**CS5224 Cloud Computing – Final Report**

Semester 2 – 2018/2019



**Chaos Monkey**

A tool to test resiliency with random shutdowns

|  |  |
| --- | --- |
| Chirumamilla, Jagadeesh | A0191509B |
| Kemper, Philipp | A0191365B |
| Maddi Kamal, Divya | A0178511E |
| Thiyagarajan, Dhakshayani | A0100461B |

Table of Contents

[Executive Summary 3](#_Toc6694132)

[Business case identification 4](#_Toc6694133)

[Business model 6](#_Toc6694134)

[SaaS Architecture 7](#_Toc6694135)

[Screenshot of working prototype 9](#_Toc6694136)

[Implementation plan 10](#_Toc6694137)

[Economic Factors 11](#_Toc6694138)

[Conclusion 12](#_Toc6694139)

[References 12](#_Toc6694140)

# Executive Summary

Failures of a system or service is inevitable in Production environment and they usually happen at most unexpected and undesired times. Chaos monkey is a tool developed by Netflix and is part of Simian Army to generate various kinds of failures and test the ability to survive a failure.

The main objective of our project is to develop a tool to test the capacity of how quickly a system or a service can be recovered from failures by-randomly disabling production instances and services in the Cloud. Our tool simulates a scenario for the consumer to understand and assess the existing systems and encourages to develop a fault tolerant system instances or rebuild an existing system. Our project aims to keep Cloud services safe, secure, and highly available by proactively testing the system responsiveness to failure conditions and thereby predicting and avoiding real time outages.

Chaos Monkey service identifies groups of instances or services and stops one of them randomly at a controlled time to simulate a random failure. Our service can be configured to define the failure scenarios and inject failures. This service is most preferably run during business hours to alert the engineers and encourages them to build resilient systems that can handle unpredictable failures. Based on the test results of this dummy crash-test tool consumers can assess the abnormal conditions, bugs and security issues for any instance.

Our SaaS tool stands out as perfect tool to test resilience with the below additional features.

* We combine traditional Chaos Monkey service with monitoring service.
* We provide statistics to understand the behaviour in case of failure.
* We predict how long it would take to recover from a failure after testing the system.
* We offer a cheap SaaS.

# Business case identification

Cloud computing is here to stay and many enterprises around the world are exploring the ways to take advantage of the Cloud. Based on a survey conducted by LogicMonitor, it was found that 83% of the enterprise workloads are expected to be in Cloud by 2020 (Columbus, 2018).

Knowing that the workload for the Cloud is going to increase over the next few years, it is essential for both the Cloud service providers such as Amazon web services, Microsoft Azure and Google Cloud platform and Cloud consumers to take additional measures to ensure that their enterprise systems are stable in the Cloud. To ensure that the system is stable in Cloud, companies constantly track various metrics such as network performance, latency monitoring and application monitoring. Apart from monitoring these metrics, companies also ensure resiliency of their system by hosting a redundant implementation of their Cloud service.

Though Cloud computing promises to provide high(er) availability through redundancy, applications still need to be engineered in a certain way to make use of high(er) availability features. Moreover, the failover and recovery process must be exercised and maintained continuously. Therefore, we saw the opportunity of building a software tool that would test the resiliency of the system by randomly killing (redundant) instances. This tool allows software engineers to perform system stability tests in a realistic manner rather than a lab-like environment. We intend to offer a ready-to-use product for our consumers who only require configuring their target systems that they would want to test.

**Chaos Monkey – A tool to test resiliency with random shutdowns**

Our tool will leverage Chaos Monkey, an open source application developed by Netflix to test their own infrastructure. It uses the concept of Chaos Engineering which is the discipline of experimenting on a system in order to build confidence in the system’s capability to withstand turbulent conditions in production (Principles of Choas Engineering, 2018). By proactively testing how the system behaves under stress, we can identify issues and fix them before it results in major outage in the production. This way we can learn how to build much more resilient systems.

We believe that our tool would be very helpful to the any industry relying heavily on IT systems for their day to day task. This not only includes Enterprise Resource Management systems, but also includes critical financial systems and healthcare systems. System failure to these critical systems can be extremely costly. For example, in Singapore, Healthcare-Cloud (H-Cloud) is a cloud computing platform supporting over 50,000 healthcare staffs to retrieve and access patient records efficiently. H-Cloud would have webservers, application servers and database servers which can be considered as potential target system for Chaos Monkey Resilient test.

