Southbridge
+
СЛЁРМ

IaC, Ansible, GitLab

Infrastructure as code на примере Ansible

Месропян Н. А.
nbw.adm+slurm@gmail.com

slurm.io

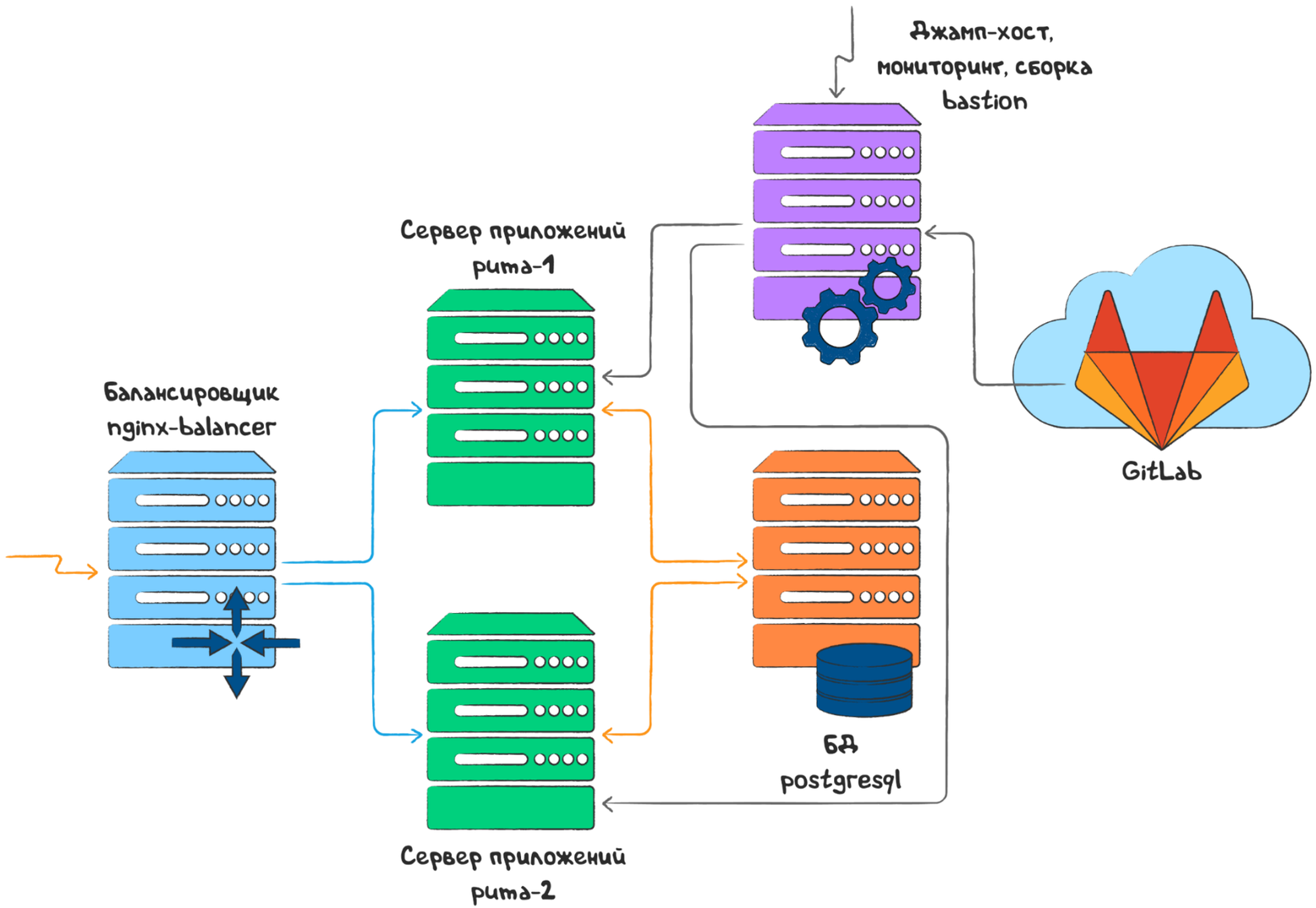
Pets or cattle?

