

# slide-python-outfit

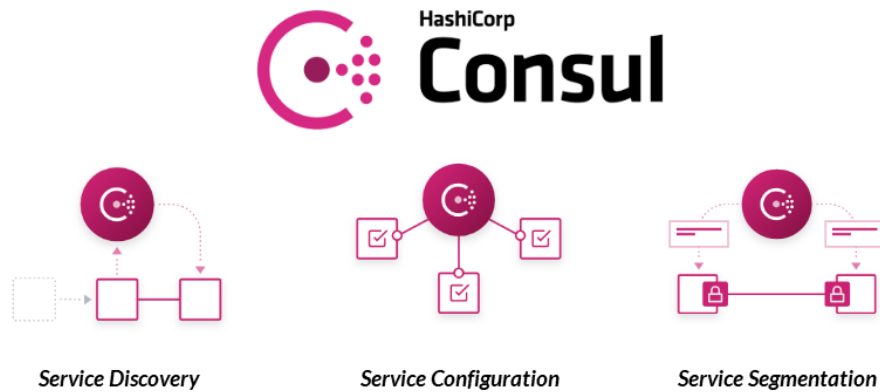
December 5, 2021

## 0.1 Managing Config in Python using Consul and Vault

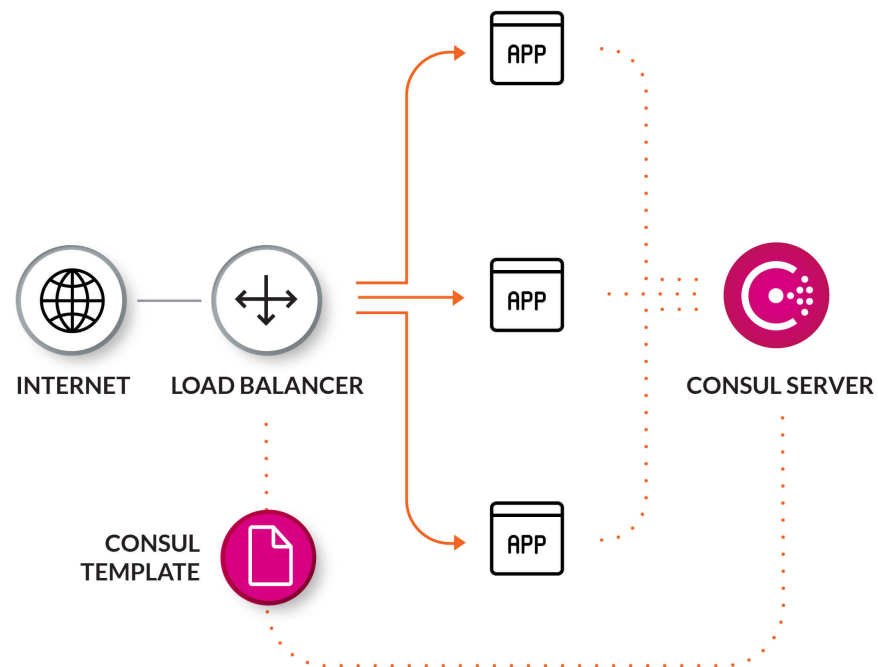
## 0.2 By Hendri Karisma

## 1 Introduction

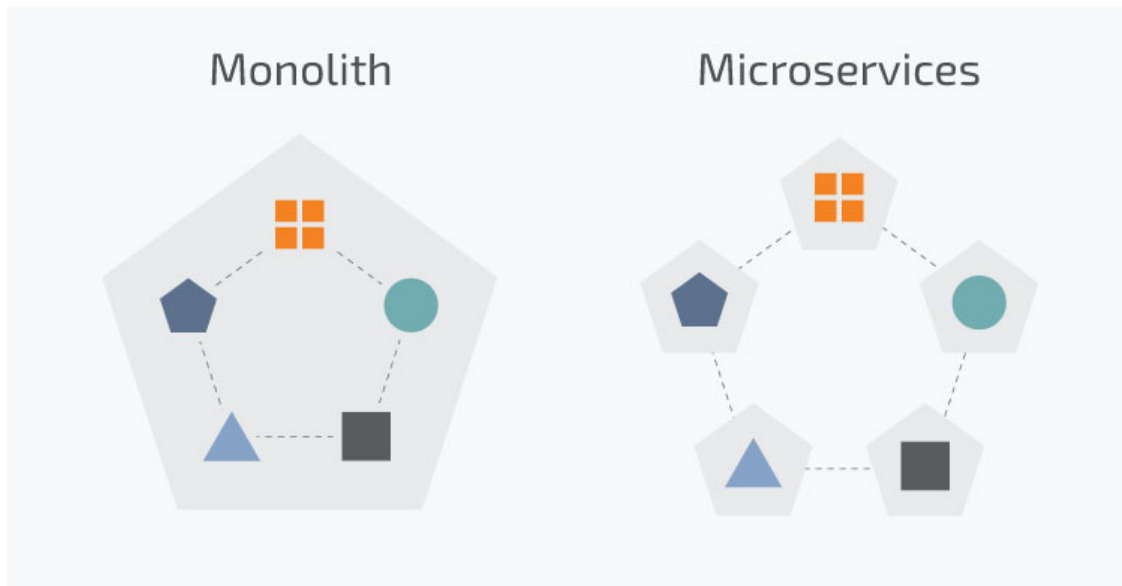
- Principal R&D Engineer at blibli.com
- Before working in AI Squad, now working in DevEx squad
- using python for AI research and in DevEx: helping engineers for several programming languages
- Selling food at Nevada Kitchen (home cuisine) you search and buy at blibli.com