**Target Users**

1. IT Service Providers (SaaS providers)

The main target users for our product is IT service providers. By using our tool, IT service providers can upsell to their consumers by marketing their product to be a resilient and stable.

2. Companies hosting their enterprise system in Cloud

We feel that Companies who are moving to Cloud would definitely invest in our product to have a sense of confidence over the system’s stability.

**Limitations**

Currently, our tool would be focusing mainly on instance shutdown. However, in the future release, we would be introducing additional forms of resilient test such latency attack that introduces artificial delays to API calls to simulate service degradation (Lafeldt, 2016). Also, our tool can only kill instances and monitor how quickly the system can be recovered. However, it does not have recovery capabilities on its own.

# Business model

We believe that our product will speak for itself and as such, we would be offering a 30-Day free trial for our customers to have hands on experience. Satisfied users can proceed to purchase Chaos Monkey service.

Our product will be charged based on two key components – the number of target system that needs to be tested and the period for which the customer requires the service.

**Price = Charge/Hour \* Service Duration \* Number of Target Systems**

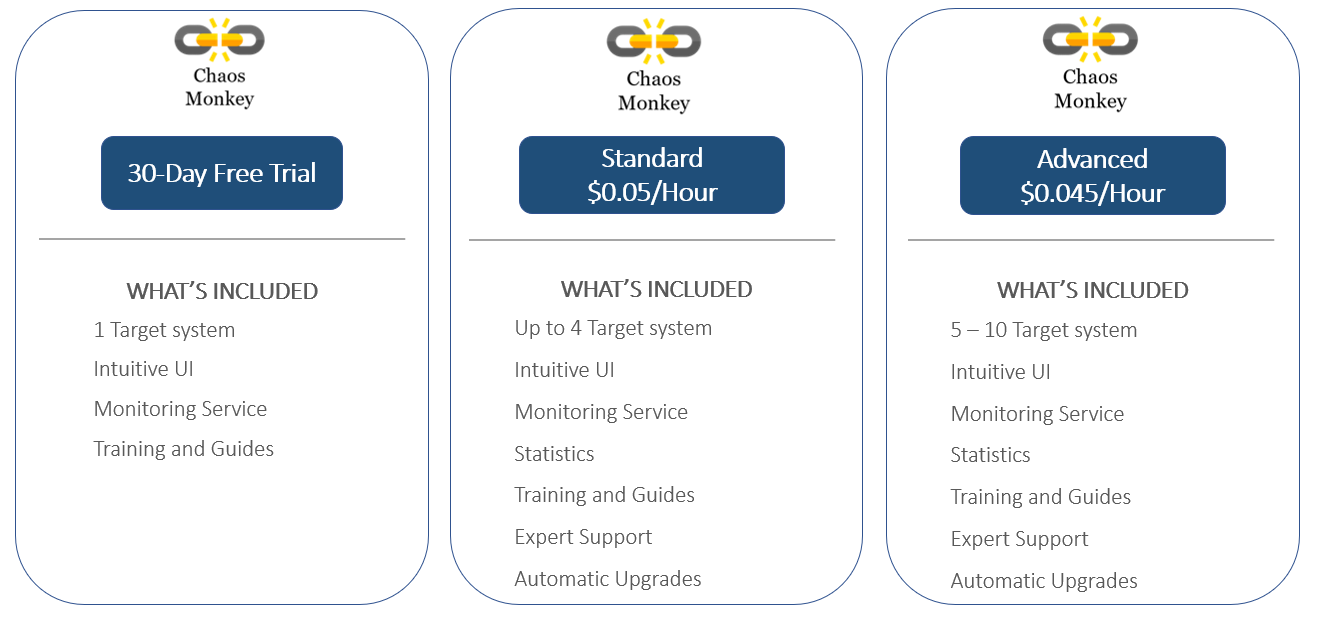


Figure 1:Chaos Monkey Pricing Model

**Pricing Insights**

A customer requiring chaos monkey service for 5 target systems over a 14-day period would be charged $75.60 based on the Advanced subscription charges.

