УДК 004.4

СИСТЕМА КОНФИГУРАЦИОННОГО УПРАВЛЕНИЯ КАК МЕТОД УМЕНЬШЕНИЯ КОЛИЧЕСТВА ОШИБОК ПРИ НАСТРОЙКЕ ВЫЧИСЛИТЕЛЬНОЙ ИНФРАСТРУКТУРЫ

Баранюк В.В., Воронцов Ю.А.

МИРЭА - Российский технологический университет, 119454, Россия, г. Москва, проспект Вернадского, 78, e-mail: valentina_bar@mail.ru, voroncov_yu@mirea.ru

В статье рассмотрен способ уменьшения количества ошибок в процессе конфигурации вычислительной инфраструктуры при помощи системы конфигурационного управления. Анализируются задачи конфигурационного управления вычислительной инфраструктуры. Выявлены причины появления ошибок в процессе изменения файлов конфигурации или же создания их с нуля. Разбирается архитектура одной из систем управления конфигураций и список реализованных в ней концепций. Приводятся преимущества внедрения подобного рода системы.

Ключевые слова: конфигурационное управление, система конфигурационного управления, проблемы управления конфигурацией, администрирование, ansible

CONFIGURATION CONTROL SYSTEM AS A METHOD TO REDUCE THE NUMBER OF ERRORS WHEN SETTING UP THE COMPUTER INFRASTRUCTURE

Baranyuk V.V., Vorontsov Yu.A.

MIREA - Russian Technological University, 119454, Russia, Moscow, Vernadsky prospect, 78,

e-mail: valentina_bar@mail.ru, voroncov_yu@mirea.ru

The article discusses a way to reduce the number of errors in the process of configuring a computing infrastructure using a configuration management system. The problems of configuration management of the computing infrastructure are analyzed. The reasons for the appearance of errors in the process of changing configuration files or creating them from scratch have been identified. The architecture of one of the configuration management systems and the list of concepts implemented in it are analyzed. The advantages of introducing such a system are given.

Keywords: configuration management, configuration management system, configuration management problems, administration, ansible

Введение

Любая вычислительная инфраструктура обеспечивает функционирование широкого набора взаимодействующих систем, решающих требуемые задачи, будь то автоматизация процессов, протекающих в компании, автоматизация хранения данных, или любая другая задача, обусловленная необходимостями соответствующей организации. Каждая подобная система требует наличия отдельного вычислительного узла, при этом узел может быть как аппаратным, то есть являться реальной вычислительной машиной, так и виртуальным, созданным при помощи технологий виртуализации.

Вне зависимости от того, каким образом организовано выделение вычислительных мощностей, необходимо организовать процесс управления конфигурациями данных вычислительных узлов с целью получения постоянного контроля над ними. При этом количество узлов в развитой вычислительной инфраструктуре исчисляется сотнями, если не тысячами, и настройка такого количества вычислительных устройств разительно отличается от настройки нескольких машин [2].

Процесс конфигурации включает в себя ряд задач, каждая из которых связана с поддержанием вычислительной инфраструктуры в рабочем состоянии. При этом каждая из задач связана с рядом трудностей при их выполнении администратором в ручном режиме без использования каких-либо средств автоматизации.

Обзор задач конфигурации

Перед рассмотрением задач необходимо определить состав узлов, расположенных в вычислительной инфраструктуре, представленных на Рисунке 1. В их число входят:

– 2 сервера с установленными на них гипервизорами Proxmox для создания виртуальных машин;

– виртуальные машины, на которых развернуты различные конечные сервисы для выполнения различного рода задач.

Перейдём к рассмотрению задач, основываясь на представленной архитектуре.

Первой задачей является создание конфигураций для новых виртуальных машин, которые необходимо запускать в эксплуатация по мере появления новых задач. При этом каждая новая машина может обладать конфигурацией схожей с уже существующей, в этом случае администратору необходимо повторно производить полную настройку с целью получения идентичной машины.

