство имен переменных**. Области могут содержать логические структуры, в том числе другие области (например: { {filename} | named.conf } ).

Области помещаются в конструкцию:

```
<area>
  <caption>
    <name>Заголовок области</name>
    <action>join|replace|drop</action>
    <quote>Начальная часть области (заголовок)</quote>
    <quote>Завершающая часть описания области</quote>
  </caption>
  [<field></field>...]
</area>
```

## Переменные

Переменные имеют запись в виде:

```
<field type="var">
  <name>имя переменной</name>
  <value>значение переменной</value>
  <action>join|replace|drop</action>
  <quote>Оригинальный текст описания</quote>
</field>
```

В некоторых конфигурационных файлах, например, /etc/openldap/slapd.conf, встречается конструкция:

```
index  cn   pres,sub,eq
       sn   pres,sub,eq
       uid  pres,sub,eq
```

В этом случае имя переменной состоит из первой и второй части, а значение - из третьей.

Стока

```
index  cn   pres,sub,eq
```

В XML это будет выглядеть так:

```
<field type="var">
  <name>indexcn</name>
  <value>pres,sub,eq</value>
  <quote>index cn
</field>
<field type="br" \>
```

## Списки

По примеру файла named.conf, блок "listen-on" может содержать одни значения - значения блока, а не переменных.

Для обозначения значений служит конструкция:

```
<field type="List">
  <name>hostallow</name>
  <value>192.168.0.0/24</value>
  <action>join|replace|drop</action>
  <quote>оригинальный текст списка</quote>
</field>
```

где внутри блока "" сохраняется оригинальный текст описания значения, без завершающего перевода строки.

### Разделённые списки

Файл настроек веб-сервера Apache может содержать инструкцию "Include", позволяющую делать исходный файл модульным. Подобные случаи описываются в "CXmlConf", как "разделённые списки".

Разделенные списки описываются следующей конструкцией:

```
<field type="SepList">
  <name>Include</name>
  <value>/etc/apache2/modules.d/* .conf</value>
  <action>join|replace|drop</action>
  <quote>оригинальный текст списка</quote>
</field>
```

### Комментарии

При объединении конфигурационных файлов комментарии оригинального файла сохраняются в ненормированном виде.

Все типы комментариев помещаются в конструкцию "". В тексте, помечаемом "", сохраняются символы комментария и перевод строки:

```
<field type="Comment">
  <quote>оригинальный текст комментария</quote>
</field>
```

Комментарии не могут быть вложенными (быть описаны в других конструкциях комментариев).

### Управляющие элементы

Для обозначения перевода строк служит конструкция:

```
<field type="Urg">
  <quote>разделительные элементы (пробелы, табуляция)</quote>
</field>
```

Где:

- "quote" содержит элементы форматирования (пробелы, табуляция)

Формат xml\_xferpanel

Формат состоит из следующих элементов: "panels", "panel", "properties", "property", "items", "item".  
Если элемент "items", то этот элемент и внутренние элементы будут заменены на элементы из шаблона.

Иначе элементы будут объединены.

## Формат XML

### Поддерживаемые форматы XML

В настоящее время реализована поддержка форматов xml\_xfce (XML файл для конфигурирования оконного менеджера XFCE), xml\_xferpanel (XML файл для конфигурирования панелей оконного менеджера XFCE), xml\_gconf (XML файл для конфигурирования GNOME).

### Отличия от формата CXmlConf

- XML файлы хранятся и обрабатываются без перевода их в другой формат.
- Все правила, функции, переменные действуют на файл XML-шаблона аналогично обычному шаблону за исключением операций объединения элементов XML.

### Объединение элементов XML

Для объединения элементов XML в тексте XML шаблона используется атрибут XML-элемента - 'action' Допустимые значения атрибута

- action="join" - элементы объединяются (действует по умолчанию)
- action="replace" - элемент шаблона замещает элемент файла
- action="drop" - элемент файла удаляется

Формат xml\_xfce

Формат состоит из элементов "channel" и "property".

В атрибутах "channel" находится имя и версия файла.

Внутри элемента "channel" находятся элементы "property".

Пример элемента "channel":

```
<channel name="xfwm4" version="1.0">
  <property name="general" type="empty">
    ...
  </property>
</channel>
```

Пример элемента "property":

```
<property name="wrap_workspaces" type="bool" value="false"/>
```

У каждого элемента "property" есть атрибут type.

Если type="array", то этот элемент и внутренние элементы будут заменены на элементы из шаблона.

Иначе элементы будут объединены.

Пример type="array":

```
<property name="workspace_names" type="array">
  <value type="string" value="Рабочее место 1"/>
  <value type="string" value="Рабочее место 2"/>
  <value type="string" value="Рабочее место 3"/>
  <value type="string" value="Рабочее место 4"/>
</property>
```

Формат xml\_xferpanel

Формат состоит из следующих элементов: "panels", "panel", "properties", "property", "items", "item".  
Если элемент "items", то этот элемент и внутренние элементы будут заменены на элементы из шаблона.  
Иначе элементы будут объединены.

элементов.

- **install\_flash\_vars** - вспомогательный метод, формирующий структуру объекта ViewInfo для методов install\_flash и install\_flash\_view;
- **Install** - основной класс установки и настройки системы.

#### Установка PXE

Установка PXE включает в себя:

- **install\_pxe** - прс метод, проверяющий входные параметры и запускающий отдельный процесс установки системы с применением [PXE](#) (Установка PXE доступна только на Calculate Directory Server);  
Входные данные метода **install\_pxe** аналогичны входным данным метода **install**:

  1. sid - идентификатор (номер) сессии;
  2. info - объект типа InstallInfo, хранящий в себе все параметры, необходимые для установки системы.

Результатыющие данные метода **install\_pxe** - список объектов типа **RetunedMessage**, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **install\_pxe\_view** - прс метод, создающий объект ViewInfo с данными о параметрах установки и передающий его клиенту;

Входные данные метода **install\_pxe\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатыющие данные метода **install\_pxe\_view** - объект типа ViewInfo, несущий информацию о всех параметрах установки, допустимых значениях и способах отображения элементов.

- **install\_flash\_vars** - вспомогательный метод, формирующий структуру объекта ViewInfo для методов install\_flash и install\_flash\_view;

• **Install** - основной класс установки и настройки системы.

#### Категория Настройка

К категории **Настройка** относятся:

- Локализация (setup\_locale);
- Загрузка (setup\_boot);
- Сеть (setup\_network);
- Видео (setup\_video);
- Настройка пакета (core\_setup);
- Система (setup\_system).

#### Локализация

Настройка локализации, заключающаяся в настройки языка и часового пояса, включает в себя:

- **setup\_locale** - прс метод, проверяющий входные параметры и запускающий отдельный процесс настройки локализации;

Необходимость указания модуля зависит от значения модуля по умолчанию. Модуль по умолчанию задается параметром env в заголовке шаблона и распространяется на все вложенные шаблоны.

Пример:

```
# Calculate env=install
```

```
Modes "#-os_install_x11_resolution-#"
```

#### Передача значения функции

Значение переменной, заключенной в #-#, вычисляется и подставляется в шаблон в первую очередь. Если переменная будет записана в параметрах функции, для вычисления последней будет использоваться рассчитанное значение переменной.

Пример:

```
<entry name="!paned_size" type="int" value="#-sum(ysize, #-os_x11_height-# / 3)-#"/>
```

В примере функция **sum** получит три значения: параметр **ysize**, пустое значение и формулу для расчёта содержимого переменной **os\_x11\_height** (разрешение экрана по вертикали), поделенное на 3.

#### Условия в заголовках шаблонов

Заголовок файла шаблона может включать условные выражения, в случае успешного выполнения которых шаблон будет использоваться либо нет при настройке системы.  
Пример заголовка шаблона с условием проверки системы:

```
# Calculate os_install_linux system==server
```

#### Условные выражения

В настоящее время поддерживаются простые конструкции условных блоков. Условное выражение, состоящее из одной или более переменных (или функций), начинается с #? и заканчивается #. Далее следует блок текста, после чего завершающая конструкция из первой переменной, заключенной в #. Пример:

```
#?hr_laptop==#
numlock on
#hr_laptop#
```

или

```
#?os_locale_language=ru&&pkg(media-gfx/clx-themes)!=#
current_theme calculate_ru
#os_locale_language#
```

Во втором примере в условном выражении участвует функция **pkg()**, и её значение сравнивается с пустотой. Условие будет выполнено, если в системе используется русский язык и установлен пакет **media-gfx/clx-themes**.

## Хранение настроек профиля пользователя

- Хранение настроек профиля пользователя
- [./calculate/init.env](#)
- [./calculate/desktop.env](#)

- **install** - grpc метод, проверяющий входные параметры и запускающий отдельный процесс установки системы;
  - **logout**
- ### **~/calculate/ini.env**
- Файл предназначен для хранения переменных функции ini(). Формат файла - samba.
- Пример:
- ```
[main]
var1 = test VAR 1
```
- Примечание:
- Возможно любое название секции (по умолчанию main);
  - Возможно любое название переменной.
- ### **~/calculate/desktop.env**
- Файл предназначен для хранения параметров клиента. Формат файла - samba.
- Пример:
- ```
[rsync]
files = <количество файлов в профиле_пользователя>
exitcode = <код_возврата_rsync_при_ошибке>

[main]
status = success
version = 2.1.11
```
- Секции:
- rsync - секция относящаяся к rsync
  - main - секция общих параметров Параметры:
  - status - состояние (error, process, success)
  - version - версия клиента
- ### **~/calculate/server.env**
- Файл предназначен для передачи параметров серверу. Формат файла - plasma.
- Пример изменения пароля для пользователя на сервере (сервисы unix, samba):
- ```
[command] [password_samba]
run=on
unix_hash=<ssha_xesh_нового_пароля_unix>
samba_lm_hash=<nt_xesh_нового_пароля_samba>
samba_nt_hash=<nt_xesh_нового_пароля_samba>
samba_nt_hash_old=<nt_xesh_старого_пароля_samba>
status=process
date=YYYY-mm-dd_NN:MM:SS
```
- Выполнение команды на сервере для создания инкрементального архива:
- ```
[command] [pack_profile]
run=on
status=process
date=YYYY-mm-dd_NN:MM:SS
```
- Входные данные метода **install**:
1. sid - идентификатор (номер) сессии;
  2. info - объект типа InstallInfo, хранящий в себе все параметры, необходимые для установки системы (класс InstallInfo объявлен в файле install/cl\_wsdl\_install.py пакета calculate-install (/usr/lib/python2.7/site-packages/calculate/install/cl\_wsdl\_install.py)).
- Результатирующе данные метода **install** - список объектов типа ReturnedMessage, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.
- **install\_view** - grpc метод, создающий объект ViewInfo с данными о параметрах установки и передающий его клиенту;
- Входные данные метода **install\_view**:
1. sid - идентификатор (номер) сессии;
  2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").
- Результатирующе данные метода **install\_view** - объект типа ViewInfo, несущий информацию о всех параметрах установки, допустимых значениях и способах отображения элементов (описание типа ViewInfo см. выше в разделе "Метод, предоставляющий дополнительную информацию").
- **install\_vars** - вспомогательный метод, формирующий структуру объекта ViewInfo;
- **Install** - основной класс установки и настройки системы.
- Установка системы на Flash
- Установка системы на Flash включает в себя:
- **install\_flash** - grpc метод, проверяющий входные параметры и запускающий отдельный процесс установки системы на Flash носителе;
- Входные данные метода **install\_flash** аналогичны входным данным метода **install**:
1. sid - идентификатор (номер) сессии;
  2. info - объект типа InstallInfo, хранящий в себе все параметры, необходимые для установки системы.
- Результатирующе данные метода **install\_flash** - список объектов типа ReturnedMessage, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.
- **install\_flash\_view** - grpc метод, создающий объект ViewInfo с данными о параметрах установки и передающий его клиенту;
- Входные данные метода **install\_flash\_view**:
1. sid - идентификатор (номер) сессии;
  2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").
- Результатирующе данные метода **install\_flash\_view** - объект типа ViewInfo, несущий информацию о всех параметрах установки, допустимых значениях и способах отображения

- `setProgress` - установить процент выполнения для текущего прогресса. Имеет три входных параметра:
  1. `perc` - процент выполнения текущей задачи;
  2. `short_message` - краткое сообщение для текущего процента выполнения;
  3. `long_message` - длинное сообщение для текущего процента выполнения;
- `getProgress` - получение текущего прогресса (процента) выполнения задачи;
- `printTable` - отображение таблицы. Имеет шесть параметров (три основных и три, использующихся при необходимости взаимодействия пользователя с таблицей в графических клиентах):
  1. `table_name` - отображаемое название таблицы;
  2. `head` - заголовок таблицы;
  3. `body` - тело таблицы;
  4. `onClick` - назначение метода, вызываемого при нажатии на строку таблицы. При этом будут переданы параметры из этой строки, описанные в поле `fields` (см. далее);
  5. `fields` - список полей, необходимых для передачи в вызываемый метод, имя которого хранится в параметре `onClick`. Список `fields` равен по длине заголовку таблицы, неиспользуемые для передачи поля равны пустым строкам. Например, при `fields = ["cl_group_name", ""]` в вызываемый метод передается параметр `cl_group_name`, значение для которого будет взято из первого столбца выбранной строки;
  6. `addAction` - добавить кнопку добавления над таблицей (кнопка с иконкой "плюс"), при нажатии на которую будет вызван метод, имя которого указано в параметре `addAction`;

- `setStatus` - установить статус процесса. 0 - процесс успешно завершён, 1 - процесс выполняется, 2 - процесс завершён с ошибкой;
  - `getStatus` - получить текущий статус процесса;
  - `askQuestion` - получить от пользователя дополнительные данные (задать вопрос введённое пользователем значение);
 

Получает один параметр `message` - текст сообщения. Вызов этого метода возвращает введённое пользователем значение;
  - `askPasswordField` - получить пароль от пользователя. Имеет два параметра: `message` (текст сообщения) и `twice` (необходимость дублирования ввода пароля).
  - `writeToFile` - записать текущее состояние процесса в файл (по умолчанию `/var/calculate/server/pids/N.pid`, где N - идентификатор процесса (pid)).
  - `askChoice` - получить от пользователя выбор из представленных вариантов
  - `askConfig` - получить от пользователя подтверждение
  - `isInteractive` - узнать интерактивно ли запущена действие в консоли (действие может из консоли может быть запущено с ключом `-f`)
- Метод должен вернуть значение `True` при успешном завершении и значение `False` в случае ошибки.

## Категория Установка

Для установки системы необходим установленный пакет `calculate-installer`. В категории **Установка** находятся три метода (при наличии необходимых прав у сертификата):

- Установка системы (`install`);
  - Установка на Flash (`install_flash`);
  - Установка PXE (`install_pxe`).
- Все методы установки системы работают через общий метод `installCommon` и описаны в файле `install/cl_wsd़l/install.py` пакета `calculate-installer` (`/usr/lib/python2.7/site-packages/calculate/install/cl_wsd़l/install.py`).

**Установка системы**  
Установка системы включает в себя:

- `action` - команда создания архива;
- `status` - состояние выполнения команды (`error`, `process`, `success`).

## ~/.logout

Устаревший формат файла. Файл предназначен для хранения состояния клиента (совместимость со старыми версиями сервера и клиента). Возможное содержимое текстового файла (`ERROR`, `PROCESS`, `SUCCESS`).

:Пример:

```
ERROR
```

Содержимое файла в случае ошибки.

## Обновление системы cl-update

- Обновление системы `cl-update`
  - Описание функциональных возможностей
  - Описание опций утилиты
  - Ключи для управления синхронизацией и кешами
  - Ключи примечания шаблонов, ревизий
  - Другие ключи
- `cl-update` - утилита, выполняющая обновление системы со всеми необходимыми сопутствующими действиями.

## Описание функциональных возможностей

Порядок обновления в общем случае следующий:

1. Синхронизация репозиториев дистрибутива
2. Если репозитории были обновлены, то выполняются действия `egencache` и `edit-update`
3. Обновление ревизии и обновление мира
4. Обновление системы
5. Обновление Python/Perl с пересборкой поврежденных пакетов при необходимости
6. Удаление ненужных пакетов
7. Пересборка модулей ядра при необходимости
8. Пересборка прочих поврежденных пакетов при необходимости
9. Пересборка пакетов для Xorg-сервера, если в этом есть необходимость
10. Выполнение `dispatch-conf`

## Описание опций утилиты

При запуске `cl-update` без параметров обновление будет происходить по общей схеме, но можно изменить поведение утилиты, используя ключи.

### Ключи для управления синхронизацией и кешами

- r REPOSITORIES, --repositories REPOSITORIES

Задает синхронизируемые репозитории дистрибутива (все по умолчанию) ('list' для отображения возможных значений). При указании опции позволяет синхронизировать лишь выбранный репозиторий.

- branch BRANCHES

Устанавливает ветки для репозиториев (`REPOSITORY:BRANCH`). Позволяет переключать между `master/develop/update` ветками репозиториев.

- force-egencache

Принудительно обновляет кэш оверлеев.

- `skip-ejectcache`
- Пропускает обновление кэша оверлеев.
- `auto-ejectcache`

Обновляет кэш оверлеев в том случае, если тог устарел (действие по умолчанию).

- `force-eix-update`

Принудительно обновляет кэш eix.

- `skip-eix-update`

Пропускает обновление кэша eix.

- `auto-eix-update`

Обновляет кэш eix, если он устарел (поведение по умолчанию).

- `o [ON/OFF]`, - `update-other [ON/OFF]`

Выполняет обновление сторонних оверлеев (не являющихся репозиториями дистрибутива). По умолчанию выключено. Для того, чтобы изменить действие по умолчанию, надо изменить значение переменной `cl_update_other_set` секции `update`; для этого можно выполнить следующую команду:

```
-s [ON/OFF], - sync-only [ON/OFF]
```

Не обновлять пакеты. Выполняет синхронизацию и применяет ревизии.

## Ключи применения шаблонов, ревизий

- `rebuild-world [ON/OFF]`

Переформирует world-файл согласно шаблонам дистрибутива на world-файл по умолчанию для данного профиля.

- `update-rev [ON/OFF]`

Обновляет ревизию дистрибутива до текущей в шаблонах.

- `T TEMPLATES, --templates TEMPLATES`

Выбор местонахождения шаблонов `calculate,distros,local,remote,clt ('list' для отображения возможных значений)`

- `usenew-conf`

Использовать новые конфигурационные файлы.

- `skip-conf`

Пропустить обновление конфигурационных файлов.

- `dispatch-conf`

Обновить конфигурационные файлы вручную (по умолчанию).

## Другие ключи

- `p [ON/OFF], --pretend [ON/OFF]`

- `values` - описание типов данных для столбцов (доступные значения, комментарии, доступность на редактирование);
- `onChanged` - зависимость значений столбца от ключевых значений (первого столбца);
- `incompatible` - булево значение доступности параметра. Если incompatible равно True, то данный параметр недоступен.
- `default` - состояние параметра, находящегося в расширенных настройках. Определется, будет ли отображено значение параметра в дополнительных настройках или оно будет пустым (авто).
- `opt` - данные для работы из консоли. Состоит из:
  - `help` - описание параметра (отображается при вызове метода с ключом `-h, --help`);
  - `metavalue` - строка, определяющая требуемое значение параметра;
  - `shortopt` - короткий (односимвольный) ключ параметра (например, `-h`);
  - `longopt` - основной ключ параметра (например, `--help`).

## Метод процесса

- Метод, выполняющий выбранный пользователем действие и запускающийся в отдельном процессе, является обычным методом класса `CoreVSDL`. Для взаимодействия с клиентами (сообщений о прогрессе выполнения процесса, ошибок, успешном выполнении и др.) он использует ряд встроенных функций (методов), описанных в пакете `calculate-core` в файле `core/server/func.py`, таких как:
- `addMessage` - основной метод добавления сообщения в стек сообщений для пользователя (сообщения имеют тип `Message`). Большинство методов являются удобными обёртками для данного метода. Имеет три входных параметра:
    1. `type` - тип передаваемого сообщения;
    2. `message` - текст передаваемое сообщение;
    3. `id` - автоматически генерируемый идентификатор объекта (номер таблицы, прогресса задания и др.);
  - `printDefault` - вывод сообщения без отметок
  - `printPre` - вывод блока неформатируемого текста
  - `printSUCCESS` - обычное сообщение, имеет один параметр `message`(строку сообщения);
  - `printWARNING` - сообщение с предупреждением, имеет один параметр `message`(строку сообщения);
  - `printERROR` - сообщение об ошибке, имеет один параметр `message`(строку сообщения).
  - Прерывает текущую задачу:
    1. `message` - название задачи;
    2. `progress` - наличие контроля прогресса выполнения задачи (отображение прогрессбара клиентами);
  - `startTask` - начало новой задачи, имеет три параметра:
    1. `message` - название задачи;
    2. `progress` - наличие контроля прогресса выполнения задачи (отображение прогрессбара клиентами);
    3. `num` - количество частей для контроля прогресса. Используется только при `progress` равном True и определяет, сколько частей от общего прогресса выполнения займет данная задача. Этот параметр не обязательен, актуален для клиента, работающего в упрощённом режиме вывода с одним прогрессом для всех задач текущего процесса.
  - `setTaskNumber` - установка числа частей для задач. Вызывается перед первым вызовом метода `startTask`. Если в вызовах `startTask` не указывается количество частей (параметр `num`), то в вызове `setTaskNumber` нет необходимости. Имеет один целочисленный параметр - число частей для задач, которое должно быть равно сумме всех значений параметра `num`, указанных в методах `startTask`;
  - `endTask` - сообщение о завершении текущей задачи. Имеет два параметра:
    1. `result` - сообщение о завершении задачи;
    2. `progress_message` - сообщение, используемое при выводе прогресса;
  - `endFrame` - сообщение пользователя о завершении работы процесса;
  - `addProgress` - добавление прогресса для задания;

Методы, предоставляющие дополнительную информацию клиентам, имеет один декоратор `rpc`, служащий для указания типа входных и возвращаемых данных. Для метода `core_setup_view` декоратор `rpc` имеет вид:

```
@rpc(Integer, ViewParams, _returns = ViewInfo)
```

Метод принимает два параметра: целое число (номер сессии) и обязательный параметр типа `ViewParams`, а также возвращает обязательный параметр типа `ViewInfo`, объект которого несёт в себе всю информацию о параметрах метода `core-setup`.

В результате метод `core_setup_view` имеет вид:

```
@rpc(Integer, ViewParams, _returns = ViewInfo)
def core_setup_view (self, sid, params):
```

#### Параметр `ViewInfo`

Класс `ViewInfo` и все ниже описанные классы (`GroupField`, `Field`) описаны в пакете `calculate-core` в файле `core/server/api/types.py`.

Параметр `ViewInfo` представляет собой массив объектов **GroupField** и булевый параметр `has_brief`, сообщающий клиенту о том, следует ли выводить информацию о значении всех параметров перед непосредственным запуском метода. Каждый объект **GroupField** несёт информацию о группе параметров (например, настройка локализации в методе установки системы) и состоит из:

- `name` - имя группы параметров для отображения;
- `last` - булевое значение, указывает ли данная группа последний (используется для методов с несколькими штагами);
- `next_label` - строка, отображаемая на кнопке, осуществляющей переход на следующий шаг или запуск метода (по умолчанию "Далее" или "Ok", используется для графических клиентов);
- `prev_label` - строка, отображаемая на кнопке, осуществляющей переход на предыдущий шаг или выход из метода (по умолчанию "Назад" или "Отмена", используется для графических клиентов);
- Массив объектов типа `Field`.

#### Тип параметра `Field`

Объект типа `Field` - описание отдельного параметра, состоящее из полей:

- `name` - имя параметра;
  - `label` - отображаемое имя параметра;
  - `element` - тип элемента параметра (`table`, `radio`, `combo`, `multoice`, `input` и др.), определяет основные характеристики и правила работы с параметром;
  - `type` - тип данных переменной;
  - `help` - дополнительная вспомогательная информация о параметре;
  - `value` - текущее значение переменной, поступившее от сервера утилит;
  - `choice` - список доступных для выбора значений (для элементов `radio`, `combo`, `comboEdit`, `multoice` и др.);
  - `comments` - список комментариев, соответствующих каждому значению из `choice`, использующийся для отображения пользователю. Как правило, значения из `comments` имеют удобный для человека вид или перевод.
  - `listValue` - список выбранных значений для элементов множественного выбора (`multoice`, `multoice_add`, `selectable`, `selectable_add`);
  - `tableValue` - информация о таблице (только для элемента `table`), состоит из:
- `head` - заголовок таблицы, определяет название столбцов и ширину таблицы;
  - `body` - тело таблицы, представляет собой данные таблицы, поступающие от сервера утилит;

Вместо действительного обновления пакетов, только отобразить, что будет установлено.

```
-w wait - another_update [ON/OFF]
```

Ждать завершения выполнения другого запущенного cl-update. По умолчанию включено.

```
--schedule [ON/OFF]
```

Учитывать график автопроверки обновлений. Если временной интервал с последней проверки ещё не пропал, то обновление выполниться не будет.

```
-v [ON/OFF], -verbose [ON/OFF]
```

Включает подробный вывод.

```
-f, --force
```

Не задавать вопросы во время процесса.