**Price = 0.045/Hour \* 24 Hours \*14 Days \* 5 Target Systems = $ 75.60**

The target users would be getting an essential tool at an affordable price. If they were to develop, install, maintain and upgrade their own resilient tool, the opportunity cost would be significantly higher than employing our service that runs throughout the year.

# SaaS Architecture

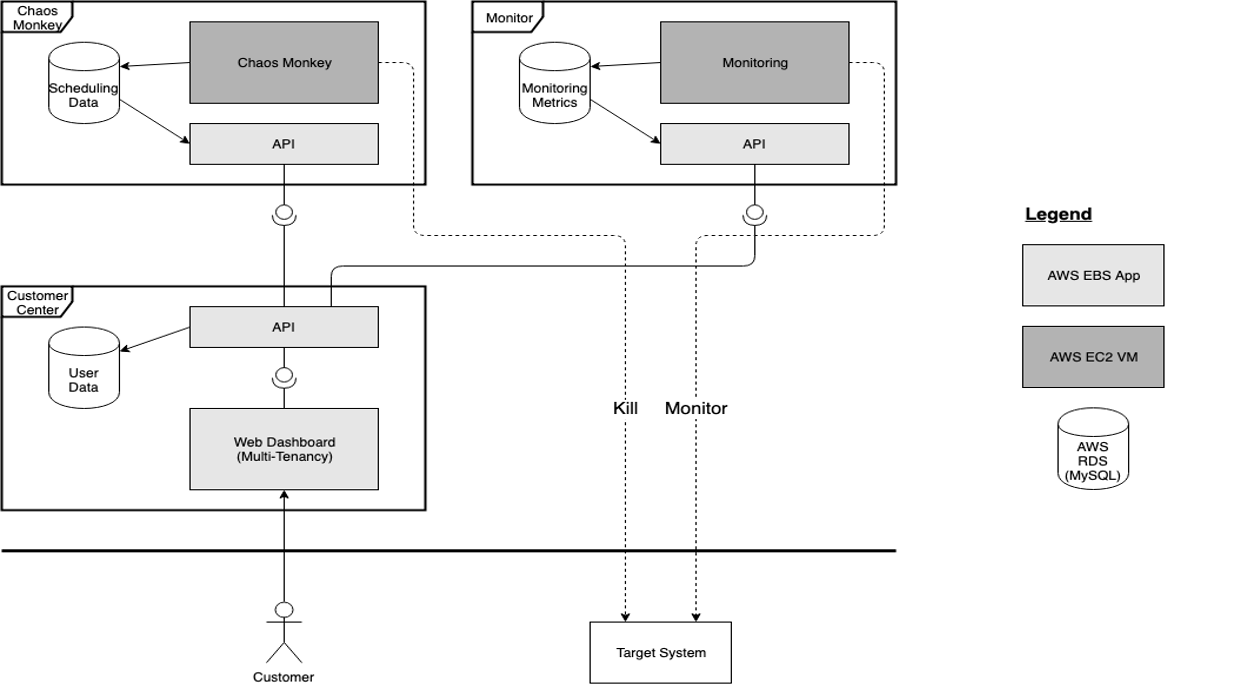
The system consists of 3 main components which are explained in greater detail.

Figure 2: System Architecture

1. **Chaos Monkey**

The Chaos Monkey an open source application developed by Netflix to test their own infrastructure. It’s written in Go and can be installed on any Linux system. The data for the scheduling the random activity and the results of the execution are stored in a MySQL database.

**Cloud Components**

* Chaos Monkey – AWS EC2 VM (Linux)
* API – AWS Elastic Beanstalk web app (Node.js)
* Scheduling database – AWS RDS (MySQL)

1. **Monitor**

The monitor constantly watches the customers target system to detect outages. It’s not connected to chaos monkey and could therefore be offered as an independent service.

**Cloud components**

* Zabbix – AWS EC2 VM (Linux)
* API – AWS Elastic Beanstalk web app (Node.js)
* Monitoring database – AWS RDS (MySQL)

1. **Customer Center**

The customer center is a web application that lets the user add his target system, schedule a chaos monkey run on the target system and later collect the monitoring data. By combining chaos monkey results with the systems health data, the user can see which component was able to stop the system from functioning. In cases of automated disaster recoveries, it can also visualize how long it took until the system was back online.

