CHAOS ENGINEERING

Javier Carrillo González – UO269412
Andrea García Cernuda – UO270115
Héctor Fernández Maillo – UO271913

Distributed software is becoming more and more complex, and being so important, developers often focus on its flexibility and speed being deployed. However, is the system trustable enough? Are we certain that this system will survive any problems that could arise? Even if the system is theoretically working as intended and all the inner parts are tested, we must consider that sometimes, these systems use services which may be not available for some time.

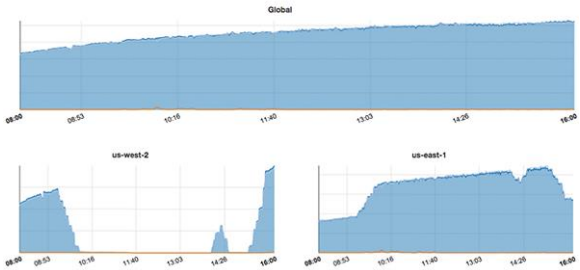
Chaos engineering is the discipline that tries different experiments within a system to make it more trustable regarding the possible outcomes that may happen in production.

History: from a monkey to a discipline

It all started in 2010 when Netflix, the very well-known streaming service, surprised the world with **Chaos Monkey**: a tool which tore apart one of the servers where the system was deployed.

A short time later (2011), Chaos Monkey became part of a set of tools, a project called **Simian Army**, and then it evolved into **Chaos Kong**. This tool not only took down a server, but a complete Amazon Web Services region. Even if this error hardly ever happens, it happened, on September 20th, 2015 actually. However, Netflix just suffered a bit of availability loss, since they quickly identified the error and sent a fix to it, thanks to all the testing and preparation Chaos Kong provided to them.

And how can they analyse if the system is becoming more resilient? They provide this example on their web. In the first chart you can see the combined view of the system, while the two lower ones show the different metrics of each of the labelled regions.



Finally, a new discipline called **Chaos Engineering** was created. This was an empirical system-based approach that assumed that chaos was going to happen in the systems for sure; thus, we should be able to build a system that will be able to withstand these conditions.

Practical steps

To apply the theoretical concepts that have been presented until now, *four* different and clearly defined moves need to be followed:

1. Describe what is called a "steady state" of the system, that is when it reflects normal behaviour.
2. Make a hypothesis stating that the previous situation will remain in two distinct groups: control and challenge.
3. Establish stress variables in the latter, like can be server failures or hardware issues.
4. Attempt to invalidate the hypothesis developed in step 2 by showing differences between both groups.

Principles

According to the Principles of Chaos Engineering, the following are the ones stated in this publication, in which other companies can take part by helping to improve it as it is a living document:

- Build a hypothesis around steady state behaviour.
- Vary real-worlds events.

- Run experiments in production.
- Automate experiments to run continuously.
- Minimize blast radius.

Pros & Cons

Advantages	Disadvantages
Antifragility and resilience are gained	Implementation for large-scale systems can cause a higher cost
Prevention of significant economic business losses and costs	Failing in the experiment process could affect the customer
No loss of client confidence	
Help in finding bugs whose solutions may be found easily and faster	

When is this discipline recommended to be applied?

We should consider applying this discipline if the applications that bring our business to life are distributed systems along with microservice architectures (**Cloud Native**).

Which companies practice Chaos Engineering?

Twilio, Netflix, LinkedIn, Facebook, Google, Microsoft, Amazon are some of the companies that right now are practicing Chaos Engineering.

Conclusion

Chaos Engineering is a strong practice which has come to stay and change the way of understanding software in terms of design and engineering.

Since the goal is to prevent and counteract the consequences of failures as much as possible, the tests described above should be performed in production.

This process is tedious and not very pleasant, but in the future, it saves time, effort and lost in services.

Bibliography

https://netflixtechblog.com/chaos-engineering-upgraded-878d341f15fa

https://www.infoq.com/articles/chaos-engineering-security-networking/

https://principlesofchaos.org/

https://www.pandasecurity.com/es/mediacenter/seguridad/ingenieria-del-caos-para-que-sirve-introducir-fallos-adrede/

https://alexmarket.medium.com/principios-de-la-ingenier%C3%ADa-del-caos-b295e0318ed2

https://www.gremlin.com/community/tutorials/chaos-engineering-the-history-principles-and-practice/

https://www.limepoint.com/blog/chaos-engineering-not-just-another-buzzword

https://sharpend.io/the-limitations-of-chaos-engineering/

https://www.knowledgehut.com/blog/devops/chaos-engineering