Второй задачей является обновление конфигурации уже существующей виртуальной машины. С течением времени требования к конфигурации вычислительных узлов могут меняться, появляется необходимость в новых пакетах, необходимость в размещении нового ПО на узлах и т.д. При этом если необходимо произвести обновление нескольких однотипных вычислительных узлов, необходимо будет повторять несколько раз один и тот же набор действий на всех вычислительных узлах. Количество манипуляций с конфигурациями в таком случае можно выразить формулой (1):

$$S = n \sum_{1}^{m} b_i \tag{1}$$

где n – количество вычислительных узлов, конфигурацию которых необходимо изменить; b_i – количество изменений для одного приложения, m – количество конфигурируемых приложений.

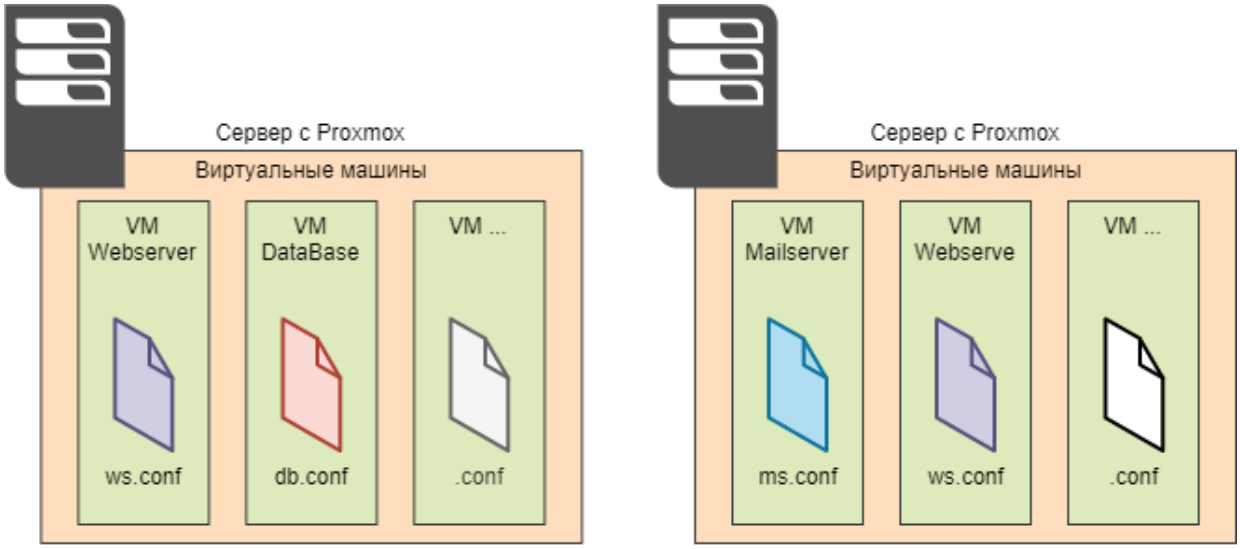

Рисунок 1 – Архитектура вычислительной инфраструктуры

Такой подход к решению задач влечет за собой ряд сложностей и проблем при обслуживании инфраструктуры.

Первая из них – это невозможность повторения конфигурации для вводимых в эксплуатацию узлов. Необходимо будет полностью вручную повторить настройку нового узла, последовательно выполняя все необходимые команды для установки и настройки ПО. Данный способ может повлечь за собой ряд ошибок, поскольку конфигурационные файлы предполагаемо одинаковых серверов на самом деле могут отличаться. В глобальной перспективе это может привести к неправильной работе разворачиваемых на вычислительных узлах сервисов, что в итоге повлечет за собой неправильную работу всей инфраструктуры в целом.

Данную проблему можно частично решить при помощи написания скриптовых файлов. Данные файлы позволяют запустить последовательность команд для установки и настройки всех необходимых пакетов. Такой подход снижает количество необходимых операций по настройке узла, однако глобально также имеется широкий

набор конфигурационных файлов, каждый из которых является источником информации для системного администратора, из этого вытекает следующая проблема.