**Cloud components**

* Dashboard – AWS Elastic Beanstalk web app (React.js, Node.js, Express.js)
* API – AWS Elastic Beanstalk web app (Node.js)
* User data database – AWS RDS (MySQL)

**Use Case**

Company A provides services to survey the customer satisfaction of a restaurant. It has four web server instances and two database instances that run redundantly to perform the surveys. Company A wants to test the availability of the service and see if there is a failure of one of the servers, will it still be able to provide the survey service.

Company A can subscribe to our SaaS service to test the resilience. Our Chaos Monkey will consider web server as one Target System and Database as another Target System and randomly terminate the instances. We will observe the fault tolerance of the Target System and we will provide the monitoring results. So, we will help the Company A to assess their existing system and will give an opportunity to improve their current system. Company A will understand and improve the fault tolerance of their system and will reduce real time outage costs and will not lose the business.

# Screenshot of working prototype

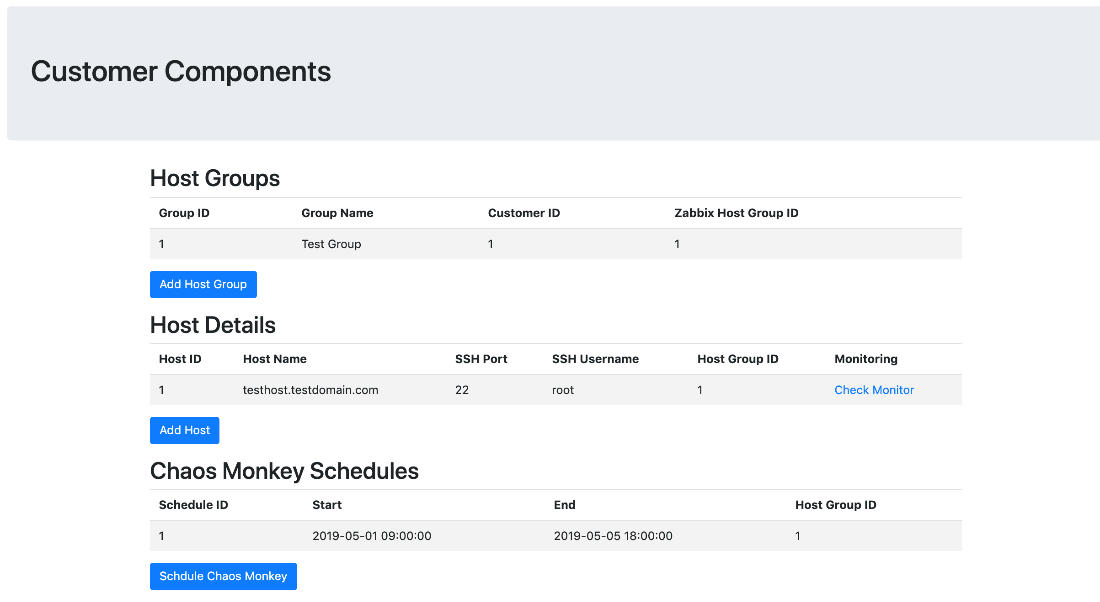


Figure 3: Screenshot of Customer Center

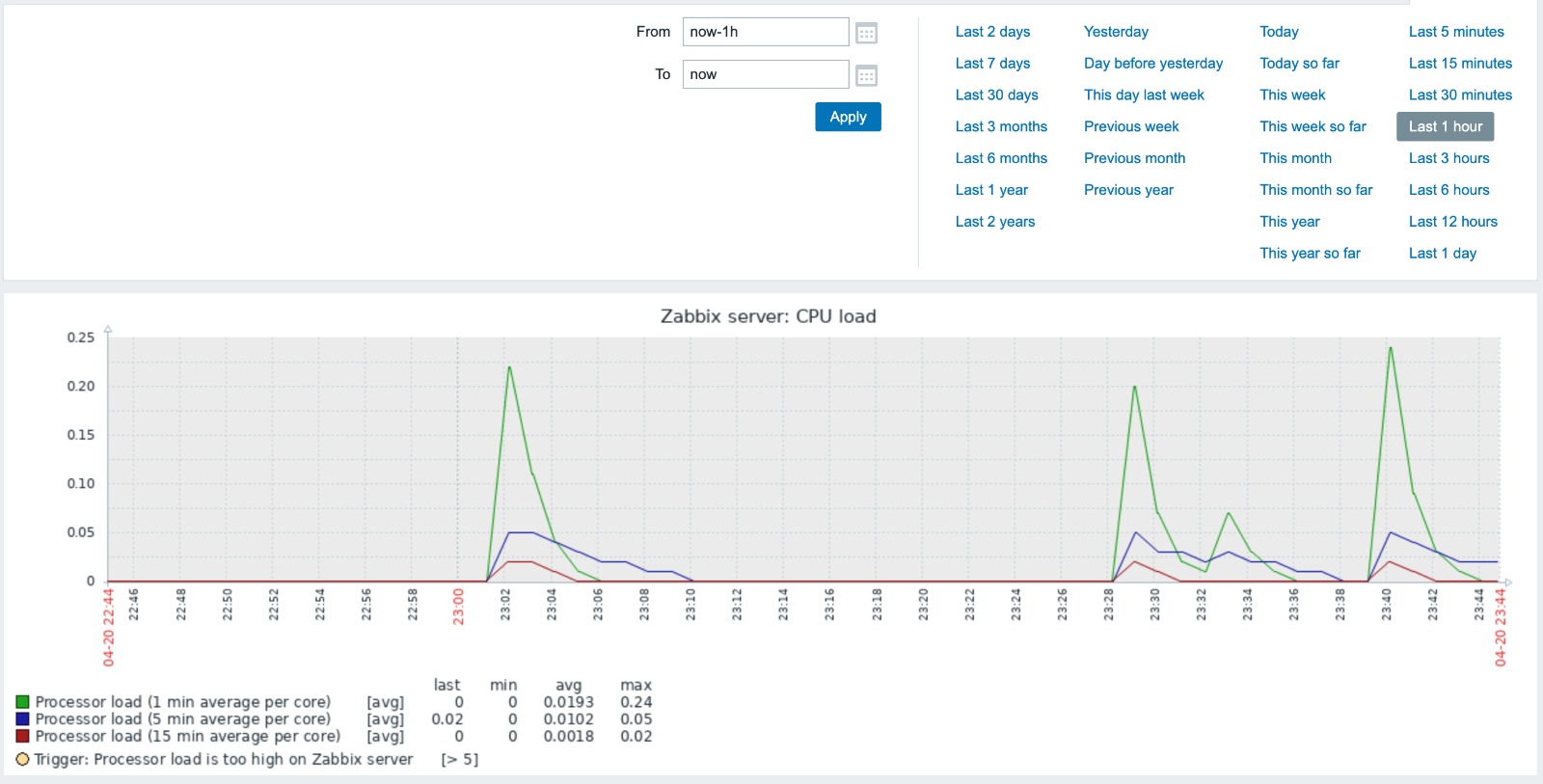


Figure 4: Screenshot of Monitoring Result

# Implementation plan

The three components can be developed in parallel. If one component needs integration before the other component is developed, stubs can be used temporarily.

