FACULDADE IMPACTA

CURSO DE PÓS GRAUDAÇÃO EM CLOUD COMPUTING

RA:1801261 - ROGÉRIO DA SILVA SANTOS

RA:1202836 – KELVYN TOMAZ MARTINS DA SILVA

RA:1802302 – KHRISTOPHER SANTOS

**SIMULAÇÃO DE POSSIVEIS PROBLEMAS**

SÃO PAULO

2019

RA:1801261 - ROGÉRIO DA SILVA SANTOS

RA:1202836 – KELVYN TOMAZ MARTINS DA SILVA

RA:1802302 – KHRISTOPHER SANTOS

**SIMULAÇÃO DE POSSIVEIS PROBLEMAS**

Trabalho elaborado para a disciplina de Infraestrutura e Interoperabilidade de Plataformas Cloud, do curso de Pós-Graduação Lato Sensu em Cloud Computing e DevOps, com o objetivo de explicar a ferramenta Chaos Monkey utilizada para testar a estabilidade do sistema aplicando falhas por meio da terminação pseudo-aleatória de instâncias e serviços dentro da arquitetura da Netflix.

SÃO PAULO

2019

Este trabalho foi desenvolvido em Grupo como exigência para a disciplina Infraestrutura e Interoperabilidade de Plataformas Cloud. Agradecemos ao professor Yuri Menezes por incentivar a pesquisa e o conhecimentos de novas ferramentas.

**Sumário**

1. Definindo Simulação5
2. Princípios Chaos Engineeing 6
3. Chaos na Prática7
4. Casos de Estudos 7
5. História Chaos Monkey......................................................................................8
6. Como funciona o Chaos Monkey.......................................................................9
7. Benefícios Chaos Monkey9
8. Visão Geral Implantação 10
9. Funcionalidades dos “monkeys” .....................................................................11
10. Conclusão.........................................................................................................12
11. Referencias Bibliograficas...............................................................................13

1. DEFININDO SIMULAÇÃO

A simulação é apresentada como uma metodologia pouco utilizada, e consiste em fazer parecer como algo real uma coisa que não é.

São obtidos os dados para criar uma simulação, com ela são feitas projeções baseadas nos dados atuais e, depois, o modelo é confirmado (ou não). Com os novos dados é possível fazer novas projeções, a serem confirmadas ou não, de modo que o aperfeiçoamento desse mesmo modelo é contínuo.

Dessa forma, a simulação se presta tanto no "contexto de descoberta" quanto no "contexto da prova".

O contexto da prova é aquele no qual se tenta provar ou refutar uma hipótese ou uma teoria. Assim, os dados estão sendo utilizados para testar o modelo. Muitos modelos são desenvolvidos com base num conjunto de dados, e quando submetidos a outro conjunto coletado sob condições diferentes podem ou não se mostrar capazes de predizer os resultados.

O contexto da descoberta é aquele no qual construímos as hipóteses. Isso acontece quando os dados refutam a teoria ou a hipótese, forçando a criação de novas hipóteses e a alteração do modelo. Tal fato pode decorrer de uma nova variável que precisa ser incorporada ou de uma função que não se comporta perfeitamente; ou seja, de uma disfunção.

Finalmente, as simulações devem ser vistas como uma ferramenta importante de investigação do mundo real, nas simulações, como na ciência em geral, deve-se buscar uma superação, processo este que se repete continuamente.

1. PRINCÍPIOS DE CHAOS ENGINEERING

Chaos Engineering é a disciplina de realizar experimentos sobre sistemas distribuídos com o intuito de construir confiança com relação a capacidade de um sistema distribuído suportar condições adversas em produção.

Avanços em sistemas de software distribuídos de larga escala estão revolucionando a engenharia de software. Como indústria, nós somos rápidos para adotar práticas que aumentam a flexibilidade do desenvolvimento e a velocidade das entregas em produção (deploys). Uma pergunta crítica surge junto com estes benefícios: Quanta confiança conseguimos ter nos sistemas complexos que nós colocamos em produção?

Mesmo quando cada um dos serviços que compõem um sistema distribuído está funcionando adequadamente, as interações entre esses serviços podem causar resultados imprevisíveis. Estas consequências, são compostas por combinações raras de eventos reais que afetam os ambientes de produção, tornando esses sistemas distribuídos inerentemente caóticos.

