CLOUD NATIVE MX

CHAOS ENGINEERING

GREMLIN

SISTEMAS DISTRIBUIDOS

- Un Sistema Distribuido es complejo
- Ejecutas tus aplicaciones en distintos servidores
- Necesitamos monitorear nuestro sistema para conocer la salud del mismo
- Observabilidad
- Métricas “si no sabes que está fallando como vas a arreglarlo”

RESILIENCIA

- La capacidad del Sistema para soportar condiciones turbulentas
- Fault tolerance
- Failure Isolation
- Scalability
- Evolvability

FALACIAS

- La red es confiable
- La latencia es cero
- El ancho de banda es infinito
- La red es segura
- La topología no cambia
- Hay un solo administrador
- El costo de transporte es cero
- La red es homogénea

CHAOS ENGINEERING

- Poner a prueba la *Resiliencia* del Sistema
- Romper las cosas a propósito
- Encontrar puntos críticos de falla
- Identificar las debilidades antes de que se manifiesten de forma Proactiva
- Experimentos controlados

HISTORIA

- 2010 Netflix se migra a AWS crea Chaos Monkey
- 2011 nace The Simian Army “La nube es acerca de la redundancia y la tolerancia a fallos, pero nadie puede garantizar un uptime del 100%, necesitamos generar una arquitectura en la nube donde los componentes puedan fallar sin afectar la disponibilidad del sistema entero”
- 2012 Chaos Monkey se sube a Github
- 2014 Netflix crea el rol de Chaos Engineer

EXPERIMENTOS

I. Plan an Experiment

Create a hypothesis. What could go wrong?

II. Contain the Blast Radius

Execute the smallest test that will teach you something.

III. Scale or squash

Find an issue? Job well done. Otherwise increase the blast radius until you're at full scale.

EXPERIMENTOS

	Knowns	Unknowns
Known	Things we are aware of and understand.	Things we are aware of but don't understand.
Unknown	Things we understand but are not aware of.	Things we are neither aware of nor understand.

EJEMPLO

Slow Response from Database Primary

Hypothesis (Expected Outcome)	No customer impact expected
Attack Condition	**Duration:** 600s (10 min) **Latency:** 400ms **Targets:** 50% of available instances
Result (Actual Outcome)	1.2 to 1.6 second latency Degraded user experience Returning cached data with 200s “That’s a real miss in alerting and metrics.”

ROMPER PRODUCCION

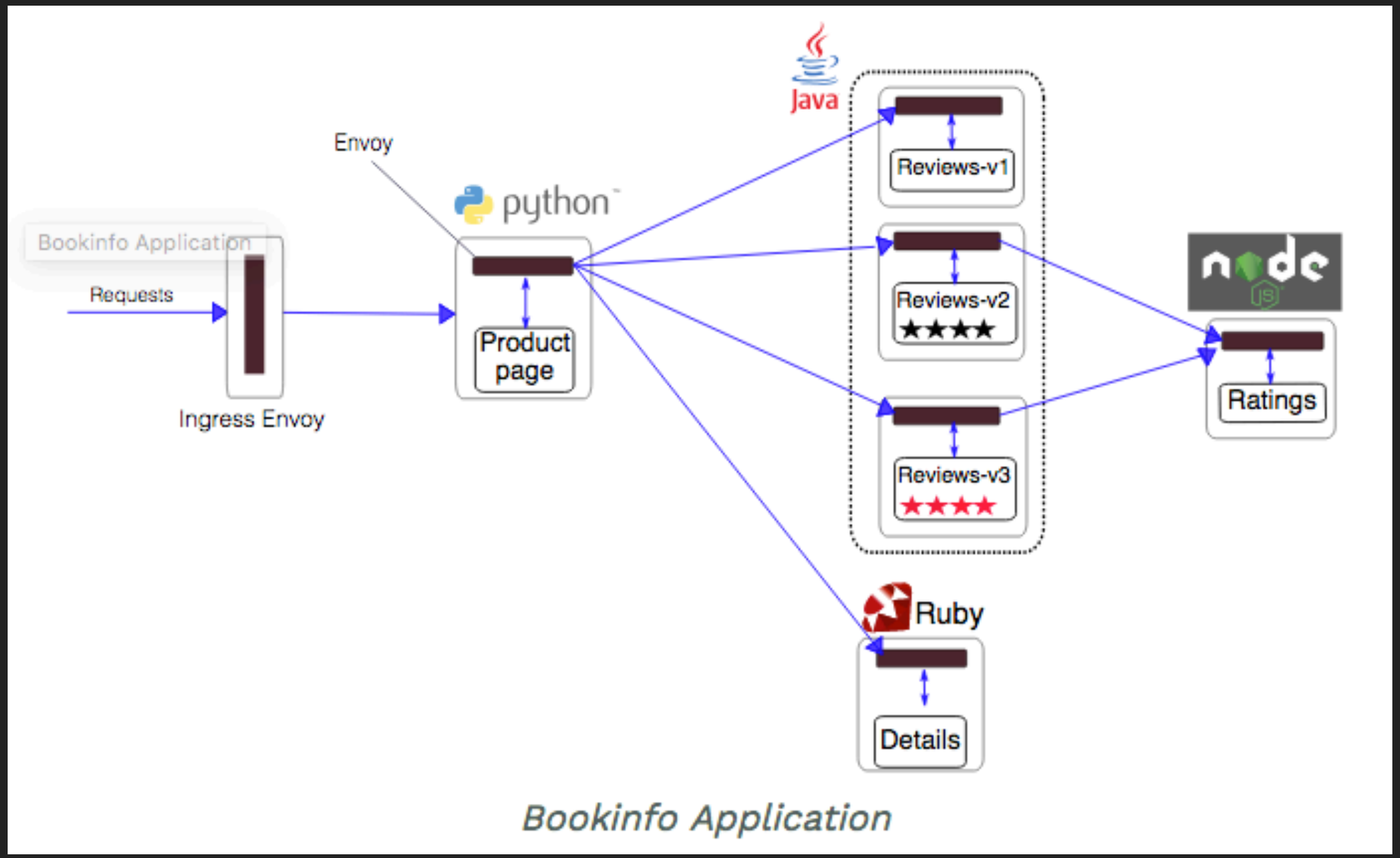
- Los sistemas de comportan diferente dependiendo del entorno
- El muestreo de trafico real es la única manera de garantizar la autenticidad de la prueba
- Diseña tus experimentos incrementando el radio
- Automatizar los experimentos
- ¡NUNCA! Hagas pruebas si sabes que vas a afectar a tus usuarios, trata siempre con problemas conocidos

GREMLIN

- Failure as a Service
- Ayuda a automatizar tus pruebas en diferentes ambientes
- Todos los ataques pueden revertirse inmediatamente, lo que le permite abortar de forma segura y volver al estado estable si las cosas salen mal
- Seguridad

DEMO

DEPLOYMENT



Bookinfo Application

REFERENCIAS

- Esta presentación https://github.com/Angelorum/chaosEngineeringTalk
- Principios del Chaos https://principlesofchaos.org/
- Artículos https://github.com/dastergon/awesome-chaos-engineering
- Gremlin https://www.gremlin.com
- Chaos Toolkit https://chaostoolkit.org