Pets → cattle

Выбор SCM

pull	push
puppet labs	ANSIBLE
CHEF	
CFEngine	
SALTSTACK	

Стенд для Ansible

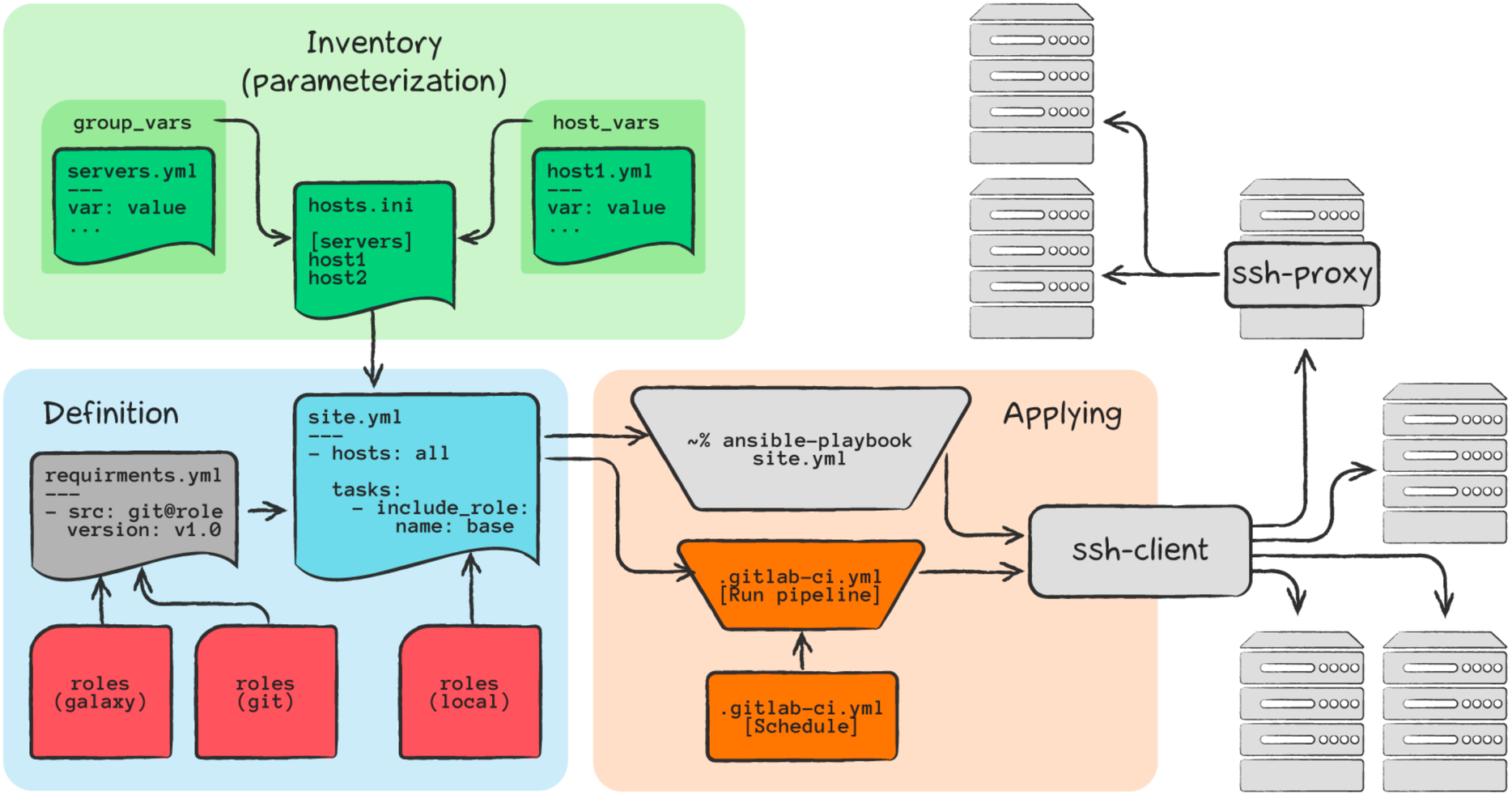


Работа с ansible-vault

Файл `~/.vpasswd`

```
ansible-vault encrypt
ansible-vault encrypt_string
```

Ansible data flow



Шаблон → переменные → результат

```
{% for key, value in base_sysctl_vars|dictsort %}
{{ key }} = {{ value }}
{% endfor %}
```

```
base_sysctl_vars:
 net.ipv4.ip_forward: 1
 net.netfilter.nf_conntrack_upd_timeout: 0
```

```
net.ipv4.ip_forward = 1
net.netfilter.nf_conntrack_upd_timeout = 0
```

Чуть более сложный пример

```
# THIS FILE IS ANSIBLE MANAGED.
# HANDS OFF!

# - Connection Settings -

listen_addresses    = '{{ postgresql_conf['listen_addresses']|default('*') }}'
port          = {{ postgresql_conf['port']|default('5432') }}
max_connections     = {{ postgresql_conf['max_connections']|default('100') }}

# - Memory -

{% set auto_shared_buffers = ansible_memtotal_mb / 100 * 20 %}
shared_buffers     = {{ postgresql_conf['shared_buffers']|default(auto_shared_buffers|round(-1)|int ~ 'MB') }}
temp_buffers       = {{ postgresql_conf['temp_buffers']|default('8MB') }}
{% set auto_work_mem = ansible_memtotal_mb / 100 * 1 %}
work_mem           = {{ postgresql_conf['work_mem']|default(auto_work_mem|round(0)|int ~ 'MB') }}
{% set auto_maintenance_work_mem = ansible_memtotal_mb / 100 * 4 %}
maintenance_work_mem    = {{ postgresql_conf['maintenance_work_mem']|default(auto_maintenance_work_mem|round(0)|int ~ 'MB')
{% if postgresql_conf['max_stack_depth'] is defined %}
max_stack_depth    = {{postgresql_conf['max_stack_depth'] }}
{% endif %}

```

Вычисляемые значения по умолчанию;

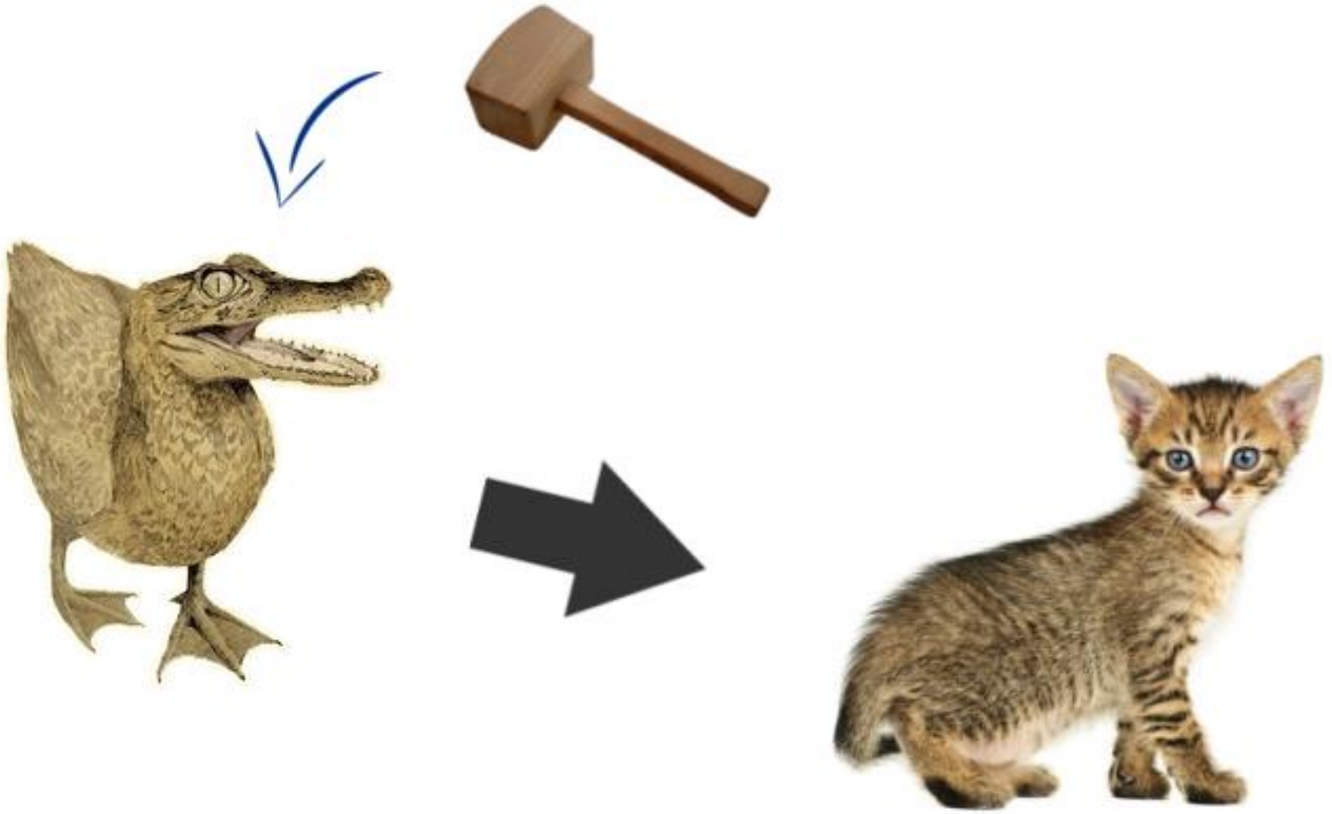
Фильтры;