Nós precisamos identificar fraquezas antes que elas ocorram por meio de comportamentos anômalos em todo o sistema. Falhas sistêmicas podem se manifestar por meio de: configurações inadequadas de fallback (opções de contingência) quando um serviço não está disponível; uma chuva de tentativas causadas por timeouts mal configurados; interrupções quando uma dependência downstream recebe muito mais tráfego do que o esperado; Falhas cascateadas quando um único ponto de falha para de funcionar; etc.. Nós devemos abordar as fraquezas mais relevantes dos nossos sistemas distribuídos de forma proativa, antes que afetem nossos clientes em produção. Precisamos de uma maneira de gerenciar o caos inerente a esses sistemas, aproveitando o aumento da flexibilidade e velocidade e ter a confiança em nossos sistemas de produção, independentemente da complexidade que elas representam.

Uma abordagem empírica, baseada em sistemas endereça o caos em sistemas distribuídos em escala e gera confiança na capacidade desses sistemas de resistir a condições reais. Nós aprendemos sobre o comportamento de um sistema distribuído observando-o durante um experimento controlado. Nós chamamos isso de Chaos Engineering (Engenharia do Caos).

1. CHAOS NA PRÁTICA

Para abordar a incerteza inerente aos sistemas distribuídos em escala, Chaos Engineering (Engenharia do Caos) pode ser pensada como a condução de experimentos para descobrir fraquezas sistêmicas. Estes experimentos seguem quatro etapas:

1. Comece definindo o que significa "sistema estável", uma medida que tem como resultado mensurável um indicativo sobre o comportamento normal do sistema.

2. Crie hipóteses este estado "sistema estável" permanecerá tanto no grupo de controle quanto no grupo onde o experimento será executado.

3. Introduza variáveis que reflitam eventos que ocorrem no mundo real, como por exemplo: servidores que travam, discos rígidos defeituosos, conexões de rede que são interrompidas, etc.

4. Tente refutar cada hipótese procurando diferenças entre o "sistema estável", o grupo de controle e o grupo experimental.

Quanto mais difícil for causar impactos no "sistema estável", mais confiança teremos no comportamento normal do sistema. Se uma fraqueza ou falha for descoberta, teremos uma nova meta de melhoria antes que esse comportamento se manifeste no sistema como um todo em produção.

1. **CASO DE ESTUDO**

Em meados de Setembro de 2015, a Amazon Sofreu um Downtime grande nos seus serviços de cloud, rimeiramento no serviço de DynamoDB, e afetando 20 outros serviços dependentes, afetando muitos usuários e causando custos de operação para solucionar.

E a Netflix, mesmo tendo a maioria da infraestrutura na amazon, não sofreu muito downtime, após alguns segundos, ela estava no ar, sem muito esforço, pois o chaos monkey já havia preparado as equipes, aplicação e o ambiente para este caso, tendo toda a sua resiliencia sem muito impacto.

1. **HISTÓRIA CHAOS MONKEY**

Enquanto supervisionava a migração da Netflix para a nuvem em 2011, Greg Orzell teve a ideia de abordar a falta de testes de resiliência adequados configurando uma ferramenta que causaria falhas em seu ambiente de produção, o ambiente usado pela Netflix clientes. A intenção era passar de um modelo de desenvolvimento que não assumisse falhas para um modelo em que as interrupções fossem consideradas inevitáveis, levando os desenvolvedores a considerar a resiliência interna como uma obrigação e não como uma opção:

"Na Netflix, nossa cultura de liberdade e responsabilidade nos levou a não forçar os engenheiros a projetar seu código de maneira específica. Em vez disso, descobrimos que poderíamos alinhar nossas equipes em torno da noção de resiliência de infraestrutura isolando os problemas criados pela neutralização de servidores e empurrando-os para o extremo. Nós criamos Chaos Monkey, um programa que escolhe aleatoriamente um servidor e desativa-lo durante suas horas habituais de atividade. Alguns vai achar que louco, mas não poderíamos depender da ocorrência aleatória de um evento para testar o nosso comportamento em face das próprias consequências deste evento sabendo que isso aconteceria freqüentemente criou um forte alinhamento entre os engenheiros para construir redundância e automação de processos para sobreviver a tais incidentes, sem afetar os milhões de usuários da Netflix. Chaos Monkey é uma das nossas ferramentas mais eficazes para melhorar a qualidade dos nossos serviços.

Ao "matar" regularmente instâncias aleatórias de um serviço de software, era possível testar uma arquitetura redundante para verificar se uma falha de servidor não impactava visivelmente os clientes.

1. **COMO FUNCIONA CHAOS MONKEY**

Chaos Monkey é um serviço que identifica grupos de sistemas e finaliza aleatoriamente um dos sistemas de um grupo, o serviço opera em um horário controlado (não funciona nos finais de semana e feriados) e intervalo (só funciona durante o horário comercial). Na maioria dos casos, projetamos nossos aplicativos para continuar trabalhando quando um ponto ficar off-line, mas, nesses casos especiais, queremos ter certeza de que há pessoas por perto para resolver e aprender com qualquer problema.

