Автоматическое развертывание кластерного исполнения AxelNAC

В данной статье приведено описание развертывания кластерного исполнения AxelNAC с помощью утилиты Ansible.

Как работает кластерное исполнение AxelNAC

Кластерное исполнение AxelNAC может состоять из нечетного количества узлов и представляет собой отказоустойчивый кластер. **Отказоустойчивые кластеры** — это группы серверов, которые спроектированы в соответствии с методиками обеспечения высокой доступности и которые гарантируют работу кластера в целом при отказе одного или нескольких его узлов. Без кластеризации, если сервер, на котором запущен AxelNAC, выходит из строя, система будет недоступна до тех пор, пока не будет устранена неполадка на сервере. Отказоустойчивый кластер исправляет эту ситуацию, обнаруживая аппаратные и программные сбои и немедленно перенаправляя трафик на другие узлы, не требуя вмешательства администратора. Кластер AxelNAC может содержать до 9 узлов в разных L2-сегментах, такой кластер называется катастрофоустойчивым.

Данный кластер работает по принципу **active-active**, т.е. один или несколько узлов выступают в качестве балансировщика, который осуществляет прием и распределение запросов между собой и оставшимися узлами кластера. В случае выхода из строя сервера, он исключается из балансировки. В случае выхода из строя узла, выполняющего роль балансировщика, эта роль и кластерный адрес передаются другому узлу кластера.

Кластер AxelNAC будет сохранять полную работоспособность при соблюдении обязательного кворума — более 50% узлов кластера должны быть доступны. При потере кворума, AxelNAC сохраняет частичную работоспособность, переходя в режим только для чтения. В данном режиме изменяется логика основных элементов системы:

- **Аутентификация RADIUS MAC:** продолжит работу и будет возвращать атрибуты RADIUS, связанные с ролью, зарегистрированной в базе данных. Если к этому устройству можно применить фильтры VLAN или RADIUS, они будут применены, но любое изменение роли не будет сохранено;
- **Аутентификация RADIUS 802.1X:** продолжит работу, и если включена функция **Dot1x переопределяет роль с портала**, она будет вычислять роль с использованием доступных источников аутентификации, но не будет сохранять ее в базе данных в конце запроса. Если этот параметр отключен, он будет вести себя так же, как аутентификация по MAC-адресу. Фильтры VLAN и RADIUS по-прежнему будут применяться к подключениям, но любое изменение роли не будет сохранено. Если какие-либо из ваших источников аутентификации являются внешними (LDAP, AD, RADIUS, ...), они должны быть доступны для успешного выполнения запроса:
  - Аутентификация существующих в системе пользователей по протоколам PEAP и EAP-TLS будет выполняться в штатном режиме;
  - Реаутентификация существующих в системе пользователей по таймеру, установленному на коммутаторе, будет выполняться в штатном режиме;
  - Реаутентификация существующих в системе пользователей по периоду реаутентификации, установленному в системе, будет выполняться в штатном режиме;
  - Для новых пользователей, соответствующих правилам, установленным в системе, будет выполняться аутентификация, но новая запись в базе данных системы создаваться не будет.
- **Captive-портал:** будет остановлен. Для пользователей будет отображено сообщение о том, что система в настоящее время находится в режиме обслуживания.
- **Прослушиватели DHCP:** будут отключены, входящие DHCP-пакеты не будут сохраняться в базе данных. Firewall SSO также будет отключен.
- **Веб-интерфейс AxelNAC:** по-прежнему будет доступен в режиме только для чтения для всех разделов и в режиме чтения-записи для раздела конфигурации.

Для реализации данного способа вам потребуется нечетное количество установленных серверов AxelNAC (от трех до девяти). Серверы должны быть расположены в пределах следующих ограничений задержки (требование для **MariaDB Galera** — службы синхронизации БД кластера):

- для небольших развертываний (от трех до пяти узлов) между узлами кластера допускается задержка не более 75 мс;
- для больших развертываний (семь узлов) допускается задержка между узлами кластера не более 50 мс.

Также, для запуска скриптов автоматической сборки кластера понадобится дополнительная виртуальная машина с которой будет происходить развертывание. После окончания сборки ее можно будет деактивировать.

Скрипты для сборки кластерной инсталляции необходимо запускать на отдельной виртуальной машине с сетевым доступом ко всем узлам, которая не будет использоваться в кластере. Данная виртуальная машина будет использоваться для задач обслуживания кластера.