Условия.

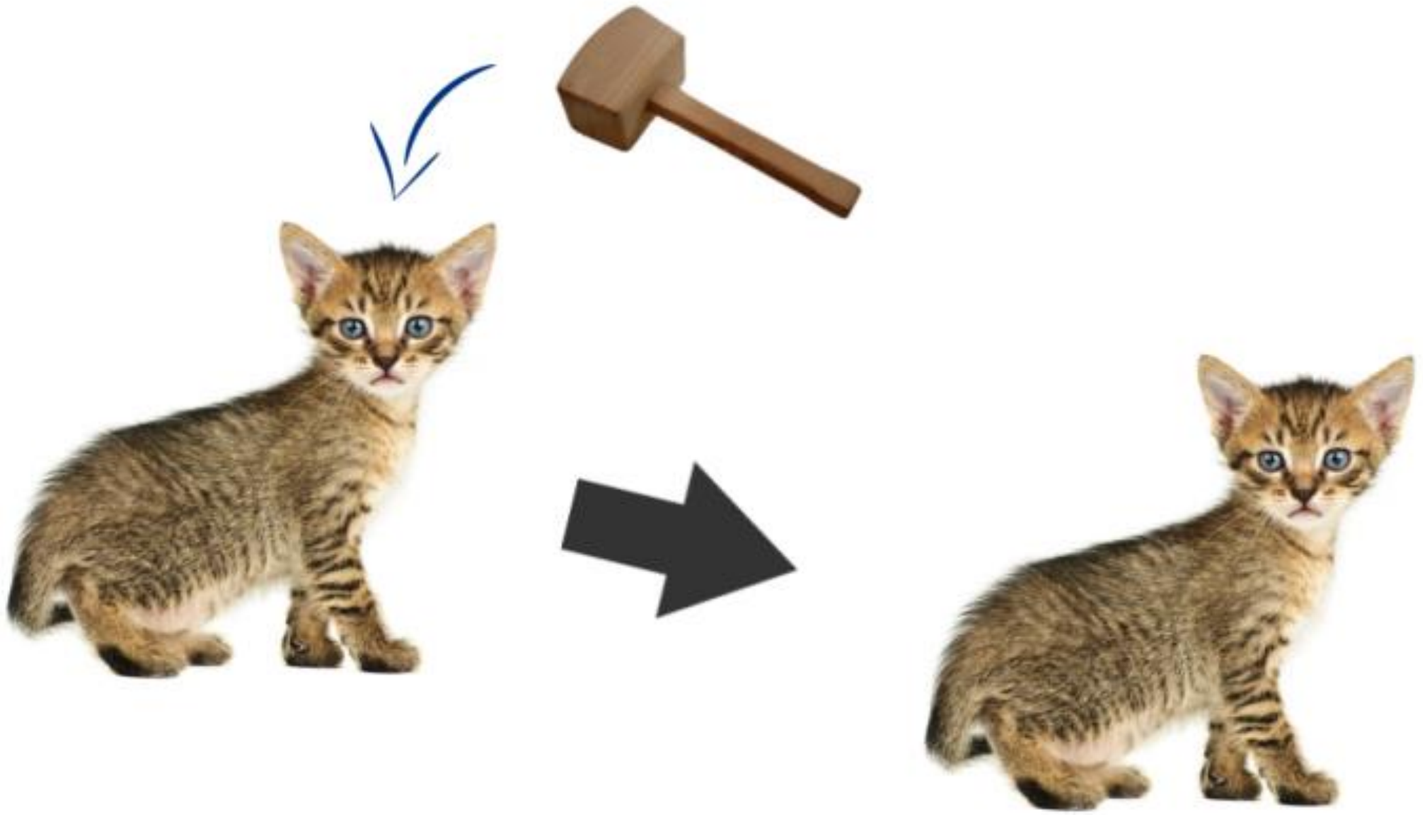
Идемпотентность | Idempotence

Свойство, означающее,
что действие может
многократно применяться к объекту
без изменения результата.