## Смена профиля системы cl-update-profile

### Описание утилиты

cl-update-profile - утилита предназначена для смены репозитория и профилей системы.

В общем виде утилита использует следующим образом:

```
cl-update-profile [-url CL_UPDATE_PROFILE_REPO] [-s [ON/OFF]] [-f [PROFILE]]
```

### Описание опций

Установить репозиторий профиля

```
-url CL_UPDATE_PROFILE_REPO
```

Репозиторий указывается следующим образом (используется либо вместе с именем профиля, либо с опцией `list`):

```
cl-update-profile -url=git://git.calculate.ru/calculate/distros.git list
```

Команда выводет список профилей данного репозитория:

Профиль системы:  
[CLD] distros:CLD/amd64  
[CLDX] distros:CLDX/amd64  
[CLS] distros:CLS/amd64  
[CMC] distros:CMC/amd64  
[CDS] distros:CDS/amd64  
[CSS] distros:CSS/amd64

Для установки профиля аналогично прсыльщай команде, но используя имя профиля:

```
cl-update-profile -url=git://git.calculate.ru/calculate/distros.git
```

Выход команды:

Репозиторий  
\* Название репозитория: distros  
Профиль

```

* Профиль системы: distros:CLDX/amd64
* Название дистрибутива: Calculate Linux Desktop 15 xfce
* Используемые репозитории:
+--+
| Название | URL
+--+
| distros | git://git.calculate.ru/calculate/distro.git|
| calculate | git://git.calculate.ru/calculate/overlay.git|
| portage | git://git.calculate.ru/calculate/portage.git|
+--+
* Список пакетов системы: Обновить
* Пропустить настройку системы: нет

```

Запустить процесс? (Yes/No): yes

\* Синхронизация репозиториев

- [ ok ] \* Синхронизация Distroс репозитория ...
- [ ok ] \* Синхронизация Calculate репозитория ...
- [ ok ] \* Синхронизация Portage репозитория ...
- [ ok ] \* Синхронизация завершена

Настройка профиля

- \* Переключение на CLDX профиль ...

[ ok ] \* Обновление профиля завершено успешно

Для сервисов github и bitbucket поддерживается укороченный формат. Пример с использованием стороннего пользовательского репозитория:

```

cl-update-profile -url=github:laure76/list

```

Синхронизировать или нет репозитории:

-s [ON/OFF], --sync [ON/OFF]

Не задавать вопросы во время процесса:

-f, --force

Подробно о создании своего профиля и использования утилиты можно прочитать в [блоге](#).

## Сборка системы

- Сборка системы
- Введение
- Необходимые требования
- Использование
- Подготовка системы к сборке
- Смена профиля системы
- Обновление системы
- Создание загруженного образа
- Прерывание сборки
- Восстановление сборки
- Создание мультизагрузочной Live USB Flash

## Возвращаемый тип ReturnedMessage

- Для сообщения клиенту об успешном запуске процесса или об ошибках сервер утилит возвращает массив типовых сообщений типа ReturnedMessage. Каждое сообщение состоит из следующих полей:
- \* type - тип сообщения. В случае успешного запуска процессы type равно строке "pid", в случае ошибки "error", при необходимости сообщить предупреждение - "warning"
  - \* message - передаваемое сообщение. В случае успешного запуска процессы значение message равно идентификатору процесса (pid), в остальных случаях message хранит строку с сообщением об ошибке или предупреждении;
  - \* field - поле с ошибкой или предупреждением. Значение равно имени параметра, в котором допущена ошибка или для которого необходимо сообщить предупреждение;
  - \* expert - устанавливается, если поле находится в дополнительных настройках.

### Декоратор core.method

- Декоратор core.method используется для указания дополнительной информации о методе для клиента и имеет шесть параметров:
- \* category - указывает в какой категории будет находиться метод (для графических клиентов);
  - \* title - указывает отображаемое название метода, как правило является строкой для перевода;
  - \* image - указывает иконки для отображения (используется для графических клиентов. Иконки могут перечисляться через запятую и клиент использует первую найденную);
  - \* gui - определяет, используется ли данный метод для графических клиентов;
  - \* command - команда для создания символьской ссылки на сервер утилит cl-cse. Также указывает, используется ли данный метод для консольного клиента;
  - \* rights - указывает список прав, необходимых для возможности запуска данного метода. Для успешного запуска метода необходимы все указанные права.

### Для метода core.setup:

```

@core_method(category=__('Configuration'), title=__('Configure package'),
              image='applications-other', gui=True, command='cl-core-
setup',
              setup',
              rights=['configure'])

```

Декораторы указываются непосредственно перед объявлением метода и выполняются один раз. В итоге метод core\_setup имеет вид:

```

class CoreWssdl:
    @rpc(Integer, CoreSetupInfo, _returns = Array(ReturnedMessage))
    @core_method(category=__('Configuration'), title=__('Configure
package'),
                 image='applications-other', gui=True, command='cl-core-
setup',
                 setup',
                 rights=['configure'])
    def core_setup(self, sid, info):
        ...

```

## Метод, предоставляющий дополнительную информацию

- Рассмотрим на примере метода настройки пакетов в системе core\_setup.
- Для каждого метода, запускающегося процессом на выполнение, должен быть метод, который сообщает клиенту информацию об используемых параметрах. Имя такого метода составляется из имени основного метода и окончания \_view, например для метода core\_setup имя метода core\_setup\_view.

## Введение

Состав API (Интерфейс прикладного программирования утилит Calculate) - набор готовых классов, функций, структур и констант утилит Calculate, предоставляемых для использования во внешних программных продуктах. Calculate API предоставляет инструменты взаимодействия различных клиентов (таких как [cl-console](#) и [cl-console-gui](#)) с сервером утилит [cl-core](#). Подробнее см. [API в Википедии](#).

Доступ к API осуществляется через протокол НГТРС, использующий криптографический протокол SSL, который зашифровывает все передаваемые данные и этим защищает их от несанкционированного доступа. Установка SSL-соединения обязательна для вызова API методов.

В утилитах Calculate взаимодействие осуществляется по протоколу [SOAP](#). Обмен данными по этому протоколу ведется посредством XML-сообщений.

## Состав Calculate API

На стороне сервера утилит API реализовано с использованием библиотеки [dev-python/soaplib](#).

Все модули и функции Calculate API можно разделить на две большие группы:

- **Основные** - служат для вызова методов, непосредственно выполняющих выбранные пользователем действия (установка системы, настройка сети и др.);
- **Вспомогательные** - служат для обеспечения взаимодействия клиента и сервера утилит (получение списка доступных методов, получение сессии, проверка прав сертификата и др.).

API реализовано с помощью [RPC](#) декораторов библиотеки [soaplib](#), осуществляющих взаимодействие и проверку параметров (их число и типы данных).

## Основные модули

Основные модули и методы Calculate API осуществляют основные действия сервера утилит. Простейшие из них представляют собой отдельный класс с именем [CoreWsdl](#) (для импортирования в сервер утилит). В этом классе находятся как минимум три метода (два из которых [RPC](#) методы), связанных между собой:

- метод, запускающий процесс на выполнение;
- метод, предоставляемый дополнительную информацию клиенту;
- метод, выполняющий выбранный пользователем действие (представляет собой отдельный процесс).

## Метод, запускающий процесс на выполнение

Рассмотрим на примере метода настройки пакетов в системе [core\\_setup](#).

Имя метода, запускающего процесс на выполнение, является именем, по которому клиент вызывает данный метод (запускает процесс), в нашем случае это [\\_core\\_setup\\_](#). Основные функции методов, запускающих процесс:

- проверка поступающих от клиента параметров и сообщение об ошибках при необходимости;
- создание глобального словаря параметров процесса для хранения всей информации о процессе;
- запуск процесса на выполнение.

Каждый такой метод обрамляется двумя обязательными декораторами (подробнее о декораторах на [ru.wikipedia.org](#)): [rpc](#) и [core\\_method](#).

Декоратор [rpc](#) служит для указания типа входных и возвращаемых данных. В декораторе метода [core\\_setup](#) два входных параметра - целочисленный (номер сессии) и тип [CoreSetupInfo](#) (объект, хранящий необходимые значения для запуска и выполнения процесса), и один возвращаемый параметр - массив элементов, каждый из которых имеет тип [ReturnedMessage](#):

```
@rpc(Integer, CoreSetupInfo, _returns = Array(ReturnedMessage))
```

## Введение

Все операции по сборке образов дистрибутива Calculate Linux выполняются при помощи утилит Calculate. По завершению сборки создаётся загрузочный образ Live USB, который можно использовать как для работы, так и для установки системы на жёсткий диск компьютера. Начиная с версии утилит 3.4, модуль сборки системы был полностью переписан. Выполненные файлы утилит были переименованы и теперь начинаются с префикса 'cl-builder', после которого следует выполняемое действие (например, 'cl-builder-update').

## Необходимые требования

Для выполнения сборки вам понадобится выход в интернет, свободное место на диске, а также любой доступный "ISO-образ системы Calculate Linux": версии 15 или выше, например, [Calculate Linux Scratch](#) или Calculate Scratch Server. Вместо ISO-образа можно указать CD-привод или USB Flash с установленным Calculate Linux. Для больших возможностей в формировании собственных сборок не плохо иметь свой [Git-репозиторий](#) с Calculate-совместимым профилем системы. Для размещения репозитория подойдёт такой хостинг как [GitHub](#), [BitBucket](#) или любой другой, например, собственный, поднятый при помощи [Gitolite](#).

Для разворачивания и обновления образа без использования слоёв вам может потребоваться около 5 Гб свободного места на жёстком диске и 15 Гб при разворачивании образа без использования слоёв (м. Подготовка системы к сборке). Для создания образа дистрибутива потребуется около 2 Гб. В случае создания бинарных пакетов дополнительно может потребоваться до 5 Гб свободного места. Сборка дистрибутива может производиться как на отдельном разделе жёсткого диска, так и в выделенном каталоге файловой системы. Готовый образ сохраняется в директории [/var/calculate/Linux](#), бинарные пакеты сохраняются в директории [/var/calculate/remote/builder](#). Позаботьтесь о наличии свободного места по этим путям.

## Использование

Пакет sys-apps/calculate-utils 3.4, входящий во все дистрибутивы Calculate Linux 15, включает в себя необходимые компоненты для сборки системы. **Основные возможности:**

- поддержка графического интерфейса и работы из командной строки;
- поддержка работы с многослойной файловой системой OverlayFS;
- поддержка работы в системе, загруженной с Live USB;
- поддержка параллельной сборки нескольких дистрибутивов;
- создание мультизагрузочных USB Flash;
- профили сборки: Calculate-совместимые;
- поддерживаемые архитектуры: i686 и x86\_64;
- поддержка сборки 32-битных дистрибутивов на 64-битной системе.

Операции по сборке системы выполняются в разделе "Сборка" графической консоли утилит Calculate:



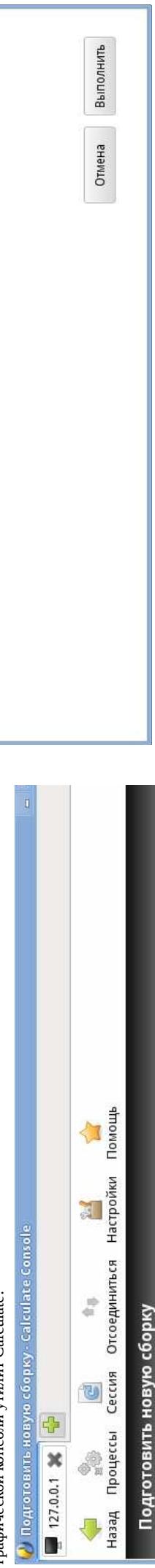
Для работы из командной строки вы можете использовать следующие утилиты:

- **cl-builder-prepare** (Подготовить новую сборку) - используется для подготовки данных для сборки;
- **cl-builder-profile** (Изменить профиль) - используется для смены профиля собираемой системы;
- **cl-builder-update** (Обновить сборку) - используется для обновления пакетов собираемой системы;
- **cl-builder-break** (Прервать сборку) - используется для прекращения сборки;
- **cl-builder-image** (Создать образ) - используется для создания загрузочного ISO-образа;
- **cl-builder-restore** (Восстановить сборку) - используется для восстановления данных сборки после перезагрузки машины;
- **cl-builder-menu** (Обновить меню загрузки) - используется для обновления мультизагрузочного меню Live USB.

Вызов утилит из консоли можно сочетать с работой графических утилит, т.к. они остаются полностью совместимыми.

### Подготовка системы к сборке

Для подготовки системы к сборке кликните по иконке "Подготовить новую сборку" в разделе "Сборка" графической консоли утилит Calculate.



### Подготовить новую сборку

#### Подготовить новую сборку

Исходный образ  
Нажмите для дополнительных настроек

В открывшейся странице выберите исходный образ системы. В случае загрузки с Live USB по умолчанию вам будет предложен загруженный образ. По необходимости используйте дополнительные настройки.

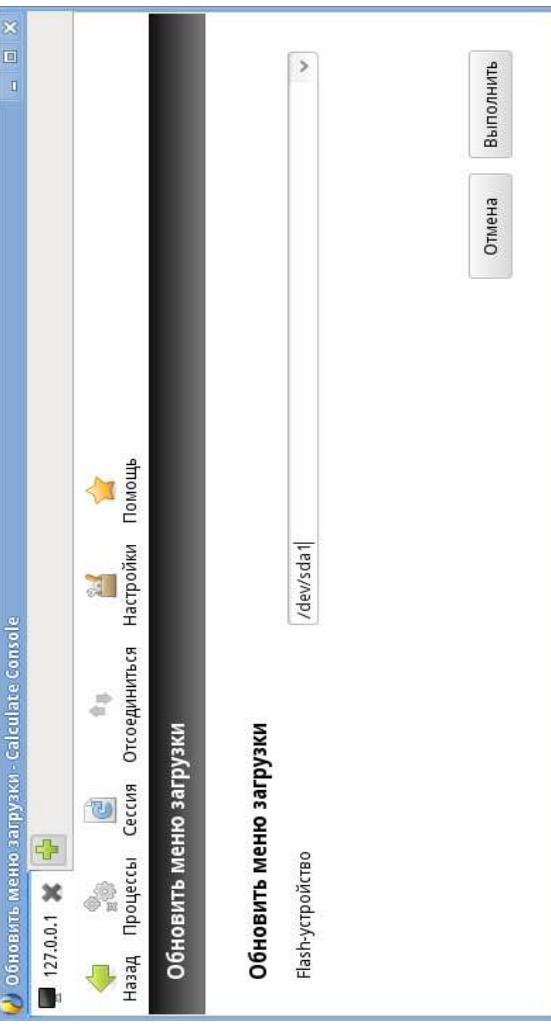
Для работы в терминале выполните:

```
cl-builder-prepare
```

#### Основные опции:

- --source SOURCE - исходный образ системы.

Во время работы с Live USB при создании образа загрузочное меню переформировывается таким образом, чтобы можно было выбрать загрузку с любого из созданных ISO-образов. Вы также можете просто скопировать в директорию /linux USB Flash другие ISO-образы Calculate Linux, а затем обновить меню загрузчика. Для этого кликните по иконке "Обновить меню загрузки" в разделе "Сборка" графической консоли утилит Calculate. После этого откроется следующая страница:



Вы можете сформировать меню флешки, загрузившись с неё, либо вставив в USB разъём и из текущей системы при помощи утилиты обновления.

При работе в терминале для обновления меню выполните:  
`cl-builder-menu -d устройство`

Вместо устройства можно указать путь, в который подключена USB Flash.

### Calculate API

- Calculate API
- Введение
- Состав Calculate API
- Основные модули
- Метод, запускающий процесс на выполнение
- Метод, предоставляемый дополнительную информацию
- Метод процесса
- Категория Установка
- Методы работы с сертификатами
- Категория Настройка
- Категория Утилиты
- Вспомогательные модули
- Методы работы с сессиями
- Методы работы с процессами
- Методы работы с кешем
- Прочие вспомогательные методы

В терминале прервать сборку можно, выполнив:

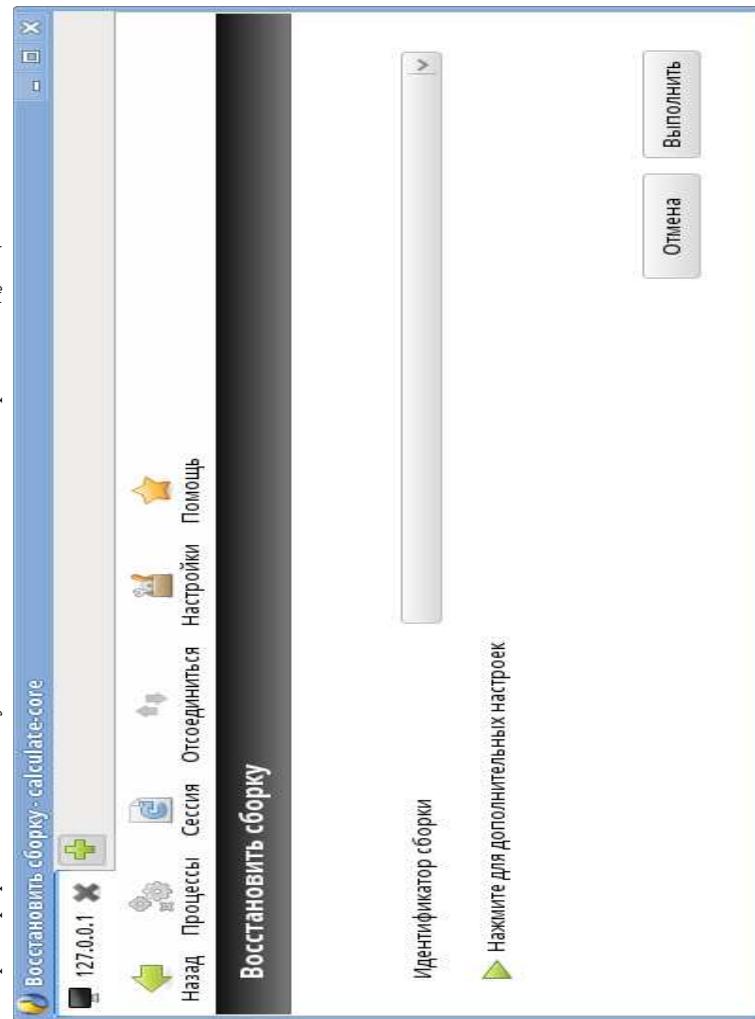
```
c1-builder-break
```

#### Параметры:

- **-id ID** - идентификатор сборки; необходим, только если производится несколько сборок одновременно.
- **-clear [ON/OFF]** - очистить данные после отключения сборки (актуально при сборке в отдельном каталоге). Опция включена по умолчанию.
- **-clear-pkg [ON/OFF]** - удалить бинарные пакеты

### Восстановление сборки

Если во время работы с дистрибутивом компьютер был перезагружен, вы можете выполнить восстановление прерванной сборки. Для этого кликните по иконке "Восстановить сборку" в разделе "Сборка" графической консоли утилит Calculate. После чего откроется следующее окно:



То же действие можно выполнить в терминале, набрав:

```
c1-builder-restore
```

В качестве параметра вы можете указать идентификатор сборки.

### Создание мультизагрузочной Live USB Flash

Вы можете полноценно работать с Live USB, создавая модификации текущего, либо любого другого ISO образа Calculate Linux. Для возможности сохранять на USB Flash данные система должна быть установлена на флешку при помощи утилит Calculate.

В качестве исходного образа может выступать ISO образ, CD-привод или Live USB с дистрибутивом Calculate Linux. Дополнительные опции:

- **-d DEST, -disk DEST** - раздел или директория для сборки. Убедитесь, что в разделе и в каталоге у вас не хранится информации, так как во время подготовки сборки она будет удалена.
- **-layers [ON/OFF]** - использовать многослойную файловую систему (OverlayFS) для подготовки сборки, при использовании этой опции образ не будет распакован в файловую систему. В файловой системе будут храниться только изменения от базового образа. Не используется при сборке в отдельном разделе.
- **-id ID** - идентификатор сборки. При одновременной сборке нескольких систем служит для обозначения сборки. По умолчанию обозначается как **имя\_репозитория\_имя\_профиля**.

Исходя из того, укажете ли вы раздел или директорию сборки, будете ли вы использовать многослойную файловую систему, у вас может быть три варианта подготовки образа:

1. Сборка в выделенной директории с использованием многослойной файловой системы.
2. Сборка в выделенной директории без использования многослойной файловой системы.
3. Сборка в выделенном разделе диска.

Сборка в выделенном разделе с использованием многослойной файловой системы не поддерживается. Для экономии времени и места, по умолчанию, при подготовке образа используется многослойная файловая система.

Во всех типах сборки доступ к корню собираемой системы можно получить в директории `/var/calculate/mount/katalog` сборки, где каталог сборки - это имя сборки с подменой символов ":" и "/" на "\_" (например: "distros\_CLSK\_apm64").

#### Во время подготовки системы к сборке выполняются следующие действия:

1. Подключение базового образа в директории `/var/calculate/mount/iso` и `/var/calculate/mount/squash`
2. Распаковка базового образа
3. Настройка шаблонами `builder/prepare` (событие `ac_builder_prepare`), `builder/setup` (событие `ac_builder_setup`)
4. Подключение точек монтирования `/dev`, `/dev/shm`, `/dev/pts`, `/proc`, `/sys`, `/var/calculate/remote`
5. Отключение базового образа системы

#### Особенности в подготовке системы к сборке при использовании выделенного раздела:

- Для развертывания образа потребуется выделенный раздел, данные в котором будут удалены при форматировании. В качестве файловой системы будет использована текущая файловая система, если она поддерживается утилитами.

#### Особенности в подготовке системы к сборке при использовании слоёв:

- Вместо распаковки исходный образ монтируется и остается промонтированным на всём протяжении сборки системы.
- Дельта выполняемых изменений хранится в директории `/var/calculate/builder/каталог_сборки`.

### Смена профилей системы

Смена профиля собираемой системы вам может понадобиться, главным образом, для выбора своего профиля, в котором вы можете описать особенности системы, такие как: состав пакетов, используемые флаги сборки, опции компилятора, настройки переменных утилит Calculate, шаблоны настройки систем, ebuild-ы пакетов и т.д.

Для переключения профиля, воспользуйтесь иконкой "Изменить профиль" в разделе "Сборка" графической консоли утилит, после чего откроется следующая страница:



По умолчанию отображаются профили репозитория distros. При использовании своего профиля, хранящегося, например, на Github, вы можете указать "github:autte76", где "github" - обозначение сервиса хостинга, "autte76" - имя вашей учётной записи. При указании сокращённого имени поиска профилей будет выполняться в репозитории "oneplay/git".

В дополнительных настройках вы можете указать "Обновить кш" для того, чтобы при наличии кэша репозитория данные были обновлены. Это может понадобиться, если вы захотите добавить новый профиль, отсутствовавший при предыдущем вызове команды.

После нажатия кнопки "Далее" откроется окно выбора профиля:



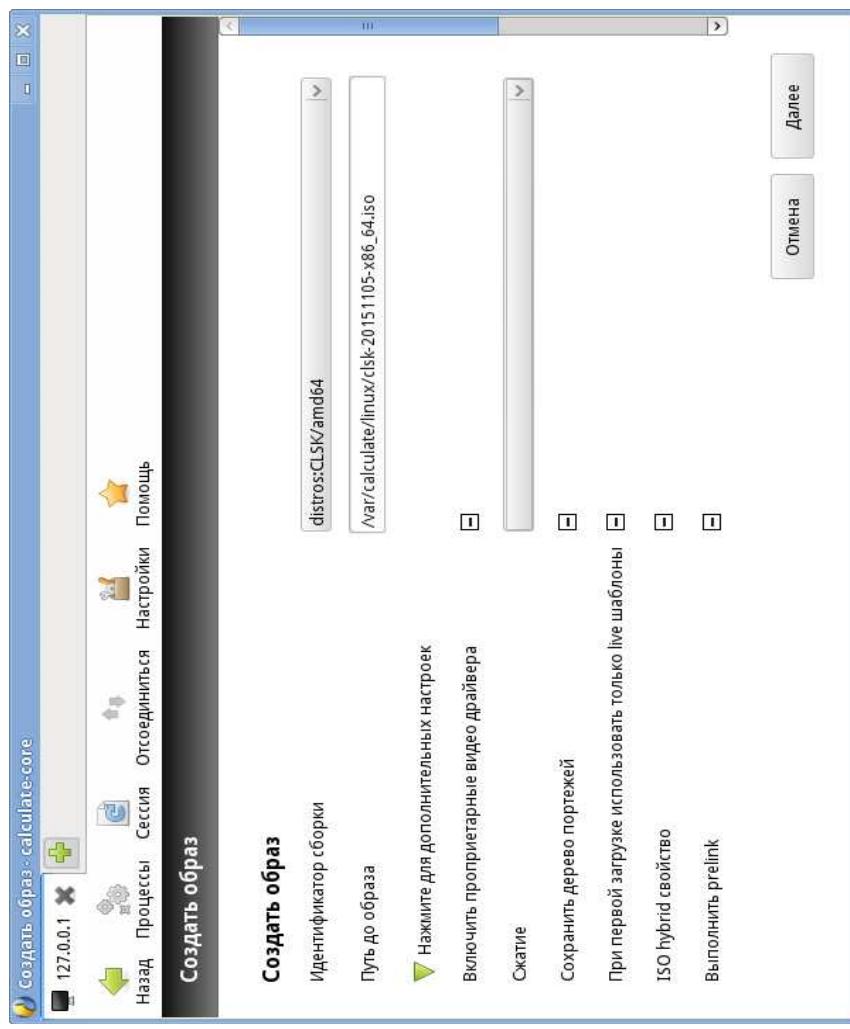
После выполнения операции все временные файлы будут удалены.