Подготовка к развертыванию кластерного исполнения

Для того чтобы произвести автоматическое развертывание кластерного исполнения AxelNAC, необходимо выполнить следующие подготовительные этапы:

Первичная настройка сетевого интерфейса на узлах

Для начала развертывания кластера необходимо настроить сетевые интерфейсы на каждом из узлов. Для этого выполните следующие шаги:

Шаг 1. Подключитесь к узлу через платформу виртуализации и задайте адрес на сетевом интерфейсе:

nano /etc/network/interfaces

Шаг 2. Измените файл конфигурации на каждом узле по данному примеру:

```
auto eth0
 iface eth0 inet static
   address <ip-address>
   netmask <net mask>
   gateway <gateway ip-address>
```

Шаг 3. Для применения конфигурации перезапустите службу сети, используя следующую команду:

```
 systemctl restart networking
```

Предварительная настройка мастер-узла AxelNAC

Для следующего этапа необходимо сконфигурировать мастер-узел. Для этого выполните следующие шаги:

Шаг 1. Подключитесь к узлу по адресу **https://<ip-address>:1443** — откроется веб-конфигуратор AxelNAC.

Шаг 2. Выберите интерфейс, который будет использоваться для управления. На странице его настройки в поле **Тип** выберите значение **Управление** и переместите переключатель **Высокая доступность** в состояние **включено**.

Шаг 3. Пройдите все оставшиеся этапы настройки в веб-конфигураторе.

Предварительная настройка ведомых узлов AxelNAC

После того как мастер-узел предварительно сконфигурирован, можно приступить к преднастройке ведомых узлов кластера. Для этого выполните следующие шаги:

Шаг 1. Подключитесь к ведомому узлу по адресу **https://<ip-address>:1443** — откроется веб-конфигуратор AxelNAC.

Шаг 2. Выберите интерфейс, который будет использоваться для управления. На странице его настройки в поле **Тип** выберите значение **Управление** и переместите переключатель **Высокая доступность** в состояние **включено**.

Необходимо пройти только первый шаг веб-конфигуратора, после чего вы можете закрыть страницу.

Скрипты для развертывания кластерного исполнения AxelNAC

В данном разделе представлены скрипты для развертывания кластерного исполнения AxelNAC версии 2.0.x:

- Скрипт для развертывания кластерного исполнения с помощью утилиты Ansible;

Установка и конфигурация утилиты Ansible

Для того чтобы автоматизировать настройку кластера, необходимо перенести скрипты для развертывания кластера на отдельную виртуальную машину с сетевым доступом ко всем узлам, которая не будет использоваться в кластере и сконфигурировать утилиту Ansible:

Шаг 1. Распакуйте набор скриптов **ansible-cluster-tools.tar.gz** на виртуальную машину, с которой будет выполняться обновление кластера, выполнив команду:

tar -xvf ansible-cluster-tools.tar

Шаг 2. Обозначьте переменную конфигурации для совместимости с плейбуком, введя следующую команду:

export ANSIBLE_CONFIG=ansible.cfg

Инструменты для развертывания кластерного исполнения

Все инструменты для развертывания кластерного исполнения расположены в директории **ansible-cluster-tools**:

- **create-cl.yml** — плейбук (набор скриптов) для сборки кластера из готовых узлов AxelNAC;
- **create-local-upgrade.sh** — скрипт для создания **docker-anac-update.tar**;
- **upgrade-cl.yml** — плейбук для обновление кластера из локального архива с обновлениями;
- **ansible.cfg** — файл локальной конфигурации Ansible;
- **multizone** — пример инвентарного файла для сборки кластера с двумя мастер-узлами;
- **normal** — пример инвентарного файла для сборки кластера с одним мастер-узлом;

- **files** — дополнительные файлы для плейбуков (например архив обновления **docker-anac-update.tar**);
- **templates** — шаблоны конфигурационных файлов для кластера и hosts для узлов;
- **roles** — роль обновления;
- **encrypt.sh** — пример скрипта для получения файла **val.cry**;
- **pass.sh** — пример скрипта генерации пароля для шифрования;
- **val.cry** — пример шифрованных данных для плейбука.

Подготовка среды

При исполнении плейбука ansible использует хранилище (vault) **val.cry** для определения паролей доступа к разделам AxelNAC. Сгенерируйте пароли для системы и БД с помощью скрипта **encrypt.sh**, в котором содержатся следующие переменные:

- **ansible_ssh_pass** — пароль от учетной записи **anac** для доступа к AxelNAC по протоколу **SSH**;
- **db_pfcluster_pass** — пароль для службы репликации баз данных между узлами кластера. Пароль должен соответствовать следующим требованиям:
  - Не содержит ни один специальный символ из списка: **', %, |, (,), /,** , а также **пробел;**
  - Длина пароля не более 16 символов.
- **web_srv_pass** — пароль от службы синхронизации конфигураций узлов кластера.