Com isso em mente, o Chaos Monkey só funciona em horário comercial com a intenção de que os engenheiros estejam alertas e sejam capazes de responder.

1. BENEFICOS CHAOS MONKEY

1. PREPARA VOCÊ PARA FALHAS DE INSTÂNCIAS ALEATÓRIAS

O Chaos Monkey permite falhas de instâncias planejadas quando você e sua equipe estão mais bem preparados para lidar com eles. Você pode agendar términos para que eles ocorram com base em um número médio de dias configurável e durante um determinado período de tempo todos os dias.

2. INCENTIVA REDUNDÂNCIA

Parte integrante de uma arquitetura distribuída, a redundância é outro grande benefício para as práticas inteligentes da Chaos Engineering. Se um único serviço ou instância for desativado inesperadamente, um backup redundante poderá salvar o dia.

3. CONSTRUÍDO EM SPINNAKER

Chaos Monkey Version 2.0 depende do Spinnaker. Isso é tanto um pró quanto um contra. Ele permite compatibilidade entre nuvens, mas exige que o usuário esteja usando o Spinnaker.

1. VISÃO GERAL IMPLANTAÇÃO

O chaos monkey não é fácil de configurar, para implantar o Chaos Monkey, você precisa:

* Configurar o suporte do Spinnaker for Chaos Monkey
* Configurar o banco de dados MySQL
* Escreva um arquivo de configuração (chaosmonkey.toml)
* Configurar um cron job que execute a programação diária do Chaos Monkey

**Como o Chaos Monkey é executado**

O Chaos Monkey não funciona como um serviço. Em vez disso, você configura um cron job que chama o Chaos Monkey uma vez por dia para criar um cronograma de término.

Quando o Chaos Monkey cria um cronograma, ele cria outro cron job para agendar terminações durante as horas de trabalho do dia

1. FUNCIONALIDADES DOS “MONKEYS”

A Netflix, com a Eficiência que ela conseguiu com o chaos monkey, ela criou outros “Monkeys” para simular diversos outros problemas no ambiente e testar a habilidade de sobreviver a todos eles

E abaixo uma lista com os “monkeys” já criados pela Netflix:

LATENCY MONKEY: ele induz delays artificiais na aplicação, simulando degradação nos serviços e mensura a resposta dos serviços para corrigir esta falha de latência.

CONFORMITY MONKEY: Ele procura instancias que não estão nas conformidades definidas por eles, por exemplo: uma instância de EC2 está fora da VPC definida a ela, e assim ela mata e dá a oportunidade de ser iniciada a instancia na conformidade.

DOCTOR MONKEY: Ele verifica a saúde das instâncias e detecta se alguma não está saudável, como por exemplo: a carga de CPU está muito alta, e remove ela para ser diagnosticada posteriormente.

JANITOR MONKEY: Ele busca serviços ou instancias que não estão sendo utilizadas, e elimina.

10-18 MONKEY: Detecta problemas de configuração de internacionalização em maquinas, como linguagem, formato de data e moeda.

CHAOS GORILLA: É similar o chaos monkey, mas ele simula uma falha de uma zona de disponibilidade (AZ) da Amazon, para verificar se a aplicação consegue re-balancear sem intervenção.

1. CONCLUSÃO

Neste trabalho abordamos conceitos sobre Chaos Engineering e princípios sobre a ferramenta Chaos Monkey.

O Chaos Engineering entendemos que é uma pratica poderosa que já está mudando a forma como o software é projetado e desenvolvido em algumas das maiores operações de TI em escala do mundo.

Enquanto a Chaos Engineering se estende muito além do escopo de uma única técnica ou ideia, o Chaos Monkey é a ferramenta mais conhecida para executar os Experimentos do Caos e é um ponto de partida comum para engenheiros começarem com a disciplina.

1. REFERÊNCIA BIBLIOGRAFIA

<https://en.wikipedia.org/wiki/Chaos_engineering>

<https://github.com/Netflix/SimianArmy/wiki/Chaos-Monkey>

<http://principlesofchaos.org/?lang=PTBRcontent>

<https://www.gremlin.com/chaos-monkey/>

<https://www.computer.org/publications/tech-news/research/realizing-software-reliability-in-the-face-of-infrastructure-instability>

https://medium.com/netflix-techblog/chaos-engineering-upgraded-878d341f15fa