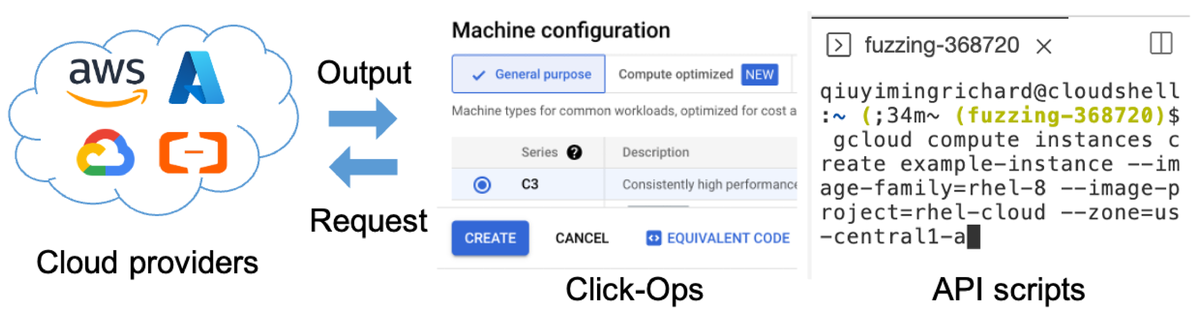
Simplifying Cloud Management with Cloudless Computing

Yiming Qiu, Patrick Tser Jern Kon, Jiarong Xing†, Yibo Huang, Hongyi Liu†
Xinyu Wang, Peng Huang, Mosharaf Chowdhury, Ang Chen

University of Michigan †Rice University

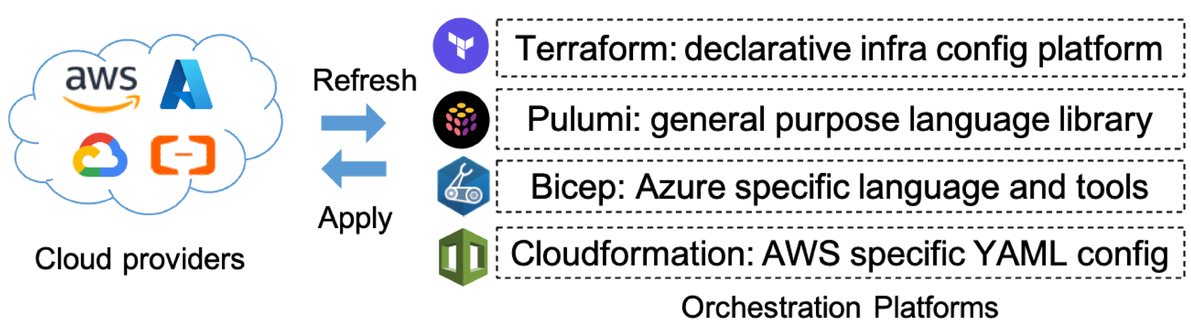
Cloud Automation

Cloud resources are difficult to configure and manage. Previously, cloud tenants have to employ extensive cloud/DevOps engineering teams to handle **obscure cloud service interfaces**:



These approaches could get the job done when operating a small development testbed, but have been reckoned extremely **inefficient and unreliable when scaling to large-scale production environments.**

Modern cloud automation frameworks arose to simplify cloud management. Most follow the **"Infrastructure as Code" (IaC) paradigm**:

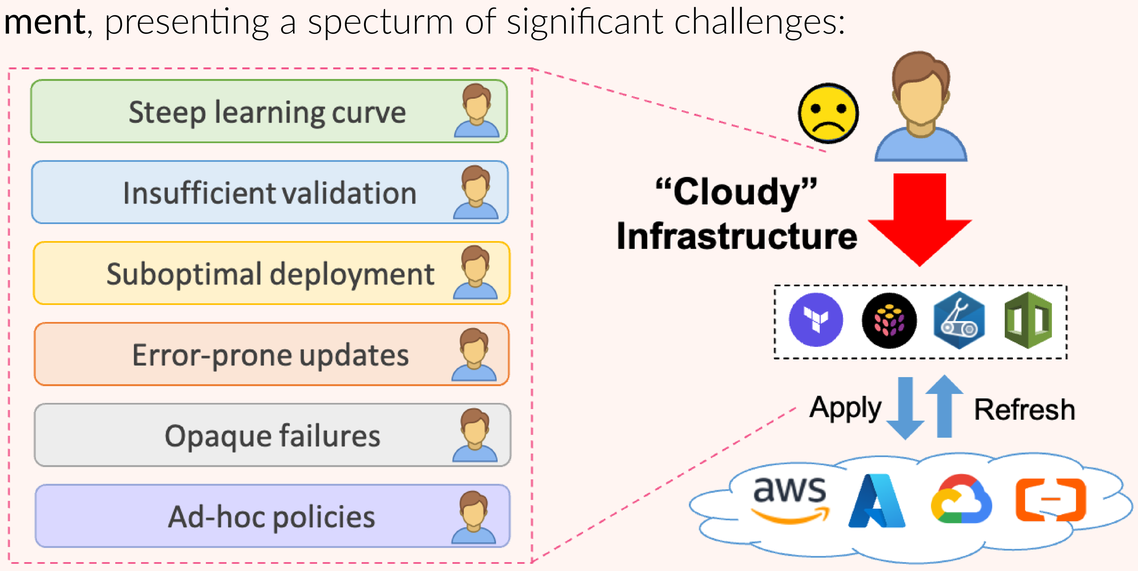


Cloud IaC enables users to write only high-level code to express their intended infrastructure, **shielding them from low-level details about how the underlying infrastructure is deployed.** Terraform code example:

```
resource "google_compute_ha_vpn_gateway" "ha_gateway1" {
  region   = "us-central1"
  name     = "ha-vpn-1"
  network  = google_compute_network.network1.id
}
resource "google_compute_network" "network1" {
  name                    = "network1"
  auto_create_subnetworks = false
}
```

Limitations of Cloud IaC

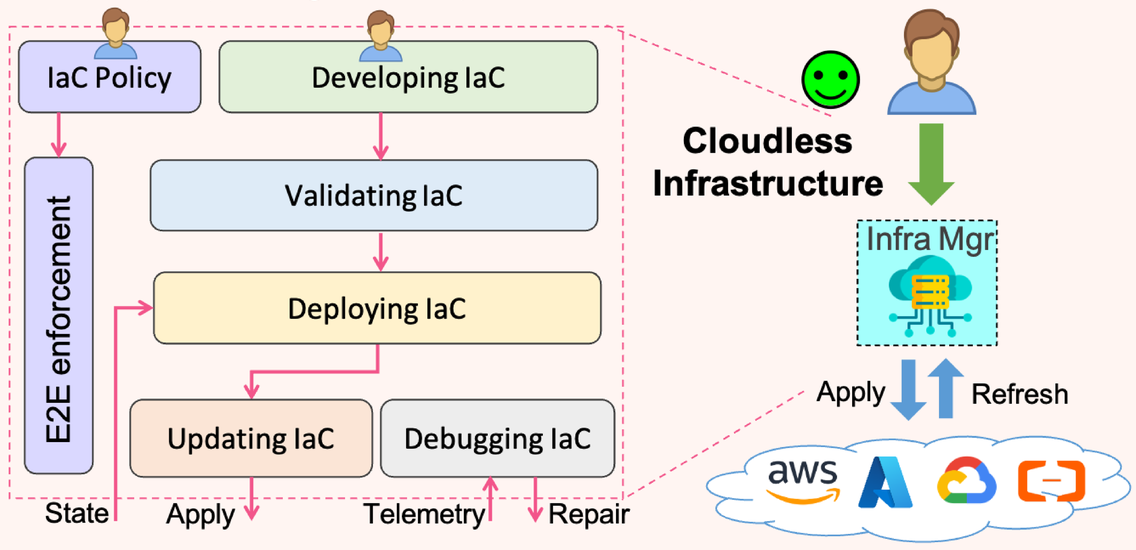
While existing IaC frameworks lower the barrier of cloud resource usage, they have **notable limitations across all stages of infrastructure management**, presenting a specturm of significant challenges:



Ad-hoc solutions exist, but so far there has been **no principled study on how IaC should be evolved to handle them in an automatic manner, while allowing users to customize policies at various decision points.**

Towards Cloudless Computing

Cloudless computing aims to address existing cloud IaC limitations, so that **the resulting infrastructure will no longer feel opaque and "cloudy,"** thus drastically improving reliability and efficiency.

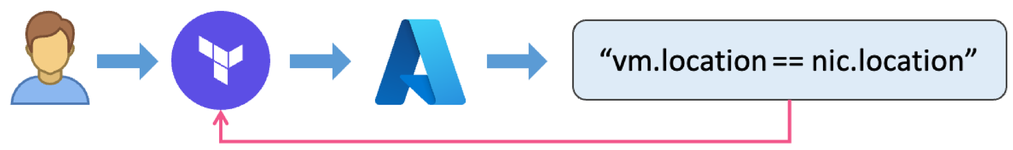


Instead of asking users to apply manual efforts or ad-hoc tools to handle various problems all through IaC life cycle, **cloudless computing only requires users to properly define their intent in IaC code.** All the low level details will be handled by a cloudless infrastructure manager, which allows **customization through end-to-end user policy enforcement.**

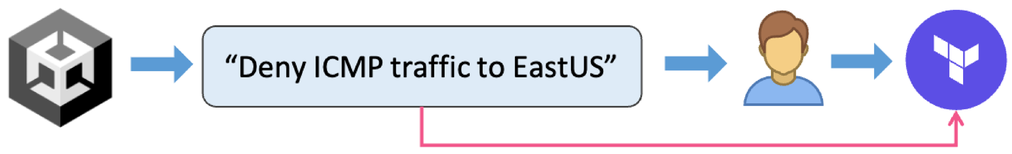
Cloudless Building Blocks

In order to realize the vision of Cloudless Computing, a broad set of building blocks are needed. We highlight a few as below:

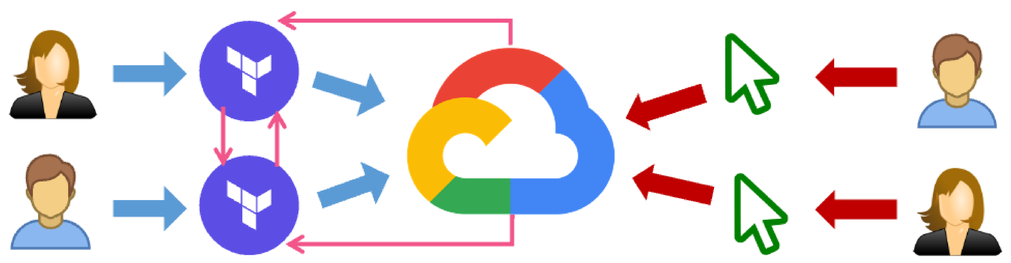
- **Validating cloud provider requirements and constraints.** Current IaC validation practices mainly targets superficial syntax checks. **Provider specific violations are mostly ignored, leading to deployment failures later on.** Cloudless would automatically translate provider constraints into static IaC checks, thus eliminating errors before actual damages.



- **Automated compliance and security check generation.** As of today, **cloud tenants have to manually port their company-wise compliance and security constraints into corresponding IaC workflow.** Cloudless would automate this process, while offering higher coverage.



- **Runtime race condition resolving and drift detection.** Large teams of **operators and developers create constant conflicts in IaC workflow.** Cloudless mitigates operator contentions through enhanced locking, and reacts to infrastructure drifts introduced by developers.



Ask us about more! Our research agenda covers all stages IaC life cycle, including development, validation, deployment, update, failure handling, and policy enforcement. Ask us about other key directions such as portability, observability, and efficiency!