На данном шаге вы должны сами задать пароли для переменных **db_pfcluster_pass** и **web_srv_pass**.

Для генерации актуального хранилища паролей выполните следующие действия:

Шаг 1. Сделайте файл **pass.sh** исполняемым, выполнив следующую команду:

```
chmod +x ./pass.sh
```

Шаг 2. Зайдите в файл **encrypt.sh** и отредактируйте находящиеся в нем значения паролей.

Шаг 3. Сгенерируйте хранилище с актуальными паролями. Для этого запустите скрипт **encrypt.sh**, используя следующую команду:

```
bash encrypt.sh
```

После выполнения команды у вас появится сгенерированный файл **val.cry**.

Запуск скрипта требует прав пользователя **sudo**, поэтому их необходимо выполнять из-под оболочки **bash**.

Шаг 4. Откройте сгенерированный файл **val.cry** и замените значения зашифрованных паролей в файле **create-cl.yml** на значения из файла **val.cry** с сохранением табуляции:

```
---
- name: (check) GET_INFO
  hosts: all
  #  serial: 7
  #  any_errors_fatal: true
  gather_facts: true
#  become: yes
#  become_user: root
  vars:
    HOST_COUNT: "{{ ansible_play_hosts | length }}"
    ansible_ssh_pass: !vault |
      $ANSIBLE_VAULT;1.1;AES256
      62643565363835666436323339326630396263383937323533656338383430613933356337653430
      65646563333765663232643636366266643631393339633300a37393531616266666333630313236 63
      31663966326666623263643465383663346662613838633464646630663937353239356336353536
      38613763363437373320a62643866373334633838303930366463393539396366666383361653831 65
      31386531366238303138393539373563311353433366238663834356338633763 8
    db_pfcluster_pass: !vault |
      $ANSIBLE_VAULT;1.1;AES256
      62643565363835666436323339326630396263383937323533656338383430613933356337653430
      65646563333765663232643636366266643631393339633300a37393531616266666333630313236 63
      31663966326666623263643465383663346662613838633464646630663937353239356336353536
      38613763363437373320a62643866373334633838303930366463393539396366666383361653831 65
      31386531366238303138393539373563311353433366238663834356338633763 8
    web_srv_pass: !vault |
      $ANSIBLE_VAULT;1.1;AES256
      31336437363537396461613830316333633933666131336434326238666165633834366461373137
      64353361383064666331343931303631626462393937353 70a37613435373864343039343163 9239
      63633766616437653936616534343862323363643632343866373264616364366464363663646261
      37323634626663639350a383966383963616333306234663136353239356662366166664530663436
      33356662316233323833333261313931323032323138366438376262633465356161
```

Запуск автоматического развертывания кластерного исполнения AxelNAC

Сборка кластера с одним узлом-балансировщиком

Для того чтобы запустить сборку кластерного исполнения с одним узлом-балансировщиком, выполните следующие шаги:

Шаг 1. В файле **normal**, в разделе **[cluster:children]** укажите имена хостов и IP-адреса узлов, а также пометьте мастер-узел, указав в конце строки параметр **master=true**.

Шаг 2. В разделе **[cluster:vars]** укажите от имени какого пользователя производятся действия: **ansible_user=anac**.

Шаг 3. Укажите виртуальный IP-адрес кластера с помощью атрибута **vip**.

Шаг 4. В файле **create-cl.yml** сконфигурируйте параметры проверки на соответствие.

Пример инвентарного файла

```
[cluster:children]
dc1

[dc1]
hostname_узла_1 ansible_host=IP-адрес_узла_1 vip=VIP-адрес_кластера master=true
hostname_узла_2 ansible_host=IP-адрес_узла_2
hostname_узла_3 ansible_host=IP-адрес_узла_3

[cluster:vars]
####### !!!!!!!!!!!! type = 'normal' or 'l3' !!!!!!!!!!! #######
type=normal

ansible_ssh_common_args='-o StrictHostKeyChecking=no -o PasswordAuthentication=yes'
ansible_user=anac
#ansible_ssh_pass=anac_pass
ansible_python_interpreter="/usr/bin/python3"
# mem >= 8 Gb
mem=7944
# cpu >= 4
cpu=4
# sda.size >= 200 Gb
hdd=100
```

Сборка кластера с двумя и более узлами-балансировщиками

Для того чтобы запустить сборку кластерного исполнения с двумя и более узлами-балансировщиками, выполните следующие шаги:

Шаг 1. В файле **multizone**, в разделе **[cluster:children]**, в секциях **dc1** и **dc2,** укажите имена хостов и IP-адреса узлов, а также пометьте мастер-узлы, указав в конце строки параметр **master=true**.

