Chaos Engineering

"Predictibilidad, ¿El aleteo de una mariposa en Brasil hace aparecer un tornado en Texas?"

Edward Norton Lorenz – Teoría del Caos

Efecto Mariposa

01

Introducción

Contexto, propósito y porque hacer Chaos Engineering

¿Cual es el costo de 1 hora de indisponibilidad?

- Pequeñas empresas: **$10K - $167/min**
- Empresas medianas: **$25K a $100K - $1,7k/min**
- Empresas grandes: **$1M a $5M** – 167k a 835K/min

ITIC 2020 Global Server Reliability Report

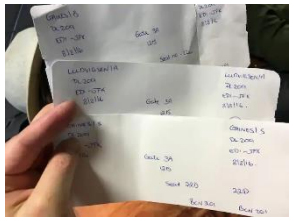
Un estudio de **101 startups** encontraron que el **29% de estas fallan** porque **se quedan sin efectivo**.

binsights

¿Cual es el costo de 1 hora de indisponibilidad?

Apple 2015

En Marzo del 2015, 12 horas **Apple Store** caída costo a la compañía **$25 millones**.



Boarding pass

En Agosto del 2016, 5 horas de caída de un centro de operaciones causo 2,000 vuelos cancelados y un estimado en perdidas de **$150 millones** a **Delta Airlines**.

En Julio del 2018, **Amazon** una hora de caída en el Prime Day costo hasta **$100 millones** en perdidas de compras.

Amazon 2018

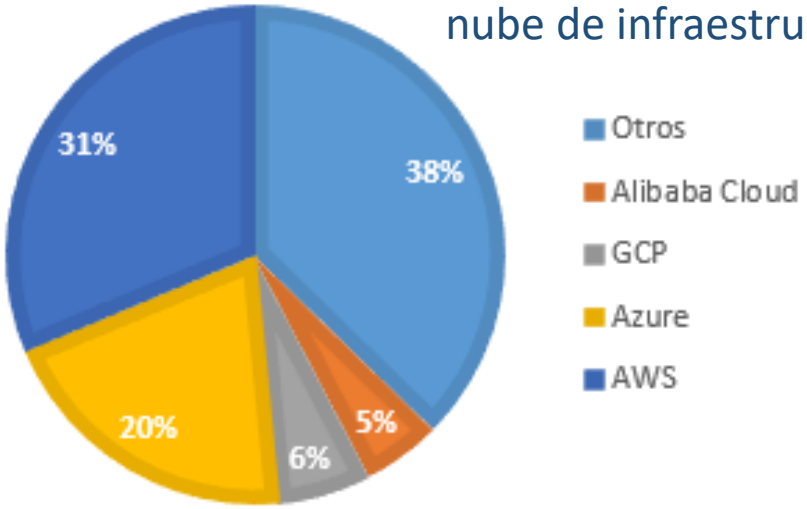
Facebook 2019

En Marzo del 2019, 14 horas de caída costaron a **Facebook $90 millones**.

¿Proveedores de nube al rescate?

Distribución de gasto mundial en servicios nube de infraestructura(Q2 2020)

Canalys



99,99% **52'/año**
Indisponibilidad

SLA disponibilidad
Azure, GCP y AWS*

En un estudio de **1,200 compañías y personas** de toma de decisión de IT revela que:

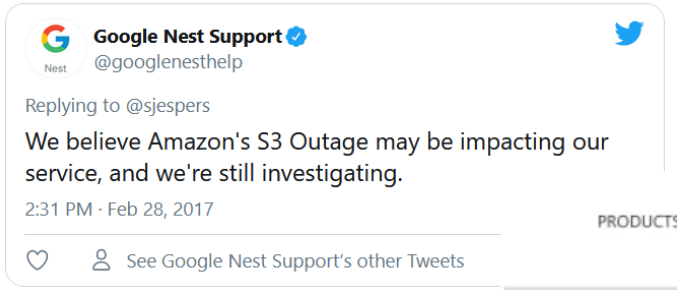
60% No han evaluado el costo de caída de la nube

