kickstart в NiceOS V

1. Введение

NiceOS V — это современная операционная система, оптимизированная для автоматизированного развертывания в виртуальных, облачных и локальных средах. Ее сердце — **Kickstart**, конфигурационный файл в формате JSON, который позволяет полностью автоматизировать процесс установки, от разметки дисков до настройки сети, пользователей и запуска Ansible-плейбуков. Этот инструмент создан для DevOps, системных администраторов и разработчиков, стремящихся к воспроизводимости, масштабируемости и минимальному ручному вмешательству.

В отличие от традиционных Kickstart-файлов в формате `.ks`, используемых в других Linux-дистрибутивах, NiceOS V применяет JSON. Этот формат легко читаем, валидируем и интегрируется с CI/CD-пайплайнами, что делает его идеальным для автоматизации инфраструктуры. JSON позволяет генерировать конфигурации из шаблонов (например, Jinja2), упрощая массовое развертывание.

Преимущества Kickstart в NiceOS V:

- **Полная автоматизация:** Настройка дисков, пакетов, сети и пользователей без ручного ввода.
- **Гибкость:** Поддержка LVM, Btrfs, шифрования, Docker и Ansible для любых сценариев.
- **Масштабируемость:** Создание образов (ISO, OVA, QCOW2, VMDK, AMI) для локальных и облачных сред.
- **Универсальность:** Единый подход для серверов, IoT-устройств, виртуальных машин и контейнеров.
- **Безопасность:** Настройка пользователей, SSH-ключей и политик прямо в конфигурации.

Ценность для пользователей:

Kickstart экономит время, устраняет ошибки ручной настройки и позволяет развертывать системы от минималистичных IoT-решений до сложных корпоративных кластеров. Это инструмент, который превращает установку в декларативный процесс,

где вы описываете желаемое состояние, а NiceOS V воплощает его в жизнь.

2. Структура Kickstart-файла

Kickstart-файл в NiceOS V — это JSON-документ, структурированный по логическим блокам, которые соответствуют этапам установки и настройки системы. Это делает конфигурацию интуитивной и удобной для автоматизации, будь то ручное редактирование или генерация в CI/CD.

Основные категории параметров:

- **Диски и разметка:** `disk`, `disks`, `partitions`, `partition_type`, `bootmode` — управление дисками, LVM, Btrfs, шифрованием и теневыми разделами.
- **Пакеты и репозитории:** `packages`, `packagelist_file`, `repos`, `additional_rpms_path`, `niceos_docker_image`, `linux_flavor` — выбор программного обеспечения и источников.
- **Пользователи и безопасность:** `password`, `shadow_password`, `user_name`, `user_shadow_password`, `user_wheel`, `public_key` — настройка учетных записей и доступа.
- **Автоматизация:** `preinstall`, `postinstall`, `preinstallscripts`, `postinstallscripts`, `ansible` — выполнение скриптов и плейбуков.
- **Сеть:** `network`, `hostname` — настройка интерфейсов, VLAN, DNS и имени хоста.
- **Файлы:** `additional_files`, `search_path` — копирование ресурсов в систему.
- **Расширенные опции:** `arch`, `insecure_repo`, `timezone`, `log_level`, `ostree` — настройка архитектуры, безопасности и логирования.

JSON-формат упрощает валидацию, интеграцию с инструментами (например, Terraform, Ansible) и повторное использование конфигураций. Каждый блок четко разделяет ответственность, делая Kickstart мощным инструментом для DevOps.

Ценность для пользователей:

Структурированный подход позволяет быстро адаптировать конфигурацию под любые задачи — от создания минимальных образов до развертывания сложных серверных инфраструктур, сохраняя читаемость и воспроизводимость.

3. Управление дисками и разметкой

Настройка дисков — ключевой этап установки NiceOS V. Kickstart предоставляет гибкие инструменты для создания разметки, поддерживая простые loop-образы, сложные схемы с LVM, Btrfs, шифрованием LUKS и теневыми разделами для

атомарных обновлений.