После завершения действия в chroot из оболочки необходимо выйти, в противном случае выполнение действий по обновлению системы, смене профиля или создания образа завершится сообщением об ошибке. Пример:

\* /bin/bash уже запущена внутри distros :CLSK/amd64

## Создание загрузочного образа

Для создания нового образа дистрибутива воспользуйтесь иконкой "Создать образ" в разделе "Сборка" графической консоли утилит Calculate. Страница создания образа будет выглядеть, как на рисунке ниже:



Большинство настроек будет по умолчанию скрыто от глаз. При желании вы можете указать, включить ли дистрибутивы proprietарных видеодрайверов в образ Live USB для того, чтобы иметь возможность однажды их работу, загрузившись с USB Flash. Вы можете выбрать метод скатия образа для того, чтобы найти оптимальный вариант между размером файла и скоростью скатия. Опция ISO hybrid позволит создавать образ диска, который можно писать на устройство при помощи прямого копирования (например, dd). Выполнение Prelink позволит синхронизовать библиотеки перед запаковкой для получения некоторого ускорения запуска программы.

После выполнения в директории /var/calculate/linux/ будет создан образ дистрибутива (с расширением .iso), файл с контрольными суммами (с расширением DIGESTS) и файл с составом программ (с расширением list).

В терминале запаковать образ можно, выполнив:

Изменить профиль сборки - Calculate Console  
■ 127.0.0.1 +  
Назад Процессы Сессия Отсоединяться Настройки Помощь

## Изменение загрузочного образа

Для изменения нового образа дистрибутива воспользуйтесь иконкой "Создать образ" в разделе "Сборка" графической консоли утилит Calculate. Страница создания образа будет выглядеть, как на рисунке ниже:



В списке доступных профилей отображаются все профили выбранного репозитория. Состав пакетов (файл /var/lib/portage/world) формируется шаблонами утилит Calculate. Вы можете выбрать один из режимов обновления файла: переформировать, объединить или обновить. По умолчанию используется полная замена списка пакетов дистрибутива.

При работе в терминале для смены профиля вы можете использовать следующую команду:

```
c1-builder -profile
```

### Основные параметры:

- **-id ID** - идентификатор сборки. Указывать данный параметр необходимо в случае одновременных сборок. Просмотреть список идентификаторов можно при помощи значения list.
- **-url URL** - репозиторий профиля. По умолчанию используется репозиторий, в котором находится текущий профиль системы.
- **-rebuild-world, --merge-world, -update-world** - действие с файлов world при переключении профиля; список заменяется списком из профиля, текущий список объединяется с новым или выполняется только обновление списка пакетов.

### Дополнительные параметры:

- **-u [ON/OFF], --update-cache [ON/OFF]** - обновить скачанный ранее репозиторий
- **-skip-setup-system [ON/OFF]** - пропустить перенастройку системы после переключения профиля. По умолчанию после переключения профиля в системе выполняется действие аналогичное c1 -setup -system, за исключением того, что в собираемой системе не будут выполнены шаблоны события 3, ас install live.

Переключение профиля в собираемой системе по сути аналогично переключению профиля в текущей системе при помощи команды c1 -update -profile.

## Обновление системы



### Обновить сборку

Идентификатор сборки

distros:CLSK/amd64

Нажмите для дополнительных настроек

Для обновления собираемой системы из терминала используйте следующую команду:

c1-builder-update

### Параметры:

- **-id ID** - идентификатор сборки. Указывать данный параметр необходимо в случае одновременных сборок.
- **-s [ON/OFF], --sync-only [ON/OFF]** - выполнить только синхронизацию репозиториев и обновление настроек
- **-o [ON/OFF], --update-other [ON/OFF]** - обновление вспомогательных репозиториев
- **-p [ON/OFF], --pretend [ON/OFF]** - вместо действительного обновления пакетов отобразить только, что будет установлено
- **-r REPOSITORIES, --repositories REPOSITORIES** - синхронизируемые репозитории. По умолчанию синхронизируются все репозитории дистрибутива, при чём если репозиторий бинарный, то синхронизация будет до необходимых ревизий, в противном случае до ветки master.
- **-e [ON/OFF], --emergelist [ON/OFF]** - отобразить список пакетов в формате emerge
- **-(rebuild|merge|update)-world** действие с файлами world: список заменщается базовым набором, текущий список объединяется с базовым или выполняются только обновление списка пакетов необходимости.
- **-(force|skip|auto)-egen cache** - обновить кэш репозиториев: принудительно, не обновлять, при необходимости.
- **-(force|skip|auto)-eix-update** - обновить кэш eix: принудительно, не обновлять, при необходимости.
- **-rebuild-changed-packages [ON/OFF]** - пересобрать пакеты, если класс файлы, используемые для сборки изменились

## -R [ON/OFF], --skip-revdep-rebuild [ON/OFF] - пропустить выполнение команды revdep-rebuild

- (опция включена по умолчанию)
- **--scan [ON/OFF]** - выполнить поиск наиболее актуального сервера бинарных обновлений
- **--clean-pkg [ON/OFF]** - удалять устаревшие архивы программ (очистка packages и distfiles от версий пакетов, которые отсутствуют в дереве portage)
- **--branch REFS** - переключить репозитории на указанные ветки или ревизии. Начиная с версии 3.4 репозитории недостаточно переключить на нужную ветку один раз. При последующем запуске без параметра **-b branch** утилиты попытается привести репозитории к ревизиям, указанным на выбранном сервере бинарных обновлений.

Функционал cl-builder-update повторяет функционал cl-upgrade за исключением того, что обновление выполняется внутри подготовляемой сборки. Список выполняемых действий:

Действия, обновляющие репозитории и настройки:

1. Настройка шаблонами **builder/prepare** (событие ac\_builder\_prepare), **builder/setup** (событие ac\_builder\_setup)
2. Синхронизация репозиториев (можно пропустить, указав **--t none**)
3. Синхронизация прочих оверлеев (можно пропустить, указав **--o off**)
4. Обновление кэша метаданных в репозиториях (можно пропустить, указав **--skip-egen-cache**)
5. Обновление кэшей связанных с репозиториями (можно пропустить, указав **--skip-eix-update**)
6. Удаление устаревших файлов из distfiles, packages (выполняется при **--clean-pkg**)
7. Исправление настроек в собираемой системе (шаблоны события ac\_update\_sync)
8. Выполнение dispatch-conf Действия, обновляющие пакеты в системе, выполняются, если не указана опция **-p**:

9. Выполнение emerge -DN --changed-deps --with-bdeps=y @world

10. Пересборка изменявшихся пакетов
11. Обновление Python пакетов
12. Обновление Perl пакетов
13. Выполнение emerge --depclean
14. Пересборка модулей ядра
15. Исправление настроек в собираемой системе (шаблоны события ac\_preserved-libs)
16. Пересборка @preserved-libs
17. Выполнение revdev-rebuild
18. Выполнение dispatch-conf
19. Исправление списка новостей
20. Отображение списка новостей

21. Проверка на устаревшие пакеты

Отдельной команды для проверки зависимостей нет, их можно проверить без обновления репозиториев при помощи cl-update -o off -t none -p.

Во время сборки будут загружены все необходимые пакеты с исходными текстами программ в директорию /var/calculate/remote/distfiles.

Помните, что вы всегда можете получить доступ к системе при помощи chroot, выполнив:  
chroot /run/calculate/mount/каталог\_сборки

Пример:

chroot /run/calculate/mount/distros\_CLDX\_amd64

Обратите внимание, что для доступа к собираемой 32-битной системе из-под 64-битной команду chroot следует выполнять, используя утилиту linux32. Пример:  
linux32 chroot /run/calculate/mount/distros\_CLDX\_i686

Для подключения к ftp откройте любой браузер, поддерживающий этот протокол, и введите в адрес страницы `ftp://вашСервер`.

## Учетные записи

Учетные записи используются для авторизованного доступа к FTP серверу. Для каждой из них может быть настроена домашняя папка, при использовании совместно с этим сервисом, разграничены права.

### Управление учетными записями

Добавление учетной записи

Для добавления учетной записи сервиса ftp используется команда `cl-useradd`:

- Создать учетную запись user1 и задать пароль

```
cl-useradd -p user1 ftp
```

- Создать учетную запись user1 с домашней директорией, задать пароль. Домашняя директория по умолчанию помещается в `pub/users/<имя_учетной_записи>` домашней директории сервиса ftp. К этой директории имеет доступ только этот пользователь.

```
cl-useradd -p -m user1 ftp
```

- Создать учетную запись user1 с указанной домашней директорией, задать пароль. Домашняя директория в этом случае будет находиться по указанному пути относительно корня домашней директории сервиса ftp.

```
cl-useradd -p -m -d pub/user1 user1 ftp
```

Смена пароля учетной записи

```
cl-passwd user1 ftp
```

Удаление учетной записи

Удаление учетной записи сервиса ftp производится командой `cl-userdel`, при этом удаляется пользовательская домашняя директория.

```
cl-userdel user1 ftp
```

P.S. Т.к. при создании учетной записи сервиса ftp, автоматически создается учетная запись сервиса unix, то при удалении учетной записи сервиса ftp, учетная запись unix остается.

### Управление правами доступа к директориям

Права доступа - это атрибуты файла или директории, которые указывают серверу, кто и что может делать с соответствующим файлом или директорией.

Файлы имеют двух владельцев: пользователя (user owner) и группу пользователей (group owner). Для каждого файла есть индивидуальные права доступа, которые разбиты на три группы: Доступ для пользователя-владельца файла (owner), Доступ для группы-владельца файла (group), Доступ для остальных пользователей (others).

Для каждой категории устанавливаются три вида доступа: (x) - право на запуск файла/право входа в директорию, (r) - право на чтение файла/директории, (w) - право на изменение (редактирование) файла, удаление/создание файлов в директории.

Входные данные метода `setup_locale`:

1. sid - идентификатор (номер) сессии;
2. info - объект типа `InstallInfo`, хранящий в себе все параметры, необходимые для установки и настройки системы.

Результатирующие данные метода `setup_locale` - список объектов типа `ReturnedMessage`, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **setup\_locale\_view** - генерирует объект `ViewInfo` с данными о параметрах настройки и передающий его клиенту;

Входные данные метода `setup_locale_view`:

1. sid - идентификатор (номер) сессии;
2. params - объект типа `ViewParams` (описание типа `ViewParams` см. выше в разделе "Метод, предоставляемый дополнительную информацию").

Результатирующие данные метода `setup_locale_view` - объект типа `ViewInfo`, несущий информацию о всех параметрах настройки локализации, допустимых значениях и способах отображения элементов (описание типа `ViewInfo` см. выше в разделе "Метод, предоставляемый дополнительную информацию").

- **setup\_locale\_vars** - вспомогательный метод, формирующий структуру объекта `ViewInfo` для настройки локализации;

• **Install** - основной класс установки и настройки системы.

Загрузка

Настройка загрузки, заключающаяся в изменении конфигурации `grub`, включает в себя:

- **setup\_boot** - генерирует входные параметры и запускающий отдельный процесс настройки загрузки;

Входные данные метода `setup_boot`:

1. sid - идентификатор (номер) сессии;
2. info - объект типа `InstallInfo`, хранящий в себе все параметры, необходимые для установки и настройки системы.

Результатирующие данные метода `setup_boot` - список объектов типа `ReturnedMessage`, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **setup\_boot\_view** - генерирует объект `ViewInfo` с данными о параметрах настройки и передающий его клиенту;

Входные данные метода `setup_boot_view`:

1. sid - идентификатор (номер) сессии;
2. params - объект типа `ViewParams` (описание типа `ViewParams` см. выше в разделе "Метод, предоставляемый дополнительную информацию").

Результатирующие данные метода `setup_boot_view` - объект типа `ViewInfo`, несущий информацию о всех параметрах настройки загрузки, допустимых значениях и способах отображения элементов (описание типа `ViewInfo` см. выше в разделе "Метод,

```
# file: \320\234\320\265\320\275\320\265\320\266\320\264\320\265\321\2000/
```

- **setup\_boot\_vars** - вспомогательный метод, формирующий структуру объекта ViewInfo для настройки загрузки;
  - **Install** - основной класс установки и настройки системы.
- Сеть**
- Настройка сети заключается в выборе менеджера сети, настройке сетевых интерфейсов, указании имени хоста, сервера времени, сервера доменных имён, доменов для поиска и таблицы маршрутизации.
- Настройка сети включает в себя:
- **setup\_network** - грс метод, проверяющий входные параметры и запускающий отдельный процесс настройки сети;

Входные данные метода **setup\_network**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа InstallInfo, хранящий в себе все параметры, необходимые для установки и настройки системы.

- **setup\_network\_view** - грс метод, создающий объект ViewInfo с данными о параметрах настройки и передающий его клиенту;

Входные данные метода **setup\_network\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующе данные метода **setup\_network\_view** - объект типа ViewInfo, несущий информацию о всех параметрах настройки сети, допустимых значениях и способах отображения элементов (описание типа ViewInfo см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **setup\_network\_vars** - вспомогательный метод, формирующий структуру объекта ViewInfo для настройки сети;

- **Install** - основной класс установки и настройки системы.

**Видео**

Настройка видео заключается в выборе видео драйвера, разрешения экрана, разрешения фреймбуфера и установке композита.

Настройка видео включает в себя:

Входные данные метода **setup\_video**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа InstallInfo, хранящий в себе все параметры, необходимые для установки и настройки системы.

Для просмотра прав из dolphin, нажмите на файле или директории правой кнопкой мыши, выберите вкладку Права, далее нажмите на кнопку Дополнительные права. Растворите открывшееся окно, чтобы увидеть все атрибуты.

## Управление доступом

Для управлением доступом используется команда **setfac1**.

Для модификации или добавления правил используется параметр -m.  
-m user : [пользователь] : права[, user : пользователь : права]  
-m group : [группа] : права[, group : группа : права]

Если пользователь пропущен, то права назначаются владельцу файла.  
Если группа пропущена, то права назначаются группе-владельцу файла.

### Пример

Добавить право на чтение/запись файла **secretinfo** пользователем ivanov и аретров:  
**setfac1 -m user:ivanov:rw, u:aretrov:rw secretinfo**

Добавить право на чтение/выполнение файла **runit** группе

## Настройка FTP сервера

FTP (англ. File Transfer Protocol --- протокол передачи файлов) --- протокол, предназначенный для передачи файлов в компьютерных сетях. FTP позволяет подключаться к серверам FTP, просматривать содержимое каталогов и загружать файлы с сервера или на сервер.

Настройка FTP сервера производится в несколько этапов:

### Установка FTP сервиса в систему

Настройка сервиса ftp требует, чтобы в системе были установлены сервисы ldap и unix, поэтому если они еще не были установлены устанавливаем их командами:

```
c1-setup ldap  
c1-setup unix
```

Установка ftp сервиса производится командой:

```
c1-setup ftp
```

После чего ftp становится доступен для пользователя **ftp**. На сервер также автоматически в ftp директории создаются папки **tmp** и **priv**.

Просмотр информации о пользователе samba сервиса:

с1-**info -U <название пользователя>** samba  
насущих информации о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **cl-info -g\_unix**  
Перечень существующих групп unix сервиса:

```
cl-info -g samba
```

Прочтение существующих групп samba сервиса:

```
cl-info -G <название группы> unix
```

Прочтение информации о группе unix сервиса:

```
cl-info -G <название группы> samba
```

Прочтение информации о группе samba сервиса:

```
cl-info -G <название группы> samba
```

## Настройка прав доступа ACL

Для создания прав доступа к файлам Samba ресурса сервера используйте **ACL (Access Control List)** --- список контроля доступа). При помощи ACL вы можете установить доступ к файлам определенным группам, либо конечным пользователям.

Стандартные команды работы с ACL - **setfacl** и **getfacl** - подробно описаны в руководстве, поэтому здесь мы ограничимся примерами.

После **настройки Samba сервера**, создаётся путь для общей точки монтирования **/var/calculate/server-data/samba/share**.

Внимание. Убедитесь, что файловая система диска, содержащего указанную директорию, поддерживает ACL. Не забудьте в опции монтирования файловой системы указать параметр "acl" (для  **nfs** не требуется).

### Новый ресурс

Для примера допустим у вас в системе есть две группы **manager** и **logist**, а также два пользователя **iivanov** и **arcticov**.

Создадим директорию **Менеджер** для доступа на чтение и записи пользователем, входящим в группу **manager**.

```
mkdir -m 700 Менеджер  
setfacl -m d:g:manager:rwx,g:manager:rwx Менеджер
```

Первая команда создаст директорию с правами чтения и изменения содержимого для пользователя root. Второй командой мы установим те же права для пользователей группы manager.

Файлы и директории, создаваемые пользователями этой группы, либо rootом, будут наследовать атрибуты доступа.

## Просмотр прав доступа

Права на доступ к директории, вы можете посмотреть из консоли, либо через файловый менеджер dolphin, с **Calculate Linux Desktop** (KDE версия).  
Увидеть права доступа из консоли можно выполнив команду:

```
getfacl Менеджер
```

Программа выведет следующий текст:

Результатирующющие данные метода **setup\_video** - список объектов типа **ReturnedMessage**, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **setup\_video\_view** - грс метод, создающий объект **ViewInfo** с данными о параметрах входные данные метода **setup\_video\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа **ViewParams** (описание типа **ViewParams** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Входные данные метода **setup\_video\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа **ViewParams** (описание типа **ViewParams** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **setup\_video\_vars** - вспомогательный метод, формирующий структуру объекта **ViewInfo** для настройки видео;

1. sid - идентификатор (номер) сессии;
2. params - объект типа **ViewParams** (описание типа **ViewParams** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Входные данные метода **setup\_video\_vars**:

Результатирующющие данные метода **setup\_video** - объект типа **ViewInfo**, несущий информацию о всех параметрах настройки видео, допустимых значениях и способах отображения элементов (описание типа **ViewInfo** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **setup\_system** - грс метод, проверяющий входные параметры и запускающий отдельный процесс настройки системы;

Входные данные метода **setup\_system**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа **InstallInfo**, хранящий в себе все параметры, необходимые для установки и настройки системы.

Результатирующющие данные метода **setup\_system** - список объектов типа **ReturnedMessage**, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **setup\_system\_view** - грс метод, создающий объект **ViewInfo** с данными о параметрах входные данные метода **setup\_system\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа **ViewParams** (описание типа **ViewParams** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Входные данные метода **setup\_system\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа **ViewParams** (описание типа **ViewParams** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующющие данные метода **setup\_system** - объект типа **ViewInfo**, несущий информацию о всех параметрах настройки системы, допустимых значениях и способах отображения элементов (описание типа **ViewInfo** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **setup\_system\_vars** - вспомогательный метод, формирующий структуру объекта **ViewInfo** для настройки системы;

1. sid - идентификатор (номер) сессии;
2. params - объект типа **ViewParams** (описание типа **ViewParams** см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **Install** - основной класс установки и настройки системы.

#### Настройка пакета

Настройка пакета заключается в выборе пакета и наложении шаблонов для него.

Настройка пакета включает в себя:

- **core\_setup** - прс метод, проверяющий входные параметры и запускающий отдельный процесс настройки пакета;

Входные данные метода **setup\_video**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа CoreSetupInfo, хранящий в себе все параметры, необходимые для настройки пакета (класс CoreSetupInfo объявлен в файле core/server/setup\_package.py пакета calculate-core).

Результатирующе данные метода **setup\_video** - список объектов типа RetunedMessage, несущих информацию о запуске процесса (идентификатор процесса rid) или сообщения с ошибками.

- **core\_setup\_view** - прс метод, создающий объект ViewInfo с данными о параметрах настройки и передающий его клиенту;

Входные данные метода **setup\_video\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующе данные метода **setup\_video\_view** - объект типа ViewInfo, несущий информацию о всех параметрах настройки пакета, допустимых значениях и способах отображения элементов (описание типа ViewInfo см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **core\_setup\_vars** - вспомогательный метод, формирующий структуру объекта ViewInfo для настройки пакета;

- **updateSystemConfigs** - основной класс настройки пакета.

#### Группы созданные по умолчанию

##### Доменные группы Доменные группы - глобальные группы, которые действуют в домене.

- *Domain Admins* - администраторы домена (полные права на компьютерах в домене).
- *Domain Guests* - гости домена (минимальные права).
- *Domain Users* - пользователи домена.
- *Domain Computers* - компьютеры домена.

##### Локальные группы

Локальные группы - группы действующие локально на данном компьютере.

Локальные группы отсутствуют.

##### Встроенные группы

Встроенные группы - группы встроенные в систему.

- *Administrators* - администраторы (полные права)
- *Account Operators* - операторы учетных записей. Создание и управление пользовательской учетной информацией, создание и управление группами, резервное копирование файлов и каталогов.
- *Backup Operators* - операторы архивов. Резервное копирование, восстановление из резервной копии, остановка системы.
- *Print Operators* - операторы печати. Управление принтерами, резервное копирование.
- *Replicators* - репликаторы. Эта группа используется службой репликации File Replication на контроллерах домена.
- *System Operators* - операторы системы. Изменение системного времени, останов системы, останов с удаленной системы, резервное копирование, восстановление из резервной копии, блокирование сервера, преодоление блокировок сервера, форматирование жесткого диска, управление сетевыми каталогами, управление принтерами.

##### Создание Samba группы

Создание доменной группы test. По умолчанию создается доменная группа, тип группы - 2.  
**cl-groupadd test samba**

Создание встроенной группы 'Power Users' - пользователя имеющие дополнительные права.  
**cl-groupadd -g 547 -rid 547 -t 5 'Power Users' samba**

Где:

- *g* - идентификатор группы 547 (Group ID)
- *rid* - уникальный идентификатор 547 (RID)
- *t* - тип группы 5 (встроенная группа)

#### Просмотр информации

Для просмотра информации о пользователях и группах сервера воспользуйтесь командой **cl-info**:

Перечень пользователей unix сервиса:  
**cl-info -u unix**

Перечень информации о пользователях и группах сервера воспользуйтесь командой **cl-accounts**:

Просмотр информации о пользователе unix сервиса:  
**cl-info -U <название пользователя> unix**

Запросов на сервере утилит,

Входные данные метода **core\_show\_request**:

## Добавление и удаление пользователей

Для работы с пользователями, используйте аналоги стандартных Unix команд: `cl-useradd`, `cl-userdel`, `cl-usermod`, `cl-passwd`, `cl-groupadd`, `cl-groupdel`, `cl-groupmod`. Синтаксис этих команд будет во многом совпадать с оригинальными.

Вместо `smbpasswd`, используйте `cl-passwd` для изменения паролей пользователей, в т.ч. администратора Windows компьютеров.

Пример добавления пользователя `test`:

```
cl-useradd test samba
```

Пример изменения пароля пользователя `test`:

```
cl-passwd test samba
```

Пример добавления пользователя с первичной группой Domain Admins - администраторы домена

```
cl-useradd -g 'Domain Admins' test samba
```

Обратите внимание на опцию `samba`, добавляемую в конце команды.

## Настройка прав доступа к файловой системе

Для настройки прав доступа к файлам на сервере используйте ACL (Access Control List --- список контроля доступа). Изменяя права на файлы, вы ограничиваете к ним доступ в равной степени как для Windows, так и для Linux клиентов.

Права доступа применяются как к файлам, так и к директориям. Вы можете указать права доступа на владельца файла, либо на группу. Если Windows клиент будет распознавать только Samba группы, то в Linux будут отображаться имена Unix и Samba групп. Поэтому, для разграничения прав доступа, предпочтительней использовать Samba группы.

Для создания Samba группы "manager", выполните:

```
cl-groupadd job unix
```

для создания Unix группы "job", выполните:

```
cl-useradd job
```

Подробно настройка прав доступа с использованием ACL описана в статье "[Настройка прав доступа к файловой системе](#)".

## Настройка прав доступа для пользователей Windows компьютеров

Настройка прав доступа к общим файлам сервера описана выше.

Для настройки дополнительных прав Windows машин, таких как: возможность устанавливать программы, возможность выхода из домена и т.д., используйте Samba группы.

Пример повышения прав пользователя `test` до администратора домена:

```
cl-groupmod -a test 'Domain Admins' samba
```

Структура Samba групп

Samba группы могут быть следующих типов:

- Доменные группы (номер типа группы 2)
- Локальные группы (номер типа группы 4)
- Встроенные группы (номер типа группы 5)

## 1. sid - идентификатор (номер) сессии;

2. info - объект типа `RequestInfo`, хранящий в себе параметры выдаваемой таблицы запросов (количество строк и смещение). Класс `RequestInfo` объявлен в файле `core/server/request.py` пакета `calculate-core`.