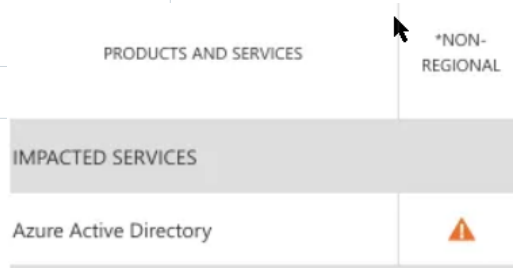
Veritas

¿Quién es responsable de producirse una caída?

*VM Services

¿Quién es responsable de producirse una caída?





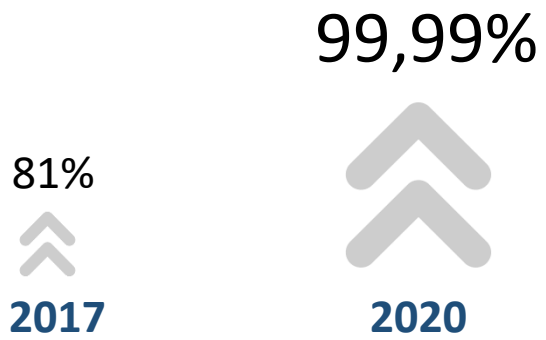
Aplicación, dispositivos, sistemas y datos

El cliente

Infraestructura y servicios

Proveedores de nube

87% de las organizaciones ahora **quieren alcanzar un mínimo de 99.99%** disponibilidad.



99.99%
("**four nines**")

A eso de le llama 99.99% o **“cuatro nueves”**

Mes: 4,4’

Año: 52’

ITIC 2020 Global Server Reliability Report

Para el **2024**, mas de **50% de compañías grandes** utilizarán la practica de **ingeniería del chaos** para alcanzar el **99.999%** disponibilidad. Gartner - 2019

99.999%

("**five nines**")

Mes: 26s

Año: 5' 15s

02

Definición, Principios, Procesos

everis
an NTT DATA Company

Introducción

La historia detrás de Chaos Engineering

2008 - Netflix planea su migración de Datacenters a cloud.

Obj: Solucionando su infraestructura con un solo puntos de fallo.
Instancias Grandes

2018: **Compañía de nube no tan madura**, donde las **instancias desaparecían** eventualmente, sin advertencia previa.

Resultado: Multiples puntos de fallos al distribuir la carga con instancias inestables.

2008 - Netflix contrata ingenieros seniors para solucionar el problema.

Obj: Solucionando el problema de inestabilidad

Resultado:
- 1° interacción: Falla porque se aplican técnicas de estabilidad conocidas
- 2° interacción: Nace el Chaos Monkey

2012 Visperas de Navidades - Netflix experimenta una caida de región en AWS. Caída de ELB.

Obj: Solucionando el problema de inestabilidad regiones

Resultado:
- Nace el Chaos Kong

¿Qué no es Chaos Engineering?

!!! Chaos Engineering es más que causar caos !!!

¿Que se quiere romper?

No es Chaos Engineering Desencadenado

No es causar dolor para quien lo ejecuta y para los usuarios

No es hacer pruebas

No es romper producción

No es mayor resiliencia (antifragilidad)

¿Que se quiere aprender?

