Google OAuth 기반 OpenStack 개발 환경 자동화

사용자 인증부터 팀 공유, 리소스 회수까지

금동하

HAMONSOFT (2010~)

다루고자 하는 내용

OpenStack 기반 개발 환경 자동화

#Auth #Cloud-init #Ansible

자동화 이후 신경 쓰면 좋은 것들

#Bation #DNS #Monitoring

A platform crafted by developers, for developers

VM은 왜 항상 모자랄까?

왜 OpenStack 이였을까?

외부 서비스들

보안 정책 준수가 어려울 수 있음

높은 비용 구조

24시간 가동 시
도입하기 어려운 가격

내부 리소스 접근 제한

기업 내부 네트워크와
자원에 대한 접근이 제한적

컨테이너가 아닌 OS가 필요

k8s , 커널파라미터 등 수정

이러한 외부 서비스들은 기업 환경에서 도입하기 어려운 한계가 있음

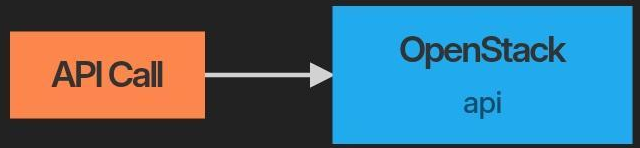
자동화의 시작은 API 로부터

API 액세스

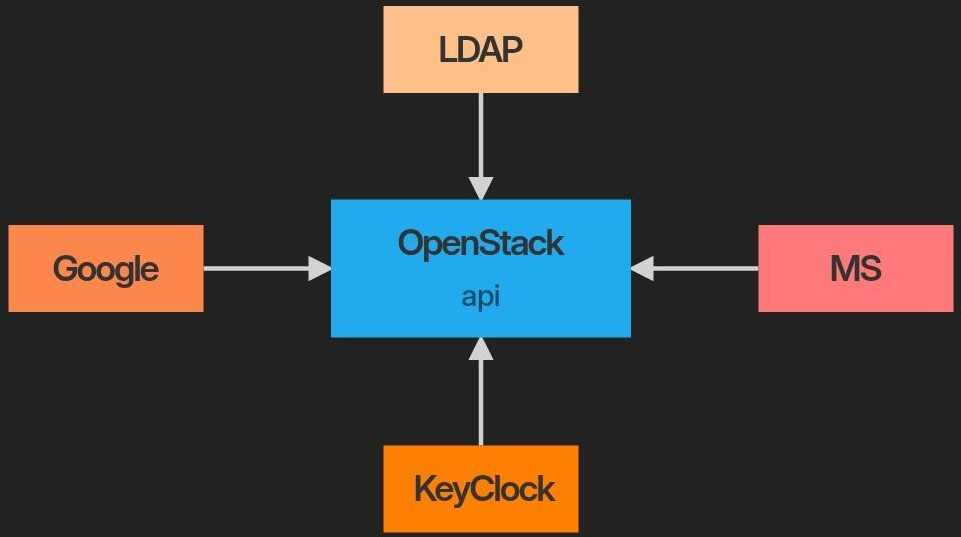
7 항목 표시

서비스	서비스 Endpoint
Block Storage	http://172.16.20.10/volume/v3
Compute	http://172.16.20.10/compute/v2.1
Compute_Legacy	http://172.16.20.10/compute/v2/1e99
Identity	http://172.16.20.10/identity
Image	http://172.16.20.10/image
Network	http://172.16.20.10/networking
Placement	http://172.16.20.10/placement