Основные параметры:

- `disk`: Указывает устройство (например, `/dev/sda` или `/dev/disk/by-path`). Устаревает, но поддерживается.
- `disks`: Описывает диски с идентификатором `default`, указывая устройство (`device`) или файл (`filename`) с размером в МБ.
- `partitions`: Список разделов с параметрами: `mountpoint`, `size` (МБ, 0 для заполнения диска), `sizepercent`, `filesystem` (`ext4`, `btrfs`, `swap`, etc.), `lvm`, `btrfs`, `ab` (теневые разделы), `encrypt`.
- `partition_type`: Тип таблицы разделов (`gpt` по умолчанию, `msdos`).

Расширенные возможности:

- **LVM:** Создание логических томов с `vg_name` и `lv_name`.
- **Btrfs:** Поддержка субтомов для гибкой организации данных.
- **Шифрование:** Включение LUKS с `encrypt: true`.
- **AB-разделы:** Теневые разделы для обновлений и откатов.

Примеры конфигураций:

3.1. Простая разметка с loop-диском

```
{
  "disks": {
    "default": {
      "filename": "rootdisk.img",
      "size": 2048
    }
  },
  "partitions": [
    {
      "mountpoint": "/",
      "size": 0,
      "filesystem": "ext4"
    },
    {
      "size": 256,
      "filesystem": "swap"
    }
  ]
}
```

3.2. Многодисковая разметка

```
{
  "disks": {
    "default": {
      "filename": "disk-root.img",
      "size": 4096
    },
    "data": {
      "filename": "disk-data.img",
      "size": 8192
    }
  },
  "partitions": [
    {
      "disk_id": "default",
      "mountpoint": "/",
      "size": 0,
      "filesystem": "ext4"
    },
    {
      "disk_id": "data",
      "mountpoint": "/mnt/data",
      "size": 0,
      "filesystem": "ext4"
    }
  ]
}
```

3.3. Btrfs с субтомами

```
{
  "partitions": [
    {
      "mountpoint": "/",
      "size": 4096,
      "filesystem": "btrfs",
      "btrfs": {
        "label": "main",
        "subvols": [
          {"name": "rootfs", "mountpoint": "/"},
          {"name": "var", "mountpoint": "/var"},
          {"name": "home", "mountpoint": "/home"}
        ]
      }
    }
  ]
```

```
}
```

3.4. LVM и шифрование

```
{
  "partitions": [
    {
      "mountpoint": "/",
      "size": 0,
      "filesystem": "ext4",
      "lvm": {
        "vg_name": "VirtualGroup1",
        "lv_name": "root"
      },
      "encrypt": true
    },
    {
      "mountpoint": "/boot/efi",
      "size": 128,
      "filesystem": "vfat"
    }
  ]
}
```

Ценность для пользователей:

Гибкая разметка дисков позволяет создавать компактные образы для IoT, надежные серверные конфигурации с LVM и шифрованием, а также системы с поддержкой атомарных обновлений, минимизируя риски сбоев.

4. Настройка загрузки

Конфигурация загрузчика в NiceOS V определяет, как система будет запускаться: через BIOS, UEFI или оба режима (dualboot). Это критично для совместимости с облачными платформами, локальными гипервизорами и IoT-устройствами.

Параметры:

- `bootmode`:
  - `bios`: Legacy-загрузка с разделом для GRUB.
  - `efi`: UEFI с FAT-разделом (ESP) и `grub.efi`.
  - `dualboot`: Поддержка обоих режимов (по умолчанию для x86_64).
- `live`: `true` для запуска на хосте, `false` для создания переносимых образов (без EFI-

записей и `machine-id`).
- `eject_cdrom`: `true` (по умолчанию, извлечение ISO) или `false`.

Пример:

```
{
  "bootmode": "efi",
  "live": false,
  "eject_cdrom": true
}
```

Ценность для пользователей:

Универсальная загрузка упрощает развертывание на любых платформах, от устаревших серверов до современных облаков, а автоматическое извлечение ISO экономит время в автоматизированных пайплайнах.