Вторая сложность при администрировании вычислительной инфраструктуры – необходимость обновления конфигурации на всех вычислительных узлах инфраструктуры вручную. То есть для изменения одного параметра конфигурации администратору необходимо последовательно обслужить каждый узел в инфраструктуре.

Даже использование скриптовых файлов не позволит решить данную задачу, поскольку необходимо будет в любом случае исполнить такой файл на каждом узле. В результате это не позволит снизить количество выполняемых администратором действий.

В случае же, если итоговой задачей конфигурации является приведение нескольких по-разному настроенных узлов к единой конфигурации, такой подход и вовсе является неприемлемым. При таком сценарии подразумевается приведение нескольких узлов, находящихся в разном исходном состоянии в единое конечное состояние, что не может быть решено набором одних и тех же команд. Следовательно, каждый узел будет приводится в требуемое состояние разными последовательностями команд, что не может быть оптимизировано при помощи написания скриптов, поскольку всё равно придется писать отдельный скрипт для настройки каждой машины.

В результате после настройки всех узлов администратор имеет набор никак не связанных между собой конфигурационных файлов. При этом данные файлы дублируются на каждом узле без какой-либо возможности централизованного управления данными файлами. При этом при добавлении нового узла в систему количество конфигурационных файлов также возрастает. Следовательно, возрастает и количество возможных ошибок при конфигурации вычислительного оборудования, которое можно выразить следующей формулой (2):

$$E = n\sum_{i=1}^{m}\sum_{j=1}^{p} a_{ij} r, \tag{2}$$

где n – количество вычислительных узлов, конфигурацию которых необходимо изменить; m – количество единиц конфигурируемого ПО; p – количество конфигурационных файлов для каждой единицы ПО; a_{ij} – количество изменений в каждом конфигурационном файле; r – среднее количество ошибок при осуществлении изменения.

Обозначенные проблемы могут быть решены при помощи систем конфигурационного управления.

Система конфигурационного управления

Конфигурационное управление позволяет осуществлять эффективный контроль вычислительной инфраструктурой любой организации. [1] В целом процесс конфигурационного управления не ограничивается только лишь вычислительной инфраструктурой. Его целью является снижение сложности протекающих процессов и управление данной сложностью, повторное использование решений и ясность данных о процессах. [3]. Если говорить о конфигурационном управлении вычислительной инфраструктурой, то подразумевается процесс управления состояниями вычислительных устройств, направленный на снижение сложности обслуживания узлов, подразумевающее изначальную настройку сервера, и дальнейшее его поддержание в рабочем состоянии при внесение различного рода изменений.

Системы управления конфигурацией позволяют определять конфигурируемые параметры инфраструктуры, а также настраивать и поддерживать состояние инфраструктуру в заданной конфигурации. Управление конфигурацией помогает администраторам отслеживать, в каком состоянии находятся в текущий момент вычислительные узлы и сервисы на них. [4]

Правильный подход к управлению конфигурацией включают в себя следующие правила:

– классификация узлов и управление ими по группам и подгруппам;

– централизованное изменение конфигурации;

– применение новых настроек для всех причастных узлов;

– автоматическая идентификация, исправление и обновление. [4]

В качестве примера системы конфигурационного управления предлагается рассмотреть систему Ansible.

Основной принцип работы Ansible заключается в подключении к узлам вычислительной инфраструктуры и отправке им небольших программы, называемые модулями. Модули используются для выполнения задач автоматизации конфигурационного управления. Данные программы написаны как ресурсные модели желаемого состояния системы, они представляют из себя набор действий, которые необходимо выполнить для перевода системы в требуемое состояние. Затем, после передачи, Ansible выполняет эти модули и удаляет их по завершении.

Ansible не имеет агентов, что означает, что узлы, которыми он управляет, не требуют установки на них какого-либо дополнительного программного обеспечения. Всё, что должно быть установлено – это интерпретатор языка Python, при помощи которого Ansible запускает передаваемые на узлы модули.