Результатирующе данные метода `core_show_request` - список объектов типа `RetumedMessage`, несущих информацию о запуске процесса (идентификатор процесса `pid`) или сообщения с ошибками.

- **core\_show\_request\_view** - грс метод, создающий объект `ViewInfo` с данными, необходимыми для формирования таблицы запросов, и передающий его клиенту;

Входные данные метода `core_show_request_view`:

1. sid - идентификатор (номер) сессии;
2. params - объект типа `ViewParams` (описание типа `ViewParams` см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующе данные метода `core_show_request_view` - объект типа `ViewInfo`, несущий информацию о параметрах таблицы запросов (описание типа `ViewInfo` см. выше в разделе "Метод, предоставляющий дополнительную информацию").

## Настройка прав доступа к файлу

- **show\_request\_meth** - грс метод, формирующий таблицу запросов на сервере утилит;

• **requestCommon** - метод, выполняющий дополнительные проверки и запускающий процесс формирования таблицы запросов на сервере утилит.

**Подробный просмотр запросов** реализован для графических клиентов и осуществляет взаимодействие метода просмотра запросов с методами подтверждения и удаления запросов на сервере утилит, включает в себя:

- **core\_detail\_request** - грс метод, проверяющий входные параметры (номер запроса). Не вызывает никаких других методов (является заглушкой);

Входные данные метода `core_detail_request`:

1. sid - идентификатор (номер) сессии;
2. info - объект типа `DetailRequestInfo`, хранящий в себе параметры запроса (номер и имя группы для подписания сертификата). Класс `DetailRequestInfo` объявлен в файле `core/server/request.py` пакета `calculate-core`.

Результатирующе данные метода `core_detail_request` - список объектов типа `RetumedMessage`, несущих информацию о запуске процесса (идентификатор процесса `pid`) или сообщения с ошибками.

- **core\_detail\_request\_view** - грс метод, создающий объект `ViewInfo` с данными запроса.

Содержит данные о запросе и кнопки вызова других методов (просмотра, подписания и удаления запросов, с передачей необходимых параметров);

Входные данные метода `core_detail_request_view`:

1. sid - идентификатор (номер) сессии;
2. params - объект типа `ViewParams` (описание типа `ViewParams` см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующе данные метода `core_detail_request` - объект типа `ViewInfo`, несущий информацию о всех параметрах запроса, допустимых значениях и способах отображения элементов, а так же кнопках перехода к другим методам и передачи им параметров (описание

типа ViewInfo см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **core\_confirm\_request** - гrpc метод, проверяющий входные параметры и вызывающий вспомогательный метод confirmRequestCommon, который запускает процесс подписания запроса на сертификат;

Входные данные метода **core\_confirm\_request**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа DetailRequestInfo, хранящий в себе номер запроса и имя группы для подписания сертификата.

Результатирующющие данные метода **core\_confirm\_request** - список объектов типа RetunedMessage, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **core\_confirm\_request\_view** - гrpc метод, создающий объект ViewInfo с данными, необходимыми для формирования таблицы запросов, и передающий его клиенту;

Входные данные метода **core\_confirm\_request\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующющие данные метода **core\_confirm\_request\_view** - объект типа ViewInfo, несущий информацию о параметрах запроса (номер запроса и имя группы для подписания), допустимых значениях и способах отображения элементов (описание типа ViewInfo см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **config\_request\_meth** - метод, выполняющий подписание сертификата;

- **configRequestCommon** - метод, выполняющий дополнительные проверки и запускающий вспомогательный метод delRequestCommon, который удаляет все данные о запросе с сервера процесса под подписания сертификата (config\_request\_meth).

Удаление запроса включает в себя:

- **core\_del\_request** - гrpc метод, проверяющий входные параметры и вызывающий вспомогательный метод delRequestCommon, который удаляет все данные о запросе с сервера утилит;

Входные данные метода **core\_del\_request**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа DetailRequestInfo, хранящий в себе номер запроса для его удаления.

Результатирующющие данные метода **core\_del\_request** - список объектов типа RetunedMessage, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

- **core\_del\_request\_view** - гrpc метод, создающий объект ViewInfo с данными, необходимыми для удаления запроса, и передающий его клиенту;

Входные данные метода **core\_del\_request\_view**:

1. sid - идентификатор (номер) сессии;

## Настройка Samba сервера

### Введение

Samba это популярный пакет программ с открытыми исходными текстами, которая предоставляет файловые и принт-сервисы Microsoft(r) Windows(r) клиентам. При помощи пакета утилит [calculate-server](#) на Samba сервере создаются пользователи, группы, ресурсы и происходят управление ими. В качестве Linux клиента может выступать [Calculate Linux Desktop](#) или любой Gentoo дистрибутив с установленным пакетом [calculate-client](#). В качестве Windows клиентов могут быть использованы операционные системы семейства Windows. Чтобы настроить сервер и клиент, воспользуйтесь статьей "[переход на использование Linux](#)". Для указания настраиваемого пакета программ в *calculate-server* используется термин - сервис.

Сервис *samba* используется для настройки *Samba*.

Сервис обязательно указывается для программ пакета *calculate-server*.

Пример добавления пользователя *test*:

```
cl-useradd test samba
```

Для пользователя сервера права на файловые ресурсы одинаковы вне зависимости от клиента - Linux или Windows.

Пакет Samba включен в поставку [Calculate Directory Server](#). Если вы используете другую Gentoo систему, пакет можно установить из портфеля, выполнив *emerge net-fs/samba*.

### Настройка сервера

Настройка сервера выполняется при помощи утилиты *calculate-server*. Для начала убедитесь что у вас [настроен LDAP сервер](#) и выполнены [настройки Unix сервера](#).

Для настройки Samba-сервера выполните команду:

```
cl -s setup [параметры] samba
```

В качестве параметров вы можете указать *netbios* и *workgroup*.

- "-n name" - устанавливает имя NetBIOS, под которым будет работать Samba сервер. По умолчанию оно устанавливается равным первому компоненту DNS имени хоста.
- "-w workgroup" - имя домена или рабочей группы NT для компьютеров, которые будут получать доступ к этому серверу.

Если Samba сервер будет выступать в качестве PDC (первичного контроллера Windows домена), вам следует задать пароль администратора сервера - пользователя с логином *admin*.

```
cl-passwd -s smb admin Samba
```

Пользователь *admin* используется только для ввода клиентского windows компьютера в домен. Он не имеет домашней директории.

Если нужен администратор домена для управления windows компьютерами, добавьте нового пользователя который будет включен в доменную группу "Domain Admins", или включите в эту группу существующего пользователя.

Пример создания администратора домена:

```
cl-useradd -p -g id "Domain Admins" -c "Администратор домена" d_admin  
samba
```

Для [подключения Unix клиентов](#), укажите пароль для служебного пользователя *client*.

```
cl-passwd -s smb client samba
```

## c1-setup ldap

2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Во время выполнения этой команды, программа выполнит настройку сервера LDAP, запустит его и пропишет в автозагрузку. Будьте внимательны, программа перезапишет базу данных LDAP. Если вы работаете с LDAP сервером до этого, выполните резервное копирование ваших данных. Пароли доступа к LDAP всех сервисов хранятся в */etc/calculate/calculate ldap*.

## Использование LDAP сервера для хранения учетных записей

### Введение

В Unix системе вся информация о пользователях хранится в файле */etc/passwd*. Это текстовый файл, право на чтение которого имеют все пользователи, а на запись только суперпользователь. Пароли пользователей хранятся в файле */etc/shadow* в зашифрованном виде, закрытом для чтения и записи.

Информация о группах хранится в файле */etc/group*.

У этого способа хранения есть ряд ограничений, из которых можно выделить сложность переноса пользователей, ограниченность в атрибутах, производительность и т.п. Однако при помощи *PAM (Pluggable Authentication Module)*, вы можете с легкостью интегрировать в системы UNIX различные технологии аутентификации, в т.ч. *LDAP (Lightweight directory Access Protocol)*.

### Настройка Unix сервера

Выполнить настройки сервера вы можете воспользовавшись программой *c1-setup*, входящей в пакет *calculate-server*. Для этого выполните с правами суперпользователя:

```
c1-setup unix
```

После выполнения этой операции, имена учетных записей пользователей, хранящиеся в базе *LDAP* сервера и имеющие системные *ID*, будут видны в системе.

### Добавление и удаление пользователей

Обратите внимание, что после настройки LDAP сервера в качестве хранилища учётных записей Unix пользователей, следует использовать альтернативные команды по управлению пользователями.

Вместо команд: *useradd (adduser)*, *userdel*, *usermod*, *passwd*, *groupadd*, *groupmod*, следует использовать альтернативные: *c1-useradd*, *c1-userdel*, *c1-passwd*, *c1-groupadd*, *c1-groupmod*. Синтаксис этих команд будет во многом совпадать с оригинальными.

Пример добавления пользователя *test*:

```
c1-useradd test unix
```

Пример изменения пароля пользователя *test*:

```
c1-passwd test unix
```

Обратите внимание на опцию *unix*, добавляемую в конце команды.

- **del\_request\_meth** - метод, выполняющий удаление запроса;
- **delRequestCommon** - метод, выполняющий дополнительные проверки и запускающий процесс удаления запроса (*del\_request\_meth*).

### Управление группами

Группа методов управления группами состоит из методов просмотра, подробного просмотра, добавления, изменения и удаления групп прав сертификатов.

- **certificateCommon** - метод, выполняющий дополнительные проверки и запускающий процесс просмотра данных о сертификате (*view\_cert\_meth*).

Результатирующущие данные метода **core\_del\_request\_view** - объект типа *ViewInfo*, несущий информацию о параметрах запроса (номер запроса), допустимых значениях и способах отображения элементов (описание типа *ViewParams* см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующущие данные метода **core\_view\_cert\_view** - объект типа *ViewInfo* с данными, необходимыми выбора сертификата для просмотра информации о нём, и передающей его клиенту:

Входные данные метода **core\_view\_cert**:

- 1. sid - идентификатор (номер) сессии;
- 2. info - объект типа *CertificateInfo*, хранящий в себе номер сертификата.

Результатирующущие данные метода **core\_view\_cert\_cert** - список объектов типа *ReturnedMessage*, несущих информацию о запуске процесса (идентификатор процесса *pid*) или сообщения с ошибками.

Результатирующущие данные метода **core\_view\_cert\_view** - группы прав сертификата, данных о подписанчке и субъекте, серийного номера и отпечатка сертификата:

Входные данные метода **core\_view\_cert\_view**:

- 1. sid - идентификатор (номер) сессии;
- 2. params - объект типа *ViewParams* (описание типа *ViewParams* см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующущие данные метода **core\_view\_cert\_view** - объект типа *ViewInfo*, несущий информацию о параметрах (номера сертификатов), допустимых значениях и способах отображения элементов (описание типа *ViewInfo* см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **view\_cert\_meth** - метод, выполняющий сбор информации о сертификате (доступных для выполнения методов, группы прав сертификата, данных о подписанчке и субъекте, серийного номера и отпечатка сертификата);
- **certificateCommon** - метод, выполняющий дополнительные проверки и запускающий процесс просмотра данных о сертификате (*view\_cert\_meth*).

- **core\_show\_groups** - грс метод, проверяющий входные параметры и вызывающий вспомогательный метод groupCommon, который запускает процесс формирования таблицы групп, существующих на сервере утилит;

Входные данные метода **core\_show\_groups**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа GroupInfo, хранящий в себе параметры выдаваемой таблицы запросов (количество строк и смещение). Класс GroupInfo объявлен в файле core/server/edit\_groups.py пакета calculate-core.

Результатирующе данные метода **core\_show\_groups** - список объектов типа RetunedMessage, несущих информацию о запуске процесса (идентификатор процесса rid) или сообщения с ошибками.

- **core\_show\_groups\_view** - грс метод, создающий объект ViewInfo с данными, необходимыми для формирования таблицы групп, и передающий этот объект клиенту;

Входные данные метода **core\_show\_groups\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

Результатирующе данные метода **core\_show\_groups\_view** - объект типа ViewInfo, несущий информацию о параметрах таблицы групп (описание типа ViewInfo см. выше в разделе "Метод, предоставляющий дополнительную информацию").

- **show\_groups\_meth** - метод, формирующий таблицу групп на сервере утилит;

• **groupCommon** - метод, выполняющий дополнительные проверки и запускающий процесс формирования таблицы групп на сервере утилит (запуск метода show\_groups\_meth).

**Подробный просмотр группы** реализован для графических клиентов и осуществляет взаимодействие метода просмотра групп с методами изменения и удаления групп на сервере утилит, включает в себя:

- **core\_detail\_group** - грс метод, проверяющий входные параметры (имя группы). Не вызывает никаких других методов (является заглушкой);

Входные данные метода **core\_detail\_group**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа DetailGroupInfo, хранящий в себе параметры группы (имя группы и список прав). Класс DetailGroupInfo объявлен в файле core/server/edit\_groups.py пакета calculate-core.

Результатирующе данные метода **core\_detail\_group** - список объектов типа RetunedMessage, несущих информацию о запуске процесса (идентификатор процесса rid) или сообщения с ошибками.

- **core\_detail\_group\_view** - грс метод, создающий объект ViewInfo с данными группы.

Содержит данные о группе и кнопки вызова других методов (просмотр, изменения и удаления группы, с передачей необходимых параметров);

Входные данные метода **core\_detail\_group\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляющий дополнительную информацию").

## Добавим группы пользователей

Создав группы, пользователи автоматически будут размещены в них в вашем Jabber клиенте, что, безусловно, облегчит использование, а также будет служить отправной точкой знакомства со структурой компании для новых сотрудников.

```
c1-groupadd "Директор" jabber
c1-groupadd "Кадры" jabber
c1-groupadd "Программист" jabber
c1-groupadd "Логист" jabber
c1-groupadd "Менеджер" jabber
```

## Добавим пользователей

Ниже приведен простой bash скрипт для добавления пользователей. Отредактируйте его, закомментировав неиспользуемые сервисы. Обратите внимание, что используемые в скрипте группы manager, audio, lp, plugdev, scanner и video должны быть предварительно созданы как описано выше.

```
#!/bin/bash

login="ipetrov"
fullname="Петров Иван"
group="manager"
groupJabber="Менеджер"
mail="ip@calculate-linux.org"

echo ">>>Настроим Samba"
/usr/bin/cl-useradd -p -c "$fullname" -g $group -G
audio,lp,plugdev,scanner,video $login samba

echo ">>>Настроим Mail"
/usr/bin/cl-useradd -c "$fullname" -p -e $mail $login mail

echo ">>>Настроим Jabber"
/usr/bin/cl-useradd -c "$fullname" -p $login jabber
/usr/bin/cl-usermod -g "$groupJabber" $login jabber
```

В приведенном примере мы создаем два почтовых адреса, т.к. сервер находится в локальной сети и не является основным почтовым сервером.

## Настройка LDAP сервера

**Введение**  
LDAP (англ. Lightweight Directory Access Protocol --- «облегчённый протокол доступа к каталогам») --- это сетевой протокол для доступа к службе каталогов. LDAP --- относительно простой протокол, использующий TCP/IP и позволяющий производить операции аутентификации (bind), поиска (search) и сравнения (compare), а также операции добавления, изменения или удаления записей. Обычно LDAP-сервер принимает входящие соединения на порт 389 по протоколам TCP или UDP. Для LDAP-сессий, инкапсулированных в SSL, обычно используется порт 636.

## Настройка

В качестве LDAP сервера в Calculate Directory Server используется [OpenLDAP](#). Настройка сервера выполняется пакетом *calculate-server* из состава утилит [Calculate 2](#).

Для настройки сервера, воспользуйтесь программой *cl-setup*, выполнив:

## Зададим пароль для входа в домен Linux- и Windows-клиентов

```
cl-passwd --smb client samba  
cl-passwd --smb admin samba
```

## Настроим использование distfiles клиентами домена

Создайте каталог /var/calculate/remote/distfiles если его нет  
`mkdir -p /var/calculate/remote/distfiles`

затем выполните команды

```
cl-groupadd --gid 250 -f portage unix  
cl-usermod -a portage client unix  
chmod 0775 /var/calculate/remote  
chmod -R 2775 /var/calculate/remote/distfiles  
chown -R root:portage /var/calculate/remote/distfiles
```

## Настройка Samba

### Создадим базовые системные группы

Базовые системные группы могут понадобиться в том случае, если вы захотите ограничить доступ пользователей к определенным ресурсам компьютера. В этом случае системные Unix-группы должны быть продублированы в LDAP-сервере. При необходимости эту операцию можно выполнить позднее.

```
cl-groupadd -f -g 7 lp unix  
cl-groupadd -f -g 10 wheel unix  
cl-groupadd -f -g 18 audio unix  
cl-groupadd -f -g 27 video unix  
cl-groupadd -f -g 35 games unix  
cl-groupadd -f -g 440 plugdev unix  
cl-groupadd -f -g 441 scanner unix  
cl-groupadd -f -g 442 vboxusers unix
```

Более подробно о назначение каждой группы вы можете прочитать [здесь](#).

### Добавим группы пользователей

Как минимум у пользователей должна быть одна группа. Тем не менее, если вы впоследствии захотите разграничить доступ к данным пользователям различных отделов, группы - наилучший способ организовать распределенный доступ.

В качестве примера взяты 5 групп:

```
cl-groupadd boss samba  
cl-groupadd job samba  
cl-groupadd it samba  
cl-groupadd logist samba  
cl-groupadd manager samba
```

## Настройка Jabber

Вы можете организовать свой Jabber-сервер. В свете недавних событий с ограничением использования ICQ сторонними клиентами это не плохой способ избавиться от зависимости от одной компании. Jabber-сервер может с успехом использоватьсь и внутри вашей локальной сети для эффективного взаимодействия сотрудников. С целью предотвращения злоупотреблений рабочим временем на сервере предусмотрено логирование сообщений.

Результатирующие данные метода **core\_detail\_group\_view** - объект типа ViewInfo, несущий информацию о параметрах группы, прав, допустимых значениях и способах отображения элементов, а так же кнопках перехода к другим методам и передачи им параметров (описание типа ViewInfo см. выше в разделе "Метод, предоставляемый дополнительную информацию").

**Добавление группы** создает новую группу прав на сервере утилит (для графического клиента - кнопка плюс "+" над таблицей групп). Включает в себя:

- **core\_add\_group** - грс метод, проверяющий входные параметры и вызывающий вспомогательный метод addGroupCommon;

Входные данные метода **core\_add\_group**:

1. sid - идентификатор (номер) сессии;
2. info - объект типа AddGroupInfo, хранящий в себе имя новой группы и список прав для неё.

Результатирующие данные метода **core\_add\_group** - список объектов типа ReturnValueMessage, несущих информацию о запуске процесса (идентификатор процесса pid) или сообщения с ошибками.

• **core\_group\_view** - грс метод, создающий объект ViewInfo с данными, необходимыми для добавления группы, и передающий его клиенту;

Входные данные метода **core\_group\_view**:

1. sid - идентификатор (номер) сессии;
2. params - объект типа ViewParams (описание типа ViewParams см. выше в разделе "Метод, предоставляемый дополнительную информацию").

Результатирующие данные метода **core\_group\_view** - объект типа ViewInfo, несущий информацию о параметрах новой группы (имя и список прав), допустимых значениях и способах отображения элементов (описание типа ViewInfo см. выше в разделе "Метод, предоставляемый дополнительную информацию").

- **add\_group\_meth** - метод, выполняющий добавление новой группы;
- **addGroupCommon** - метод, выполняющий дополнительные проверки и запускающий процесс добавления группы (add\_group\_meth).

## Вспомогательные модули

Вспомогательные модули служат для обеспечения взаимодействия клиента и сервера утилит. По выполняемым функциям методы этих модулей можно разделить на пять типов:

- работа с сертификатами, запросами и списками отзывов сертификатов;
- работа с сессиями;
- работа с процессами;
- работа с кэшем сессий, методов и процессов;
- прочие методы.

## Методы работы с сертификатами

К вспомогательным методам работы с сертификатами относятся:

- **post\_client\_request** - отправить клиентский запрос на подпись сертификата. Вызов этого метода передает на сервер утилит от клиента запрос на подпись сертификата.

Имеет четыре входных параметра:

- `request` - запрос на подпись сертификата (в виде строки с содержимым запроса);
- `ip` - IP-адрес (unikальный сетевой адрес узла в компьютерной сети);
- `mac` - MAC-адрес (unikальный идентификатор, присваиваемый каждой единице оборудования компьютерных сетей);
- `client_type` - тип клиента (console или gui).

Возвращает номер запроса или значение "-1" в случае ошибки.

- `post_server_request` - отправить серверный запрос на подпись сертификата, Вызов этого метода передаёт на сервер утилит запрос на подпись сертификата от другого сервера утилит.

Имеет три входных параметра:

- `request` - запрос на подпись сертификата (в виде строки с содержимым запроса);
- `ip` - IP-адрес;
- `mac` - MAC-адрес;

Возвращает номер запроса или значение "-1" в случае ошибки.

- `get_client_cert` - забрать подписанный сертификат клиента с сервера утилит.

Имеет два входных параметра:

- `req_id` - номер запроса (unikальный номер запроса на сервере утилит, выдаётся при вызове метода `post_client_request`).
- `request` - запрос на подпись сертификата. Необходим для проверки с ранее сохранённым.

Возвращает значение "1", если запрос был отвергнут или не найден на сервере, значение "2", если сертификата не существует (запрос ещё не был подписан). В случае подписания возвращает список из двух сертификатов - сертификат клиента и корневой сертификат сервера.

- `get_server_cert` - забрать подписанный сертификат сервера утилит.

Имеет аналогичные с методом `get_client_cert` входные и возвращаемые параметры (см. выше).

- `get_crl` - получить список отзыва сертификатов.

Возвращает содержимое списка отзыва сертификатов или значение "".

- `post_cert` - проверка сертификата на сервере. Вызывается при установке соединения для проверки наличия сертификата клиента на сервере и проверки сертификата на срок годности. Используется для графических клиентов, для консольных используется метод `int_session`. Не имеет входных параметров. Возвращает список целочисленных значений: номер сертифика и значение годности сертификата (принимает значения: "-2" - срок годности истёк, "-1" - срок годности более 2-х месяцев или количество дней до истечения срока годности). Если у клиента нет сертификата возвращает значение [-3], если сертификата клиента не найден на сервере - значение [-4].

## 5. Настройка сервера

1. Перенос учётных записей пользователей в Calculate Directory Server
2. Настройка LDAP сервера
3. Использование LDAP сервера для хранения учётных записей
4. Настройка Samba сервера
5. Настройка прав доступа ACL
6. Настройка FTP сервера
7. Настройка Jabber сервера
8. Настройка почтового сервера
9. Настройка Proxy сервера
- 10.Настройка DNS сервера
- 11.Настройка DHCP сервера
- 12.Настройка PXE
- 13.Настройка репликации Samba серверов
- 14.Настройка репликации почтовых серверов
- 15.Настройка сервера шлюза
- 16.Настройка Asterisk сервера
- 17.Обзор структуры LDAP сервера
- 18.Управление клиентскими машинами
- 19.Система виртуализации QEMU
- 20.Настройка gitolite
- 21.Резервное копирование

## Перенос учётных записей пользователей в Calculate Directory Server

Рассмотрим более подробно все шаги по переносу пользователей в Calculate Directory Server.

### Настройка серверов

Перед настройкой обновите пакет calculate-server до последней версии:  
`emerge calculate-server`

### Настройка сервисов

Перед настройкой сервисов проверьте, что у вас поднята сеть. Для настройки проводной сети воспользуйтесь следующим [руководством](#).

В приведённом примере мы настроим и запустим все поддерживаемые сервисы:

```
c1-setup ldap
c1-setup unix
c1-setup samba
c1-setup mail
c1-setup jabber
```

Обратите внимание: при настройке вы можете указывать дополнительные параметры. Перед запуском вызовите справку по каждой команде:

```
c1-setup --help
c1-setup -help_jabber
```

Метод `clear_method_cache` имеет два входных параметра:

- `sid` - номер сессии.
- `method_name` - имя метода.

Возвращает 1 в случае ошибки и 0 при успешной очистке кэша метода.

- `clear_pid_cache` - очищает кэш процесса. После завершения работы процесс хранит все данные о своей работе, чтобы пользователь мог просмотреть результаты.
- Метод `clear_pid_cache` имеет два входных параметра:
  - `sid` - номер сессии;
  - `pid` - номер процесса.

Возвращает значение 1 в случае ошибки, значение 2 при отсутствии номера процесса в списке номеров процессов сессии и 0 при успешной очистке кэша процесса.

## Прочие вспомогательные методы

К методам, не попавшим не в одну из предыдущих групп относятся:

- `get_methods` - получение всех доступных (по правам сертификата) методов на сервере утилит.
- Имеет два входных параметра:
  - `sid` - номер сессии;
  - `client_type` - тип клиента (gui или console).

Метод `get_methods` возвращает список данных о каждом методе, каждое из которых в свою очередь тоже являются списком из трёх элементов: название символьической ссылки на метод (для запуска через cli-core), название груп метода (для запуска с помощью клиентов) и метода. Например:

```
[ [c1-install, install, установка системы],  
[c1-core-devel-request, core-devel_request, удаление запроса],  
... ]
```

- `get_server_host_name` - получение имени хоста, на котором находится сервер утилит, записанное в его сертификате.