5. Пакеты и репозитории

Выбор программного обеспечения определяет функциональность и размер образа NiceOS V. Kickstart позволяет точно указать пакеты, подключить внешние репозитории, локальные RPM и даже предустановить Docker-образы.

Параметры:

- `packages`: Список пакетов для установки.
- `packagelist_file`: JSON-файл со списком пакетов.
- `repos`: Дополнительные RPM-репозитории с настройками `name`, `baseurl`, `enabled`, `gpgcheck`.
- `additional_rpms_path`: Путь к локальным RPM-пакетам.
- `niceos_docker_image`: Docker-образ (например, `niceos:5.0`).
- `ostree`: Атомарная установка с выбором репозитория (`default_repo`, `repo_url`, `repo_ref`).

Примеры:

5.1. Установка пакетов

```
{
```

```
  "packages": ["linux", "openssh", "bash", "initramfs"]
}
```

5.2. Подключение репозиториев

```
{
  "repos": {
    "core": {
      "name": "NiceOS Core",
      "baseurl": "https://repo.niceos.local/core/x86_64",
      "enabled": 1,
      "gpgcheck": 0
    },
    "updates": {
      "name": "NiceOS Updates",
      "baseurl": "https://repo.niceos.local/updates/x86_64",
      "enabled": 1,
      "gpgcheck": 1
    }
  }
}
```

5.3. Атомарная установка с OSTree

```
{
  "ostree": {
    "default_repo": false,
    "repo_url": "http://repo.niceos.local/repo",
    "repo_ref": "niceos/5.0/x86_64/minimal"
  }
}
```

5.4. Docker-образ

```
{
  "niceos_docker_image": "niceos:5.0"
}
```

Ценность для пользователей:

Гибкость в выборе ПО и источников позволяет создавать минималистичные образы для IoT или полнофункциональные серверные системы с контейнерами, идеально подходящие для корпоративных сред.

6. Сетевые настройки

Сетевые параметры в Kickstart для NiceOS V используют современный синтаксис, совместимый с netplan и cloud-init, поддерживая DHCP, статические IP, VLAN и DNS. Это упрощает интеграцию с облачными и локальными сетями.

Параметры:

- `network`: Настройка интерфейсов (`ethernets`, `vlans`) с параметрами `match` (по имени или MAC), `dhcp4`/`dhcp6`, `addresses`, `gateway`, `nameservers`.
- `hostname`: Имя хоста системы.

Примеры:

6.1. DHCP на интерфейсе

```
{
  "network": {
    "version": "2",
    "hostname": "niceos-vm",
    "ethernets": {
      "id0": {
        "match": {"name": "eth0"},
        "dhcp4": true
      }
    }
  }
}
```

6.2. Статический IP и VLAN

```
{
  "network": {
    "version": "2",
    "hostname": "vlan-vm",
    "ethernets": {
      "eth0": {
        "match": {"name": "eth0"},
        "dhcp4": false,
        "addresses": ["10.0.0.2/24"]
      }
    },
    "vlans": {
```

```
        "vlan100": {
            "id": 100,
            "link": "eth0",
            "addresses": ["10.0.100.2/24"],
            "gateway": "10.0.100.1",
            "nameservers": {"addresses": ["9.9.9.9"]}
        }
    }
  }
}
```

Ценность для пользователей:

Быстрая и надежная настройка сети упрощает развертывание в сложных инфраструктурах, от облаков до изолированных корпоративных сетей.

7. Пользователи и безопасность

Настройка пользователей и доступа — важная часть Kickstart, обеспечивающая безопасность и удобство управления системой.

Параметры:

- `password`: Пароль root (`text`, `crypted`, `age`).
- `shadow_password`: Зашифрованный пароль root (короткая форма).
- `user_name`: Имя дополнительного пользователя.
- `user_shadow_password`: Зашифрованный пароль пользователя.
- `user_wheel`: Добавление пользователя в группу `wheel`.
- `public_key`: SSH-ключ для root-доступа.