Информация об управляемых узлах хранится в специальных файлах, называемых inventory. Данные файлы содержат в себе все необходимые данные по управляемым узлам, а именно: как к ним можно подключиться, какие дополнительные параметры при подключении необходимо указать и т.д. Изначально существует только один подобный файл, из которого Ansible получает информацию, однако администратор может самостоятельно его задать и определить все необходимые узлы.

Ansible использует протокол SSH для осуществления безопасного подключения к узлам и последующего выполнения задач. После подключения Ansible передает модули, необходимые для выполнения указанной команды или playbook'a, на управляемую машину.

Playbook – это файл, написанный в формате YAML, содержащий один или несколько сценариев, и используемый для определения желаемого состояния системы. В отличии от модуля Ansible, который представляет собой скрипт, то есть набор исполняемых инструкций, который можно использовать внутри Ansible Playbook.

Сценарии состоят из упорядоченного набора задач, которые необходимо выполнить в зависимости от выбора группы узлов, заданной в файле inventory. Задачи – это элементы, и вызывающие модули Ansible, которые будут производить необходимые действия на узлах.

Выполнение задач происходит в том же порядке, в котором они написаны, поэтому следует соблюдать порядок, если выполнение одной задачи зависит от другой. При выполнении сценария Ansibl имеет возможность отслеживать состояние системы.

Если при сканировании системы будет обнаружено несоответствие целевого состояния, описанного при помощи сценария, и фактического состояния системы, то будут произведены все необходимые изменения для устранения найденных несоответствий. В противном случае, когда узел находится в необходимом согласно сценарию состоянии, никаких изменений производиться не будет.

Помимо проверки состояния системы в Ansible также предоставляется возможности проверки непосредственно playbook'ов и специальных команды перед внесением каких-либо изменений в конфигурационные файлы узлов. Данная особенность позволяет оценить изменения, которые будут проводиться на управляемых узлах без непосредственного применения этих изменений.

Ещё одной возможностью Ansible являются обработчики (handlers). Обработчики в Ansible используются для запуска определенной задачи только после того, как основной сценарий отработает и на узлы будут внесены все необходимые изменения. Они запускаются один раз и после выполнения всех сценариев в playbook'е. Обработчики позволяют, к примеру, запустить выполнение установленного в процессе выполнения сценария приложения, активировать автозапуск этого приложение и т.д.

Среди концепций Ansible также существует еще одна, называемая переменные. Данная концепция позволяет изменять выполняемый playbook в зависимости от параметров узла, на котором он исполняется. Переменные используются для учета различий между узлами, таких как версии пакетов или пути к файлам. Данная концепция позволяет более гибко настраивать выполнение сценариев, опираясь на текущее состояние конкретного узла в вычислительной инфраструктуре. [5]

Рассмотренные инструменты Ansible позволяют облегчить выполнение задач администратора по конфигурации узлов. В результате архитектура вычислительной инфраструктуры претерпевает небольшие изменения. В её составе появляется управляющий узел, хранящий все необходимые файлы для конфигурации вычислительных узлов. Абстрактно новая архитектура представлена на рисунке 2.

Система конфигурационного управления позволяет решить все перечисленные выше проблемы.

Администратору нет необходимости вручную прописывать конфигурации для новых вычислительных устройств, достаточно лишь включить новый узел в уже существующий конфигурационный пул и отправить на этот узел конфигурацию, всю дальнейшую настройку возьмет на себя система конфигурационного управления.

Вторая проблема конфигурации, связанная с необходимостью обновления каждого узла вычислительной инфраструктуры вручную, то есть с отсутствием централизованного управления конфигурациями, также решается. Администратору достаточно обновить один общий конфигурационный сценарий, хранящийся на управляющем узле, а затем распространить его на все управляемые узлы для их переконфигурации в соответствии с новыми требованиями.