7 항목 표시



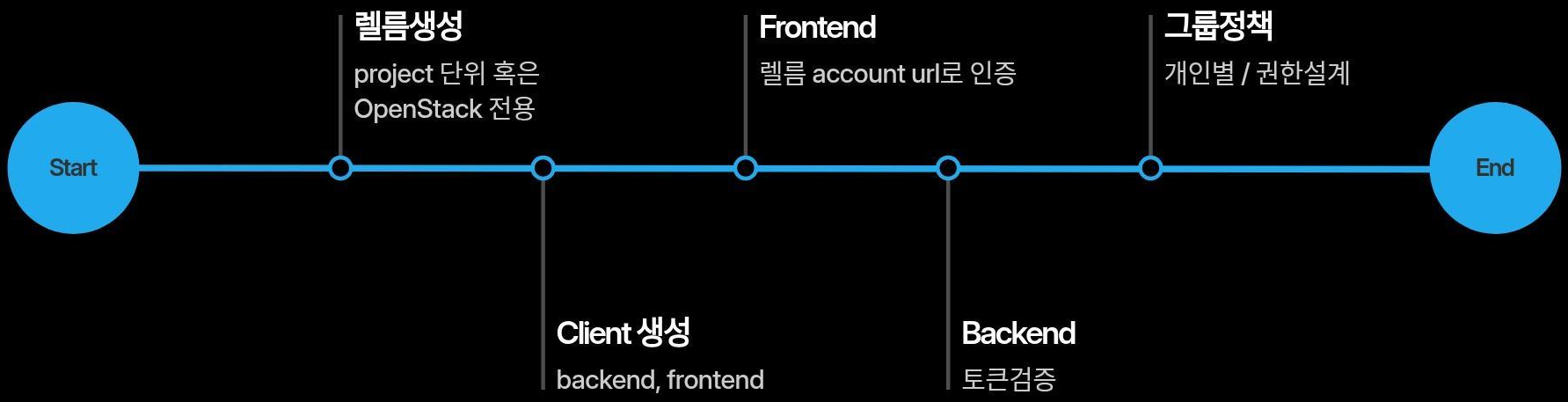
어느 인증 정보를 사용하는게 좋을까



Keyclock 연동

- **높은 커스터마이징 가능성**
  특정 역할 매핑이나 사용자 속성 추가
- **독립적인 사용자 관리 가능**
  외부 서비스 장애에 영향을 받지 않음
- **다양한 프로토콜 지원**
  SAML2, OAuth2 등 여러 인증 프로토콜을 지원
- **멀티 테넌시 지원**
  Realm 기능을 활용하여 논리적 구분

KeyClock을 연동한다면



Google OAuth 연동

```
const authOptions = {
  providers: [
    KeycloakProvider({
      clientId: "nextjs-app",
      clientSecret: "-",
      issuer: "https:///realms/openstack-
dev",
      authorization: {
        params: {
          kc_idp_hint: "google",
          scope: "openid email profile",
        },
      },
    }),
  ]
}
```

Google과 KeyClock 조합시 kc_idp_hint를 사용하면
KeyClock 로그인 페이지 없이 구글로만 인증 된 것처럼 보임

- **취약한 비밀번호 관리 방지**
  비밀번호 공유 혹은 만료시 한 글자만 바꾸기
- **Google WorkSpace 통합**
  조직이 Google 기반의 앱을 사용할때 효과적

사용자 팀 매핑

외부 DB 설계를 통한 사용자 관리

사용자 정보를 별도의 외부 데이터베이스에서 관리하여 유연성과 확장성을 확보합니다.

팀 단위 권한 제어

사용자를 팀 단위로 분류하여 각 팀별로 차등화된 리소스 접근 권한을 부여합니다.

동적 권한 조정

사용자의 소속 팀 변경 시 자동으로 권한이 조정되어 신속하게 대응할 수 있습니다.

팀 변경에 따른 자동 처리

팀 변경 시 리소스 할당, 접근 권한 등이 자동으로 업데이트되어 효율적인 관리가 가능합니다.

가시성 제어 구현

사용자 및 팀별로 리소스 현황을 확인할 수 있는 대시보드를 제공하여 투명성을 확보합니다.

Cloud-init

VM 초기 설정 자동화

표준 설정을 수행

VM 생성 시점에 컨트롤

SSH KEY 및 ID , 네트워크 설정

패키지 설치 자동화

개발에 필요한 패키지를
자동 설치

개발 도구 사전 구성

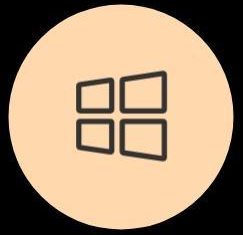
IDE, 버전관리 시스템 등 개발에
필요한 도구들을 구성

Cloud-init enables automated development environments, delivering standardized infrastructure and boosting developer productivity.