Шаг 2. В разделе **[cluster:vars]** укажите от имени какого пользователя производятся действия: **ansible_user=anac**.

Шаг 3. Укажите виртуальные IP-адреса кластера с помощью атрибута **vip**.

Шаг 4. В файле **create-cl.yml** сконфигурируйте параметры проверки на соответствие.

Пример инвентарного файла

```
[cluster:children]
dc1
dc2

[dc1]
hostname_1_узла_1 ansible_host=IP-адрес_узла_1 vip=VIP-адрес_кластера_1 master=true
hostname_2_узла_1 ansible_host=IP-адрес_узла_2
hostname_3_узла_1 ansible_host=IP-адрес_узла_3

[dc2]
hostname_1_узла_2 ansible_host=IP-адрес_узла_1 vip=VIP-адрес_кластера_2
hostname_2_узла_2 ansible_host=IP-адрес_узла_2

[cluster:vars]
####### !!!!!!!!!!!! type = 'normal' or 'l3' !!!!!!!!!!! #######
type=l3

ansible_ssh_common_args='-o StrictHostKeyChecking=no -o PasswordAuthentication=yes'
ansible_user=anac
#ansible_ssh_pass=anac_pass
#ansible_ssh_pass=123123
ansible_python_interpreter="/usr/bin/python3"
# mem >= 8 Gb
mem=7944
# cpu >= 4
cpu=4
# sda.size >= 200 Gb
hdd=100
```

Работа с плейбуком

Описание стадий

Плейбук для автоматической сборки кластерного исполнения разделен на стадии, по которым можно запускать/повторять отдельно серию задач при необходимости. Для этого используются теги:

Тег	Описание
init	• Вносит необходимые правки в **sysctl**; • Устанавливает недостающее ПО; • Вносит изменения в **hostname** согласно инвентарному файлу и обновляет записи **/etc/hosts**; • Создает пользователя реплики; • Перезапускает узлы.
galera1	• Формирует конфигурационные файлы кластера; • Перезапускает службы на мастер-узле.
galera2	• Переводит мастер-узел в режим создания кластера; • Отключает службу **iptables**.
galera3	• Синхронизирует конфигурационные файлы ведомых узлов с мастер-узлом.
galera4	• Скачивает конфигурационные файлы для проверки их идентичности.
galera5	• Перезагружает службы на узлах.
galera6	• Возвращает мастер-узел в обычный режим; • Включает службу **galera-autofix**.
galera7	• Перезагружает службы на узлах; • Перезагружает ведомые узлы.

Запуск плейбука

Перед запуском сборки кластерной инсталляции необходимо выполнить проверку корректности введенных параметров с помощью команды:

ansible-playbook -i <**normal**|**multizone**> --vault-id ./pass.sh create-cl.yml --skip-tags check -Kb --check

Для того чтобы начать сборку кластерной инсталляции, запустите плейбук:

ansible-playbook -i <**normal**|**multizone**> --vault-id ./pass.sh create-cl.yml --skip-tags check -Kb

Дополнительные варианты запуска плейбука

Плейбук может быть запущен в различных режимах. Ниже приведено описание каждого из дополнительных режимов с примерами:

Если вы запускаете скрипт развертывания с узла, находящегося в кластере, необходимо после первой перезагрузки запустить плейбук с пропуском шага **init**, выполняющего перезагрузку узлы кластера:

ansible-playbook -i normal --vault-id ./pass.sh create-cl.yml -t --skip-tags init,check

Запуск конкретного шага:

ansible-playbook -i <**normal**|**multizone**> --vault-id ./pass.sh create-cl.yml -t <ТЕГ_СТАДИИ>

Такой командой вы можете запустить определенную стадию плейбука (например, если во время исполнения плейбука произошла остановка операций, или пришлось прервать его работу вручную).

Запуск конкретного шага на конкретном узле:

ansible-playbook -i <**normal**|**multizone**> --vault-id ./pass.sh create-cl.yml -t <ТЕГ_СТАДИИ> -l <IP-АДРЕС_УЗЛА>

Такой командой вы можете запустить определенную стадию плейбука на определенном узле.