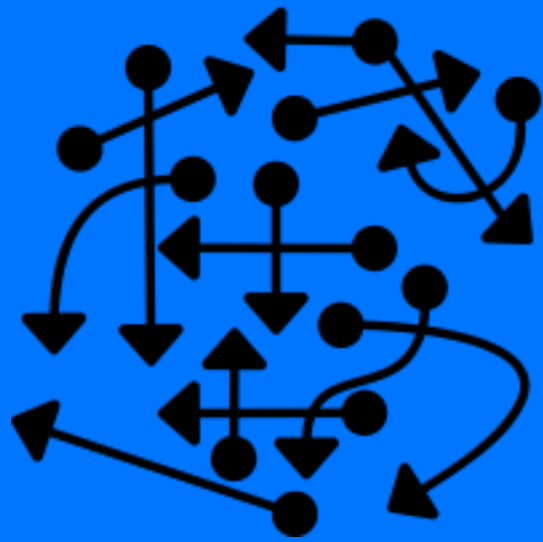
Debe ser controlado

Debe tener un propósito que se demuestra durante la experimentación

Debe enseñar de caos de sistema a los equipos para ganar resiliencia

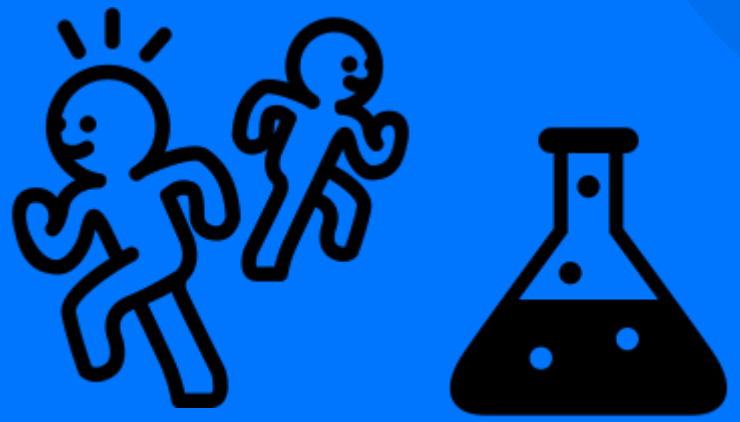
¿Qué es Chaos Engineering?

Una definición formal: “La ingeniería del caos **es una disciplina** de experimentación **cuyo propósito es construir un Sistema distribuido confiable capaz de mantenerse estable en condiciones de turbulencia en producción**.” Esto se establece a traves de la experimentación, que se encuentra fuera de las pruebas.



Chaos Engineering es “**El apoyo a experimentos para descubrir debilidades de los sistemas**.”

Chaos Engineering en la práctica



Definir un “estado de quietud”

Plantear la hipótesis de que el “estado de quietud” continuará en grupos controlados y experimental.

Introducir variables del mundo real

Tratar de refutar la hipótesis que el “estado de quietud” se mantendrá en el grupo de controlado y experimental.

03

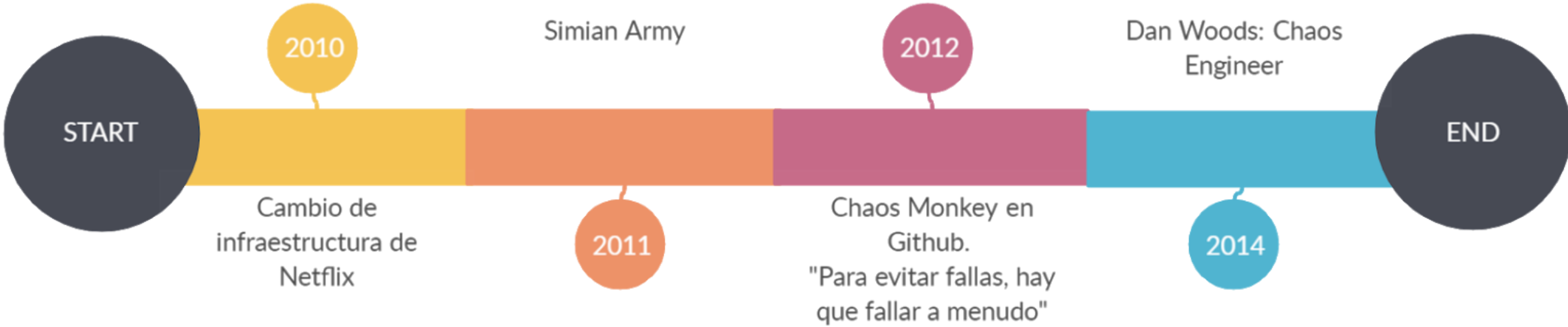
Aspectos prácticos

¿Está lista la aplicación para producción?
¿Sobrevivirá el sistema a las fallas de otras empresas de las que depende?¿Sobrevivirá el sistema al fallo de sus propias configuraciones?