Примеры:

7.1. Настройка пароля root

```
{
  "password": {
    "crypted": false,
    "text": "changeme",
    "age": 0
  }
}
```

7.2. Создание пользователя

```
{
  "user_name": "admin",
  "user_shadow_password": "$6$encrypted$hash",
  "user_wheel": true,
  "public_key": "ssh-rsa AAAAB3NzaC1yc2E..."
}
```

Ценность для пользователей:

Простая настройка пользователей и SSH-доступа повышает безопасность и упрощает администрирование, особенно в автоматизированных сценариях.

8. Автоматизация с Ansible

Встроенная поддержка Ansible позволяет запускать плейбуки сразу после установки, автоматизируя настройку окружения, сервисов и приложений. Это делает NiceOS V идеальной платформой для Infrastructure-as-Code.

Параметры:

- `ansible`: Список плейбуков с параметрами: `playbook`, `verbosity` (0–5), `logfile`, `extra-vars`, `tags`, `skip-tags`.

Примеры:

8.1. Запуск плейбуков

```
{
  "ansible": [
    {"playbook": "/usr/share/ansible/setup.yml"},
    {
      "playbook": "/usr/share/ansible/hardening.yml",
      "logfile": "ansible-harden.log",
      "verbosity": 2,
      "extra-vars": "@/tmp/vars.yml"
    }
  ]
}
```

8.2. Плейбук для Docker

```
---
- name: Setup Docker
  hosts: localhost
  tasks:
    - name: Start Docker service
      systemd:
        name: docker
        enabled: true
        state: started
    - name: Run Nginx container
      docker_container:
        name: nginx
        image: nginx:latest
        state: started
        ports:
          - "80:80"
```

Ценность для пользователей:

Ansible автоматизирует сложные настройки, от hardening до деплоя приложений, сокращая время и усилия на настройку инфраструктуры.

9. Preinstall и Postinstall

Скрипты `preinstall` и `postinstall` позволяют выполнять команды до и после установки, обеспечивая гибкую настройку окружения.

Параметры:

- `preinstall`/`postinstall`: Списки shell-команд.
- `preinstallscripts`/`postinstallscripts`: Списки внешних скриптов, ищутся в `search_path`.

Примеры:

9.1. Preinstall для определения диска

```
{
   "disk": "$DISK",
   "preinstall": [
```

```
      "#!/bin/sh",
      "ondisk=$(ls -lh /dev/disk/by-path/ | grep 'scsi-0:0:1:0' | awk '{print $9}')",
      "export DISK=\"/dev/disk/by-path/$ondisk\""
    ]
}
```

9.2. Postinstall для SSH

```
{
  "postinstall": [
      "#!/bin/sh",
      "mkdir -p /root/.ssh",
      "echo 'ssh-rsa AAAAB3NzaC1yc2E...' >> /root/.ssh/authorized_keys",
      "chmod 600 /root/.ssh/authorized_keys",
      "systemctl enable sshd"
  ]
}
```

9.3. Внешние скрипты

```
{
  "search_path": ["/tmp/scripts"],
  "preinstallscripts": ["find_disk.sh"],
  "postinstallscripts": ["enable_services.sh"]
}
```

Ценность для пользователей:

Автоматизация до и после установки минимизирует ручную работу, упрощая настройку и интеграцию с CI/CD.

10. Управление файлами

Параметры `additional_files` и `search_path` позволяют копировать файлы, скрипты и конфигурации в целевую систему, упрощая настройку сервисов.

Параметры:

- `additional_files`: Список пар {источник, назначение} для копирования.
- `search_path`: Директории для поиска файлов.

Пример:

```
{
  "search_path": ["/home/user/custom"],
  "additional_files": [
    {"resizefs.sh": "/usr/local/bin/resizefs.sh"},
    {"resizefs.service": "/lib/systemd/system/resizefs.service"}
  ],
  "postinstall": [
    "chmod +x /usr/local/bin/resizefs.sh",
    "systemctl enable resizefs.service"
  ]
}
```

Ценность для пользователей:

Легкое добавление пользовательских ресурсов ускоряет настройку сложных систем и их интеграцию в инфраструктуру.