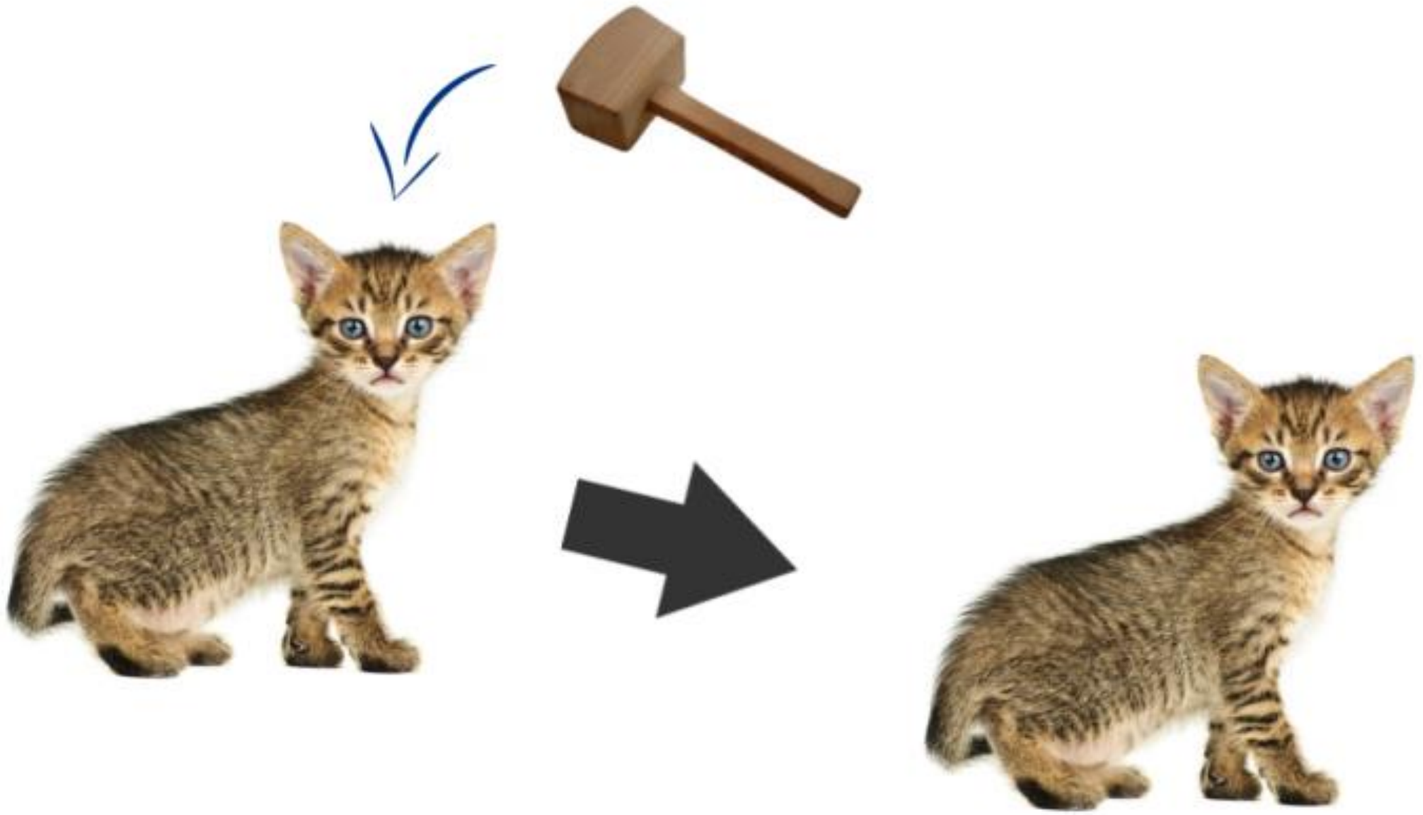
Идемпотентность | Idempotence



Идемпотентность | Idempotence



Идемпотентность | Idempotence



Антипаттерны

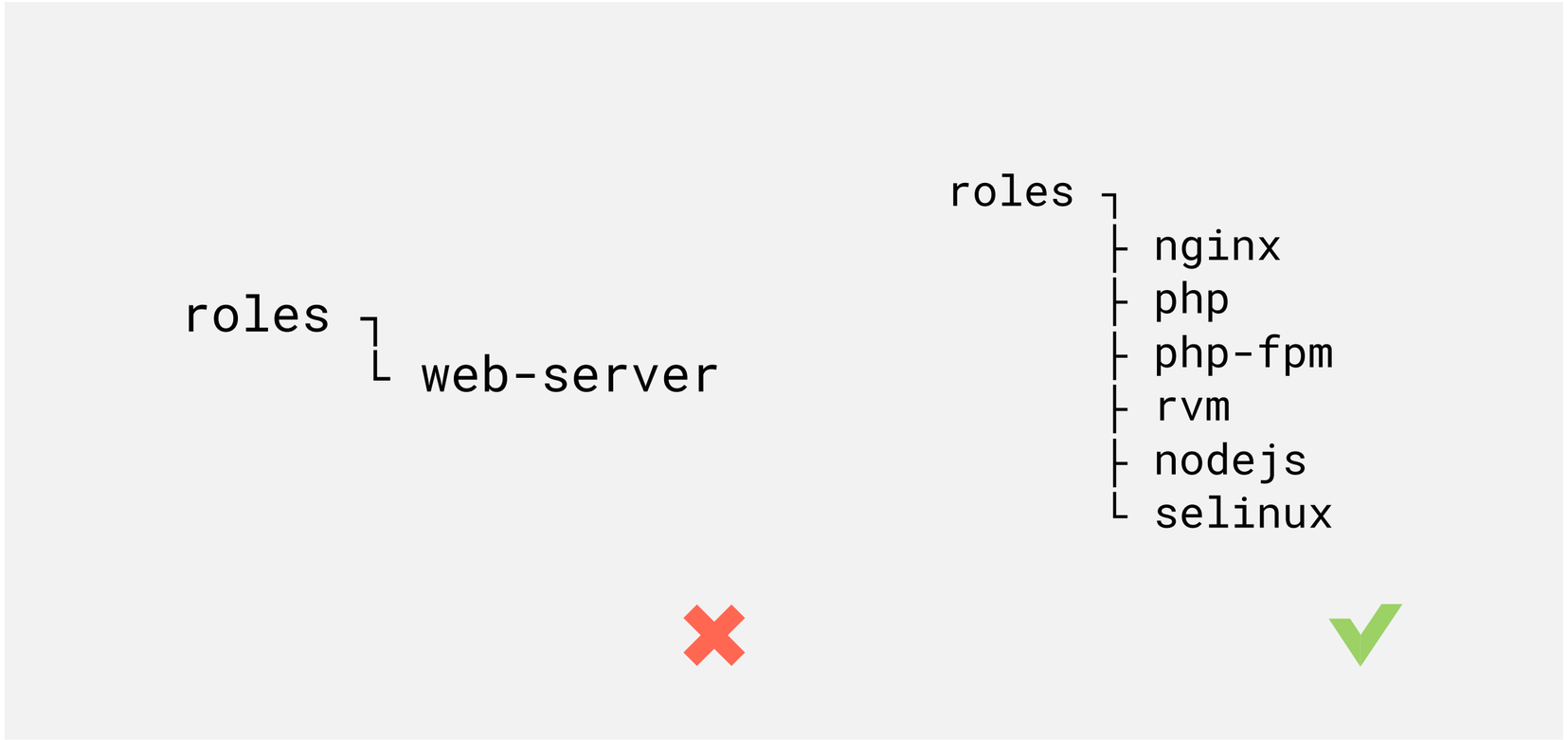
Антипаттерн – это распространенный подход
к решению класса часто встречающихся проблем, являющийся неэффективным
и имеющий высокий риск вызвать некорректное или отличное от ожидаемого поведение программ.
Антипаттерны относятся к *ошибкам проектирования*.