Не имеет входных параметров, возвращает имя хоста (hostname) сервера утилит, записанное в его сертификате.

- `active_client` - метод используется для графических клиентов, позволяет определить наличие постоянного соединения с каждой из сторон, а так же был ли перезагружен сервер (при этом необходимо переустановить соединение).

Имеет один входной параметр `sid` - номер сессии.

Возвращает:

- значение 0, если обновление информации о сессии прошло успешно.
- значение 1 в случае ошибки;
- значение 2, если клиентом послан неверный идентификатор сессии.

## Методы работы с сессиями

К вспомогательным методам работы с сессиями относятся:

- `post_sid` - получение номера сессии с сервера утилит или продолжение незакрытой предыдущей сессии. Также вызов метода устанавливает текущий язык для локализации. Используется графическими клиентами.

Имеет три входных параметра:

- `sid` - номер сессии, хранящийся у клиента. В случае, если сессия не была закрыта, она будет продолжена;
  - `cert_id` - номер сертификата;
  - `lang` - текущий язык клиента.

Возвращает список из двух целочисленных значений: номера сессии и признака продолжения старой сессии (1 - новая сессия, 0 - старая сессия).

- `del_sid` - закрытие (удаление) сессии на сервере утилит.
- Имеет единственный входной параметр `sid` - идентификатор (номер) сессии.

Возвращает значение ['0'] при успешном закрытии сессии.

- `get_sessions` - получения списка текущих сессий на сервере утилит.
  - Имеет один входной параметр `sid` - номер своей сессии, необходимый для проверки соответствия сертификату.
- Возвращает значение [''], если сессия не соответствует сертификату, иначе список всех активных сессий на сервере утилит.
- `sid_info` - получить информацию по выбранной сессии.
  - Имеет один входной параметр `sid` - номер сессии.
- Возвращает список значений: номер сертификата, дату подписания сертификата, ip-адрес, мас-адрес и тип клиента при указанные при генерации сертификата, а также значение, актив на ли сессия ('0' - активна, иначе - нет).

- `init_session` - установка начального соединения консольного клиента. Выполняет проверку сертификата клиента и получает номер сессии (или продолжает незакрытую сессию). Аналогичен действиям методов `post_sid` и `post_cert` для графических клиентов.
- Имеет два входных параметра:

- `sid` - номер сессии, хранящийся у клиента. В случае, если сессия не была закрыта, она будет продолжена;
- `lang` - текущий язык клиента.

Возвращает кортеж (список) из двух списков:

- первый - список целочисленных значений: номер сертификата и значение годности сертификата (принимает значения: "-2" - срок годности истек, "-1" - срок годности более 2-х месяцев или количество дней до истечения срока годности). Если у клиента нет сертификата возвращает значение [-3], если сертификат клиента не найден на сервере - значение [-4].
- второй список из двух целочисленных значений: номера сессии и признака

продолжения старой сессии (1 - новая сессия, 0 - старая сессия).

## Методы работы с процессами

К вспомогательным методам работы с процессами относятся:

- **list\_pid** - получить список запущенных процессов (а так же завершивших работу, но принадлежащих данной сессии).

Имеет один входной параметр **sid** - номер текущей сессии.

Возвращает список номеров запущенных процессов или значение [0].

- **pid\_info** - информация по текущему процессу.

Имеет два входных параметра:

- **sid** - номер текущей сессии;
- **pid** - номер процесса, по которому необходимо просмотреть информацию.

Возвращает список значений: идентификатор (номер) процесса, статус процесса ('1' - активен, '0' - завершён, иначе прерван или завершён с ошибкой), время запуска процесса, имя процесса (запустивший его функция), имя труса метода процесса.

**pid\_kill** - завершить (прервать) выполняющийся процесс.

Имеет два параметра, аналогичных методу **pid\_info**.

Возвращает значение -2, если сессия не соответствует сертификату, значение 3, если нет такого процесса на сервере утилит, значение 1, если не удалось послать процессу сигнал завершения и значение 0, если процессу послан сигнал завершения.

Для получения информации от выполняющихся процессов используются следующие методы: **get\_entire\_frame**, **get\_frame**, **get\_table** и **send\_message**. Методы **get\_entire\_frame**, **get\_frame** и **send\_message** возвращают данные типа **Message** (или массив таких данных). Возвращаемый сервером утилит тип **Message** служит для сообщения клиенту информации о работе запущенного процесса. Добавление таких сообщений процессом описано ранее в разделе "Основные модули" => "Метод процесса". Каждый объект типа **Message** состоит из следующих полей:

1. **type** - тип сообщения ("егор", "warning", "table", "startTask" и другие);
2. **message** - передаваемое сообщение;
3. **id** - уникальный номер (идентификатор), служащий для указания номера таблицы, номера значения прогресса или количества частей всех заданий и количество частей для каждого задания.

4. **get\_entire\_frame** - получить список всех сообщений процесса.

Имеет два входных параметра:

- **sid** - номер сессии;
- **pid** - номер выполняющегося процесса.

Возвращает массив объектов типа **Message**.

- **get\_frame** - получить список новых (неполученных ранее) сообщений процесса.
- **get\_progress** - получение значения аналогичных параметров метода **get\_entire\_frame**.

Входные и возвращаемые значения аналогичны параметрам метода **get\_entire\_frame**.

Имеет три входных параметра:

- **sid** - номер сессии;
- **pid** - номер процесса;
- **id** - идентификатор прогресса задачи.

Возвращает объект типа **ReturnProgress**, состоящий из полей:

- **percent** - целочисленное значение прогресса (процент);
- **short\_message** - короткое сообщение для текущего прогресса;
- **long\_message** - полное сообщение для текущего прогресса.

- **get\_table** - получить таблицу с сервера утилит.

Имеет аналогичные с методом **get\_progress** параметры, где **id** - идентификатор таблицы для процесса.

Возвращает объект типа **Table**, включающий в себя поля:

- **head** - шапка таблицы (список строк);
- **onClick** - имя метода, вызываемого при нажатии на строку таблицы.
- **fields** - имена полей (список строк), для передачи в вызываемый метод, имя которого указано в поле **onClick**. Номер поля в списке соответствует номеру столбца, из которого будет присвоено значение. Указывается только совместно с полем **onClick**;
- **body** - тело таблицы (двумерный массив строк);
- **addAction** - имя метода, отвечающего за добавление строки таблицы. Поле используется для графических клиентов, добавляет кнопку плюс "+" над таблицей, при нажатии на которую вызывается метод.
- **values** - список объектов типа **ChoiceValue**, в методах **printTable** и **get\_table** не используется (поле используется только в таблицах, передаваемых в составе объекта **ViewInfo**);

- **send\_message** - Отослать сообщение на сервер утилит. Постылает ответ на запросы процесса, осуществляемые вызовом методов **askQuestion** и **askPassword**.

Имеет три входных параметра:

- **sid** - номер сессии;
- **pid** - номер процесса;
- **text** - текст ответа.

Возвращает объект типа **Message**

## Методы работы с кэшем

Для работы с **кэшем** существует три метода, позволяющих на разных уровнях очистить текущие данные:

- **clear\_session\_cache** - очищает кэш для всей сессии.

Имеет один параметр **sid** - номер сессии.

Возвращает 1 в случае ошибки и 0 при успешной очистке кэша сессии.

- **clear\_method\_cache** - очищает кэш метода, принадлежащего данной сессии. Кэш метода создается до запуска процесса и хранит значение параметров, передаваемых от клиента серверу утилит.

## Поддерживаемые параметры

- Язык (для получения списка, выполните `cl-install -v -filter os_lang`)
- `DEFAULTTPARAM calculate=lang:ro_RO`
- Раскладка клавиатуры - `DEFAULTTPARAM calculate=keymap:ru_RU`
- Часовой пояс - `DEFAULTTPARAM calculate=tzonezone:Europe/Istanbul`
- Разрешение экрана - `DEFAULTTPARAM calculate=resolution:2560x1600`
- Видео драйвер - `DEFAULTTPARAM calculate=video:nvidia`
- Графическое ускорение (on/off) - `DEFAULTTPARAM calculate=composite:on`
- Домен - `DEFAULTTPARAM calculate=domain:192.168.2.3`
- Пароль для домена - `DEFAULTTPARAM calculate=domain_pw:secret`
- В качестве разделителя используется запятая. Пример: `DEFAULTPARAM calculate=domain:192.168.2.3, domain_pw:secret`

Примечание. Если указать только домен, пароль будет запрошено во время загрузки.

## Настройка репликации Samba серверов

- Настройка репликации Samba серверов
  - Введение
  - Настройка и запуск репликации
  - Резервное копирование
  - Настройка репликации сервисов
  - Перенос настроек репликации на другие серверы
  - Применение настроек репликации на других серверах
  - Отключение репликации
  - Поддержка со стороны клиента

### Введение

Часто возникает необходимость организации функционирования информационной системы с единой базой данных на нескольких серверах. Причиной этого становится, как правило, наличие на предприятиях удаленных подразделений (территориально-распределенная структура предприятия). В этом случае приходится решать проблему организации передачи данных. Для решения данной проблемы используется механизм репликации.

Репликация --- механизм синхронизации содержащего нескольких копий объекта (база данных LDAP). Репликация --- это процесс, под которым понимается копирование данных из одного источника на множество других и наоборот.

При репликации изменения, сделанные в одной копии объекта, могут быть распространены в другие копии.

Репликация Samba подразумевает, что на серверах, включенных в репликацию, будут одинаковые списки пользователей, и изменения, прогоденные на одном из этих серверов, касающиеся учетных записей unix и samba, будут автоматически перенесены на остальные сервера.

Поддержка репликации LDAP включена в пакет `calculate-server` начиная с версии 2.0.7.

## Настройка и запуск репликации

### Резервное копирование

Перед тем как начинать настраивать репликацию, рекомендуется сделать резервную копию настроек, чтобы в случае некорректных действий не потерять данные:

```
cl-backup
```

Для восстановления данных из последней резервной копии используйте команду:

### Управление unix группами

Так как ftp сервис тесно связан с unix сервисом, то управление группами, в которые входит пользователь осуществляется через unix сервис.

#### Добавление группы

Добавление группы осуществляется командой `cl-groupadd`:

```
# установить группу-владельца pubwriter
chgrp pubwriter pub
# установить разрешение писать в директорию pub для группы
chmod g+w pub
```

#### Удаление группы

Удаление группы осуществляется командой `cl-groupdel`:

```
# удалить группу pubwriter
cl-groupdel test unix
```

#### Добавление/удаление учетных записей в группу

Добавление и удаление учетных записей производится двумя способами:

- с использованием команды `cl-usermod`:

```
# добавить пользователя guest в группу pubwriter
cl-usermod -a pubwriter guest unix
# заменить пользователя guest группы на guest
cl-usermod -G guest guest unix
```
- с использованием команды `cl-groupmod`:

- с использованием команды `cl-groupmod`:

```
# удалить пользователей guest1 и guest2 из группы pubwriter
cl-groupmod -d guest1,guest2 pubwriter unix
# добавить пользователей guest1,guest2 в группу pubwriter
cl-groupmod -a guest1,guest2 pubwriter unix
```

Для определения прав доступа к директориям сервис ftp взаимодействует с unix сервисом: каждой учетной записи ftp соответствует одноименная учетная запись unix. Таким образом, пользователь, входя под своей учетной записей на ftp сервер, получает определенный доступ к файлу в зависимости от того, является ли он пользователем-владельцем файла, и входит ли в группу-владельца файла. Если пользователь не является владельцем и не входит в группу-владельца, то доступ определяется по правам `others`. В случае анонимного доступа к ftp, права также определяются по `others`.

### Установка прав доступа

Установка прав осуществляется на сервере командной chmod

```
# установить права user=все group=чтение/запись others=чтение
chmod u=rwx, g=rw, o=r file
```

# установить группу-владельца pubwriter

```
chgrp pubwriter pub
# установить разрешение писать в директорию pub для группы
chmod g+w pub
```

## Настройка Jabber сервера

XMPR (ранее известный как Jabber) --- Extensible Messaging and Presence Protocol (англ. расширяемый протокол обмена сообщениями и информацией о присутствии), это основанный на XML открытый, свободный для использования протокол для мгновенного обмена сообщениями и информацией о присутствии в режиме, близкому к режиму реального времени.

Настройка jabber сервера производится в несколько этапов:

### Установка Jabber сервиса в систему

Jabber сервис, настраиваемый с помощью пакета Calculate-Server, требует, чтобы в системе был установлен сервис Idap. Если сервис не был ранее установлен, установим его командой:

```
c1 -setup ldap
```

Установка jabber сервера производится командой

```
c1 -setup jabber
```

В этом случае сервис будет установлен с параметрами по умолчанию: имя хоста jabber сервиса - имя хоста машины, лог сообщений пользователей вестись не будет.

Для указания дополнительных jabber хостов (например jabber .myhost .ru) используется параметр --hosts:

```
c1 -setup -h hosts jabber .myhost .ru jabber
```

Установка сервиса с ведением лога сообщений пользователей, производится командой:

```
c1 -setup -h history on jabber
```

Сообщения будут сохраняться в директорию /var/log/jabber

При установке сервиса потребуется пароль для учетной записи admin.

### Учетные записи

Каждый пользователь в сети имеет уникальный идентификатор --- Jabber ID (сокращенно JID). Адрес JID, подобно адресу электронной почты, содержит имя пользователя и доменное имя сервера, на котором зарегистрирован пользователь, разделенные знаком @. Например, пользователь user, зарегистрированный на сервере example.com, будет иметь адрес: user@example.com.

### Управление учетными записями

#### Добавление учетной записи

Добавление учетной записи сервиса jabber осуществляется командой c1-useradd

```
# добавить пользователя guest@домен с Nickname Гость
c1-useradd -p -с "Гость" guest jabber
# добавить пользователя guest@домен с Nickname Гость, и установить для
его фотографию
c1-useradd -p -с "Гость" -i pic/guest.png guest jabber
```

Поддерживаемые форматы изображений определяются возможностями ImageMagick. Если в системе его нет - доступен только формат jpg.

#### Смена пароля

Смена пароля учетной записи сервиса jabber осуществляется командой c1-passwd

```
c1 -passwd guest jabber
```

## Информация о сети

```
c1 -info -N <ip_и_маска_сети> dhcp
```

Пример:

```
c1 -info -N 10.0.0.0/24 dhcp
```

Информация о сети 10.0.0.0/24

## Информация о всех статических хостах==

```
c1 -info --hosts dhcp
```

## Информация о статическом хосте

```
c1 -info -H <название_хоста> dhcp
```

Пример:

```
c1 -info -H computer dhcp
```

Информация о статическом хосте computer.

## Настройка PXE

### Введение

PXE --- среда для загрузки компьютеров с помощью сетевой карты без использования жестких дисков, компакт-дисков и других устройств, применяемых при загрузке операционной системы. PXE-код, прописанный в сетевой карте, получает загрузчик из сети, после чего передает ему управление.

Для организации загрузки системы в PXE используется протоколы IP, UDP, DHCP и TFTP. Система, загружаемая по сети, является аналогом загрузки с liveCD.

### Основные требования

Настройка PXE может быть выполнена на Calculate Directory Server с пакетом sys-apps/calculate-install версии 3.1.8 или выше.

### Настройка

Перед настройкой PXE на сервере должен быть настроен DHCP сервис.

Настройка PXE выполняется при помощи команды c1-install-pxe:

```
# установка iso образа
c1-install-pxe -iso /var/calculate/linux/cld-13.6.2-i686.iso
```

Образ дистрибутива развертывается в каталог /var/calculate/pxe, настраиваются необходимые службы TFTP, NFS и DHCP. Поддерживается одновременное использование только одной версии дистрибутива.

После завершения установки все машины в сети смогут воспользоваться PXE загрузкой.

### Настройка параметров по умолчанию

Вы можете изменить настройки загрузки PXE образа, такие как язык, часовой пояс, видео карта и другие при помощи параметра DEFAULT PARAM в файле /var/calculate/pxe/rhelinux.cfg/default. Запись должна быть добавлена после строки DEFAULT calcmenu.c32.

- -dnames - доменные имена передаваемые клиентскому компьютеру в качестве доменных имен для поиска, разделитель имен запятая. Первое имя используется для создания DNS домена на серверном компьютере в который будут добавлены компьютеры подключающиеся к сети.
  - --range - диапазон адресов из которого клиентский компьютер получает свой ip, два ip адреса разделенные запятой.
  - --net - сеть в которой будут находиться клиентские компьютеры
  - --dnsip - ip адрес DNS сервера передаваемый клиентскому компьютеру
- Пример:  
`c1-dhcp-netadd --router 10.0.0.1 --dnames domain.ru, domain.org --range 10.0.0.20,10.0.0.100 --net 10.0.0.0/24 --dnsip 10.0.0.5`

#### Модификация параметров сети DHCP

Для модификации параметров сети используется команда  
`c1-dhcp-netmod [параметры] ip_и_маска_сети`

Параметры:

- --router - измененный ip адрес передаваемый клиентскому компьютеру в качестве шлюза по умолчанию.
- -dnames - измененные доменные имена передаваемые клиентскому компьютеру в качестве доменных имен для поиска, разделитель имен запятая. Первое имя используется для создания DNS домена на серверном компьютере в который будут добавлены компьютеры подключающиеся к сети.
- --range - измененный диапазон динамических адресов которые раздаются клиентским компьютерам.
- --dnsip - измененный ip адрес DNS сервера передаваемый клиентскому компьютеру

Пример:

```
c1-dhcp-netmod --range 10.0.0.50,10.0.0.255 10.0.0.0/24
c1-dhcp-netmod --router 10.0.0.1 10.0.0.0/24
```

#### Удаление сети DHCP

Если существует только одна сеть DHCP то ее удаление приведет к невозможности запуска DHCP сервера добавление новой сети вернет работоспособность.

Для удаления сети используется команда

```
c1-dhcp-netdel -net ip_и_маска_сети
```

Параметры:

- --net удаляемая сеть

Пример:

```
c1-dhcp-netdel -net 10.0.0.0/24
```

#### Информация о DHCP сервисе

Для получения информации о записях и зонах DNS сервиса используется команда `c1-info`.

#### Информация о всех сетях

```
c1-info -n dhcp
```

#### Блокировка и удаление учетной записи

Удаление учетной записи из сервиса jabber осуществляется командой `c1-userdel`

```
c1-userdel guest jabber
```

Блокировка учетной записи производится командой `c1-usermod -L`

```
c1-usermod -L guest jabber
```

Разблокировать учетную запись можно командой `c1-usermod -U`

```
c1-usermod -U guest jabber
```

#### Группы

Группа - набор ID, использующийся для рассылки сообщений нескольким пользователям одновременно. При подключении пользователя к jabber сервису он автоматически получит список групп и их участников. Пользователей без групп в списке контактов добавлять придется вручную. Пользователь может входить только в одну группу.

#### Управление группами

##### Создание группы

Создание группы сервиса jabber осуществляется командой `c1-groupadd`

```
# создать группу с назначением "Тестовая группа"
c1-groupadd "Тестовая группа" jabber
```

Имя группы будет отображаться у пользователей в списке контактов.

##### Удаление группы

Удалить группу из сервиса jabber можно командой `c1-groupdel`

```
# удалить "Тестовую группу"
c1-groupdel "Тестовая группа" jabber
```

Удаляется только группа, все пользователи, которые были включены в эту группу остаются без группы.

##### Переименование группы

Переименовать существующую группу сервиса jabber можно командой `c1-groupmod`

```
# назначить группе 'My test' новое имя 'Тестовая группа'
c1-groupmod -n 'Тестовая группа' 'My test' jabber
```

##### Изменение состава группы

Изменять состав jabber группы можно при помощи команд `c1-groupmod` и `c1-usermod`

```
# Поместить пользователя guest в группу 'Guest group'
c1-usermod -g "Guest group" guest jabber
# Удалить пользователя guest из группы 'Guest group'
c1-groupmod -d guest "Guest group" jabber
# Добавить пользователя guest, guest2 в группу 'Guest group'
c1-groupmod -a guest, guest2 "Guest group" jabber
```

## Ограничения

Если на вашем сервере больше одного сетевого интерфейса, и вы желаете чтобы он работал только на одном из них, внесите в его конфиг соответствующую директиву {ip, {xxx, xxx, xxx, xxx}}, в раздел "listen". Обратите внимание! Разряды IP отделяются не точками, а запятыми!

```
{5223, ejabberd_c2s, [
    {access, c2s},
    {shaper, c2s_shaper},
    {ip, {192, 168, 1, 6}},
    {certfile, "/etc/jabber/ssl.pem"}, tls,
    {max_stanza_size, 65536}
]},
```

Соответственно, эту директиву можно прописать во всех службах ejabberd: ejabberd\_s2s, ejabberd\_http, и т.д.

## Настройка почтового сервера

### Установка почтового сервиса в систему

Почтовый сервис mail требует, чтобы в системе были установлены сервисы LDAP и Unix, поэтому если они еще не были установлены установите их командой:

```
c1-setup ldap
c1-setup unix
```

Установка почтового сервиса производится командой:

```
# установка сервиса с параметрами по умолчанию
c1-setup mail
```

Для указания имени почтового хоста, отличного от используемого по умолчанию, служит параметр "--host":

```
# mymail.mydomain.com почтовый хост (не указывайте одно короткое имя,
например: mymail)
c1-setup --host mymail.mydomain.com mail
```

Использование протокола pop3 и/или imap управляется через параметр "--type":

```
# устанавливает сервис с поддержкой двух протоколов pop3 и imap с
шифрованием TLS
c1-setup --type imap,pop3 mail
# устанавливает сервис с поддержкой только pop3 и шифрованием TLS
c1-setup --type pop3 mail
```

Шифрование (или отмена его использования) указывается параметром "--crypt":

```
# будет использовано шифрование TLS, и по умолчанию будет использоваться
только IMAP
c1-setup --crypt tls mail
# без шифрования, по умолчанию используется только IMAP
c1-setup --crypt none mail
```

Если необходимо указать используемые протоколы и шифрование, выполним команду с параметрами следующего вида:

## Управление DHCP сервисом

### Управление статическими хостами DHCP сервиса

Создание статического хоста DHCP

Для создания статического используется команда

```
c1-dhcp-hostadd --host <название хоста> --ip <ip адрес> --mac <mac адрес>
```

Параметры:

- --host - название клиентского компьютера, (название без домена)
- --ip - ip адрес который будет назначен клиентскому компьютеру
- --mac - mac адрес клиентского компьютера (можно узнать командой ifconfig)

Пример:

```
c1-dhcp-hostadd --host test --ip 10.0.0.20 --mac 00:17:31:c2:88:82
```

Модификация параметров статического хоста

Для модификации статического хоста используется команда

```
c1-dhcp-hostmod [параметры] название_хоста
```

Параметры:

- --ip - измененный ip адрес который будет назначен клиентскому компьютеру
- --mac - измененный mac адрес клиентского компьютера

Пример:

```
c1-dhcp-hostmod --ip 10.0.0.25 test
```

Удаление статического хоста

Для удаления статического хоста используется команда

```
c1-dhcp-hostdel --host <название хоста>
```

Параметры:

- --host - название клиентского компьютера, (название без домена)

Пример:

```
c1-dhcp-hostdel --host test
```

### Управление сетями DHCP сервиса

Создание сети DHCP

На каждый сетевой интерфейс может быть создана только одна сеть.

Для создания сети используется команда

```
c1-dhcp-netadd --router <ip роутера> --dnames <имена доменов> --range
<диапазон ip для динамических хостов> --net <ip сети с маской
<ip DNS сервера> dhcp
```

Параметры:

- --router - ip адрес передаваемый клиентскому компьютеру в качестве шлюза по умолчанию

Результат при нормальной работе сервиса DNS:

```
Using domain server:
Name: 10.0.0.5
Address: 10.0.0.5#53
Aliases:
```

```
calculate.domain.ru has address 10.0.0.54
```

## Настройка DHCP сервера

### Установка DHCP сервиса в систему

Поддержка DHCP сервиса появилась в пакете calculate-server 2.1.4.

В качестве сервера используется dhcpr.

#### Перед установкой

Убедитесь что у вас поднят сетевой интерфейс, интерфейсу присвоен ip находящийся в сети которая будет указана при установке сервиса DHCP (параметр командной строки установки сервиса -net).

Убедитесь что у вас в системе установлен сервис dns, если он не был установлен, выполните установку командой:

```
c1-setup dns
```

```
# без шифрования, поддержка imap и pop3
c1-setup none -t type imap,pop3 mail
```

После выполнения команды `c1-setup`, с требуемым набором параметров - почтовый сервис конфигурируется и запускается. Результат можно проверить, посмотрев открытые порты:

```
> netstat -tln
...
tcp      0      0    * :imaps          *:*
tcp      0      0    * :pop3s          *:*
tcp      0      0    * :smtp           *:*
...
LISTEN
LISTEN
LISTEN
```

Для предотвращения спама для системных почтовых пользователей `/etc/mail/aliases` не создается почтовая директория.

