Secure Continuous Deployment for the Cloud

PROGRAMME
DE RECHERCHE
CLOUD

Ouail Derghal *(IMTA, P4S)*
<*ouail.derghal@imt-atlantique.fr*>

23/05/2024

Plan

1. General Information
2. Objectives
3. Infrastructure-as-Code
4. Self-Adaptive Systems
5. Potential Contributions

General Information

- **Title:** Secure Continuous Deployment for the Cloud
- **Keywords:** *DevOps, DevSecOps, Cloud Computing*
- **PEPR Project:** *Trust in Cloud*
- **Start Date:** *January 15th, 2024 (~4months)*
- **Thesis Direction:**
  - **Supervisor:** Fabien Dagnat *(IMTA)*
  - **Co-supervisor:** Jean-Cristophe Bach *(IMTA)*

Objectives

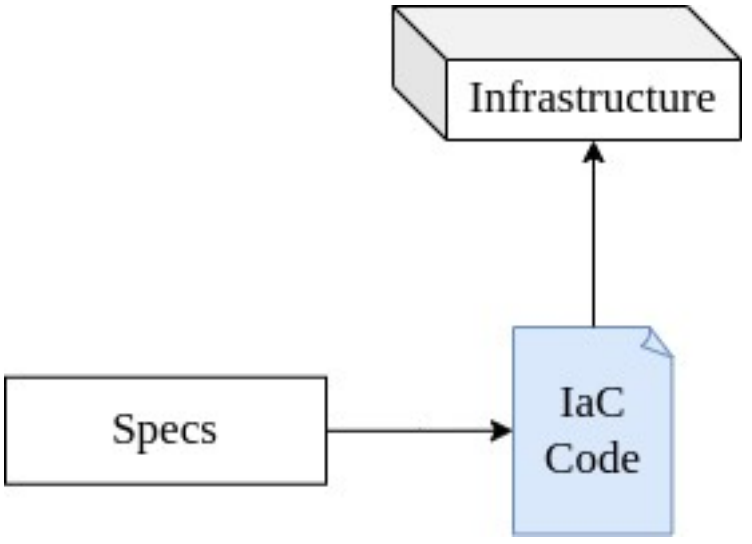
- Exploration of the DevOps, DevSecOps and deployment on the cloud.
- Proposition of an innovative approach that enables applications to detect and adhere to predefined requirements (security or other)
- Simplification of the adaption of applications to contextual changes (infrastructure evolution, vulnerability detection, ...)

Infrastructure-as-Code

- Raise of agile software and lean techniques resulted in a knowledge gap between developers and operators → DevOps Field.
- IaC is the main DevOps tool to manage large-scale infrastructures.
- Two main operations that can be performed on infrastructures using IaC tools: **provisioning** and **management**.
- Two distinct approaches to define and configure infrastructures as code:
  - **Code-centric approach**
  - ***Model-driven approach***

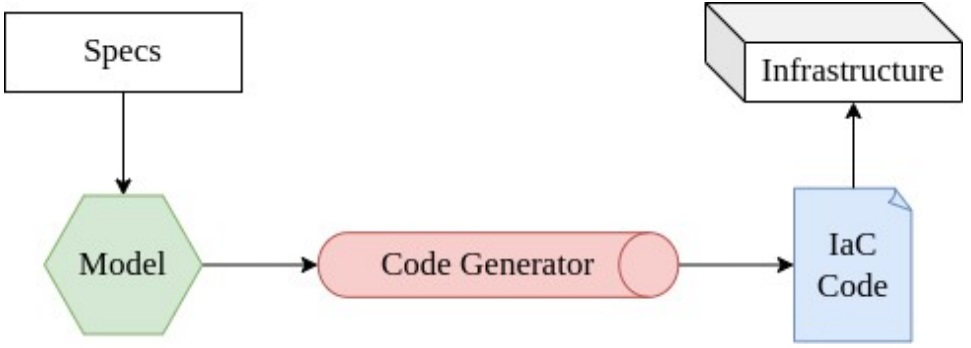
Code-centric Approach

- Define the desired state of the infrastructure in machine-readable code files.
- Used by engineers responsible on managing large-scale infrastructures.