Неоправданное использование command/shell

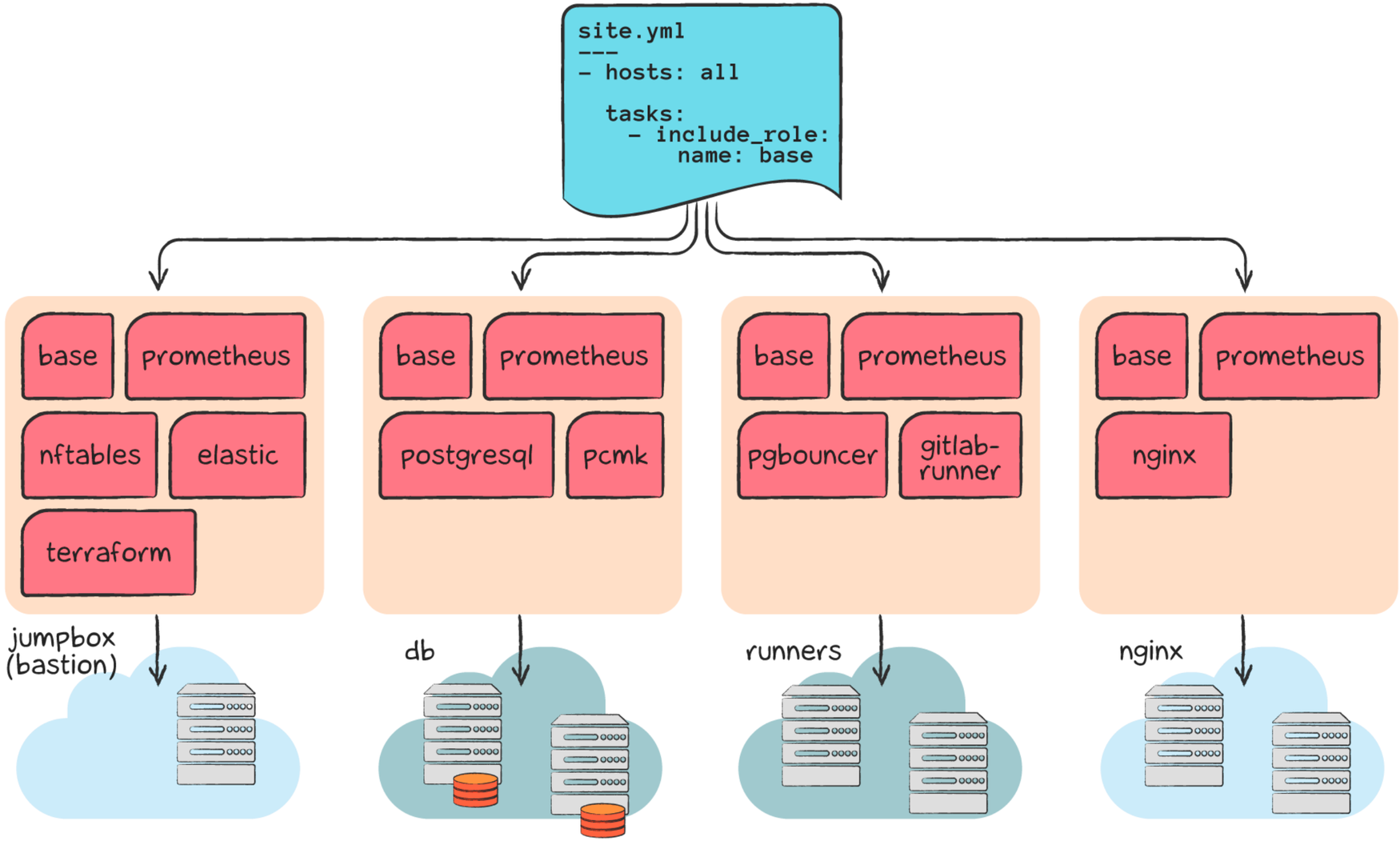

```
- name: Incorrect hostname setting
  command: hostnamectl set-hostname node1.example.com
```

```
- name: Correct hostname setting
  hostname:
      name: node1.example.com
```

Всеобъемлющие роли



Кирпичики конфигурации



lineinfile вместо template

```yaml
- name: Incorrect sudo setup
  lineinfile:
    path: /etc/sudoers
    regexp: '%wheel.*NOPASSWD'
    line: '%wheel ALL=(ALL) NOPASSWD: ALL'
```

✖

```yaml
- name: Correct sudo setup
  template:
      src: etc/sudoers.j2
      dest: /etc/sudoers
      owner: root
      group: root
      mode: "0660"
```

✔

```
# This file managed by ansible
# sudo/templates/etc/sudoers.j2
Defaults !visiblepw
Defaults env_reset
{% if sudo_notrequiretty %}
Defaults !requiretty
{% endif %}
root   ALL=(ALL) ALL
%wheel ALL=(ALL) NOPASSWD: ALL
```

Неиспользование групп хостов

```
[foo]
bar1.example.com
bar2.example.com
```

```yaml
- name: Play for all hosts
  hosts: all

  tasks:
   - name: >
       Include role bar
       conditionally
     include_role:
       name: bar
     when: foo_var is defined
```

```yaml
- name: Play for foo hosts
  hosts: foo

  tasks:
   - name: Import role bar
     import_role:
        name: bar
```

