InfraMedia Manual

Операционная система медиасерверных ПАК

Максим Лапшин

© Эрливидео *2026*

Table of contents

© Эрливидео 2026

1. Продукты

© Эрливидео 2026

2. InfraMedia

2.0.1 InfraMedia

Flussonic InfraMedia (далее — InfraMedia) — операционная система аппаратных медиасерверов. Это линукс с неизменяемым корнем, который поставляется одним образом вместе с прикладной программой, и демон управления, который настраивает машину по HTTP.

Назначение: превратить сервер в законченный прибор, который работает без администрирования. Машину настраивает не человек в консоли, а программа, ради которой прибор и поставлен, — через программный интерфейс конфигуратора.

Устройство системы описано на странице техническая архитектура, установка — на странице установка, обновление и откат — на странице обновление прошивки.

Зачем неизменяемая система

Системные и исполняемые файлы InfraMedia нельзя изменить во время эксплуатации: корень собран из готовых образов squashfs и смонтирован только для чтения. Отсюда всё остальное.

- **Машина в стойке ровно такая, какой её выпустили.** Изменить или подменить системный файл невозможно, поэтому невозможны и накопленные за годы правки, о которых уже никто не помнит. Две машины одной версии одинаковы.
- **Операционная система и прикладная программа едут одной поставкой.** Они собраны и проверены вместе, а стык между ними держит поставщик, а не эксплуатация.
- **Версии при этом раздельные.** Прикладная программа лежит в собственном слое, и обновить её, не трогая систему, — обычное дело; обратное тоже верно.
- **Обновление встаёт целиком или никак.** Новая версия кладётся рядом с работающей и включается одной строкой в загрузчике. Состояния «наполовину обновились» не бывает, а откат — это та же строка, возвращённая обратно.
- **Внешние репозитории не нужны.** В рабочем режиме пакеты не ставятся: всё, что есть в системе, приехало в образе.

Из чего состоит система

Часть	Назначение
Базовый слой	линукс с systemd, демон управления, веб-интерфейс, модули ядра
Ядро и initrd	ядро собственной сборки под целевую машину и сценарий начальной загрузки
Конфигуратор	демон управления: HTTP API, настройка машины через systemd
Веб-интерфейс	обзор машины, интерфейсы, система, обновления, питание
Слой прикладной программы	медиасервер (например, Mcaster) отдельным слоем поверх базового
Слой `postgres`	общий компонент для продуктов, которым нужна база

Две версии

У операционной системы свой номер версии, у прошивки сервера — свой. Прошивка это InfraMedia такой-то версии плюс прикладная программа такой-то версии; обе записаны внутри загрузочного файла и видны в интерфейсе машины. Номер операционной системы не меняется от того, что обновилась прикладная программа.

© Эрливидео 2026

Навигатор по документации

Эта документация поможет вам:

- Разобраться, как устроена система
- Установить прошивку на сервер с флешки
- Обновить прошивку и откатиться на прежнюю версию

© Эрливидео 2026

2.1 Техническая архитектура

InfraMedia собирается образом и не изменяется на машине. Корень файловой системы составлен из слоёв, каждый слой — образ squashfs, упакованный при сборке. Всё, что машина пишет, лежит на отдельных разделах.

2.1.1 Слои прошивки

Слой	Содержимое
Базовый	линукс с systemd, демон управления, веб-интерфейс, модули ядра
`postgres`	сервер базы данных для продуктов, которым он нужен
Прикладной	медиасервер: исполняемые файлы и то, что им нужно сверх базового слоя

Слой прикладной программы — приращение над базовым: в него попадает только разница с базовым слоем. Библиотеки, которые уже есть внизу, наверх не едут.

2.1.2 Разделы диска

Установщик размечает диск на три раздела.

- **FIRMWARE** — загрузочный раздел EFI: загрузочные файлы версий, образы слоёв, меню загрузчика.
- **SETTINGS** — настройки машины: хранилище конфигуратора и оверлей `/etc`.
- **VAR** — изменяемые данные: журналы, каталог данных базы, рабочие файлы прикладной программы.