Ansible

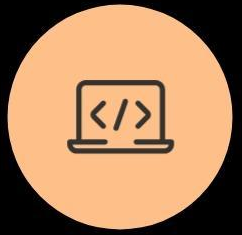
구성 관리 자동화

Ansible을 통해 일관된 인프라 구성 관리 가능



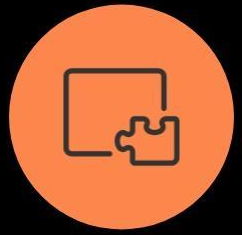
일관된 환경 설정

Ansible 플레이북으로 개발 환경 표준화 및 일관성 확보



스크립트 기반 프로비저닝

선언형 설정으로 신속한 VM 프로비저닝 가능

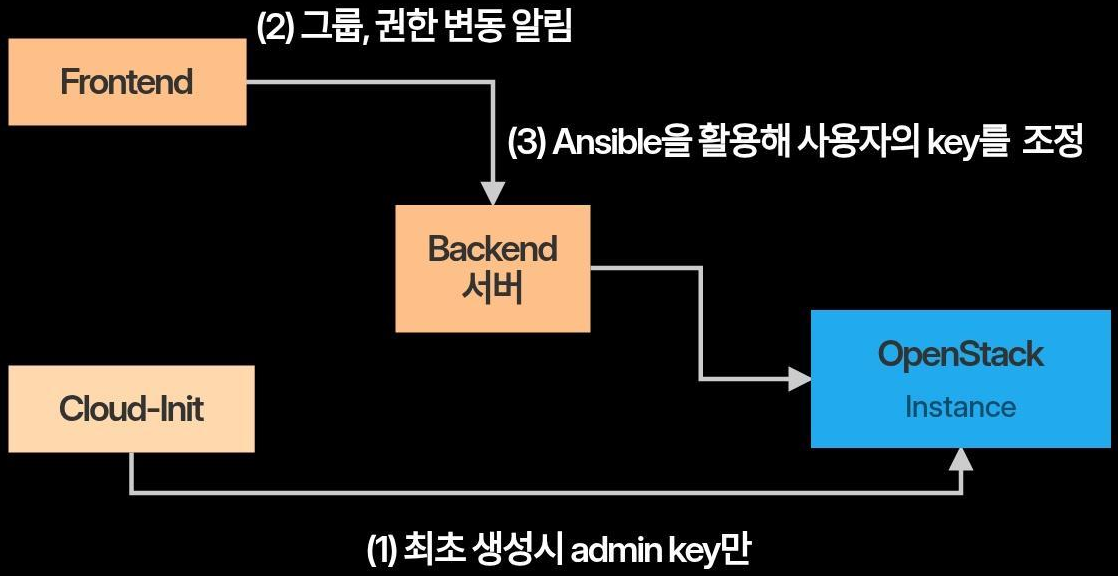


확장 가능한 구조

모듈화된 Ansible 설계로 추가 기능 쉽게 확장 가능

Ansible enables automated setup of development environments and scalable systems.

Cloud-init + Ansible



Bastion 호스트

보안 접근 통제

네트워크 및 리소스에 대한 접근을 엄격하게 제어하여 보안을 강화합니다.

네트워크 분리

개발 환경과 운영 환경을 물리적으로 분리하여 보안 경계를 명확히 합니다.