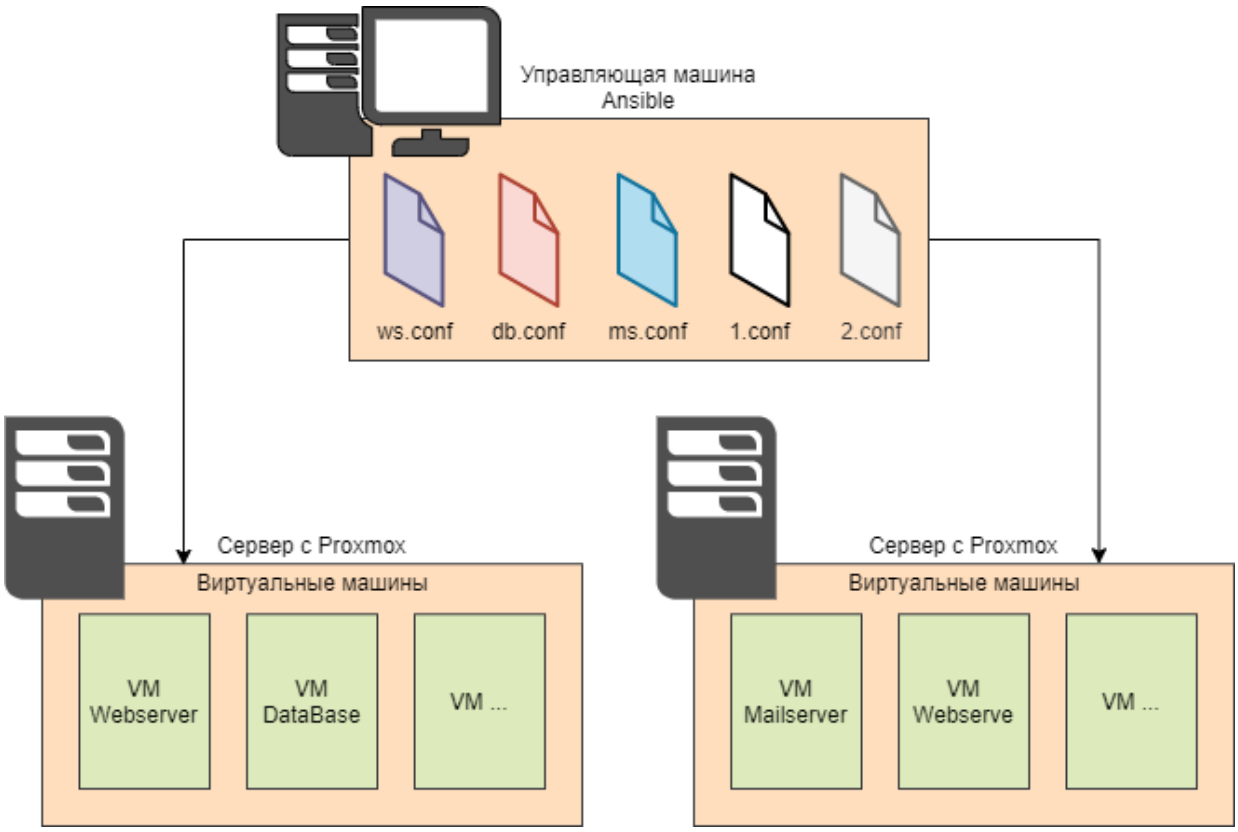

Рисунок 2 – Архитектура вычислительной инфраструктуры после внедрения Ansible

На основании этого сокращается количество возможных ошибок, которые может допустить администратор при настройке инфраструктуры. В результате представленная выше формула (2) будет иметь следующий вид (3):

$$E = \sum_{i=1}^{m} a_i r, \tag{3}$$

где m – количество единиц конфигурируемого ПО; a_i – количество изменений в конфигурации; r – среднее количество ошибок при осуществлении изменения.