Запуск нескольких шагов:

ansible-playbook -i <**normal**|**multizone**> --vault-id ./pass.sh create-cl.yml -t <ТЕГ_СТАДИИ>,<ТЕГ_СТАДИИ>,<ТЕГ_СТАДИИ>

Такой командой вы можете запустить определенные стадии плейбука (например, при запуске плейбука с узла, который обновляется в данный момент).

Проверка работоспособности кластера

Для того чтобы проверить работоспособность кластера необходимо открыть веб-интерфейс AxelNAC. После ввода учетных данных откроется раздел**Статус → Система**, в котором можно будет увидеть подключенные узлы.

IP-адреса всех узлов кластера также должны быть отображены в файле **/usr/local/pf/conf/cluster.conf** и соответствовать содержимому, указанному в файле **create-cl.yml**.

Пример файла для кластера с двумя VIP-адресами из 5 узлов:

```
# MULTI ZONE CLUSTER CFG
[general]
multi_zone=enabled

[dc1 CLUSTER]
management_ip=10.21.209.41

[dc1 CLUSTER interface eth0]
ip=10.21.209.41

[dc1 deb-to-astra-dc1-1]
management_ip=10.21.209.36

[dc1 deb-to-astra-dc1-1 interface eth0]
ip=10.21.209.36

[dc1 deb-to-astra-dc1-2]
management_ip=10.21.209.37

[dc1 deb-to-astra-dc1-2 interface eth0]
ip=10.21.209.37

[dc1 deb-to-astra-dc1-3]
management_ip=10.21.209.38

[dc1 deb-to-astra-dc1-3 interface eth0]
ip=10.21.209.38

[dc2 CLUSTER]
management_ip=10.21.209.42

[dc2 CLUSTER interface eth0]
ip=10.21.209.42

[dc2 deb-to-astra-dc2-1]
management_ip=10.21.209.39

[dc2 deb-to-astra-dc2-1 interface eth0]
ip=10.21.209.39

[dc2 deb-to-astra-dc2-2]
management_ip=10.21.209.40

[dc2 deb-to-astra-dc2-2 interface eth0]
ip=10.21.209.40
```

Также статусы можно проверить с помощью консоли управления. Для этого подключитесь по**SSH** к любому узлу и выполните следующую команду:

```
/usr/local/pf/bin/pfcmd service pf status
```

Еще один способ проверки — проверка статусов через MySQL. Чтобы просмотреть все активные узлы в кластере выполните следующую команду:

```
show status like '%wsrep%';
```

Активные узлы кластера будут перечислены в строке **wsrep_incoming_addresses**.

Дополнительные команды Ansible

Проверка доступа узлов

ansible -i <**normal**|**multizone**> all --vault-id ./pass.sh -m ping

Запуск команд на удаленных узлах

ansible -i <**normal**|**multizone**> all --vault-id ./pass.sh -m shell -a "/usr/local/pf/bin/pfcmd service pf restart"

Отображение информации о хосте в виде дерева

ansible -i <**normal**|**multizone**> all -m ansible.builtin.gather_facts --tree

Отображение информации о хосте в общем виде

ansible -b -i <**normal**|**multizone**> all -m gather_facts

Отображение полной информации о хосте

ansible -i <**normal**|**multizone**> all --vault-id ./pass.sh -m setup

Просмотр журнала ansible

ansible-playbook -i <**normal**|**multizone**> --vault-id ./pass.sh create-cl.yml -t init -l dc1-anac5-n1 -vvv

Возможные ошибки и пути их решения

Остановка службы mariadb

Некорректная остановка службы **maridb** не приводит к остановке выполнения плейбука и может быть проигнорирована.

Не определяется sda (HDD) и memory (RAM)

Пример вывода:

fatal: [ds-net-axelnac-01]: FAILED! => {"msg": "The task includes an option with an undefined variable. The error was: 'dict object' has no attribute 'sda'. 'dict object' has no attribute 'sda'\n\nThe error appears to be in

Решение:

Дальнейшие действия выполняются по усмотрению пользователя и могут повлечь за собой риски.

Выполните команду для запуска нескольких шагов, пропустив тег **check**.

```
ansible-playbook -i normal --vault-id ./pass.sh create-cl.yml -t init,galera1,galera2,galera3,galera4,galera5,galera6,galera7
```

ID статьи: 1040
Последнее обновление: 4 мар., 2026
Обновлено от: Егоров В.
Ревизия: 33
База знаний AxelNAC -> Документация -> Система контроля доступа к сети «AxelNAC». Версия 2.1.0 -> AxelNAC. Руководство по настройке кластера -> Автоматическое развертывание кластерного исполнения AxelNAC
https://docs.axel.pro/entry/1040/