Для того чтобы получать письма посланные на адреса указанные в `/etc/mail/aliases` выполните следующие команды:

```
mkdir /var/calculate/server-data/mail/nobody
chown nobody:nobody /var/calculate/server-data/mail/nobody
chmod 0700 /var/calculate/server-data/mail/nobody
```

Письма для системных почтовых пользователей будут находиться в `/var/calculate/server-data/mail/nobody`

## Учетные записи

Учётная запись - это запись, которая содержит сведения, необходимые для идентификации пользователя при подключении к системе, информацию для авторизации и учёта. В данном случае для подключения к почтовому серверу или почтовому ящику.

Почтовый ящик - логически выделенная часть дискового пространства, предназначенная для хранения электронных почтовых сообщений, которая обозначается электронным почтовым адресом. Почтовый ящик может иметь несколько почтовых адресов, называемых синонимами почтового адреса ( псевдонимами почтового адреса).

### Управление учетными записями

Добавление учетных записей

Добавление учетной записи пользователя почтового сервиса производится командой `c1-useradd`:

```
c1-useradd -p -e <почтовый псевдоним один или несколько через запятую>
<учетная запись> mail
```

```
# добавляем пользователя guest с псевдонимом для почты
guestmail@mail.mydomain.com guest mail
c1-useradd -p -e guestmail@mail.mydomain.com guest mail
```

Смена пароля

```
Для смены пароля учетной записи почтового сервиса используется команда c1-passwd:
c1-passwd guest mail
```

Удаление учетных записей

Удаление учетной записи пользователя почтового сервиса производится командой `c1-userdel`

```
cl-userdel guest mail
```

```
cl-info -z domain.ru dns
```

## Почтовые группы

Почтовая группа --- набор почтовых адресов, использующийся для рассылок почты нескольким корреспондентам. Письмо, отправленное на адрес группы, рассылается для всех почтовых учетных записей, входящих в эту группу.

### Управление почтовыми группами

#### Добавление почтовой группы

```
Добавление почтовой группы производится командой cl-groupadd
```

```
# добавить почтовую группу guestgroup с альтернативным почтовым адресом  
gg@mydomain.com  
cl-groupadd -e gg@mydomain.com guestgroup mail
```

#### Удаление почтовой группы

```
Удаление почтовой группы производится командой cl-groupdel:
```

```
# удалить почтовую группу guestgroup  
cl-groupdel guestgroup mail
```

#### Добавление и удаление почтовых учетных записей в группу

```
Добавление и удаление учетных записей производится двумя способами:
```

- с использованием команды cl-usermod

```
# добавить пользователя guest в группы guesttest и guestgroup  
cl-usermod -a guesttest, guestgroup guest mail
```

```
# заменить пользователя guest группы на guesttest
```

```
cl-usermod -G guesttest guest mail
```

- с использованием команды cl-groupmod

```
# удалить пользователей guest1 и guest2 из группы guesttest  
cl-groupmod -d guest1, guest2 guesttest mail
```

```
# добавить пользователей guest1 и guest2 в группу guesttest  
cl-groupmod -a guest1, guest2 guesttest mail
```

## Проверка работы сервера

Проверить работоспособность сервера можно при помощи программы telnet, при этом сервер должен быть сконфигурирован без шифрования ("--с罂ут поН"). В конечном рабочем варианте, для безопасности, шифрование следует оставить включенным.

1. устанавливаем сервис

```
cl-setup --type imap, pop3 --scrypt none mail
```

1. вводим пароль

```
cl-useradd -p -e guest@mymail.mydomain.org guest mail
```

1. добавляем пользователя guest

```
host calculate.domain.ru 10.0.0.5
```

Результат:

```
Information about master DNS zone domain.ru  
+-----+  
| Field | Value |  
+-----+  
| Zone name | domain.ru |  
| Master authoritative server | domain.ru |  
| NS record | domain.ru |  
| A record | 10.0.0.5 |  
| Email administrator | root@domain.ru |  
| Serial number | 3 |  
| Refresh | 8H |  
| Update | 2H |  
| Expiry | 2W |  
| Minimum | 2H |  
+-----+
```

(10 rows)

```
Information about A records in master DNS zone domain.ru  
+-----+  
| Domain | ip |  
+-----+  
| localhost.domain.ru | 127.0.0.1 |  
| calculate.domain.ru | 10.0.0.54 |  
+-----+
```

(2 rows)

Используя любую из существующих A записей проверьте работоспособность DNS сервера с помощью команд:

```
nslookup имя_A_записи ip_DNS_сервера
```

или

```
host имя_A_записи ip_DNS_сервера
```

Пример:

ip адрес проверяемого DNS сервера 10.0.0.5 информация о зоне domain.ru приведена в предыдущем примере.

Выполняем проверку при помощи nslookup:

```
nslookup calculate.domain.ru 10.0.0.5
```

Результат при нормальной работе сервиса DNS:

```
Server: 10.0.0.5  
Address: 10.0.0.5#53
```

```
Name: calculate.domain.ru  
Address: 10.0.0.54
```

```
Выполним проверку при помощи host:  
host calculate.domain.ru 10.0.0.5
```

**Примеры:**

```
c1-info -Z 10.0.0.0/24 dns
```

Информация о обратной зоне 0.0.10.in-addr.arpa (сеть 10.0.0.0/24)

```
c1-info -Z test.ru dns
```

Информация о прямой зоне test.ru

**Информация о записи**

```
c1-info -r <имя_записи или ip> dns
```

**Примеры:**

```
c1-info -r 10.0.0.5 dns
```

Информация о записи в обратной зоне 5.0.0.10.in-addr.arpa (ip 10.0.0.5)

```
c1-info -r test.test.ru dns
```

Информация о записи в прямой зоне test.test.ru

**Пример создания зоны и записей вней**

Необходимо создать зону test.ru а также доменные имена:

- test.ru - ip 10.0.0.1 - WEB сервер, DNS сервер.
- www.test.ru - ip 10.0.0.1 - WEB сервер (CNAME запись, тот же сервер что и test.ru)
- ftp.test.ru - ip 10.0.0.5, FTP сервер
- user1.test.ru - 10.0.0.100, компьютер пользователя

Для этого выполняем следующие команды после установки сервиса DNS:

1. Создаем зону test.ru с A записью (test.ru --> 10.0.0.1) и обратной зоной для сети 10.0.0.0/24

```
c1-dns-zoneadd -n test.ru --server test.ru --ipserver 10.0.0.1
```

1. Создаем CNAME запись (www.test.ru --> test.ru)

```
c1-dns-recadd -t cname --host www.test.ru --cname test.ru
```

1. Создаем A и PTR записи для FTP сервера

```
c1-dns-recadd --host ftp.test.ru --ip 10.0.0.5
```

1. Создаем PTR запись для компьютера пользователя

```
c1-dns-recadd -n host user1.test.ru --ip 10.0.0.100
```

**Проверка работоспособности DNS сервера**

Для проверки работоспособности DNS сервера используйте утилиты nslookup или host.

После того как вы создали DNS зону и добавили в нее записи, нужно посмотреть существующие записи в зоне с помощью команды:

```
c1-info -Z имя_зоны dns
```

**Пример:**

Ранее была создана зона test.ru и записи вней.

Выполним:

**2. запускаем telnet**

```
> telnet
```

```
1. подключаемся к smtp
> open localhost 25
Trying 127.0.0.1...
Connected to mymail.mydomain.org.
Escape character is '^'.
220 mymail.mydomain.org ESMTP
> EHLO "mymail"
250-mymail.mydomain.org
250-PIPELINING
250-SIZE 100000000
250-ETRN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
> MAIL FROM:<noname@mailer.org>
250 2.1.0 Ok
> RCPT TO:<guest@mymail.mydomain.org>
250 2.1.5 Ok
> DATA
354 End data with <CR><LF> .<CR><LF>
> Hello
> .
250 2.0.0 Ok: queued as D42932B91
> QUIT
221 2.0.0 Bye
Connection closed by foreign host.
```

```
1. теперь запускаем telnet и подключаемся по port3
> telnet
> open localhost 110
Trying 127.0.0.1...
Connected to mymail.mydomain.org.
Escape character is '^'.
+OK Dovecot ready.
> USER guest
+OK
> PASS 111
+OK "Аутентифицироваться удалось"
> LIST
+OK 1 messages: "На сервере есть одно сообщение"
1 529
QUIT
+OK Logging out.
Connection closed by foreign host.
```

**Настройка Proxy сервера**

Настройка Proxy сервера производится в несколько этапов:

## Установка Proxy сервиса в систему

Поддержка proxy сервиса появилась в пакете *Calculate-server* в версии 2.0.13. В качестве сервера используется наиболее распространенный proxy сервер *Squid*.

Перед установкой, убедитесь что у вас в системе установлен сервис LDAP, если он не был установлен, выполните установку командой:

```
cl-setup ldap
```

Установка proxy сервиса производится командой

```
cl-setup proxy
```

В этом случае сервис будет установлен с параметрами по умолчанию: имя хоста proxy сервиса - имя хоста машины, порт для соединения - 8080.

Установка сервиса с установкой доверительных сетей:

```
cl-setup -a proxy
```

При установке сервиса будут созданы базовые группы:

- *http* - доступный порт 80
- *ftp* - доступный порт 21
- *https* - доступный порт 443
- *gopher* - доступный порт 70
- *wais* - доступный порт 210
- *unregistered* - диапазон доступных портов 1025-65535
- *http-mgmt* - доступный порт 280
- *gss-ftp* - доступный порт 488
- *filemaker* - доступный порт 591
- *multiling* - доступный порт 777
- *swat* - доступный порт 901

Интервал обновления кеша изменений в правах доступа пользователей по умолчанию составляет 5 минут.

## Управление учетными записями

### Добавление учетной записи

Добавление учетной записи пользователя proxy сервиса осуществляется командой *cl-useradd*

```
# добавим пользователя _ivan_ с полным именем в комментарии
```

```
cl-useradd -p -c "Ivanov Ivan" ivan proxy
```

### Смена пароля

Смена пароля учетной записи осуществляется командой *cl-passwd*, пример:

```
cl-passwd ivan proxy
```

### Удаление/блокировка учетной записи

Удаление учетной записи осуществляется командой *cl-userdel*, пример

```
cl-userdel ivan proxy
```

Блокировка учетной записи производится командой *cl-usermod -L*, пример

```
cl-usermod -L ivan proxy
```

## Модификация или создание MX записи

### Пример 1:

```
cl-dns-rcmod -mx mail1.test.ru,mail2.test.ru test.test.ru
```

Заменяет а в случае отсутствия добавляет MX записи в A записи test.test.ru.

#### Исходная запись:

```
A запись test.test.ru - MX запись mail.test.ru (приоритет 10)
```

#### Запись после модификации:

```
A запись test.test.ru - MX запись mail1.test.ru (приоритет 10), MX запись mail2.test.ru (приоритет 20)
```

### Пример 2:

```
cl-dns-rcmod -mxmod mail2.test.ru,mailnew.test.test.ru test.test.ru
```

Изменяет MX записи.

#### Исходная запись:

```
A запись test.test.ru - MX запись mail1.test.ru (приоритет 10), MX запись mail2.test.ru (приоритет 20)
```

#### Запись после модификации:

```
A запись test.test.ru - MX запись mail1.test.ru (приоритет 10), MX запись mailnew.test.test.ru (приоритет 20)
```

## Удаление DNS записи

Для удаления DNS записи используется команда *cl-dns-recdel*.

Удаление A или CNAME записи

Удаление PTR записи

### Пример:

```
cl-dns-recdel --host test.test.ru
```

Будет удалена A или CNAME запись test.test.ru

Удаление PTR записи

### Пример:

```
cl-dns-recdel -ip 10.0.0.20
```

Будет удалена PTR запись 20.0.0.10.in-addr.apr (10.0.0.20 соответствует test.test.ru)

Удаление MX записи из A записи

### Пример:

```
cl-dns-recdel -mx -host test.test.ru
```

Будет удалены все MX записи из A записи test.test.ru

## Информация о DNS сервисе

Для получения информации о записях и зонах DNS сервиса используется команда *cl-info*.

## Информация о всех зонах

```
cl-info -z dns
```

## Информация о зоне

```
cl-info -Z <имя_зоны или сеть> dns
```

Записи после модификации:  
PTR запись 10.0.0.6 соответствует oldname.test.ru  
A запись oldname.test.ru соответствует 10.0.0.6

Разблокировать учетную запись можно командой `cl-usermod -U`, пример  
`cl-usermod -U ivan proxy`

#### Изменение доменного имени PTR записи

**Пример:**

```
cl-dns-recmod -t ptr --automod off --host newname.test.ru oldname.test.ru
```

или

```
cl-dns-recmod -t ptr --automod off --host newname.test.ru 10.0.0.5
```

Изменяет доменное имя oldname.test.ru на newname.test.ru

*Исходная запись:*

PTR запись 10.0.0.5 соответствует oldname.test.ru

Запись после модификации:

PTR запись 10.0.0.5 соответствует newname.test.ru

*Изменение IP PTR записи*

**Пример:**

```
cl-dns-recmod -t ptr --automod off --ip 10.0.0.6 10.0.0.5
```

или

```
cl-dns-recmod -t ptr --ip 10.0.0.6 oldname.test.ru
```

Изменяет IP на 10.0.0.6 для доменного имени oldname.test.ru

*Исходная запись:*

PTR запись 10.0.0.5 соответствует oldname.test.ru

Запись после модификации:

PTR запись 10.0.0.6 соответствует oldname.test.ru

#### Модификация CNAME записи

**Пример 1:**

```
cl-dns-recmod --cname calculate.ru cn.test.ru
```

Изменяет CNAME запись.

*Исходная запись:*

CNAME запись cn.test.ru соответствует acoola.ru

Запись после модификации:

CNAME запись cn.test.ru соответствует calculate.ru

**Пример 2:**

```
cl-dns-recmod -t cname --host cname.test.ru cn.test.ru
```

Изменяет CNAME запись.

*Исходная запись:*

CNAME запись cn.test.ru соответствует calculate.ru

Запись после модификации:

CNAME запись cname.test.ru соответствует calculate.ru

## Группы

Группа - набор учетных записей для применения правил доступа.  
У группы есть параметр определяющий диапазон сетевых портов.

Пользователю разрешен доступ к ресурсу только в том случае, если доступ к порту, по которому происходит обращение, описывается одной из групп пользователя.

## Управление группами

### Создание группы

Создание группы proxy сервиса осуществляется командой `cl-groupadd`, пример создания группы "ads!":  
`cl-groupadd -p 80 83,2000-3000 ads1 proxy`

Пользователи добавленные в эту группу будут иметь доступ к сетевому порту 80 и 83, а также к диапазону сетевых портов 2000-3000, по которым может работать определенный сервис. Условно мы его называем "ads!".

### Удаление группы

Удалить группу из сервиса можно командой `cl-groupdel`, пример:  
`cl-groupdel ads1 proxy`

### Переименование группы

Переименовать существующую группу сервиса можно командой `cl-groupmod`, пример смены имени группы "ads!" на "ads12":  
`cl-groupmod -n ads12 ads1 proxy`

### Изменение состава группы

Изменить описание группы можно при помощи команд "cl-groupmod" и "cl-usermod". Примеры:  
*Пример 1:*  
`cl-usermod -G http ivan proxy`

Удалим пользователя ivan из группы 'http' (закроем доступ к сайтам)  
`cl-groupmod -d ivan http proxy`

Добавим пользователя ivan в группу 'http' (откроем пользователю доступ к сайтам)  
`cl-usermod -G http ivan proxy`

Для того чтобы дать пользователю доступ к ftp, он должен иметь доступ к порту 21 и порту proxy сервера (по умолчанию 8080).  
*Пример 2:*  
`# Дать пользователю guest доступ к ftp  
cl-usermod -G ftp,unregistered guest proxy`

## Настройка DNS сервера

- Настройка DNS сервера

## • Установка DNS сервиса в систему

## • Управление DNS сервисом

### • Создание DNS зоны

### • Модификация DNS зоны

### • Удаление MX записи для зоны

### • Удаление A записи для зоны

### • Создание DNS записи

### • Модификация записи

### • Удаление DNS записи

### • Информация о DNS сервисе

### • Информация о всех зонах

### • Информация о зоне

### • Информация о записи

### • Пример создания зоны и записей в ней

### • Проверка работоспособности DNS сервера

Настройка DNS сервера производится в несколько этапов:

## Установка DNS сервиса в систему

Поддержка DNS сервиса появилась в пакете [calculate-server 2.1.4](#).

В качестве сервера используется наиболее распространенный DNS сервер BIND.

Перед установкой убедитесь что BIND скомпилирован с поддержкой *sdb-ldap*.

Перед установкой, убедитесь что у вас в системе установлен сервис *ldap*, если он не был установлен, выполните установку командой:

```
c1 -setup ldap
```

Установка dns сервиса выполняется командой

```
c1 -setup dns
```

Примечание: Интервал времени жизни DNS записи - *ttl*, составляет 178600 секунд.

## Управление DNS сервисом

### Термины:

- *DNS зона* - сегмент пространства доменных имен.
- *master DNS зона* - основная зона хранения записей.
- *slave DNS зона* - подчиненная основной зоне хранения записей.
- *Обратная DNS зона* - зона хранения записей соответствия IP адреса доменному имени.
- *Авторитативный сервер* - сервер для хранения DNS зоны, записи которого считаются авторитетными для других DNS серверов.
- *SOA запись* - запись описания зоны.
- *NS запись* - доменное имя авторитативного сервера.
- *A запись* - соответствие доменного имени IP адресу.
- *PTR запись* - соответствие IP адреса доменному имени.
- *CNAME запись* - соответствие одного доменного имени другому.
- *MX запись* - соответствие доменного имени почтовых серверов.

### или

```
c1-dns-recmod -a automod off -h host newname.test.ru 10.0.0.5
```

Изменяет доменное имя *oldname.test.ru* на *newname.test.ru*

### Исходные записи:

А запись, *oldname.test.ru* соответствует 10.0.0.5

### Запись после модификации:

А запись, *newname.test.ru* соответствует 10.0.0.5

### Изменение IP A записи

### Пример:

```
c1-dns-recmod -a automod off -h ip 10.0.0.6 10.0.0.5
```

или

```
c1-dns-recmod -ip 10.0.0.6 oldname.test.ru
```

Изменяет IP на 10.0.0.6 для доменного имени *oldname.test.ru*

### Исходные записи:

А запись *oldname.test.ru* соответствует 10.0.0.5

### Запись после модификации:

А запись *oldname.test.ru* соответствует 10.0.0.6

### Модификация PTR записи

### Изменение доменного имени PTR записи и A записи

### Пример:

```
c1-dns-recmod -t ptr -h host newname.test.ru oldname.test.ru
```

или

```
c1-dns-recmod -t ptr -h host newname.test.ru 10.0.0.5
```

Изменяет доменное имя *oldname.test.ru* на *newname.test.ru*

### Исходные записи:

PTR запись 10.0.0.5 соответствует *oldname.test.ru*

А запись *oldname.test.ru* соответствует 10.0.0.5

### Запись после модификации:

PTR запись 10.0.0.5 соответствует *newname.test.ru*

А запись *newname.test.ru* соответствует 10.0.0.5

### Изменение IP PTR записи и A записи

### Пример:

```
c1-dns-recmod -t ptr -h ip 10.0.0.6 10.0.0.5
```

или

```
c1-dns-recmod -ip 10.0.0.6 oldname.test.ru
```

Изменяет IP для доменного имени *oldname.test.ru*

### Исходные записи:

PTR запись 10.0.0.5 соответствует *oldname.test.ru*

А запись *oldname.test.ru* соответствует 10.0.0.5

## Создание PTR записи

Пример создания PTR записи. Сначала должна быть создана обратная зона 0.0.10.in-addr.arpa.

```
c1-dns-recadd -t ptr -r 10.0.0.67 -h host.test.ru
```

- Будет создана запись в обратной зоне 0.0.10.in-addr.arpa, 10.0.0.67 соответствует host.test.ru

### Создание CNAME записи

Пример создания CNAME записи. Сначала должна быть создана прямая зона test.ru.

```
c1-dns-recadd -t cname -h host.test.ru -s calculate.ru
```

- Будет создана запись в прямой зоне test.ru, host.test.ru соответствует calculate.ru

### Модификация записи

Для модификации DNS записи используется команда cl-dns-recmod.

#### Модификация A записи

Изменение доменного имени A записи и PTR записи

Пример:

```
c1-dns-recmod -h host.newname.test.ru oldname.test.ru
```

или

```
c1-dns-recmod -h host.newname.test.ru 10.0.0.5
```

Изменяет доменное имя oldname.test.ru на newname.test.ru

Исходные записи:

A запись, oldname.test.ru соответствует 10.0.0.5

PTR запись, 10.0.0.5 соответствует oldname.test.ru

Записи после модификации:

A запись, newname.test.ru соответствует 10.0.0.5

PTR запись, 10.0.0.5 соответствует newname.test.ru

Изменение IP A записи и PTR записи

Пример:

```
c1-dns-recmod -i ip 10.0.0.6 10.0.0.5
```

Изменяет IP для доменного имени oldname.test.ru

Исходные записи:

A запись oldname.test.ru соответствует 10.0.0.5  
PTR запись 10.0.0.5 соответствует oldname.test.ru

Записи после модификации:

A запись oldname.test.ru соответствует 10.0.0.6  
PTR запись 10.0.0.6 соответствует oldname.test.ru

Изменение доменного имени A записи

Пример:

```
c1-dns-recmod -a automod off -h newname.test.ru oldname.test.ru
```

## Создание DNS зоны

Для создания DNS зоны используется команда cl-dns-zoneadd.

### Создание master DNS зоны

- Создание зоны с авторитативным сервером в созданной зоне.  
`cl-dns-zoneadd -n <имя зоны> -s server <имя авторитативного сервера>`

### Создание зоны с авторитативным сервером в другой зоне.

- `cl-dns-zoneadd -n <имя зоны> -s server <имя авторитативного сервера>`

**Примеры:**  
`cl-dns-zoneadd -n test.ru -s server test.ru -i pserver 10.0.0.34`

- Будет создана прямая зона test.ru
- Если не существует, будет создана обратная зона 0.0.10.in-addr.arpa
- Будет создана в записи зоны test.ru, A запись - test.ru соответствует 10.0.0.34
- Будет создана в записи зоны test.ru NS запись - test.ru
- Если обратная зона 0.0.10.in-addr.arpa была создана, для нее будет создана NS запись - test.ru
- Будет создана обратная зона 0.0.10.in-addr.arpa

`cl-dns-zoneadd -n test.ru -s server ns.test.ru -i pserver 10.0.0.34`

- Будет создана прямая зона test.ru
- Если не существует, будет создана обратная зона 0.0.10.in-addr.arpa
- Будет создана в зоне test.ru, A запись - ns.test.ru соответствует 10.0.0.34
- Будет создана в записи зоны test.ru NS запись - ns.test.ru
- Если обратная зона 0.0.10.in-addr.arpa была создана, для нее будет создана NS запись - test.ru
- В обратной зоне 0.0.10.in-addr.arpa будет создана PTR запись - 10.0.0.34 соответствует ns.test.ru, если такая запись не существует.

`cl-dns-zoneadd -n 10.0.10.0/24 -s server test.ru`

- \* Будет создана обратная зона для сети 10.0.10.0/24 - 10.0.10.in-addr.arpa \* Будет создана в записи зоны 10.0.10.in-addr.arpa NS запись - test.ru

### Создание slave DNS зоны

#### Создание DNS зоны.

- `cl-dns-zoneadd -t slave -p <имя зоны> -s servers <ip серверов хранения master зоны для этой зоны>`

**Примеры:**

`cl-dns-zoneadd -t slave -n slave.ru -s servers 10.0.0.3,10.0.10.5`

- Будет создана подчиненная прямая зона slave.ru, данные для которой будут получены из основной зоны slave.ru находящейся на DNS серверах с адресами 10.0.0.3, 10.0.10.5

`cl-dns-zoneadd -t slave -n 10.0.0.0/24 -s servers 10.0.0.3`

- \* Будет создана подчиненная обратная зона для сети 10.0.0.0/24 - 0.0.10.in-addr.arpa, данные для которой будут получены из основной зоны 0.0.10.in-addr.arpa находящейся на DNS сервере с адресом 10.0.0.3

## Модификация DNS зоны

Для модификации DNS зоны используется команда `cl-dns-zonemod`.

Модификация параметров зоны возможна только для master зоны.

```
cl-dns-zonemod -n <имя зоны или сеть> <параметры>
```

имя зоны - модификация прямой зоны  
сеть - модификация обратной зоны

**Параметры модификации зоны:**

- `--server` - изменение доменного имени главного авторитативного сервера зоны
- `--ip` - изменение или добавление в случае отсутствия, ip адреса для зоны (модификация или добавление A записи)
- `--ptr` - замена или добавление в случае отсутствия, MX записи для зоны (модификация или добавление доменных имен почтовых серверов)
- `--ptrhop` - замена одного доменного имени почтового сервера на другой в MX записи для зоны (модификация доменного имени почтового сервера)
- `--email` - изменение почтового адреса администратора зоны (по умолчанию root@имя\_зоны)
- `--servers` - изменение списка всех авторитативных серверов зоны (NS записи зоны)
- `--refresh` - интервал времени после которого будет обновлена зона в секундах или число + (M - минуты, H - часы, D - дни, W - недели).
- По умолчанию 8H - 8 часов.
- `--update` - интервал времени после неудачного обновления зоны после которого будет сделано новое обновление зоны.
- По умолчанию 2H - 2 часа.
- `--expiry` - интервал времени после которого данные зоны устареют на вторичных DNS серверах в случае невозможности соединения с главным DNS сервером.
- По умолчанию 2W - 2 недели.
- `--minimum` - интервал времени хранения данных неудачных запросов для этой зоны.
- По умолчанию 2H - 2 часа.