Корень при этом остаётся только для чтения. Домашний каталог `root` — tmpfs и живёт до перезагрузки; то, что должно её пережить, лежит на разделе изменяемых данных.

2.1.3 Загрузка

Машина грузится через systemd-boot. Ядро, initrd, командная строка и **манифест состава** склеены в один загрузочный файл. Это сделано ради одного свойства: выбор загрузочной записи выбирает ядро, initrd и список слоёв вместе, рассинхронизировать их нельзя. Две версии прошивки на диске — это две записи в меню, и откат возвращает состав целиком.

Порядок загрузки такой.

1. Прошивка UEFI передаёт управление загрузчику, тот запускает загрузочный файл версии, выбранной в меню.
2. initrd читает манифест, находит на загрузочном разделе образы слоёв и **сверяет их sha256** с манифестом.
3. Слои подключаются и собираются в корень; сверху подключается оверлей `/etc` с раздела настроек.
4. Управление передаётся systemd, который поднимает демон управления и прикладную программу.

initrd — сценарий на busybox: он не зависит от прикладной программы и собирается один раз на архитектуру.

2.1.4 Конфигуратор

Конфигуратор — демон управления машиной. Он слушает HTTP на порту 8090 и отдаёт программный интерфейс: обзор машины, сетевые интерфейсы, имя машины и серверы времени, прошивка, перезагрузка и выключение. Он же раздаёт веб-интерфейс — отдельного порта под интерфейс нет.

Воздействие на систему идёт через вызовы штатных программ и запись конфигурационных файлов — `ip`, `networkctl`, `resolvectl`, `hostnamectl`, `timedatectl`, `systemctl` и файлы в `/etc/systemd/`.

© Эрливидео 2026

Хранилище настроек

Единственный источник правды о том, как настроена машина, — хранилище настроек конфигуратора. Файлы systemd рендерятся из него и **никогда не читаются обратно**.

Из этого следуют два практических правила.

- Править сгенерированные файлы руками бессмысленно: следующий рендеринг вернёт заданное значение.
- Заданное значение и фактическое состояние — разные вещи. Заданное берётся из хранилища, фактическое конфигуратор снимает с работающей системы машиночитаемым выводом системных утилит, и они могут расходиться.

Сессия изменения

Изменения сети применяются в режиме «подтверди или откачу». Изменение открывает сессию, и пока она открыта, хранилище заморожено целиком. Если подтверждения в срок не пришло, хранилище возвращается к сохранённой копии, а настройки — к прежним. Открытых сессий не более одной.

Смысл механизма прикладной: ошибка в настройке интерфейса больше не означает потерю доступа к машине.

Ограниченный режим

Если хранилище прочитать не удалось — файл повреждён или его версия новее самого демона, — конфигуратор переходит в ограниченный режим: читающие операции работают, изменения отклоняются, рендеринг не выполняется. Машина продолжает работать на том, что уже отрендерено.

2.1.5 Профиль железа

Профиль железа описывает машины, на которые едет прошивка этой архитектуры: конфигурацию ядра с их драйверами и правила, приводящие физические интерфейсы к каноническим именам по пути устройства. Драйверы всех поддерживаемых машин встроены в одно ядро, и на любой из них грузится один и тот же образ.

Канонические имена назначаются по роли интерфейса — `manage` для управляющих, `streaming` для потоковых, с порядковым номером. Настраивать через программный интерфейс можно только интерфейс, получивший каноническое имя; физический интерфейс без него виден, но не настраивается.

2.1.6 Веб-интерфейс

Веб-интерфейс едет базовым слоем и раздаётся самим демоном — он обязан работать на машине, где прикладной программы нет вовсе или она не поднялась. Разделов пять: **Обзор**, **Интерфейсы**, **Система**, **Обновления**, **Питание**.

Экраны собраны отдельно от оболочки: они не несут ни темы, ни маршрутизатора, ни авторизации и не знают, по какому адресу лежит программный интерфейс. Поэтому у них два входа — приложение на порту конфигуратора и раздел внутри консоли прикладной программы, которая подгружает экраны с самой машины. Оператор настраивает шасси там же, где работает с медиасервером.

