УДК 004.771

УДАЛЕННОЕ АДМИНИСТРИРОВАНИЕ ЛОКАЛЬНЫХ КОМПЬЮТЕРОВ С ПРИМЕНЕНИЕМ ПЛАТФОРМЫ С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ ANSIBLE

Е. С. Пискун, А. Д. Стрельцов

Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь, strelsov92@gmail.com

Key factors in system settings and configurations that primarily affect the efficiency and stability of corporate networks when using Ansible have been experimentally established. Certain configuration parameters allow identifying errors in the configuration of network devices and servers, as well as inefficient resource allocation, which helps improve infrastructure management and process automation.

Введение

В современном мире корпоративные сети и ИТ-инфраструктуры постоянно растут и усложняются. С увеличением числа серверов, сетевых устройств и приложений управление конфигурациями вручную становится все более трудоемким и подверженным ошибкам. Это создает необходимость внедрения автоматизированных инструментов для эффективного управления конфигурациями, которые позволяли бы оптимизировать рабочие процессы, повысить производительность и снизить риски, связанные с человеческим фактором.

Одним из таких инструментов является *Ansible* – платформа с открытым исходным кодом, предназначенная для автоматизации процессов управления конфигурациями, развертывания приложений и управления ИТ-инфраструктурой [1]. Благодаря простоте использования, модульной архитектуре и широкому набору возможностей, *Ansible* быстро завоевал популярность в среде системных администраторов и *DevOps*-инженеров.

Цель статьи заключается в исследовании возможностей *Ansible* для управления конфигурациями в корпоративных сетях. которое строится на обзоре основных принципов работы платформы, ее архитектуры, подходов к автоматизации задач, а также преимуществ и ограничений использования данного инструмента в корпоративной среде [1].

Архитектура и принципы работы Ansible

Архитектура *Ansible* основана на модели клиент-сервер, где платформа управляет удаленными системами без необходимости установки агентов на управляемых узлах. Это достигается благодаря использованию *SSH* (или *WinRM* для *OS Windows*) для удаленного доступа, что упрощает процесс внедрения и администрирования [2].

Ключевыми компонентами архитектуры Ansible являются [1]:

– Контроллер (англ. *Control Node*) – основной управляющий сервер, на котором установлен *Ansible*. Он управляет выполнением плейбуков, координацией задач и взаимодействием с управляемыми узлами;

– Управляемые узлы (англ. *Managed Nodes*) – серверы, сетевые устройства и другие компоненты ИТ-инфраструктуры, которые находятся под управлением *Ansible.* Узлы не требуют установки дополнительного программного обеспечения, поскольку взаимодействие происходит через стандартные протоколы;

– Плейбуки (англ. *Playbooks*) – основной механизм автоматизации в *Ansible*. Плейбуки написаны на языке *YAML* и описывают последовательность задач, которые должны быть выполнены на управляемых узлах. Плейбуки также могут включать условные операторы и циклы для выполнения более сложных сценариев;

– Инвентарь (англ. *Inventory*) – это список управляемых узлов, сгруппированных по категориям, что позволяет *Ansible* выбирать конкретные узлы для выполнения задач. Инвентарь может быть статическим (файлы) или динамическим (генерация через *API* или другие источники данных);

– Модули (англ. *Modules*) – независимые блоки, которые выполняют конкретные задачи, такие как установка пакетов, изменение конфигураций, управление пользователями и так далее. Модули могут быть встроенными или написанными пользователем для удовлетворения уникальных потребностей.

Одним из ключевых принципов *Ansible* является гарантия неизменности, что означает, что повторное выполнение одной и той же задачи не приведет к изменению системы, если она уже находится в нужном состоянии. Это важно для управления конфигурациями, так как гарантирует стабильность и предсказуемость изменений.

Еще одним важным принципом является декларативный подход к описанию состояния системы. В плейбуках *Ansible* описывается не пошаговый процесс выполнения задач, а конечное состояние, к которому необходимо прийти. Это упрощает понимание и поддержку кода автоматизации.