"лишняя группа лучше дополнительной логики"

import_role с параметрами

```yaml
# play.yml
---
- hosts: localhost

      vars:
          foo: 1
          ansible_connection: local

      tasks:
         - debug:
                var: foo

         - import_role:
                name: print
           vars:
                foo: 2

         - import_role:
                name: print
           vars:
                foo: 3

         - debug:
                var: foo
```

```yaml
# roles/print/tasks/main.yml
---
- name: print var
  debug:
    var: foo
```

~~OUTPUT: 1, 2, 3, 1~~

OUTPUT: 3, 2, 3, 3

"Всё, что вы делаете, имеет глобальный сторонний эффект"

Проверить регистрацию раннеров

Проверяем регистрацию раннеров в своей группе проектов:

« Settings → CI/CD → Runners »

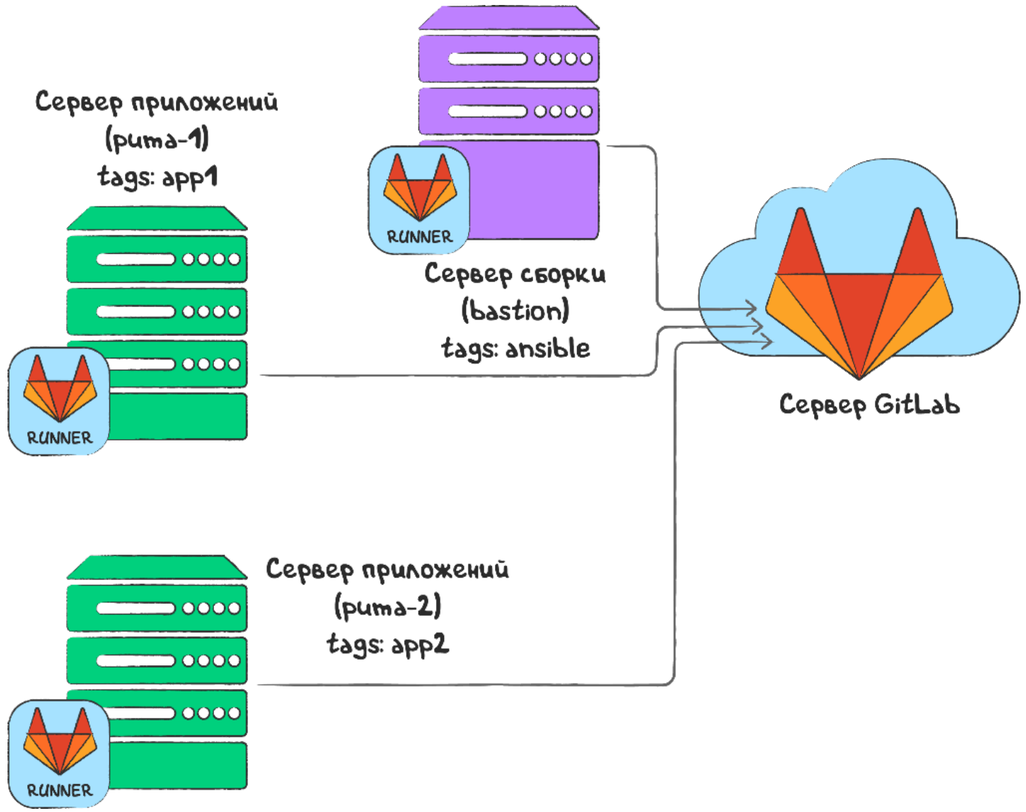
Теги :

`ansible` для сервера сборки

`app1` для первого сервера приложений

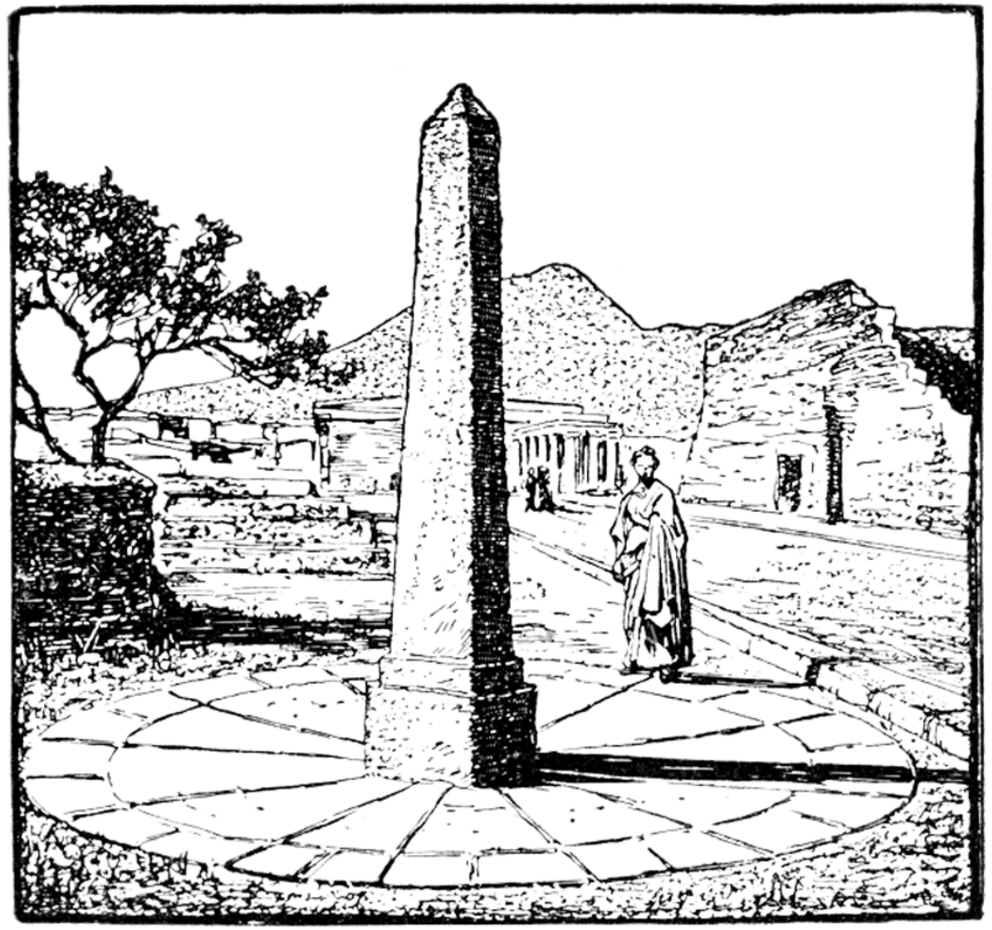
`app2` для второго.

