АВТОМАТИЗАЦИЯ СОЗДАНИЯ СЕРВЕРНОЙ ИНФРАСТРУКТУРЫ В ЛАБОРАТОРНОЙ ПРАКТИКЕ

Ушаков Ю.А. канд. техн. наук, доцент, Ушакова М.В.

Оренбургский государственный университет

В настоящее время довольно существенной частью учебного процесса в информационных технологиях является работа с виртуальными машинами, на которых установлены разнообразные операционные системы. На некоторых предметах, например «операционные системы» или «информационные системы», приходится иметь несколько запущенных виртуальных машин, соединенных в сеть, имеющих запущенные сервисы DHCP, выход в интернет. Однако на обычных компьютерах запустить такие инфраструктуры довольно проблематично из-за требуемых ресурсов и ограничений сети университета.

Самый необходимый инструмент для изучения операционных систем и сетей в современной учебной практике – виртуальные машины – в произвольных топологиях и количествах доступны при реализации частных облаков. Однако для реализации частного облака необходимы высококвалифицированные администраторы, которые настраивают хостовые операционные системы и сети. Самыми популярными системами для реализации частного облака типа IaaS являются OpenStack, OpenNebula, CloudStack. Другие продукты, такие как Vmware vCloud, Microsoft System Center, ProxMox не предоставляют в бесплатном варианте требуемый функционал.

Установка и настройка любой opensource платформы состоит из множества шагов, которые не так просто повторить на произвольном оборудовании. Кроме установки самой системы, необходимо настраивать беспарольный удаленный вход, сетевые мосты, права доступа, различные службы, доступ к системе хранения и прочее. Это останавливает преподавателей и системных администраторов от разворачивания учебных облаков и использования открытых систем в пользу пропиетарных технологий.

Процесс обучения с использованием частного облака представлен следующим образом. Преподаватель подготавливает образы операционных систем, создает учетную запись для группы, создает нужное количество виртуальных машин, соединённых изолированными сетями. Затем студенты пользуются заранее созданными машинами (или создают свои). После того, как необходимость использования закончилась, машины выключаются до следующего раза или удаляются. Для автоматизации этих процессов часто используют программные интерфейсы (API), а для изоляции сети удобнее всего использовать OpenvSwitch.

Для разворачивания такой облачной системы можно использовать бездисковую загрузку Linux серверов с заранее преднастроенного образа облачно-

го узла. Для этого используется Linux bootstrap по классической схеме загрузки PXE и присоединения корня файловой системы к NFS [1].

Однако при использовании NFS в качестве корня операционной системы для многих серверов, приходится использовать NFS каталог в режиме «только для чтения» в следующем варианте:

```
/var/lib/nfsroot *(ro,async,no_root_squash,no_subtree_check,no_all_squash)
```

При этом большинство функций может быть нарушено из-за запрета на запись. Существует два пути решения проблемы множественного доступа к корня: использование tmpfs/squashfs - создание отражение корня операционной системы в оперативной памяти, вся запись ведется в отдельный раздел, и использование технологии overlayfs из контейнерных систем для работы с образом. Второй вариант автоматизирован через пакет overlayroot [2] и позволяет объединять несколько образов в один, записывать изменения в отдельный слой, позволяет сохранять изменения на отдельный раздел или сохранять изменения в оперативной памяти только на время работы сервера.

Учебные задания как правило подразумевают временное использование типовых конфигурации, установку и настройку программного обеспечения. Лучше всего использовать в этом случае вариант tmpfs, в котором все изменения стираются при перезагрузке. Для этого необходимо выполнить установку overlayroot в режиме tmpfs внутри файловой системы собираемого образа сервера:

```
sudo apt-add-repository ppa:cloud-initramfs-tools/ppa
sudo apt-get install -y overlayroot
echo 'overlayroot="tmpfs"' >> /etc/overlayroot.conf
```

В данном режиме корень сервера поставляется из сервера хранения по NFS в режиме «только для чтения», а все изменения записываются в отдельный слой. Однако все требуемые каталоги для работы облачной системы поставляются по NFS с самого сервера в режиме «shared storage», то есть общего хранилища. Управление в системе OpenNebula происходит по выполнению скриптов из общего хранилища по SSH, в системе Openstack – по REST запросам управляющего сервиса.

Поскольку загрузка серверов происходит по сети по протоколу PXE, требуется обеспечить начальную загрузку ядра для монтирования по NFS корня системы. Для этого в стандартной pxelinux загрузке [3] после установки и настройке DHCP и TFTP сервера необходимо прописать опции монтирования NFS и прочие параметры. Для этого нужно скопировать текущее ядро и initrd из каталога /boot в корень TFTP сервера, добавить ссылку на них и прописать все остальные параметры в строке «append»:

DEFAULT linux
LABEL linux
initrd initrd.img-4.4.0-109-generic
kernel vmlinuz-4.4.0-109-generic
append root=/dev/nfs nfsroot=192.168.0.2:/var/lib/nfsroot ip=dhcp rw ipv6.disable=1 net.ifnames=0 biosdevname=0 aufs=tmpfs acpi=force reboot=pciroot

здесь:
net.ifnames=0 biosdevname=0 - параметры для отмены именования интерфейсов в новой нотации Ubuntu 16;
aufs=tmpfs – режим работы OverlayFS;
ip=dhcp – указание взять адрес по DHCP;
root=/dev/nfs – использовать NFS как корень файловой системы;
nfsroot=192.168.0.2:/var/lib/nfsroot – путь до корня запускаемой файловой системы;
acpi=force reboot=pciroot – параметры работы ACPI.