2.1.7 Две версии

У операционной системы и у прошивки сервера разные номера. Прошивка — это InfraMedia такой-то версии плюс прикладная программа такой-то; обе записаны в манифесте загрузочного файла. Прикладная программа собирается на опубликованных частях операционной системы, а не пересобирает её.

© Эрливидео 2026

3. Администратору

3.1 Запуск

3.1.1 Установка

Прошивка ставится на сервер с флешки-установщика. Установщик не задаёт вопросов и не ждёт нажатий: всё, что ему нужно, он берёт из текстового файла на самой флешке. Человек записывает образ, заполняет файл, втыкает флешку в сервер и включает питание.

Флешка-установщик

Образ флешки несёт ту же прошивку, что ляжет на диск машины, и два раздела.

- **INSTALLER** — загрузочный раздел с прошивкой.
- **CONFIG** — обычный раздел данных с файлом `install.txt`. macOS и Windows показывают его сами, загрузочный раздел они прячут.

Образ ни от каких секретов не зависит: он собирается с шаблоном `install.txt`, а заполняют файл уже на самой флешке, в любом редакторе. Переводы строк и метку порядка байтов, которые добавляют Windows и macOS, установщик отбрасывает сам.

Запишите образ на флешку и переткните её — появится том `CONFIG`.

Файл install.txt

Откройте `install.txt` на томе `CONFIG` и замените заглушки в угловых скобках.

Поле	Значение
`SALT`	соль парка: секрет, из которого выводится пароль `root` машины. Остаётся на флешке, на диск не попадает
`EDIT_AUTH_LOGIN`	имя учётной записи для консоли прикладной программы и настроек шасси
`EDIT_AUTH_PASSWORD`	её пароль
`LICENSE_KEY`	лицензионный ключ прикладной программы
`HOSTNAME`	имя машины
`INSTALL_DISK`	диск для установки. Если не указан, берётся самый большой диск, кроме самой флешки

Оставшаяся заглушка — это пустое значение. Без соли или без пароля установщик отказывает **до того, как тронет диск**.

Установка

Вставьте флешку в сервер и загрузитесь с неё — в меню загрузки это запись `UEFI: <флешка>`.

Дальше установщик работает сам:

1. размечает диск машины на разделы FIRMWARE, SETTINGS и VAR — **стирая его целиком**;
2. копирует на него прошивку;
3. оставляет на диске файл начальных значений: учётную запись консоли, лицензионный ключ и хеш пароля `root`;
4. печатает на консоли выбранный диск и аппаратный адрес машины;
5. выключает машину.

© Эрливидео 2026

Флешку можно не вынимать: при следующей загрузке начальный сценарий видит на диске ту же версию и грузит диск. Флешка с другой версией ставит её заново — так же делается и переустановка.

Пароль root

Пароль `root` каждой машины выводится из соли парка и аппаратного адреса, который напечатал установщик. Соль лежит только на установочном носителе, машине достаётся хеш, из которого пароль обратно не восстанавливается.

Отсюда главное свойство: пароль конкретной машины воспроизводим тем, у кого есть носитель и идентификатор машины, — без базы данных и без записей о самой машине.

Первая загрузка

После установки включите сервер. Машина поднимает демон управления, а за ним прикладную программу.

- Конфигуратор и его веб-интерфейс — на порту **8090** машины.
- Консоль прикладной программы — на её обычном порту.

Учётные данные берутся из файла, оставленного установщиком. Пароль превращается в хеш при первом заполнении хранилища настроек, и правка файла на уже установленной машине ничего не изменит.

> ⚠ **Без учётных данных в машину не войти**
>
> Если поля учётной записи в `install.txt` остались пустыми, демон отвергает **любой** пароль, а не только неверный. Учётных данных в хранилище нет, и появиться им неоткуда: машину придётся установить заново.

© Эрливидео 2026

3.2 Обслуживание

3.2.1 Обновление прошивки

Установленная машина обновляется без флешки. Пакет обновления встаёт **рядом** с работающей версией, а не поверх неё: настройки, данные и пароли обновление переживают, а прежняя версия остаётся на загрузочном разделе для отката.

Машина ничего не скачивает и не ставит сама. Установка обновления — всегда явный запрос: с экрана интерфейса, из программного интерфейса или руками с консоли.