Model-driven Approach

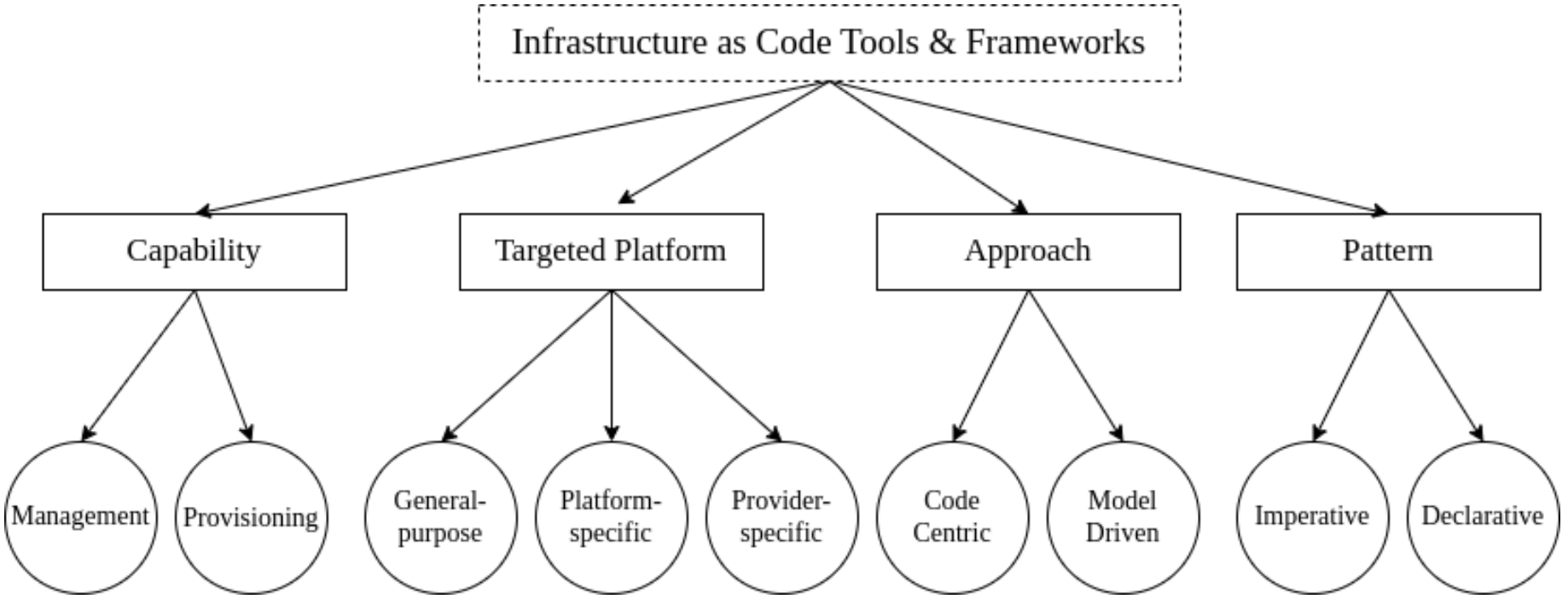
- Describe the desired state of the infrastructure as a model.
- Models can be checked and verified before the execution.
- Proposed by researchers to reduce the complexity and the technical expertise required to use IaC tools.