Simian army

- Chaos Monkey - Apaga servidores 2016
- Latency Monkey - Retrasos artificiales
- Conformity Monkey - Apaga servidores sin buenas practicas
- Doctor Monkey - Health check kpi server
- Janitor Monkey - Entorno libre de basura
- Security Monkey - Finaliza instancias vulnerables e inseguras
- 10-18 Monkey - Localización e internacionalización
- Chaos Gorilla - Disponibiliza toda la zona de disponibilización

Aspectos práctivos



Chaos Monkey

NETFLIX

Chaos Monkey

Ventajas

- Planificación de fallas de instancias aleatorias
- Fomenta la redundancia
- Visibiliza las debilidades de sistema
- Anticipa errores en producción

Desventajas

- No tiene capacidad de recuperación
- Sin interfaz de usuario
- Herramientas auxiliares limitadas
- Esconder errores

Implementaciones

Name	Open or Commercial	URL
Byteman	Open Source	https://byteman.jboss.org
Chaos Monkey	Open Source	https://github.com/Netflix/chaosmonkey
ChaosIQ	Commercial	https://www.chaosiq.io
Gremlin	Commercial	www.gremlin.com
Jepsen	Open Source	http://jepsen.io
Mangle	Open Source	https://github.com/vmware/mangle
Simian Army	Open Source	https://github.com/Netflix/SimianArmy
Spinnaker	Open Source	https://github.com/spinnaker
Verica/ChaoSlinger	Open Source	https://www.verica.io

Failure Injection Testing

Lo que se necesita es una forma de limitar el impacto de las pruebas de fallas sin dejar de romper cosas de manera realista.

- Llenar espacio de disco
- Acaparar CPU y RAM
- Sobrecargar IO
- Manipulación avanzada de tráfico de red
- Terminar procesos
- Mas ...

Kube-monkey

- Black list (pods, namespaces)
- Programado (cada 1hr, 3hr, …)
- Tipos de kills
- Numero de pods

- kube-monkey/enabled
- kube-monkey/mtbf
- kube-monkey/identifier
- kube-monkey/kill-mode

Chaos monkey en Spinnaker

- Configurar Spinnaker para el soporte de Chaos Monkey
- Configurar la BD de Mysql
- Escribir un archivo de configuración (chaosmonkey.toml)
- Ejecute un cronJob que ejecute la programación diaria del Chaos Monkey

```
[chaosmonkey]
enabled = true
schedule_enabled = true
leashed = false
accounts = ["production", "test"]

[database]
host = "dbhost.example.com"
name = "chaosmonkey"
user = "chaosmonkey"
encrypted_password = "securepasswordgoeshere"

[spinnaker]
endpoint = "http://spinnaker.example.com:8084"
```