Общая схема работы показана на рисунке 1.

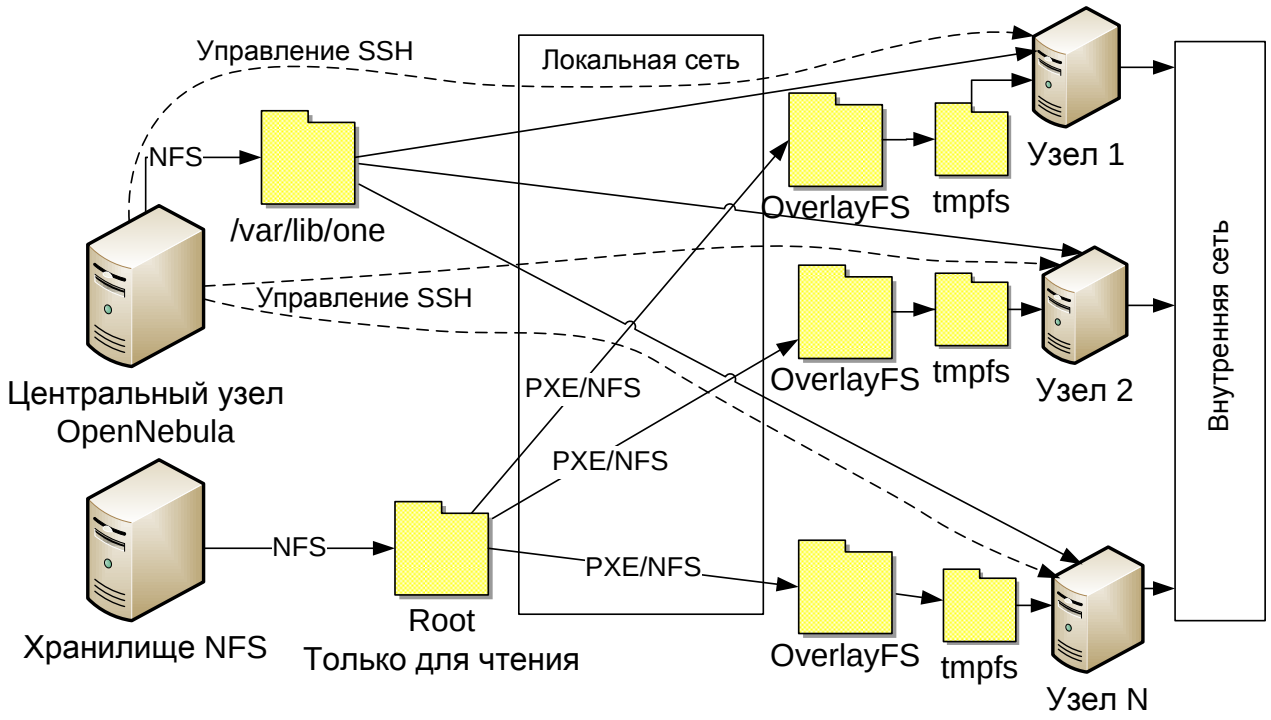

Рисунок 1 – Схема работы учебного кластера с автозагрузкой PXE

Для примера был создан образ узла OpenNebula с установленными режимом KVM, виртуальный коммутатор OpenvSwitch, подключенный ко второму интерфейсу, корень общего хранилища перенесен на сервер хранения NFS на

базе Openmediavault. Основное условие работоспособности серверов – совпадение версии ядра в папке /tftpboot и ядра внутри образа сервера.

Данный способ позволил создать один образ менее чем за час, после чего загрузить с него 20 серверов и успешно запустить первую виртуальную машину на вновь созданном облаке. Этот путь создания инфраструктуры гораздо экономичнее и быстрее, чем предлагают создатели Ubuntu (MaaS), Ansible, pupput, Kickstart, поскольку не происходит инсталяция сервера, а он загружается с заранее заданной конфигурацией. Обновление образа сервера автоматически передается на все запущенные сервера (но нужно сделать переключение снапшотов на системе хранения, чтобы все файлы одновременно обновились). Такой подход также экономит время адинистратора, а, при скачивании готового образа сервера позволяет запускать произвольные кластеры и облачные решения на любом количестве оборудования, главное что бы оно было одной архитектуры (x86_64, aarch64б sparc64 и подобное).

Статья подготовлена при поддержке РФФИ проект № 18-07-01446 и 17-07-01584.

Список литературы

1. Gavin Thomas. Boot Linux from an NFS server. https://www.gadgetdaily.xyz/boot-linux-from-an-nfs-server/ 2015

2. Justin Kulesza. Protecting the Root Filesystem on Ubuntu with Overlayroot. https://spin.atomicobject.com/2015/03/10/protecting-ubuntu-root-filesystem.- 2015.

3. Installing Debian using network booting. https://wiki.debian.org/PXEBootInstall.- 2013.