Netflix's SimianArmy

Group 5 Open-Source Project

Basic Information

- https://github.com/Netflix/SimianArmy
- Size : 22K locs
  (this number already disregarded 9K locs that are under '/test')
- Language: Java 99.5% Other 0.5%
- Contributions: Mar 18, 2012 – Mar 28, 2017 (long history of development)
- Inspired by hearing story about AWS shutdown and Netflix in class.

Background Information

- Created by Netflix after moving to AWS to improve availability & reliability.
- A cloud architecture, where individual components fail but do not affect the availability of the entire system, was greatly needed.
- Infrastructure is never 100% but stream*ing* must be non-stop.
- SimianArmy deploys “monkeys” to make cloud service less fragile and better able to support continuous service when some parts have issues.
- Potential weaknesses and/or problems could be detected and addressed.

Analogy by Netflix

- Is your spare tire properly inflated when necessary?
- How do you know? Do you have the tools to change it? Can you?
- One way to guarantee the above questions is to poke a hole in your tire once a week and go through the drill of replacing it.
- Expensive and time-consuming in the real world.
- Almost **free and automated in the cloud.**
- This philosophy led to developing of a tool, namely monkeys, that randomly disables instances to make sure this type of failure can be withheld without any or minimal negative impact.

*Source: < http://techblog.netflix.com/2011/07/netflix-simian-army.html >

Members of the SimianArmy

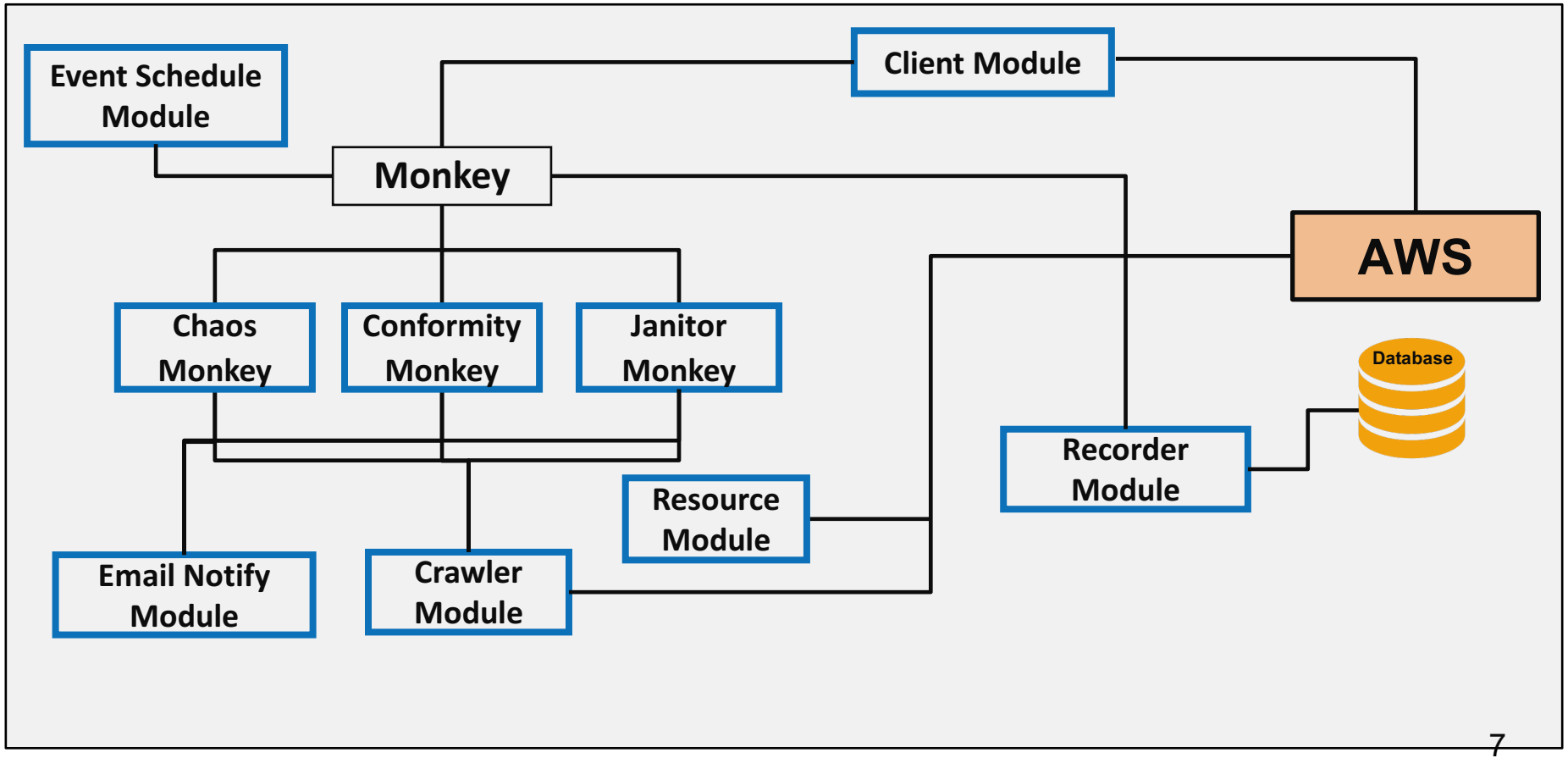
- Chaos Monkey – randomly shuts down VMs to ensure that small disruptions will not affect the overall service.
- Conformity Monkey – detects instances that aren't coded to best-practices and shuts them down, giving the service owner the opportunity to re-launch them properly.
- Janitor Monkey – searches for unused resources and discards them.
- *Security Monkey* – searches out security weaknesses, and ends the offending instances.
- *Doctor Monkey* – performs health checks on each instance and monitors other external signs of process health such as CPU and memory usage.
- *Latency Monkey* – simulates a degradation of service and checks to make sure that upstream services react appropriately.

Architecture Expression

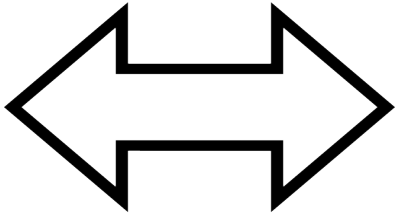
- Main functionalities of the system and system architecture:

Module	Descriptions
Client Module	Provides functionality/interface that allows the monkeys to interact with the cloud.
Resource Module	Provides functionality of getting the common properties of a resource and the methods to add and retrieve additional properties of one.
Chaos Monkey Module	Provides functionality of creating different types of chaos on AWS, thus causing small disruptions and testing ability to withstand such unexpected events.
Janitor Monkey Module	Provides functionalities of searching for unused resources, marking of them, and cleaning up.
Conformity Monkey Module	Provides functionalities of getting clusters, as well as applying a set of rules to perform conformity checks. A cluster is the basic unit of conformity check.
Crawler Module	Provides functionality of getting auto-scaling-groups from AWS.
Event Schedule Module	Provides functionalities of when and how-frequently the monkeys should be deployed.
Recorder Module	Provides functionalities of storing and finding events in data-storage.
Email Notify Module	Provides functionality of notifying users via email used by all monkeys.

System Architecture Diagram



Although, infrastructure (AWS) is never 100% stable, software (SimianArmy) can minimize damages caused instability to 0.001%, thus protecting valuable services (Netflix).



amazon
web services™