안전한 SSH 접근

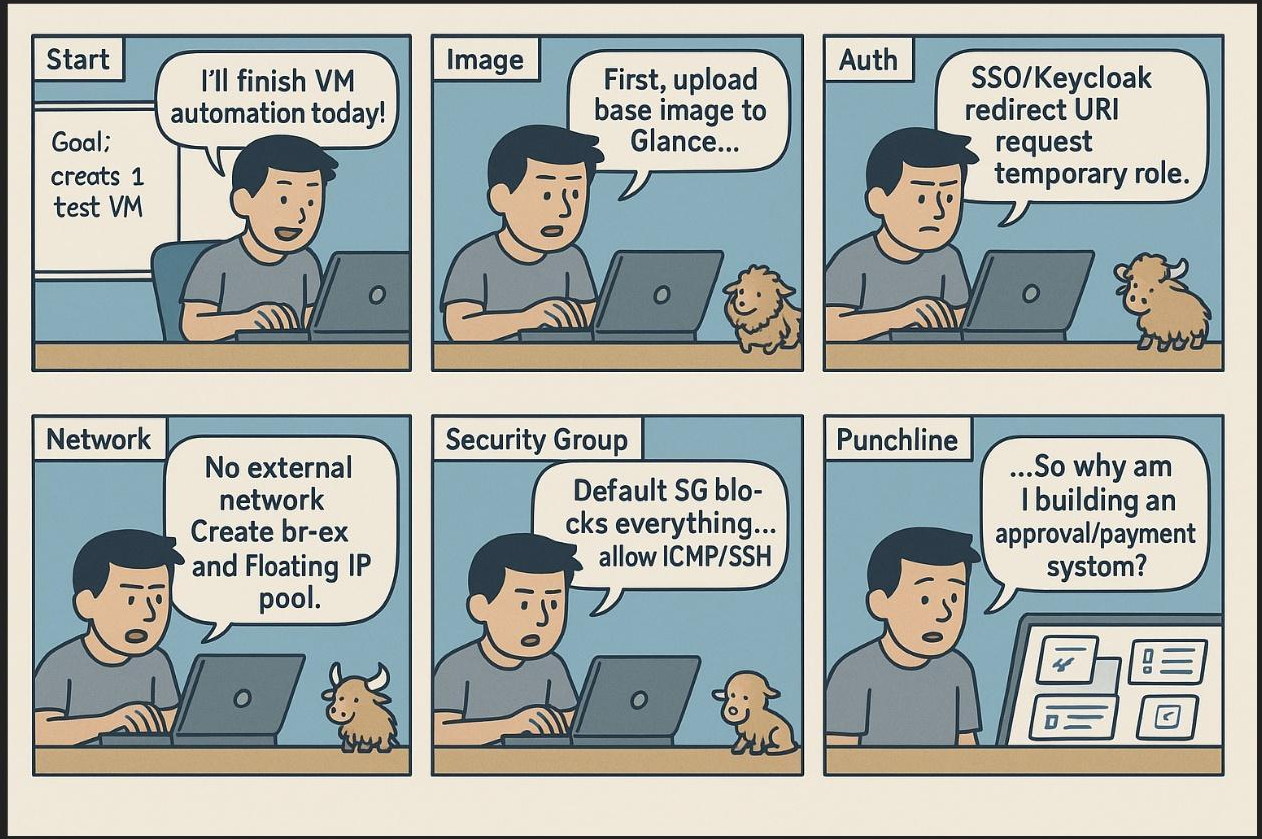
SSH 접근을 중앙에서 관리하고 강력한 인증 체계를 적용합니다.

중앙집중식 접근 관리

모든 접근을 Bastion 호스트를 통해 통제하여 보안을 강화합니다.

Bastion 호스트는 보안 접근 통제, 네트워크 분리, 안전한 SSH 접근, 중앙집중식 접근 관리 등의 기능을 제공하여 개발 환경의 보안성을 크게 향상시킵니다.

Yak Shaving?


Start
I'll finish VM automation today!
Goal: creats 1 test VM
Image
First, upload base image to Glance...
Auth
SSO/Keycloak redirect URI request temporary role.
Network
No external network Create br-ex and Floating IP pool.
Security Group
Default SG blo-cks everything... allow ICMP/SSH
Punchline
...So why am I building an approval/payment systom?

DNS 연동

BIND9 : 오픈소스

Public 서버에 설치하여 DNS 레코드를 조정한다

porkbun : API 제공

도메인만 구입하면 API가 공짜 (분당 60회 정도)

IP가 아니라 도메인으로 인스턴스를 접근하고 싶다면 DNS를 연동

porkbun 연동 예시

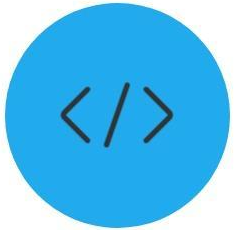
```python
import requests

API_KEY = "your_api_key"
SECRET_KEY = "your_secret_key"
DOMAIN = "your_domain"
BASE_URL = "https://api.porkbun.com/api/json/v3/dns"
AUTH = {"apikey": API_KEY, "secretapikey": SECRET_KEY}

# DNS 레코드 조회
def retrieve_records():
    response = requests.post(f"{BASE_URL}/retrieve/{DOMAIN}", json=AUTH)
    return response.json()["records"] if response.status_code == 200 else None

# 서브도메인 추가
def create_subdomain(subdomain, record_type, value, ttl=600):
    payload = {**AUTH, "name": subdomain, "type": record_type, "content": value,
"ttl": str(ttl)}
    response = requests.post(f"{BASE_URL}/create/{DOMAIN}", json=payload)
    return response.json()["id"] if response.status_code == 200 else None
```

웹 UI 소개

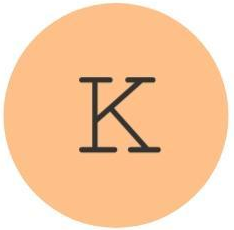


직관적인 관리 인터페이스

사용자 친화적인 UI로 어려운 설정 없이도 쉽게 인프라를 관리할 수 있습니다.