Categorization of IaC Tools

- **By capability:** management, provisioning
- **By targeted platform:** general-purpose, platform-specific, provider specific
- **By approach:** code-centric, model-driven
- **By pattern:** imperative, declarative

Categorization of IaC Tools



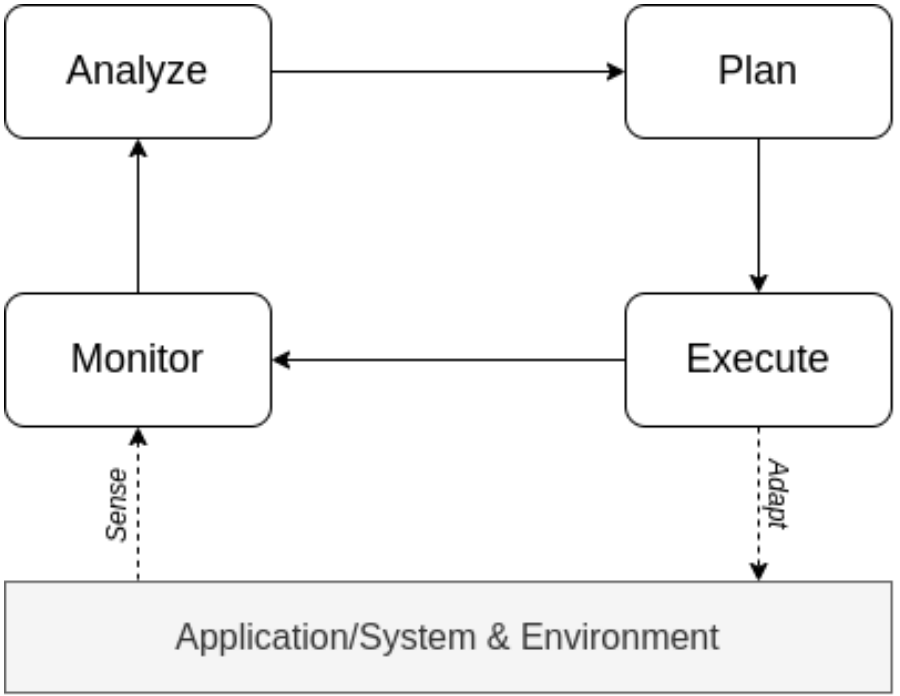
Self-Adaptive Systems

- Systems that can autonomously adjust their behavior in response to changes in their environment or internal state.
- This kind of systems can operate without human intervention.
- Adapt dynamically based on real-time data.
- Ensure resilience by maintaining functionality despite changes or disruptions.
- Improve performance through self-adjustments.

Core Components of SASs

- **Monitoring:** Continuously observe internal and external conditions.
- **Analysis:** Evaluate monitored data to detect anomalies.
- **Planning:** Devise strategies for adaptation based on analysis.
- **Execution:** Implement the planned adaptations.

The MAPE Cycle



RELAX Language

- Specialized language designed to handle the specification of requirements for self-adaptive systems.
- Capable to define and handle uncertainty.
- Introduces flexibility and dynamic requirement specification for unpredictable systems.
- The language has been defined in terms of **temporal fuzzy logic**.

Proof-of-concept Application

- SAL: self-adaptive logger
- The app is connected to a PostgreSQL database.
- The app compromises a feedback loop that continuously monitor and report changes in the environment.
- The app functions in 2 modes:
  1) Normal (*unencrypted)* mode
  2) Secure (*encrypted*) mode
- If the app detects a change in the environment, it will switch automatically to the secure mode.

DEMO: Self-Adaptive Logger

Future Improvements

- More-complex use-case scenario.
- Extract the adaption loop as a separate process.
- Define the adaption requirements using RELAX (*implement a RELAX language interpreter, MAYBE!*).
- In the case of the SAL application, the single adaption requirement can be defined as follows:
  - *“The system **SHOULD** encrypt **AS MUCH AS POSSIBLE** of the data in the database **IF** the environment fingerprint changes **ELSE** decrypt **AS MUCH AS POSSIBLE** the data”*
- Use IaC tools to trigger adaption scenarios (requires a more complex use-case).

Potential Contributions

- Extension and/or implementation of the RELAX language to define requirements for ***security-aware*** applications.
- Automation of the deployment process with a focus on **adaption** and **security**.
- Proposition of a standardized architecture/framework for *“security-aware applications”* development.
- Extension of an existing IaC solution (potentially *Ansible*) to support self-adaptive applications.

END.