Экспериментальная оценка производительности *Ansible* при массовой автоматизации

Цель эксперимента – оценить производительность *Ansible* в корпоративной сети, состоящей из нескольких виртуальных машин (*VMs*), при выполнении следующих задач:

1. Установка веб-сервера;
2. Настройка конфигурации веб-сервера;
3. Обновление конфигураций;
4. Повторное выполнение для проверки идемпотентности.

Исходные данные:

– количество виртуальных машин – 10;

– операционные системы:

- 5 машин на базе *Ubuntu* 20.04;
- 5 машин на базе *CentOS* 7.0.

Задачи:

– установить программное обеспечение с открытым исходным кодом для создания легкого и мощного веб-сервера *Nginx*;

– настроить конфигурации веб-сервера (разные порты и доменные имена);

– обновить конфигурации;

– совершить повторный запуск задач.

Метрики оценки производительности:

– время выполнения задачи (в секундах);

– количество внесенных изменений;

– количество ошибок.

Результаты выполнения эксперимента

Шаг 1 – Установка веб-сервера

В таблице ниже приведено время выполнения установки веб-сервера *Nginx* на 10 узлах:

Узел №	ОС	Время выполнения (с)	Количество изменений, шт	Ошибки
1-5	*Ubuntu*	10-12	3	0
6-10	*CentOS*	13-15	3	0

Среднее время выполнения – 12.5 с.

Шаг 2 – Настройка конфигураций веб-сервера

В нижеприведенной таблице представлены результаты по настройке конфигурации веб-сервера (например, указание доменного имени и порта для каждого узла):

Узел №	ОС	Время выполнения (с)	Количество изменений, шт	Ошибки
1-5	*Ubuntu*	4-6	2	0
6-10	*CentOS*	6-8	2	0

Среднее время выполнения – 6 с.

Шаг 3 – Обновление конфигураций

Плейбук был выполнен для обновления конфигураций (например, изменение порта). Результаты выполнения приведены в таблице ниже:

Узел №	ОС	Время выполнения (с)	Количество изменений, шт	Ошибки
1-5	*Ubuntu*	5-7	1	0
6-10	*CentOS*	8-9	1	0

Среднее время выполнения – 7 с.

Шаг 4 – Повторный запуск плейбуков (проверка идемпотентности)

Для проверки устойчивости к повторным операциям был выполнен повторный запуск плейбуков. Результаты приведены в таблице ниже.

Узел №	ОС	Время выполнения (с)	Количество изменений, шт	Ошибки
1-5	*Ubuntu*	2	0	0
6-10	*CentOS*	3	0	0

Среднее время выполнения – 2.5 с.

Общее время выполнения всех этапов показало, что *Ansible* эффективно управляет большими группами узлов, выполняя задачи за короткое время.

Количество изменений сократилось при повторном выполнении плейбуков, что подтверждает правильную работу *Ansible* к повторным операциям – система не вносит изменения, если узлы уже настроены корректно.

Ошибки отсутствовали на всех этапах, что демонстрирует стабильность работы платформы.

В результате экспериментов установлено, что использование *Ansible* значительно повышает эффективность автоматизации управления конфигурациями по сравнению с ручными операциями. Например, установка веб-сервера на 10 узлах вручную составила бы около 10-15 минут на каждый узел, в то время, как установка при помощи Ansible составляет 10-15 секунд.

Заключение

Экспериментально установлено, что *Ansible* представляет собой мощный инструмент, который обеспечивает удобство, эффективность и надёжность в процессе управления сложной инфраструктурой, состоящей из множества серверов и устройств.

Показано, что одним из важнейших аспектов Ansible является его устойчивость к повторным операциям – способность при повторных запусках плейбуков гарантировать, что система останется в корректном состоянии без внесения ненужных изменений.

Список использованных источников

1. Ansible [Электронный ресурс] // URL: https://www.ansible.com (дата обращения 16.10.2024).

2. Костромин Р.О. Сравнительный обзор средств управления конфигурациями ресурсов вычислительной среды функционирования цифровых двойников //Информационные и математические технологии в науке и управлении. 2021. № 1 (21). С. 131-145. DOI:10.38028/ESI.2021.21.1.011