03

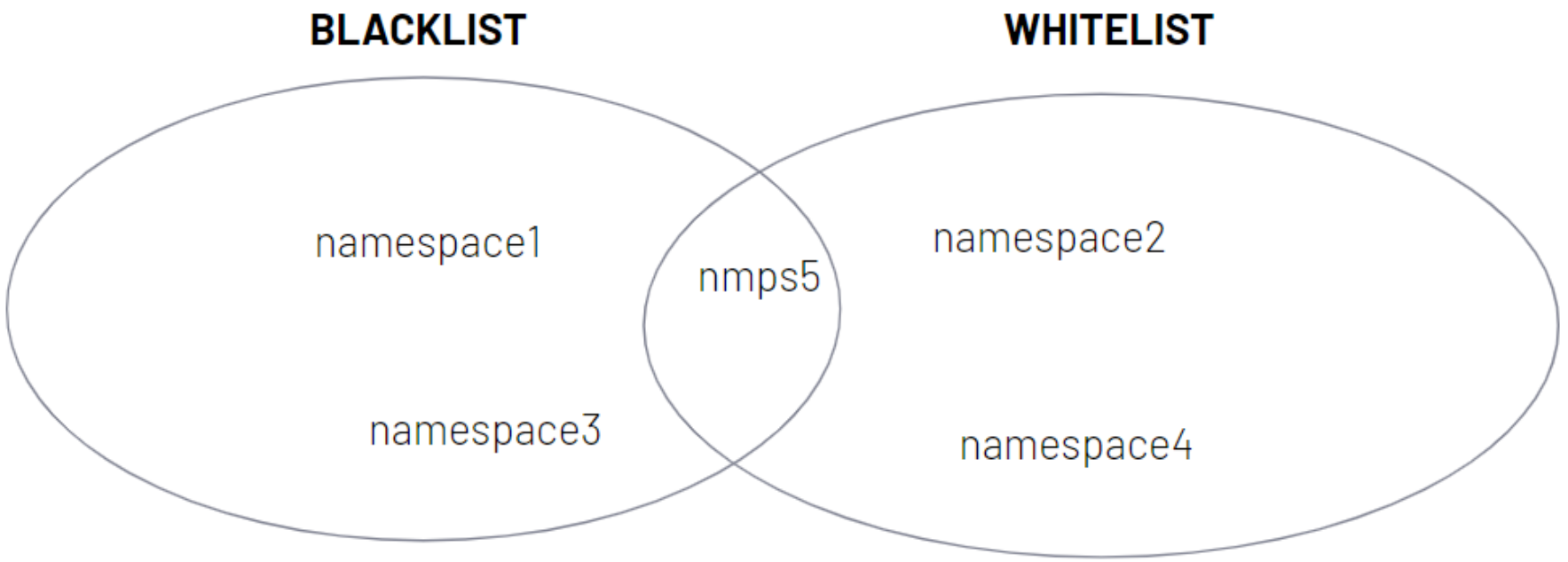
Kube Monkey

Kube-monkey

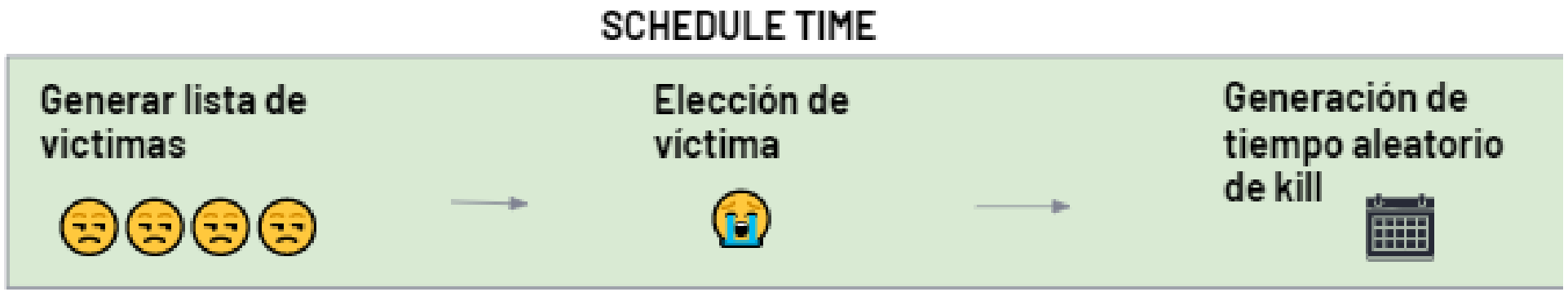
Es una **implementación** del Chaos Monkey de Netflix **para clústeres de Kubernetes** . Elimina aleatoriamente los **pods** de Kubernetes (k8s) en el clúster, fomentando y validando el desarrollo de servicios resistentes a fallas.