Пакет обновления

Пакет — это архив с тем, что ложится на загрузочный раздел: загрузочный файл новой версии, образы её слоёв, запись меню загрузчика, описание продукта и манифест контрольных сумм. Рядом с пакетом публикуется его собственная контрольная сумма.

Канал обновлений

Откуда машина узнаёт о новых версиях — из индекса канала обновлений, адрес которого **вшит в сборку**. Машина читает индекс и перечисляет версии своего продукта и своей архитектуры; те, что уже стоят на разделе, помечены.

Сборка без канала о нём не знает вовсе, и ставить на неё можно только по адресу или файлом.

Установка

ИЗ ВЕБ-ИНТЕРФЕЙСА

Раздел **Обновления** первым делом называет две версии: какая работает и какая загрузится следующей. Рядом кнопка перезагрузки — когда следующая версия отличается от работающей, она подсвечена, и экран говорит, что именно сменится при перезагрузке.

Индекс канала читается при открытии интерфейса: значок пункта меню несёт метку, когда в канале есть версия, которой на разделе нет. Кнопка **Проверить обновления** перечитывает индекс.

ЧЕРЕЗ ПРОГРАММНЫЙ ИНТЕРФЕЙС

Ответ приходит сразу, установка идёт в фоне.

Поставить пакет файлом:

```
curl -u admin:пароль -H 'content-type: application/x-tar' \
    --data-binary @mcaster-<версия>-amd64.update.tar http://<машина>:8090/system/firmware
```

Поставить пакет по адресу:

```
curl -u admin:пароль -H 'content-type: application/json' \
    -d '{"url":"https://…/mcaster-<версия>-amd64.update.tar"}' \
    http://<машина>:8090/system/firmware/install
```

Посмотреть ход операции, установленные версии и ту, что загрузится следующей:

```
curl -u admin:пароль http://<машина>:8090/system/firmware
```

Перечислить версии канала обновлений:

```
curl -u admin:пароль http://<машина>:8090/system/firmware/available
```

Перезагрузиться в новую версию:

```
curl -u admin:пароль -X POST http://<машина>:8090/system/reboot
```


© Эрливидео 2026

Что происходит при установке

1. Демон распаковывает пакет на раздел изменяемых данных.
2. Сверяет каждый файл с манифестом контрольных сумм, а продукт, архитектуру и профиль железа — с работающими.
3. Только после этого перемонтирует загрузочный раздел на запись и копирует недостающее. Образы слоёв, общие с уже установленными версиями, не копируются.
4. Переводит запись загрузчика по умолчанию на новую версию.

После установки на разделе остаются работающая версия и новая. Остальные убираются вместе с образами слоёв, которые больше никому не нужны.

Новая версия начинает работать только после перезагрузки — установка сама машину не перезагружает.

Откат

Откат — это выбор прежней версии и перезагрузка:

```
curl -u admin:пароль -X POST http://<машина>:8090/system/firmware/<версия>/select
curl -u admin:пароль -X POST http://<машина>:8090/system/reboot
```

Удалить ненужную версию:

```
curl -u admin:пароль -X DELETE http://<машина>:8090/system/firmware/<версия>
```

Работающую и выбранную версии удалить нельзя.

Когда демона нет

Если демон не поднялся или до него не достучаться, тот же путь проходится руками с консоли под `root`. Программа `firmware-update` лежит в базовом слое, зависит только от оболочки и утилит слоя и делает ровно три вещи — скачать, переложить, переключить — с теми же проверками, что и демон. Она ничего не удаляет.

```
firmware-update install https://…/mcaster-<версия>-amd64.update.tar   # или путь к файлу
firmware-update list                    # версии на разделе: работающая и следующая
firmware-update select <версия>         # следующая загрузка с прежней версии
systemctl reboot
```

Чего пока нет

Автоматического отката при неудачной загрузке новой версии нет: если машина не поднялась, загрузочную запись по умолчанию возвращают руками из меню загрузчика. Машины ранних выпусков несут загрузочный раздел на одну версию, и обновление по месту туда не помещается — такие машины один раз переустанавливают с флешки.

© Эрливидео 2026