**Примеры:**

```
cl-dns-zonemod -n test.ru --email admin@test.ru
```

Модификация почтового адреса администратора зоны

```
cl-dns-zonemod -n test.ru --refresh 10H
```

Модификация интервала времени обновления зоны (10 часов)

## Удаление DNS зоны

Для удаления DNS зоны используется команда `cl-dns-zonedelete`.

```
cl-dns-zonedelete -n <имя зоны или сеть>
```

имя зоны - удаление прямой зоны  
сеть - удаление обратной зоны

**Примеры:**

```
cl-dns-zonedelete -n test.ru
```

Будет удалена прямая зона test.ru

```
cl-dns-zonedelete -n 10.0.0.0/24
```

Будет удалена обратная зона 0.0.10.in-addr.arpa

## Удаление MX записей для зоны

Пример.

```
cl-dns-zonedelete -m test.ru
```

Будут удалены MX записи для зоны test.ru (доменные имена почтовых серверов для зоны)

## Удаление A записи для зоны

Пример.

```
cl-dns-zonedelete -i ip -n test.ru
```

Будет удалена A запись для зоны test.ru (ip зоны)

## Создание DNS записи

Для создания DNS записи используется команда `cl-dns-recadd`.

Для создания записи необходимо создание master зоны в которую будет добавлена эта запись.

Для A записи (host.test.ru --> 10.0.0.4) необходимо создание master прямой зоны test.ru.

Для PTR записи (10.0.0.4 --> host.test.ru) необходимо создание master обратной зоны 0.0.10.in-addr.arpa

## Создание A записи

- Примеры создания записи:** Создание A записи и PTR записи. Сначала должны быть созданы прямая и обратная зоны, test.ru и 0.0.10.in-addr.arpa.
- Будет созданна запись в прямой зоне test.ru, host.test.ru соответствует 10.0.0.66.
  - Будет создана запись в обратной зоне 0.0.10.in-addr.arpa, 10.0.0.66 соответствует host.test.ru.

## СозданиеPTR записи

- Создание только A записи. Сначала должна быть создана прямая зона test.ru.
- Будет созданна запись в прямой зоне test.ru, host.test.ru - -ip 10.0.0.66

## Создание MX записи

- Создание A записи, MX записи и PTR записи
- Пример создания A записи, MX записи и PTR записи. Сначала должны быть созданы прямая и обратная зоны, test.ru и 0.0.10.in-addr.arpa.
- Будет создана запись в прямой зоне test.ru, host2.test.ru (приоритет 10), mail1.test.ru (приоритет 20)
  - Будет создана запись в обратной зоне 0.0.10.in-addr.arpa, 10.0.0.69 соответствует host2.test.ru

## Создание A записи и MX записи

- Пример создания A записи и MX записи. Сначала должна быть создана прямая зона test.ru.
- Будет созданна запись в прямой зоне test.ru, host2.test.ru соответствует 10.0.0.69.
  - Будет создана MX запись в прямой зоне test.ru, host2.test.ru соответствует двум почтовым серверам mail1.test.ru (приоритет 10), mail2.test.ru (приоритет 20)
  - Будет созданна запись в обратной зоне 0.0.10.in-addr.arpa, 10.0.0.69 соответствует host2.test.ru

## Создание A записи и MX записи

- Пример создания A записи и MX записи. Сначала должна быть создана прямая зона test.ru.
- Будет созданна запись в прямой зоне test.ru, host2.test.ru соответствует 10.0.0.69.
  - Будет создана MX запись в прямой зоне test.ru, host2.test.ru соответствует двум почтовым серверам mail1.test.ru (приоритет 10), mail2.test.ru (приоритет 20)

- net-misc/dahdi-2.4.1
- net-misc/dahdi-tools-2.4.1

## Установка сервера Asterisk

Для начала работы нам необходимо установить в систему Calculate Directory Server пакет сервера Asterisk и сопутствующие пакеты.

### Конфигурация USE-флагов для устанавливаемых пакетов

Создадим файл /etc/poportage/package.use/asterisk --- в нем мы пропишем все необходимые USE-флаги для нужных нам пакетов.

```
net-misc/alsa caps iconv jabber ldap samples speex ssl vorbis
dahdi span
net-misc/asterisk-core-sounds alaw g722 g729 gsm siren14 siren7 sln16 ulaw
wav
net-misc/asterisk-extra-sounds alaw g722 g729 gsm siren14 siren7 sln16 ulaw
wav
net-misc/asterisk-moh-opsound alaw g722 g729 gsm siren14 siren7 sln16 ulaw
media-libs/speex sse ogg
```

Для пакета net-misc/asterisk мы добавляем флаги dahdi (для работы с платой телефонии), span (для факсов) и vorbis (оционально для поддержки кодека vorbis). У остальных пакетов мы прописываем поддержку всех доступных кодеков.

### Установка пакетов

Теперь, когда мы установили необходимые USE-флаги, приступим к установке.  
Вводим команду:

```
emerge -a asterisk dahdi dahdi-tools
```

Мы получим такой список пакетов для установки:

These are the packages that would be merged, in order:

```
Calculating dependencies... done!
[ebuild N       ] net-libs/libpri-1.4.11.4
[ebuild N       ] sys-libs/slang-2.2.2 USE="pcre png readline zlib -cj"
[ebuild N       ] dev-libs/iksemel-1.3 USE="gnutls"
[ebuild N       ] media-libs/spandsp-0.0.6_pre12-r1 USE="mmx sse sse3
-doc (-fixed-point) -static-libs
[ebuild N       ] net-misc/dahdi-2.4.1 USE="flash"
[ebuild N       ] dev-libs/newt-0.52.12 USE="gpm nls -tc1"
[ebuild N       ] net-misc/dahdi-tools-2.4.1 USE="-ppp"
[ebuild N       ] net-misc/asterisk-1.6.2.17.3 USE="alsa caps dahdi iconv
jabber ldap samples speex ssl vorbis -doc -freetds -lua -newt -oss
-postgres -radius -snmp -sqlite"
[ebuild N       ] net-misc/asterisk-extra-sounds-1.4.11 USE="alaw g722
g729 gsm siren14 siren7 sln16 ulaw wav" LINGUAS="fr"
[ebuild N       ] net-misc/asterisk-moh-opsound-2.03 USE="alaw g722 g729
gsm siren14 siren7 sln16 ulaw wav" [ebuild N       ] net-misc/asterisk-core-
sounds-1.4.19 USE="alaw g722 g729 gsm siren14 siren7 sln16 ulaw wav"
LINGUAS="fr"
```

Would you like to merge these packages? [Yes/No]

## Настройка репликации сервисов

Добавление samba сервера в репликацию и указание серверов производится командой  
cl-replication -r <полные имена серверов через запятую> samba

Для получения полного имени сервера используйте команду hostname -f

При указании списка серверов репликации samba сервиса, для них автоматически добавляется репликация Unix сервера.

### Перенос настроек репликации на другие серверы

Для того, чтобы перенести настройки репликации на другие серверы необходимо:  
\*на сервере, где настроена репликация, сделать резервную копию настроек:

```
cl-backup
scp /var/calculcate/server-backup/ldap/<последний бэкап>.tar.bz2
root@otherserver:/var/calculcate/server-backup/ldap/
```

### Применение настроек репликации на других серверах

Для применения настроек репликации на других серверах необходимо выполнить на них команду:  
cl-rebuild -rrep1

### Отключение репликации

Для отключения репликации сервиса на сервере используется команда:  
cl-replication -off <сервис>

### Поддержка со стороны клиента

Репликацию поддерживает пакет calculate-client начиная с версии 2.0.13. Последовательность действий клиентом:

1. при входе в сеанс клиент синхронизирует профиль с текущим сервером;
2. определяет включен ли репликация на сервере;
3. если включена репликация, считывает из LDAP информацию о местоположении последней копии профиля и при необходимости синхронизирует профиль пользователя с удаленным сервером;
4. при выходе из сеанса профиль записывается на текущий сервер;
5. в случае успешной синхронизации с удаленным сервером, делается пометка в LDAP записи (реплицируемой на все сервера), о новом расположении последней копии профиля.

## Настройка репликации почтовых серверов

6. Настройка репликации почтовых серверов
7. Определения
8. Схемы использования репликации
9. 1. Почтовый домен без использования глобальной сети
10. 2. Почтовый домен с использованием глобальной сети
11. 3. Почтовый домен с почтовым реестром
12. Настройка и запуск репликации
13. Резервное копирование

- 14. Настройка репликации сервисов
- 15. Перенос настроек репликации на другие серверы
- 16. Применение настроек репликации на других серверах
- 17. Пример создания почтового ящика
- 18. Описание реализации

## Настройка Asterisk сервера

- Настройка Asterisk сервера
- Постановка задачи
- Установка сервера Asterisk
- Конфигурация USE-флагов для устанавливаемых пакетов
- Установка пакетов
- Базовая настройка SIP
- Создание учетных записей
- Базовая настройка плана набора
- Конфигурация внутренних вызовов
- Настройка связи между двумя серверами и вызовов между ними
- Установка связи между двумя Asterisk-серверами по протоколу SIP
- Конфигурация плана набора для связи между серверами
- Настройка платы Digium AE804E и конфигурация плана набора
- Настройка системы для работы с платой телефонии
- Объяснение понятий FXO и FXS
- Настройка FXO-каналов на Asterisk-сервере
- Конфигурация плана набора для приема вызовов
- Создание очередей вызова
- Установка своей музыки (сообщения) ожидания (Music On Hold, MOH)
- Конфигурация плана набора для совершения исходящих вызовов
- Заключение

Поддержка почтовой репликации включена в пакет calculate-server начиная с версии 2.0.9.

### Определения

Инtranet - в отличие от сети интернет, это внутренняя частная сеть организации. Как правило, инtranet -- это интернет в миниатюре, который построен на использовании протокола IP для обмена и совместного использования некоторой части информации внутри этой организации.

Почтовый домен - это совокупность почтовых ящиков, групп и списков рассылки, идентифицируемая единственным доменным именем. Это имя называется основным именем почтового домена. Все почтовые ящики, группы и списки рассылки, относящиеся к одному почтовому домену имеют в адресе электронной почты общую часть после символа "@".

Репликация почтовых серверов - служит для возможности создания почтовых доменов внутри сети инtranet, состоящей из нескольких почтовых серверов. Серверы, участвующие в репликации, могут выступать в роли почтового сервера либо почтового релея.

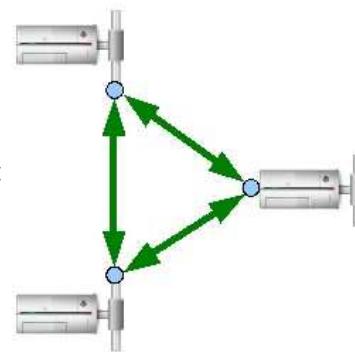
Почтовый релей --- сервер, занимающийся получением/пересылкой электронной почты. Релей обеспечивает прием письма, временное хранение (от нескольких минут до недели), пересылку сообщения почтовому серверу (или следующему почтовому релею).

Почтовый сервер - сервер, который может выполнять не только функции почтового релея, но также хранит пользовательскую почту и предоставляет доступ к этой почте по протоколам POP3 и/или IMAP.

### Схемы использования репликации

Ниже приведены три схемы настройки репликации почтовых серверов. На приведенных схемах, синими стрелками обозначена входящая из интэрнета почта, фиолетовыми - исходящая в интэрнет, зелеными - пересылка почты внутри сети инtranет.

### 1. Почтовый домен без использования глобальной сети



#### Описание

В данной схеме почтовые серверы осуществляют пересылку почты только в пределах сети инtranет. Почтовые сервера объединены в виртуальный почтовый домен (на примере: maildomain.org), благодаря чему все почтовые адреса выглядят как: "<имя почтового ящика>@maildomain.org".

#### Настройка схемы

Для настройки заданной схемы:

- Для того чтобы настроить Calculate Directory Server в качестве IP-АТС используется Asterisk. Asterisk --- компьютерная АТС под лицензией GPL поддерживающая большое количество VoIP протоколов.
- Настройка Asterisk производится путем редактирования файлов находящихся в директории /etc/asterisk.

Разберем настройку Asterisk-сервера на примере несложной конфигурации.

### Постановка задачи

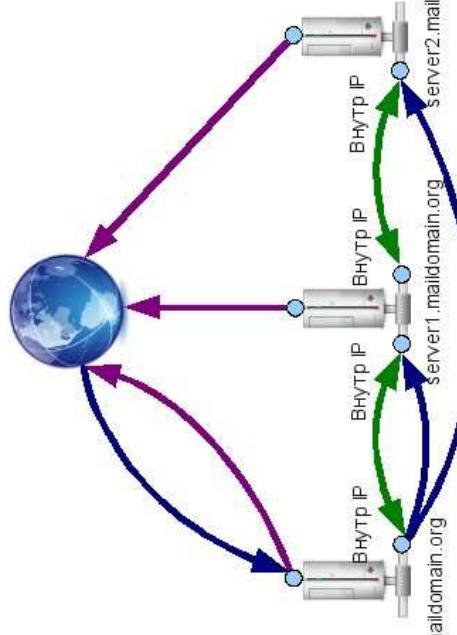
- Мы имеем один настроенный Asterisk-сервер в первом офисе, нам надо настроить второй сервер для нового офиса
- На наш новый сервер мы получаем две телефонные линии по аналогу
- Одна линия используется для телефонных разговоров, вторая для факсов
- В сервере установлена плата Digium AE804E --- PCIe-плата на 8 портов, в которую установлен один модуль на 4 FXO-порта и модуль экспандирования
- Имеющийся сервер находится в Санкт-Петербурге, новый будет в Москве
- Сервера должны быть связаны между собой и иметь единую систему нумерации внутренних номеров
- На московском сервере должна быть следующая схема очередей звонков при поступающем вызове на основную линию:

- Поступил звонок → пронгрываем приветствие и предлагаем набрать добавочный номер → если добавочный номер не был набран, то переводим звонок на секретаря в СПБ → если секретарь не ответил, то переводим звонок на абонентов московского сервера → если московские абоненты не ответили, то звонок на петербургских абонентов.
- При звонках поступающих на факсовую линию --- они должны адресоваться на факс
- Для подключения факса по SIP нам необходим FXS-шлюз, например, Linksys SPA2102
- Исходящие звонки должны совершаться по первой свободной линии. Если основная телефонная линия занята, то звонок идет через факсовую.
- Звонки с московского сервера в Санкт-Петербург (код 812) должны проходить через сервер в Петербурге

- Все нижеприведенные настройки тестируются и применяются для следующих версий пакетов:
- net-misc/asterisk-1.6.2.17.3

- выполните настройку на одном из почтовых серверов ([Unix](#) и [Samba](#)) сервисов.
- настройте репликацию Unix и Mail сервисов, указав полные имена всех почтовых серверов, участвующих в репликации.
- установите настройки репликации на остальные почтовые серверы участвующие в репликации.

## 2. Почтовый домен с использованием глобальной сети



Определяем по новым правила, по которым трафик будет отнесен в тот или иной класс. Правим /etc/shorewall/mangle:

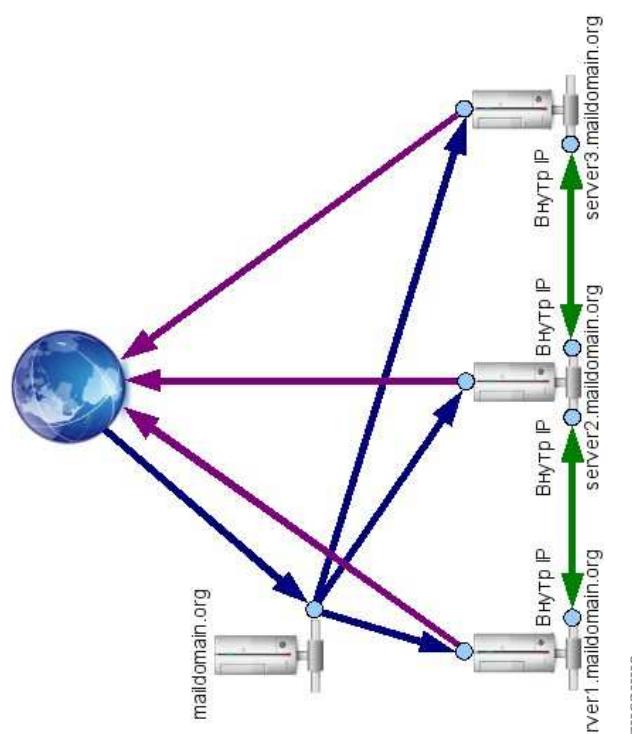
```
#####
#MARK SOURCE DEST PORT(S) SOURCE PORT(S)
USER LENGTH TOS CONNBYTES PROTO DEST PORT(S) PORT(S)
# SIP internet
CLASSIFY(10:101) 0.0.0.0/24 0.0.0.0/24 udp 5060 5060
CLASSIFY(10:101) 0.0.0.0/24 0.0.0.0/24 udp - -
CLASSIFY(10:101) 0.0.0.0/24 0.0.0.0/24 all - -
# SIP tunnel
CLASSIFY(30:301) 0.0.0.0/24 0.0.0.0/24 sip 5060 5060
CLASSIFY(30:301) 0.0.0.0/24 0.0.0.0/24 udp - -
CLASSIFY(30:301) 0.0.0.0/24 0.0.0.0/24 all - -
# RDP tunnel
CLASSIFY(20:201) 192.168.3.0/24 192.168.1.0/24 udp 5060 5060
CLASSIFY(30:302) 192.168.1.0/24 192.168.3.0/24 udp - -
CLASSIFY(10:102) 192.168.3.0/24 192.168.1.0/24 udp - -
CLASSIFY(30:302) 192.168.1.0/24 192.168.3.0/24 udp - -
CLASSIFY(10:102) 192.168.3.0/24 192.168.1.0/24 all - -
# high priority FTP
CLASSIFY(20:203) 0.0.0.0/24 0.0.0.0/24 all - -
CLASSIFY(40:403) 0.0.0.0/24 0.0.0.0/24 ftp all - -
# low priority http, https
CLASSIFY(20:204) 0.0.0.0/24 0.0.0.0/24 tcp http, https
CLASSIFY(40:404) 0.0.0.0/24 0.0.0.0/24 tcp http, https
```

### Настройка схемы

Настройка заданной схемы выполняется аналогично первой схеме.

```
#####
#MARK SOURCE DEST PORT(S) SOURCE PORT(S)
USER LENGTH TOS CONNBYTES PROTO DEST PORT(S) PORT(S)
# SIP internet
CLASSIFY(10:102) 192.168.3.0/24 192.168.1.0/24 udp 5060 5060
CLASSIFY(30:302) 192.168.1.0/24 192.168.3.0/24 udp - -
CLASSIFY(10:102) 192.168.3.0/24 192.168.1.0/24 udp - -
CLASSIFY(30:302) 192.168.1.0/24 192.168.3.0/24 udp - -
CLASSIFY(10:102) 192.168.3.0/24 192.168.1.0/24 all - -
# SIP tunnel
CLASSIFY(20:201) 192.168.3.0/24 192.168.1.0/24 tcp 3389 3389
CLASSIFY(40:401) 192.168.1.0/24 192.168.3.0/24 tcp 3389 3389
CLASSIFY(20:201) 192.168.3.0/24 192.168.1.0/24 tcp - -
CLASSIFY(40:401) 192.168.1.0/24 192.168.3.0/24 tcp - -
# high priority FTP
CLASSIFY(20:203) 0.0.0.0/24 0.0.0.0/24 all - -
CLASSIFY(40:403) 0.0.0.0/24 0.0.0.0/24 ftp all - -
# low priority http, https
CLASSIFY(20:204) 0.0.0.0/24 0.0.0.0/24 tcp http, https
CLASSIFY(40:404) 0.0.0.0/24 0.0.0.0/24 tcp http, https
```

### 3. Почтовый домен с почтовым релеем



Описание

Особенностью схемы является наличие почтового релея, который берет на себя функции переадресации почты приходящей с внешних адресов. Благодаря репликации, почтовый сервер всегда знает хосты серверов всех почтовых ящиков обслуживаемых доменов.

Отправка писем во внешние адреса осуществляется каждым сервером минуя почтовый релей.

#### Настройка схемы!

Для настройки заданной схемы:

- выполните настройку на одном из почтовых серверов ([Linux](#) и [Samba](#)) сервисов.
- настройте репликацию Unix и Mail сервисов, указав полные имена всех почтовых серверов участвующих в репликации, за исключением почтового релея, на котором репликация Unix сервиса будет отсутствовать.
- установите настройки репликации на остальные почтовые серверы участвующие в репликации.

## Настройка и запуск репликации

### Резервное копирование

Перед тем как начинать настраивать почтовую репликацию, рекомендуется сделать резервную копию настроек, чтобы в случае некорректных действий не потерять данные:

```
c1-backup
```

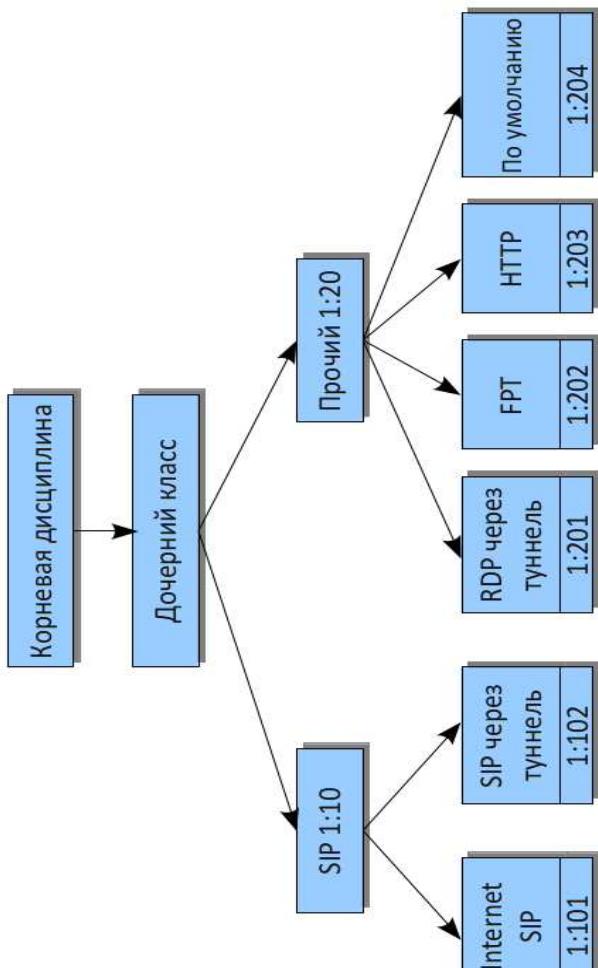
Для восстановления данных из последней резервной копии используйте команду:

```
c1-backup -r
```

### Настройка реплицирования сервисов

Добавление mail сервиса в репликацию и указание серверов производится командой

В приведенной настройке есть особенность: так как все классы подчинены напрямую корневой дисциплине, то при отсутствии sip трафика допустим RDP и http в сумме могут занять весь канал. Если необходимо, чтобы канал выделенный под sip трафик не занимался другими трафиками - необходимо использовать иерархическую структуру подобно этой:



Для того, чтобы использовать подобную классификацию необходимо указать classify параметр для каждого интерфейса. Модифицируем /etc/shorewall1/tc/devices:

```
#####
#NUMBER:           IN-BANDWIDTH      OPTIONS          REDIRECTED
#INTERFACE          -                OUT-BANDWIDTH    INTERFACES
eth0               -                5mbit          classify
eth1               -                5mbit          classify
```

Модифицируем /etc/shorewall1/tc/classes в соответствии со структурой представлений выше.

```
#####
#INTERFACE:CLASS   MARK   RATE:   CEIL   PRIORITY
OPTIONS          # INBOUND          # SIP
#INBOUND          # SIP
#NUMBER:          1:10   eth1:10:101  -       1256kbit 1
#INTERFACE:        1:10   eth1:10:102  -       256kbit  1
#OPTIONS          # OTHER            1:20   eth1:20:201  -       1mbit   1
#OTHER            1:20   eth1:20:202  -       3768kbit 2
#INBOUND          # SIP
#NUMBER:          1:20   eth1:20:201  -       1mbit   3
#INTERFACE:        1:20   eth1:20:202  -       256kbit  2
#OPTIONS          # SIP
#NUMBER:          1:20   eth1:20:201  -       3768kbit 2
#INTERFACE:        1:20   eth1:20:202  -       3768kbit 3
#OPTIONS          # SIP
#NUMBER:          1:20   eth1:20:201  -       3768kbit 2
```

```
cl-replication -r <полные имена серверов через запятую> mail
```

```
#####
#INTERFACE:CLASS          MARK      RATE:          CEIL      PRIORITY
#   OPTIONS                DMAX:UMAX
#   INBOUND
eth1           1       256kbit    256kbit    1
eth1           2       1mbit     1mbit     1
eth1           3       1mbit     3768kbit   3
eth1           4       256kbit    3768kbit   2
eth1           5       256kbit    3768kbit   4
eth1           256      256kbit    3768kbit   3

# OUTBOUND
eth1           1       256kbit    256kbit    1
eth1           2       1mbit     1mbit     1
eth1           3       1mbit     3768kbit   3
eth1           4       256kbit    3768kbit   2
eth1           5       256kbit    3768kbit   4
eth1           256      256kbit    3768kbit   3

# shoremall/mangle:
/etc/shoremall/mangle:
```

Определяем правила, по которым трафик будет отнесен в тот или иной класс. Правим /etc/shoremall/mangle:

```
#####
#MARK      SOURCE      LENGTH    TOS      DEST      CONNBYTES      PROTO      DEST      SOURCE      USER      PORT(S)      PORT(S)
#TEST
#   SIP internet
#   MARK(1) 0.0.0.0/24      0.0.0.0/24      udp      5060      5060      -
#   MARK(1) 0.0.0.0/24      0.0.0.0/24      udp      -        5060      -
#   MARK(1) 0.0.0.0/24      0.0.0.0/24      all     -        -        -
#   -
#   SIP tunnel
#   MARK(2) 192.168.3.0/24  192.168.1.0/24  udp      5060      5060      -
#   MARK(2) 192.168.1.0/24  192.168.3.0/24  udp      -        5060      -
#   MARK(2) 192.168.3.0/24  192.168.1.0/24  udp      -        5060      -
#   MARK(2) 192.168.3.0/24  192.168.1.0/24  all     -        -        -
#   -
#   RDP tunnel
#   MARK(3) 192.168.3.0/24  192.168.1.0/24  tcp      3389      3389      -
#   MARK(3) 192.168.1.0/24  192.168.3.0/24  tcp      -        3389      -
#   MARK(3) 192.168.3.0/24  192.168.1.0/24  tcp      -        3389      -
#   MARK(3) 192.168.1.0/24  192.168.3.0/24  tcp      -        3389      -
#   -
#   high priority FTP
#   MARK(4) 0.0.0.0/24      0.0.0.0/24      all     -        -        -
#   -
#   low priority http, https
#   MARK(5) 0.0.0.0/24      0.0.0.0/24      tcp      http,https
```

```
cl-replication -r <полные имена серверов через запятую> mail

Для получения полного имени сервера используйте команду:
hostname -f
```

При репликации только mail сервисы все перечисленные серверы будут почтовыми релейями. Для того чтобы сделать их (все или некоторые) почтовыми серверами, необходимо добавить их в репликацию unix сервиса. Это делается командой:

```
cl-replication -r <полные имена серверов через запятую> unix
```

## Перенос настроек репликации на другие сервера

Для того, чтобы перенести настройки репликации на другие серверы необходимо:

- на сервере, где настроена репликация, сделать резервную копию настроек:
- scp /var/calculcate/server-backup/ldap/<последний бэкап>.tar.bz2 root@otherserver:/var/calculate/server-backup/ldap/

## Применение настроек репликации на других серверах

Для применения настроек репликации на других серверах необходимо выполнить на них команду cl-rebuild -rrep1

## Пример создания почтового ящика

После настройки репликации серверов, вы можете создавать пользователей на любом из почтовых серверов. Для этого используйте команду cl-useradd с параметром "-e" (альтернативный почтовый адрес), где в качестве параметра следует указывать почтовый домен.