11. Расширенные параметры

Для нестандартных сценариев NiceOS V предлагает дополнительные параметры, обеспечивающие гибкость и адаптацию.

Параметры:

- `arch`: Целевая архитектура (`x86_64`, `aarch64`).
- `insecure_repo`: Отключение SSL-проверки для репозиториев (`true`/`false`).
- `timezone`: Часовой пояс (по умолчанию `Europe/Moscow`).
- `linux_flavor`: Тип ядра (`linux`, `linux-aws`, `linux-secure`, etc.).
- `linux_flavor_<arch>`: Ядро для конкретной архитектуры.
- `log_level`: Уровень логирования (`error`, `warning`, `info`, `debug`).
- `ui`: Включение графического интерфейса установщика.

Пример:

```
{
  "arch": "aarch64",
  "insecure_repo": true,
  "timezone": "Asia/Yekaterinburg",
  "linux_flavor": "linux-aws",
```

```
  "log_level": "debug",
  "ui": true
}
```

Ценность для пользователей:

Расширенные параметры поддерживают сложные сценарии, такие как кросс-компиляция для IoT, работа в закрытых сетях и интеграция с облаками.

12. Рекомендации по использованию

Для эффективного и безопасного использования Kickstart следуйте этим рекомендациям:

Валидация и тестирование

- Проверяйте JSON с помощью `jq . config.json`.
- Используйте `log_level: "debug"` для диагностики.
- Тестируйте конфигурации в сухом режиме, если поддерживается инструментами сборки.

Безопасность

- Используйте зашифрованные пароли (`password.crypted: true`).
- Добавляйте SSH-ключи через `public_key` вместо паролей.
- Ограничивайте доступ к конфигурационным файлам с чувствительными данными.

Организация конфигураций

- Храните файлы в Git, разделяя:
  - `configs/` — Kickstart-файлы.
  - `playbooks/` — Ansible-плейбуки.
  - `scripts/` — Скрипты pre/postinstall.
  - `vars/` — Переменные для Ansible.
- Исключайте из Git чувствительные данные:

```
*.img
*.qcow2
*.ova
```

```
*.vmdk
id_rsa
passwords.json
```

- Автоматизируйте сборку через GitLab CI или GitHub Actions.

Пример выноса переменных

```
{
  "ansible": [
    {
      "playbook": "/usr/share/ansible/harden.yml",
      "extra-vars": "@/tmp/vars/vars.json"
    }
  ]
}
```

```
{
  "timezone": "Europe/Moscow",
  "ntp_server": "ntp.niceos.local"
}
```

Ценность для пользователей:

Эти практики повышают надежность, безопасность и масштабируемость, позволяя управлять инфраструктурой как кодом.

13. Заключение

Kickstart в NiceOS V — это не просто инструмент установки, а мощная платформа для автоматизации инфраструктуры. JSON-формат, поддержка Ansible, гибкая разметка дисков и сетевые настройки делают его идеальным для DevOps, облачных и локальных сценариев. От IoT-устройств до корпоративных кластеров — NiceOS V обеспечивает воспроизводимость, безопасность и скорость развертывания.

Почему стоит выбрать NiceOS V?

- **Экономия времени:** Автоматизация устраняет ручную настройку.
- **Надежность:** Декларативный подход гарантирует одинаковый результат.
- **Гибкость:** Поддержка любых платформ и сценариев.
- **Интеграция:** Легкая совместимость с CI/CD, Docker и облачными API.

Ценность для пользователей:

Kickstart позволяет сосредоточиться на ваших задачах, а не на настройке инфраструктуры, предоставляя надежный и масштабируемый инструмент для создания современных IT-решений.

Для примера конфигурации смотрите sample_ks.cfg.