## 1.1 Service Discovery



## 1.2 What happens inside monolith and Microservices



Monolith

Microservices

## 2 What is vault

### 2.0.1 a tool that help us to manage :

- Secret to be Centralized with ACL
- Dynamic secret and unique secret for each client
- Protecting data with encryption feature (high leve API or encryption as a service)

\*\* Secret could be : username or password, app credential, api token, TLS Cert, etc

## 3 Centralized Secret

## 4 Centralized Secret

## 5 How python integrate with consul and vault ?

- Using different libraries for vault and consul
- hvac for vault client and python consul for consul

## 6 How to integrate consul and vault in python ?

1. Install hvac python client and python consul library

2. for python consul : # could use env var to load token client = consul.Consul(token=token)

```
client.kv.get('foo')           # OK
client.kv.put('foo', 'bar2')   # raises ACLPermissionDenied

client.kv.get('private/foo')   # returns None, as though the key doesn't
                                # exist - slightly unintuitive
client.kv.put('private/foo', 'bar2') # raises ACLPermissionDenied
```

3. for python vault :

```
# could use env var for the config info
client = hvac.Client(
    url='https://localhost:8200',
    token=os.environ['VAULT_TOKEN'],
    cert=(client_cert_path, client_key_path),
    verify=server_cert_path,
)
client.is_authenticated()
create_response = client.secrets.kv.v2.create_or_update_secret(
    path='foo',
    secret=dict(baz='bar'),
)

# Read the data written under path: secret/foo
read_response = client.secrets.kv.read_secret_version(path='foo')
print('Value under path "secret/foo" / key "baz": {val}'.format(
```

```

        val=read_response['data']['data']['baz'],
    ))

```

## 7 How java spring integrate with consul and vault ?

1. Add spring cloud dependency
2. Add spring.cloud.vault.\* or spring.cloud.consul.\* (could use yaml or .properties)
3. create class/bean to load configuration done

## 8 Sample bootstrap.properties in springboot

```

#Vault Configuration
spring.cloud.vault.enabled=${VAULT_ENABLE:true}
spring.cloud.vault.host=${VAULT_HOST:localhost}
spring.cloud.vault.port=${VAULT_PORT:8300}
spring.cloud.vault.fail-fast=true
spring.cloud.vault.token=${VAULT_TOKEN}
spring.cloud.vault.scheme=${VAULT_SCHEME:http}
spring.cloud.vault.generic.backend=secret/config

#Consul Configuration
spring.cloud.consul.enabled=${CONSUL_ENABLE:true}
spring.cloud.consul.host=${CONSUL_HOST:localhost}
spring.cloud.consul.port=${CONSUL_PORT:8500}
spring.cloud.consul.config.fail-fast=true
spring.cloud.consul.config.format=properties
spring.cloud.consul.config.prefix=config

```

## 9 Python Outfit

### 9.1 what is python-outfit?

- help to integrate python with consul and vault easily
- just add single file for vault and consul configuration
- add config file for logging

### 9.2 How to integrate with Consul and Vault using python-outfit ?

- add yaml file that contain configuration information for vault and consul that will read by python-outfit

```

vault:
  host: ${VAULT_HOST}
  port: ${VAULT_PORT}
  scheme: ${VAULT_SCHEME}
  token: ${TOKEN_VAULT}
  path: ${VAULT_PATH}
consul:

```

```

    host: ${CONSUL_HOST}
    port: ${CONSUL_PORT}
    scheme: ${CONSUL_SCHEME}
    token: ${TOKEN_CONSUL}
    path: ${CONSUL_PATH}
logconfig:
    mode: ${LOG_MODE}
    source_type: ${LOG_TYPE}
    source_location: ${LOG_LOCATION}
    default_type: yaml_file
    default_location: conf/logging.yaml

```

### 9.2.1 add 5 to 6 lines for load configuration for consul and vault data

```

from outfit import Outfit
from outfit import ConsulCon, VaultCon
from outfit import Logger

if __name__ == '__main__':
    Outfit.setup('conf/configuration.yaml')
    con_consul = ConsulCon()

    # get the information such as config file from consul kv then will be returned as python d
    config_dict = con_consul.get_kv()

    con_vault = VaultCon()

    # get the secret information in vault secret kv then will be returned as python dictionary
    secret_dict = con_vault.get_secret_kv()

    # if you want to merge the vault data and consul data
    main_config = merge_dict(config_dict, secret_dict)

```

## 10 Outfit, ConsulCon, VaultCon, Logger

there 5 features that already added in python-outfit: \* Outfit \* ConsulCon \* VaultCon \* Logger  
 \* Dict merger function

### 10.1 Outfit

```
{ from outfit import Outfit }
```

The main static class to load the configuration file (yaml)

### 10.2 ConsulCon

```
{ from outfit import ConsulCon }
```

Consul class to load key value class

### 10.3 VaultCon

```
{ from outfit import VaultCon }  
Consul class to load the secret data
```

## 11 DEMO SAMPLE SIMPLE APP

[https://github.com/situkangsayur/staff\\_app](https://github.com/situkangsayur/staff_app) (to open the swagger : <http://localhost:5000/api/v1>  
) <https://github.com/situkangsayur/async-fastapi-mongo>

## 12 Thank you

Contribute to python-outfit : <https://github.com/situkangsayur/python-outfit> Contacs: \* email  
: [situkangsayur@gmail.com](mailto:situkangsayur@gmail.com) \* office email : [hendri.karisma@gdn-commerce.com](mailto:hendri.karisma@gdn-commerce.com) \* github :  
<https://github.com/situkangsayur> \* twitter : @infoHendri \* telegram : @siganteng