Horizon 대체 솔루션

OpenStack Horizon 대신 더 편리하고 강력한 관리 솔루션을 제공합니다.



사용자 친화적 설계

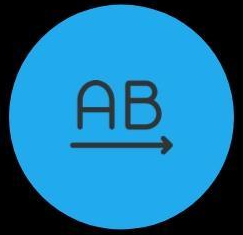
개발자의 관점에서 설계되어 개발 워크플로우와 완벽하게 통합됩니다.

팀별 리소스 현황

팀 단위로 리소스 사용량을 한눈에 파악할 수 있어 효율적인 자원 관리가 가능합니다.

이를 통해 개발자는 복잡한 설정 없이도 편리하게 개발 환경을 관리할 수 있습니다.

노코드 접근



개발자 중심 설계

개발자의 편의성과 생산성을 최우선으로 하는 설계

복잡한 설정 불필요

별도의 복잡한 설정 없이 쉽게 사용 가능

원클릭 환경 생성

한번의 클릭으로 개발 환경을 빠르게 구축

즉시 사용 가능한 개발환경

준비 없이 바로 개발에 착수할 수 있는 환경

개발자 중심의 노코드 접근으로 복잡한 설정 없이도 빠르고 편리한 개발 환경을 제공합니다.

보안 모니터링

내부자 위협 대응

직원들의 부적절한 행동이나 악의적인 접근 시도에 대한 탐지 및 차단

접근 로그 추적

사용자의 시스템 접근 기록을 상세히 추적하여 이상 행위 모니터링

이상 행위 탐지

비정상적인 로그인 시도, 데이터 유출 시도 등 이상 징후 감지 및 경고

보안 정책 준수

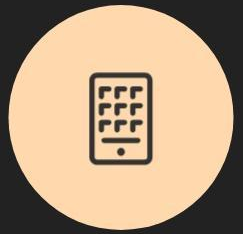
사내 보안 정책에 맞는 접근 제어 및 권한 관리를 통해 보안성 강화

내부 사용자에 의한 위협으로부터 시스템을 보호하고 보안 정책을 준수하는 것이 중요합니다. 이를 위해 접근 로그 추적, 이상 행위 탐지 등 다양한 보안 모니터링 기능이 필요합니다.

리소스 모니터링

TTL 기반 자동 회수

VM의 수명을 제한하여 자동으로 회수하는 기능



SMS 알림 시스템

리소스 사용량 임계치 초과 시 관리자에게 알림

리소스 사용량 추적

VM의 CPU, 메모리, 디스크, 네트워크 사용량을 실시간 모니터링

효율적인 자원 관리

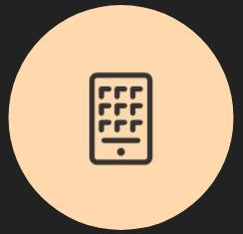
자동 회수와 모니터링을 통해 리소스 활용도를 높이고 비용을 절감

리소스 사용 현황을 실시간으로 모니터링하고, TTL 기반의 자동 회수 기능을 통해 개발 환경을 효율적으로 관리합니다.

내부자 위협에 대한 모니터링

TTL 기반 자동 회수

VM의 수명을 제한하여 자동으로 회수하는 기능



SMS 알림 시스템

리소스 사용량 임계치 초과 시 관리자에게 알림

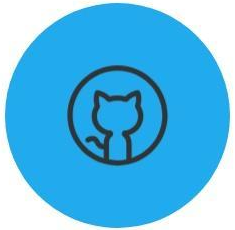
리소스 사용량 추적

VM의 CPU, 메모리, 디스크, 네트워크 사용량을 실시간 모니터링

효율적인 자원 관리

자동 회수와 모니터링을 통해 리소스 활용도를 높이고 비용을 절감

소스 공개



GitHub에 소스코드 공개

오픈소스로 소스코드를 공개하여 누구나 확인하고 기여할 수 있게 합니다.

커뮤니티 기여

오픈소스 커뮤니티와 협력하여 지속적인 개선과 발전을 이루어 나갑니다.

지속적인 개선

사용자 피드백과 커뮤니티 참여를 통해 시스템을 지속적으로 개선해 나갑니다.

투명한 개발 프로세스

모든 개발 과정을 투명하게 공개하여 신뢰성과 투명성을 높입니다.

오픈소스로 공개하여 커뮤니티와 함께 지속적인 발전을 이루어 나갈 것입니다.

감사합니다