Пример:

```
cl-useradd -e user1@maildomain.org user1 mail
```

## Описание реализации

Репликация mail сервиса заключается в том, что реплицируются только почтовые алиасы. Аlias --- запись (в данном случае в LDAP), которая говорит о том: какие альтернативные почтовые адреса соответствуют реальным почтовым адресам (в данном случае один реальный почтовый адрес). При отправке письма почтовый сервер просматривает по LDAP, принаследжит ли адресат письма текущему почтовому домену. Если принадлежит, то по записи LDAP определяется почтовый сервер получатель этого письма, и письмо отправляется на определенный почтовый сервер в интернете. Если же адресат не принадлежит текущему почтовому домену, то письмо отправляется в интернет.

## Настройка сервера шлюза

- Настройка сервера шлюза
- Используемый программный пакет
- Программные пакеты расширяющие функционал
- Настройки файрвола
- Объявление зон
- Связывание интерфейса с определенной зоной

- Связывание IP-адресов с определенной зоной
- Задание действий для пакетов по умолчанию
- Добавление маскарадинга
- Добавление правил
- Определение туннелей
- Добавление запуска Shorewall при загрузке
- Настройка управления пропускной способностью
- Включение управления трафиком на интерфейсе.
- Определение классов для трафика
- Настройка маркировки трафика
- Примеры настроек
- Пример настройки шлюза
- Добавление ip-телефонии
- IPSEC туннели

#### • Пример управления трафиком

Для настройки Calculate Directory Server в качестве шлюза используется Shorewall. Shorewall или более точно Shorewall Firewall --- инструмент для настройки файрвола (межсетевого экрана). Технически является надстройкой над подсистемой Netfilter (iptables/iptables) ядра Linux и обеспечивает упрощённые методы конфигурирования данной подсистемы. Он предоставляет более высокий уровень абстракции для отыскания правил работы файрвола.

Программа не является демоном, то есть не работает постоянно. Правила хранятся в текстовых файлах, при запуске shorewall считывает свои файлы конфигурации и преобразует их в настройки понятные iptables/iptables, после чего данные настройки файрвола могут действовать до перезапуска операционной системы.

## Используемый программный пакет

ne -t firewall/shorewall  
ne -t -firewall/shorewall\_1

Для включения примеров и документации необходимо собрать пакет с USE флагом "doc".

### Программные пакеты расширяющие функционал

- net-firwall-xtables-addons - расширения не добавленные в kernel/iptables (для определения p2p трафика)
- net-misc/17-filter-userspace - классификация пакетов по содержимому (для определения трафика по содержимому)
- net-misc/linux-igd - для добавления функционала UPnP

## Настройки файрвола

- **Зоны**

Shorewall видит сеть, в которой он работает, как состоящую из набора зон (zones). Поэтому настройка начинается с определения одной или нескольких зон в файле /etc/shorewall/zones. Зоны представляют собой отдельные IP-адреса хостов, подсети, или входящие/исходящие пакеты для конкретного интерфейса. Они могут относиться к внешней сети, внутренней сети и DMZ.

#### • Интерфейсы

Зоны распознаются либо по подключенному к ним сетевому интерфейсу, определенному в файле /etc/shorewall/interfaces, либо по IP-адресу подсети, указанному в файле /etc/shorewall/hosts. У одной зоны может быть несколько интерфейсов, а у одного интерфейса -- несколько зон. Заметим, что Shorewall рассматривает систему файервола как свою собственную зону.

#### • Действия

Описываем туннель: ipsec, туннель проходит через зону net, IP адрес удаленного шлюза. Записываем в /etc/shorewall/tunnels

```
#TYPE ZONE GATEWAY GATEWAY ZONE
ipsec net 1.2.3.5

#INTERFACE : DEST SOURCE ADDRESS PROTO PORT (S)
#IPSEC MARK USER/
#GROUP
eth0:192.168.3.0/24 192.168.1.0/24 1.2.3.4 # Измененная строка
```

Отключаем маскарадинг пакетов отправляемых из локальной сети (loc) в удаленную подсеть (vpn).

Изменяем /etc/shorewall/masq:

```
#####
#INTERFACE : DEST SOURCE ADDRESS PROTO PORT (S)
#IPSEC MARK USER/
#GROUP
eth0:192.168.3.0/24 192.168.1.0/24 1.2.3.4 # Измененная строка
```

Описываем политику, разрешая соединения vpn с локальной зоной loc. Изменяем /etc/shorewall/policy:

```
#####
#SOURCE DEST POLICY LOG LEVEL LIMIT: CONNLIMIT: MASK
#phone net DROP # Новая строка
vpn loc ACCEPT
loc all ACCEPT
$FW all ACCEPT
net all DROP
all all REJECT info
```

## Пример управления трафиком

Есть канал шириной 5 мбит, необходимо разбить его следующим образом:

- SIP из интернета 256кбит гарантированно и максимально, максимальный приоритет
- SIP по туннелю 1мбит гарантированно и максимально, максимальный приоритет
- RDP по туннелю 1мбит гарантированно и весь канал минус SIP максимально (5мбит-1мбит-256кбит) обычный приоритет
- http трафик - низкий приоритет, 256кбит гарантированно и весь канал минус SIP максимально
- ftp трафик - высокий приоритет, 256кбит гарантированно и весь канал минус SIP максимально
- оставной трафик обычный приоритет, 256кбит гарантированно и весь канал минус SIP максимально

Устанавливаем ширину входящего и исходящего канала. Правим /etc/shorewall/tcdevices:

```
#####
#NUMBER# #INTERFACE IN-BANDWIDTH OUT-BANDWIDTH OPTIONS REDIRECTED INTERFACES
eth0 - 5mbit -
eth1 -
```

Определяем классы трафика. Так как у нас eth0 смотрит в интернет, то исходящий трафик управляется на нем, eth1 смотрит в локальную сеть - входящий на нем. Правим /etc/shorewall/tcclasses:

Определив зоны, нужно задать действие по умолчанию в файле `/etc/shorewall/policy` (например, ACCEPT или DROP), применяя к трафику между каждой исходной зоной и зоной назначения.

- **Политики**

Наконец, в файле `/etc/shorewall/rules` определяются подробности исключений из политики, разрешающие доступ к заданным портам и т.д.

### Объявление зон

Объявление зон производится в файле `/etc/shorewall/zones`. Для того, чтобы объявить зону необходимо ее название (ZONE) и ее тип (TYPE) (`firewall`, `fw`) - стандартная зона).

Порядок описания зон важен, так как именно в таком порядке будут вызываться цепочки `iptables`, проверяющие пакеты направлений из одной зоны в другую. То есть если пропустит пакет адресованный файерволу, то в соответствии с вышеописанным примером пакет вначале проверяется на принадлежность зоне `net`, а затем уже зоне `loc`. Это правило является важным если одна из зон включает в себя другую. Например в зоне `loc` можно создать зону `phone`, которая представляет собой несколько ip адресов из зоны `loc`. Таким образом если вначале отписать зону `loc`, а затем `phone`, то получится, что политика описанная для зоны `phone` никогда не будет выполняться, так как трафик будет попадать в цепочки для `loc`. Данная проблема может быть решена указанием зоны `phone` как подзоны `loc` (`phone:loc` `loc` `iprule4`). Приимер настройки с двумя сетевыми картами:

- сам маршрутизатор (`fw`) обозначается как `firewall`
- локальная сеть (`loc`) использует `ipv4`

```
#####
#ZONE  TYPE          OPTIONS           IN      OUT      OPTIONS
#     fw           firewall          ipv4   ipv4
net
loc
#####
#ZONE  TYPE          OPTIONS           IN      OUT      OPTIONS
#     fw           firewall          ipv4   ipv4
loc
```

- Удаленный шлюз устанавливает ipsec туннель с удаленным шлюзом через интернет, подключая сегмент

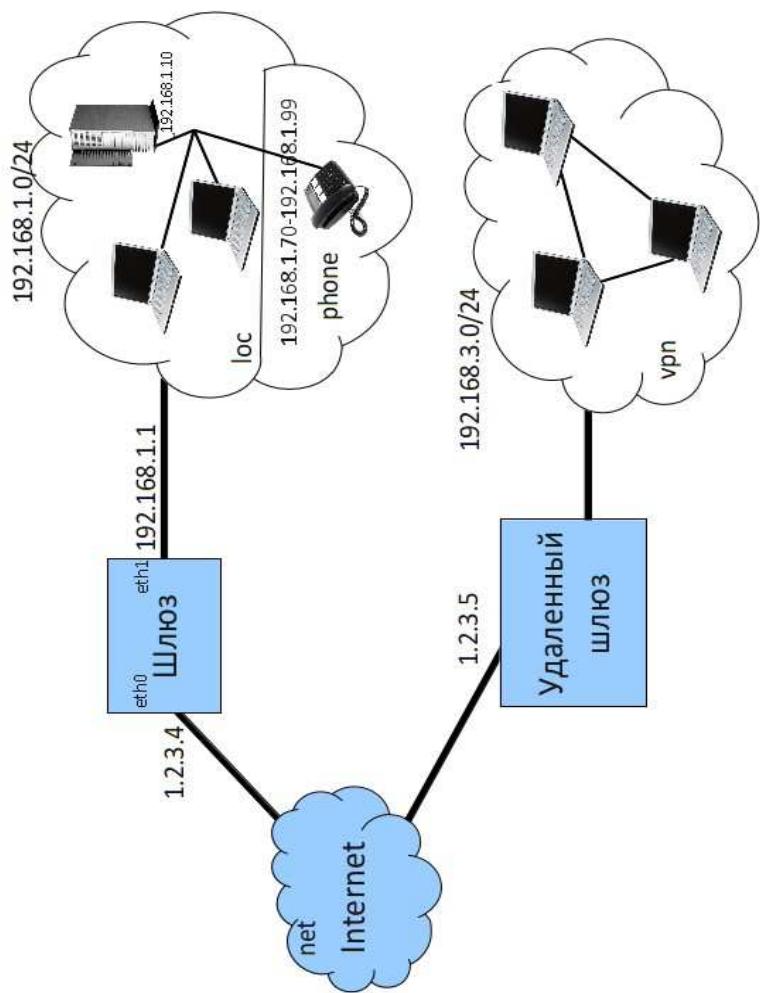
`192.168.3.0/24`

• Добавляем зону `vpn` для удаленной подсети. Изменяем `/etc/Shorewall/zones`.

```
#####
#ZONE  TYPE          OPTIONS           IN      OUT      OPTIONS
#     fw           firewall          ipv4   ipv4
vpn
net
phone
loc
#####
#ZONE  HOST(S)      OPTIONS
#     phone
vprn
eth1:192.168.1.70-192.168.1.99
eth0:192.168.3.0/24 # Новая строка
```

Определяем зону `vprn` (пакеты, поступающие `eth0` и находящиеся в подсети `192.168.3.0/24`). Изменяем `/etc/shorewall/hosts`

```
#####
#ZONE  HOST(S)      OPTIONS
#     phone
vprn
#####
#ZONE  HOST(S)      OPTIONS
#     phone
vprn
eth1:192.168.1.70-192.168.1.99
eth0:192.168.3.0/24 # Новая строка
```



Более подробно о настройке этого файла можно прочитать, выполнив `man shorewall-zones`.

### Связывание интерфейса с определенной зоной

Для связывания интерфейса с определенный зоной используется файл `/etc/shorewall/interfaces`, так же в этом файле можно указать какими фильтрами будут обрабатываться входящие пакеты.

Чтобы назначить интерфейс для определенной зоны, необходимо указать название этой зоны (ZONE), сам интерфейс (INTERFACE), широковещательный адрес (BROADCAST) (рекомендуется указать `detect`), а также три необходимости перечислить фильтры (OPTIONS).

Пример с двумя сетевыми картами:

В файле указываем, что `eth0` выходит в интернет (`net`), `eth1` в локальную сеть (`loc`). `tcpflags`, `routefilter`, `nosmurf`, `logmartialS` - применяем фильтры для входящего трафика.

- `tcpflags` - пакеты полученные этим интерфейсом проверяются на некорректное сочетание TCP флагов;
- `routefilter` - добавить anti-spoofing проверку (один из видов атак);
- `nosmurf` - не пропускать пакет с широковещательным исходящим адресом;
- `logmartialS` - логирование пакетов, которые содержат невозможный исходящий адрес.

```
#ZONE INTERFACE BROADCAST OPTIONS
net eth0 detect
tcpflags,routefilter,nosmurfs,logmartians
loc eth1 detect
tcpflags,routefilter,nosmurfs,logmartians
```

Начиная с версии 4.5.3 shorewall поддерживает второй формат в котором нет необходимости указывать BROADCAST

```
?FORMAT 2
#####
#ZONE INTERFACE OPTIONS
net eth0 tcpflags,routefilter,nosmurfs,logmartians
loc eth1 tcpflags,routefilter,nosmurfs,logmartians
```

Более подробно о настройке этого файла можно прочитать, выполнив man shorewall-interfaces.

### Связывание IP-адресов с определенной зоной

Для связывания IP-адресов или подсетей с зоной используется файл /etc/shorewall/hosts. Для связывания хостов/подсетей с зоной необходимо указать название зоны (ZONE), и сетевой интерфейс с подсетью или IP адресами (HOST).

Пример определения зон IP-адресов для туннеля с другим шлюзом, который подключен к сегменту 10.0.0.24.

```
#####
#ZONE HOST(S) OPTIONS
vpn eth0:10.0.0.0/24
```

Пример выделения IP-адресов для телефонов (множество IP адресов 192.168.0.60-192.168.0.80) в отдельную зону phone. Это может быть удобно для управления доступом по зонам, а не исключением.

```
#####
#ZONE HOST(S) OPTIONS
phone eth1:192.168.0.60-192.168.0.80
```

Более подробно о настройке этого файла можно прочитать, выполнив man shorewall-hosts.

### Задание действий для пакетов по умолчанию

Действие по умолчанию (политика), применяемое к трафику между каждой исходной зоной и зоной назначения указывается в файле /etc/shorewall/policy. Правило описывается следующим образом: исходная зона (SOURCE), зона назначения (DEST), правило по умолчанию (POLICY) (DROP,ACCEPT,REJECT), и если необходимо уровень логирования (например info).

Политика рассматривает пакеты, которые создают новые соединения. RELATED/ESTABLISHED пакеты (пакеты установленного соединения) по умолчанию пропускаются. Для примера установим следующие политики:

- разрешить доступ из локальной сети в интернет
- разрешить доступ из фаервола во всходу
- сбрасывать все пакеты пришедшие из интернета + вести лог

```
#ZONE TYPE OPTIONS
# fw firewall
net ipv4
phone ipv4 # Новая строка
loc ipv4
```

Выделим телефоны из зоны loc в зону phone. Поместим в /etc/shorewall/hosts.s

```
#####
### HOST(S)
#ZONE HOST(S)
phone eth1.192.168.1.70-192.168.1.99
```

Опишем политику запрещающую телефонам (phone) доступ к интернету (net). Изменим /etc/shorewall/policy

```
#####
#SOURCE DEST POLICY LOG LIMIT:
# phone net DROP LEVEL BURST
# loc all ACCEPT
$FW all ACCEPT
net all DROP
all all REJECT
info
```

Сделаем проброс SIP и IAX портов на 192.168.1.10. Добавим к /etc/shorewall/rules

```
#####
#ACTION SOURCE DEST PROTO DEST SOURCE
ORIGINAL RATE USER/ MARK CONNLIMIT TIME
HEADERS # DEST LIMIT GROUP
# SECTION ESTABLISHED
#SECTION RELATED
SECTION NEW
::: SMTP/DNAT net
# SIP DNAT # IAX DNAT
net net net
PORT PORT(S)
```

```
loc:192.168.1.10
loc:192.168.1.10 udp 5060
loc:192.168.1.10 udp 4569
```

### IPSEC туннели

Модифицируем сеть предыдущего примера, в соответствии с рисунком (добавление удаленный подсети):

- запретим выход в интернет с диапазона адресов 192.168.1.100-192.168.1.254
- перенаправим порты IMAP/IMAPS для доступа к почтового сервера из интернета

B /etc/shorewall/rules поместим записи:

```
#####
#SOURCE DEST POLICY LOG LIMIT: CONNLIMIT:
#
loc net ACCEPT $FW all ACCEPT net all DROP info all all REJECT info

More подробно о настройке этого файла можно прочитать, выполнив man shorewall-policy.

LEVEL BURST MASK

Добавление маскарадинга

Маскарадинг --- тип трансляции сетевого адреса, при которой адрес отправителя подставляется динамически, в зависимости от назначенного интерфейсу адреса.

Определение dynamic NAT (Masquerading) и Source NAT (SNAT) производится в /etc/shorewall/masq.

Для добавления маскарадинга к конкретному интерфейсу необходимо его указать (INTERFACE,DEST), затем указать подсеть, для которой будет применяться данный маскарадинг (SOURCE), исходящий адрес (ADDRESS) для SNAT. Если исходящий адрес не указывать, то будет использоваться динамич. NAT.

Для примера укажем, что для всех пакетов уходящих на eth0 с нашей внутренней сети (192.168.0.0/24) применять SNAT (24.56.78.21).

#####
#INTERFACE : DEST      SOURCE      ADDRESS      PROTO      PORT (S)
#IPSEC   MARK      USER/
#GROUP
eth0

192.168.0.0/24  24.56.78.21

GROUP

#SECTION RELATED
SECTION NEW
Ping/ACCEPT      net      $FW
SSH/ACCEPT       net      $FW
REJECT          loc:192.168.1.100-192.168.1.254 net
IMAP/DNAT        net      loc:192.168.1.10
IMAPS/DNAT       net      loc:192.168.1.10
SMTP/DNAT        net      loc:192.168.1.10

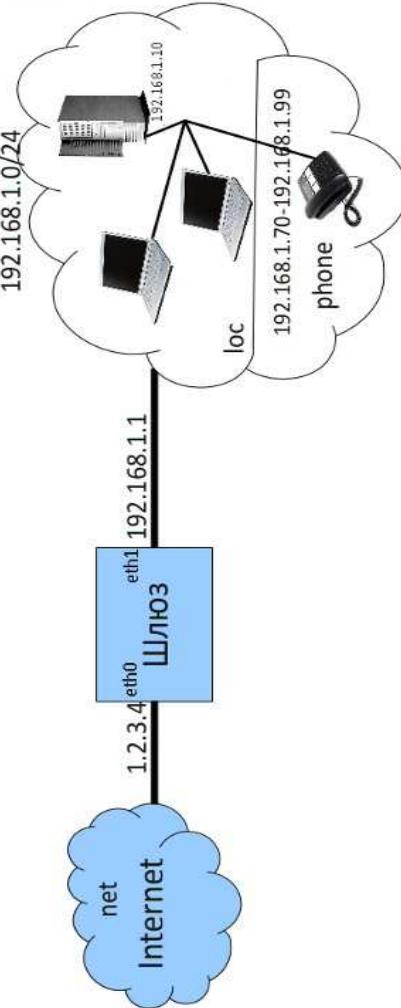
#
#SECTION ESTABLISHED
#SECTION RELATED
SECTION NEW
Ping/ACCEPT      net      $FW
SSH/ACCEPT       net      $FW
REJECT          loc:192.168.1.100-192.168.1.254 net
IMAP/DNAT        net      loc:192.168.1.10
IMAPS/DNAT       net      loc:192.168.1.10
SMTP/DNAT        net      loc:192.168.1.10

#
#SECTION ESTABLISHED
```

Разрешаем запуск shorewall, для этого устанавливаем параметр STARTUP\_ENABLED=yes в файле /etc/shorewall/shorewall.conf.

### Добавление ip телефонии

Модифицируем сеть предыдущего примера, добавив в нее ip телефоны:



- телефоны подключены к основной сети и их ip адреса находятся в диапазоне 192.168.1.1-192.168.1.10
- на сервер 192.168.1.10 устанавливается asterisk

Объявим новую зону phone для телефонов. Изменим /etc/shorewall/zones (зона phone должна быть объявлена обязательство перед зоной loc):

```
#####
#ZONE          INTERFACE  SUBNET
#loc           eth0      192.168.1.0/24
#loc           eth1      192.168.1.1
#phone         eth0      192.168.1.10
#phone         eth0      192.168.1.99
#Internet     eth0      192.168.1.1
```

При необходимости можно исключить NAT при отправке пакетов в некоторые подсети, например для настройки ipsec туннеля.

Для примера отключим NAT при отправки пакета в подсеть 192.168.2.0/24:  
eth0 !192.168.2.0/24 192.168.0.0/24 24.56.78.21

Более подробно о настройке этого файла можно прочитать, выполнив man shorewall-masq.

### Добавление правил

Правила необходимы для описания исключений из политик, применяемых к трафику. Правила помещаются в файл /etc/shorewall/rules.

Файл содержит три секции: RELATED,ESTABLISHED,NEW, соответствующие критериям состояния соединения. Критерий определяется при помощи критерия conntrack, который может классифицировать пакеты на основании их отношения к соединениям. В частности, состояние NEW позволяет выделять только пакеты, открывающие новые соединения, состояние ESTABLISHED --- пакеты, принадлежащие к установленным соединениям (например, соединению RELATED соответствует пакеты, открывавшие новые соединения, логически связанные с уже установленными (например, соединение данных в пассивном режиме FTP). Состояние INVALID означает, что принадлежность пакета к соединению установить не удалось. (INVALID также относится к секции NEW). Как уже описывалось ранее политика определяет действие для секции NEW, к RELATED и ESTABLISHED по умолчанию применяется ACCEPT.

Ниже представленные примеры записываются в секцию NEW.