Тип – shell



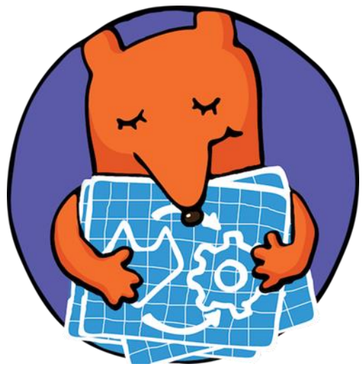
Запуск по расписанию

Для поддержания постоянной конфигурации можно запускать site.yml по расписанию

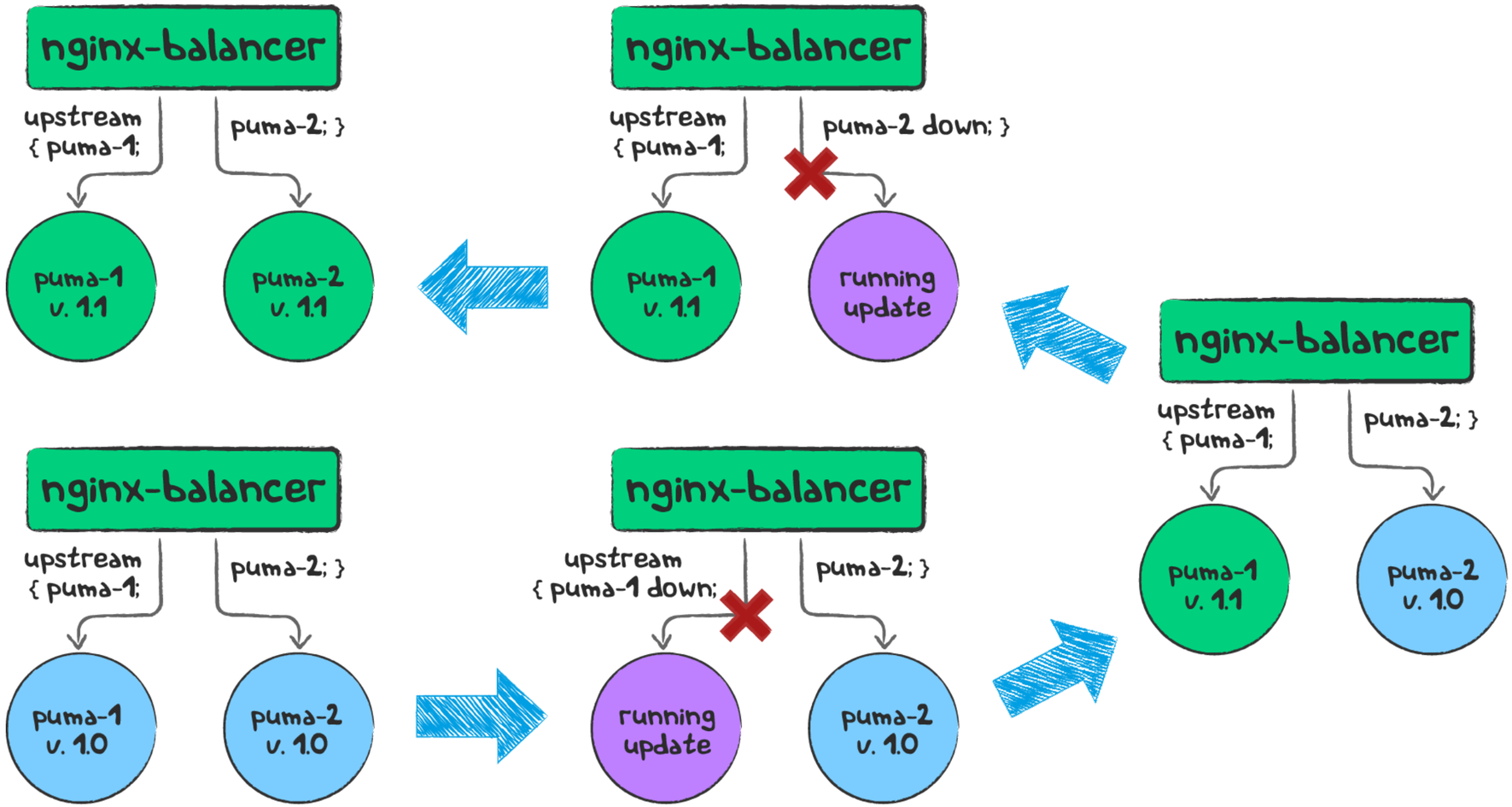


Цели пайплайна (пример)