Применение технологий конфигурационного управления позволяет сократить количество необходимых операций для настройки вычислительных узлов и приведения их к требуемому состоянию. Вследствие чего сокращается количество возможных ошибок, возникающих при настройке вычислительной инфраструктуры. В свою очередь это позволяет сократить количество возможных ошибок при функционировании всей инфраструктуры в целом. Помимо этого, внедрением системы конфигурационного управления позволяет улучшить дополнительные характеристики процесса управления конфигурацией, полный список которых представлен в таблице 1.

Таблица 1 – сравнение характеристик конфигурационного управления до и после внедрения системы конфигурационного управления

Характеристика	**Показатели без использования системы**	**Показатели при использовании системы**
Количество ошибок	$n\sum_{i=1}^{m}\sum_{j=1}^{p} a_{ij} r$	$\sum_{i=1}^{m} a_i r$
Количество вносимых изменений в конфигурацию всех узлов	$n\sum_{1}^{k} b_i$	$\sum_{1}^{k} b_i$
Количество источников информации	$n\sum_{1}^{k} d_i$	c

где n – количество вычислительных узлов, конфигурацию которых необходимо изменить; m – количество единиц конфигурируемого ПО; p – количество конфигурационных файлов для каждой единицы ПО; a_{ij} – количество изменений в каждом конфигурационном файле; r – среднее количество ошибок при осуществлении изменения; b_i – количество изменений на каждом узле; c – количество сценариев для группы узлов; d_i – количество конфигурационных файлов на узле.

Вывод

Система конфигурационного управления позволяет уменьшить количество ошибок, которые могут быть допущены администратором вычислительной инфраструктуры. Достигается это централизацией внесения изменений в конфигурационные файлы, а также снижением количества непосредственно конфигурационных файлов. Как результат, система позволяет облегчить процесс администрирования вычислительных узлов и облегчить процесс управления конфигурацией.

Список литературы

1. Serrano, João & Pereira, Rúben. (2020). Improvement of IT Infrastructure Management by Using Configuration Management and Maturity Models: A Systematic Literature Review and a Critical Analysis. Organizacija. 53. 3-19. 10.2478/orga-2020-0001.

2. Bellovin, Steven & Bush, Randy. (2009). Configuration management and security. Selected Areas in Communications, IEEE Journal on. 27. 268 - 274. 10.1109/JSAC.2009.090403.

3. Patrick Müller, Configuration Management – A Core Competence for Successful through-life Systems Engineering and Engineering Services, Procedia CIRP, Volume 11, 2013, Pages 187-192, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2013.07.032

4. https://www.redhat.com/en/topics/automation/what-is-configuration-management

5. https://www.redhat.com/en/topics/automation/learning-ansible-tutorial

6. Ali, Usman & Kidd, Callum. (2013). Configuration Management Process Capabilities. Procedia CIRP. 11. 169-172. 10.1016/j.procir.2013.07.043.

7. https://searchitoperations.techtarget.com/definition/configuration-management-CM

Reference

1. Serrano, João & Pereira, Rúben. (2020). Improvement of IT Infrastructure Management by Using Configuration Management and Maturity Models: A Systematic Literature Review and a Critical Analysis. Organizacija. 53. 3-19. 10.2478/orga-2020-0001.

2. Bellovin, Steven & Bush, Randy. (2009). Configuration management and security. Selected Areas in Communications, IEEE Journal on. 27. 268 - 274. 10.1109/JSAC.2009.090403.

3. Patrick Müller, Configuration Management – A Core Competence for Successful through-life Systems Engineering and Engineering Services, Procedia CIRP, Volume 11, 2013, Pages 187-192, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2013.07.032

4. https://www.redhat.com/en/topics/automation/what-is-configuration-management

5. https://www.redhat.com/en/topics/automation/learning-ansible-tutorial

6. Ali, Usman & Kidd, Callum. (2013). Configuration Management Process Capabilities. Procedia CIRP. 11. 169-172. 10.1016/j.procir.2013.07.043.

7. https://searchitoperations.techtarget.com/definition/configuration-management-CM