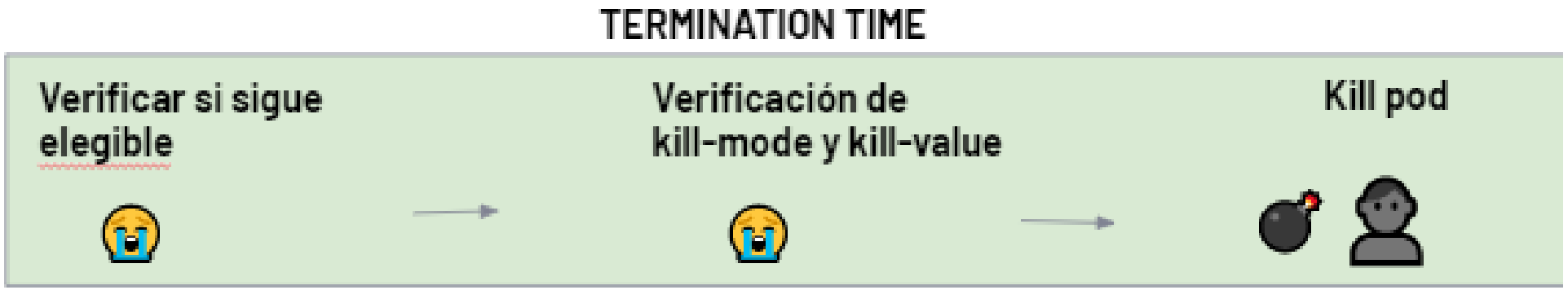
Namespaces

Definición del espacio de nombres donde se aplicará la terminación.



¿Cómo funciona?





Configuración de pods afectados

Lo que se necesita es una forma de limitar el impacto de las pruebas de fallas sin dejar de romper cosas de manera realista.

Black list (pods, namespaces)	kube-monkey/enabled
Programado (cada 1hr, 3hr, ...)	kube-monkey/mtbf
Tipos de kills	kube-monkey/identifier
Numero de pods	kube-monkey/kill-mode

Variables de ambiente

Definimos los siguientes variables de ambientes en formato Toml ubicados en la siguiente ruta: /etc/kube-monkey/config.toml

```
KUBEMONKEY_DRY_RUN                  = true
KUBEMONKEY_RUN_HOUR                 =  8
KUBEMONKEY_START_HOUR               = 10
KUBEMONKEY_END_HOUR                 = 16
KUBEMONKEY_BLACKLISTED_NAMESPACES   = kube-system
KUBEMONKEY_TIME_ZONE                = America/New_York
```

```yaml
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: kube-monkey-config-map
  namespace: kube-system
data:
  config.toml: |
    [kubemonkey]
    run_hour = 5
    start_hour = 6
    end_hour = 7
    blacklisted_namespaces = ["kube-system"]
    whitelisted_namespaces = [ "default"]
    time_zone = "America/Lima"
```

Controladores

```
---
  apiVersion: extensions/v1beta1
  kind: Deployment
  metadata:
    name: kube-monkey
    namespace: kube-system
  spec:
    replicas: 1
    template:
      metadata:
        labels:
          app: kube-monkey
      spec:
        containers:
          -  name: kube-monkey
             command:
               - "/kube-monkey"
             args: ["-v=5", "-log_dir=/var/log/kube-monkey"]
             image: ayushsobti/kube-monkey:v0.3.0
             volumeMounts:
               - name: config-volume
                 mountPath: "/etc/kube-monkey"
        volumes:
          - name: config-volume
            configMap:
              name: kube-monkey-config-map
```

Definición de permisos

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: monkey-role
rules:
- apiGroups:
  - ""
  - apps
  - extensions
  resources:
  - '*'
  verbs:
  - '*'
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: monkey-role-binding
  resourceVersion: "241143"
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: monkey-role
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
```

Thanks!

Miguel Martínez Espichan
Líder de Proyecto DevSecOps y GC DevOps
miguel.angel.martinez.espichan@everis.com
+51994755224