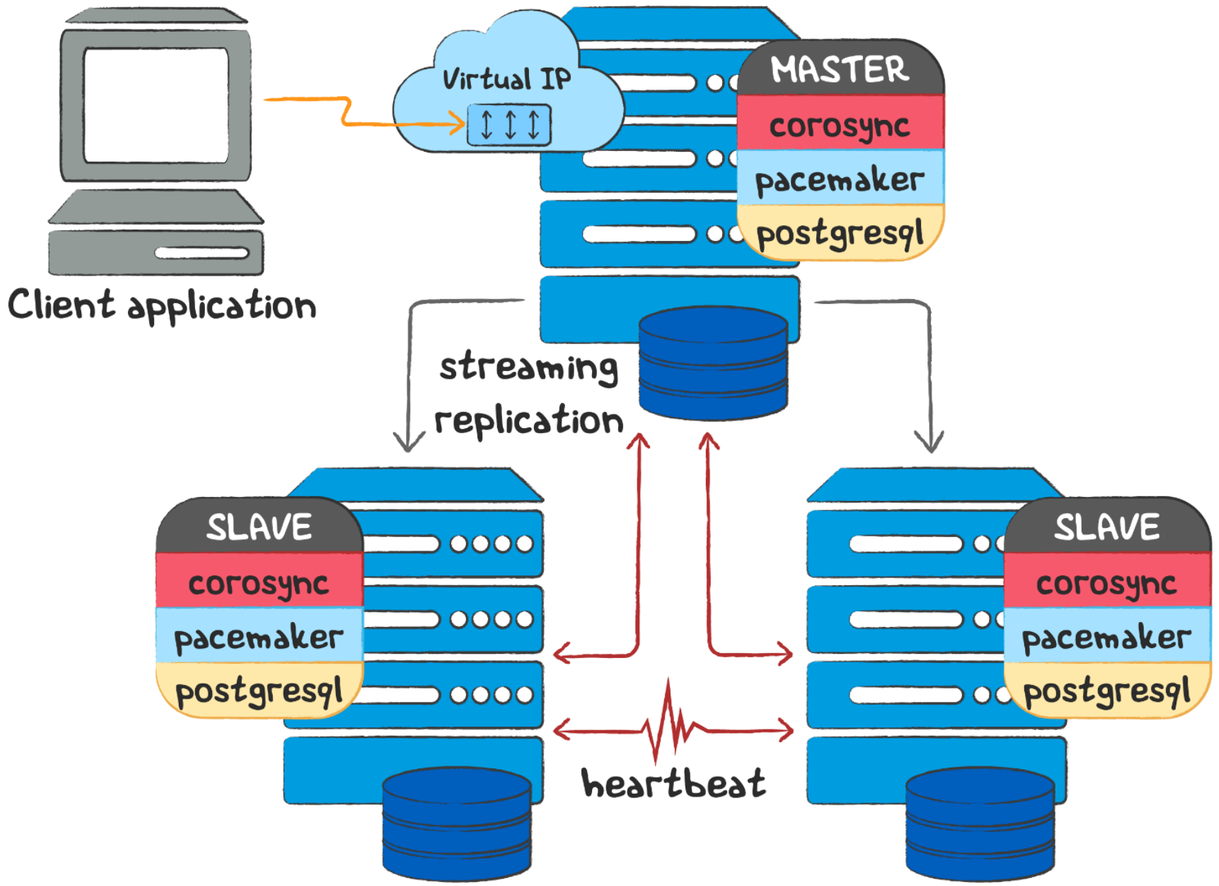
- **скачивание зависимостей, формирование ассетов (CSS, JS);**
- **тестирование работоспособности сборки;**
- **обновление версии приложения на площадке без перерыва в обслуживании;**
- **отсутствие простоя в случае сбоя обновления.**



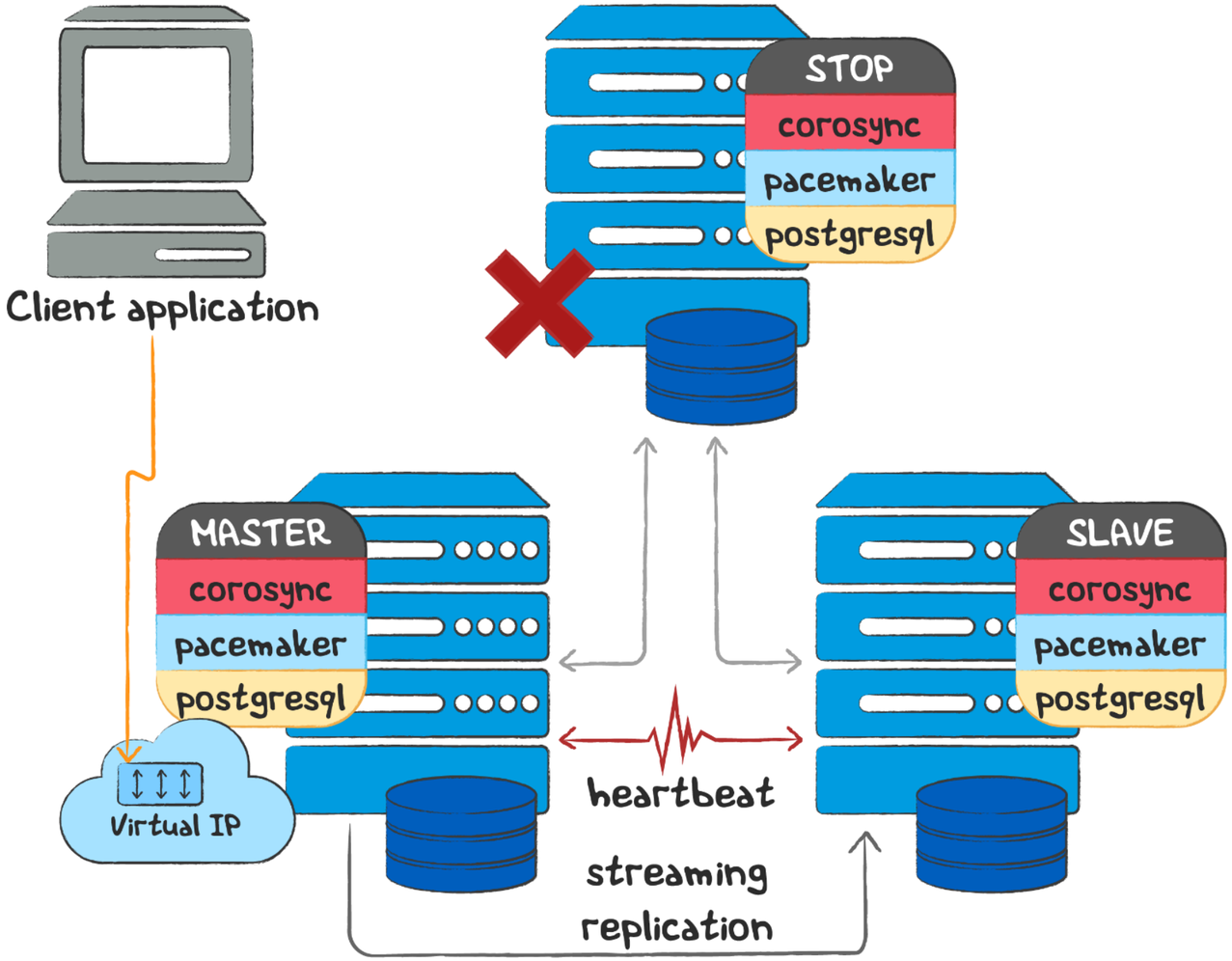
Rolling Update



Кластер PostgreSQL-over-Pacemaker



Кластер PostgreSQL: обработка отказа



Вывод

Ansible позволяет реализовывать IaC