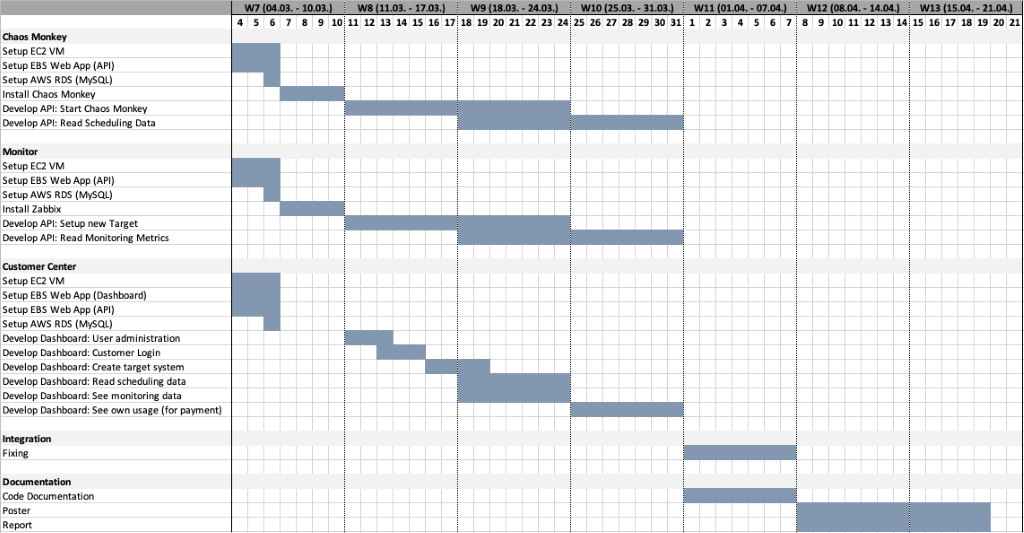


Figure 5: Implementation Plan

# Economic Factors

**Economic benefits and Key considerations:**

In our project, we can optimize the resource utilization by using multi-tenancy as we are providing a SaaS service that do not deal with customer’s critical data.

Other key considerations include providing easy user interface for the customers. The customer instances are isolated from other business consumers and the business process can be customized for each business consumer separately. Tenant isolation is also considered in our project to make sure the business consumers should have the access to the data and configuration information that it owns.

**Pricing Model**

Our SaaS is charged based on the hourly charge, service duration and the number of target systems requiring the service. We considered Separate Pricing model because we are not dealing with the customer data and the underlying infrastructure. Based on the calculation in Figure 6, the estimated Total Cost of Ownership is low ($1153 per annum) as our SaaS service is based on multi tenancy and we are expecting profits in this Pricing Model.

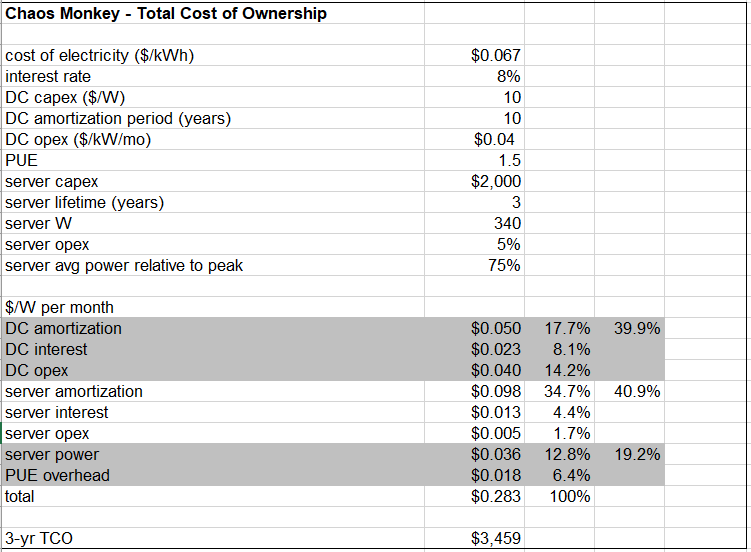


Figure 6: Total Cost of Ownership

**Trade-off between cost and SLA**

Our service tests the fault tolerance of target groups and thereby it saves a lot of time and effort to deal with real – time outages. So, by overall our service saves the outage costs and maintains the SLA of the consumers as they can avoid downtime by performing real time tests and encouraging the customer to fix the issues.

# Conclusion

With our tool, the target users identified in this report would be able to focus on their core competency as we help to monitor the health of their system. We strongly encourage our target users to subscribe our service and make their systems more resilient. It is important to take proactive measures to detect potential issues before it becomes a major outage resulting in lost business, diminished brand reputation and major financial cost.

# References

Columbus, L. (7 Jan, 2018). *83% Of Enterprise Workloads Will Be In The Cloud By 2020*. Retrieved from Forbes: https://www.forbes.com/sites/louiscolumbus/2018/01/07/83-of-enterprise-workloads-will-be-in-the-cloud-by-2020/#6565fb246261

Lafeldt, M. (4 May, 2016). *Chaos Monkey for Fun and Profit*. Retrieved from Medium.com: https://medium.com/production-ready/chaos-monkey-for-fun-and-profit-87e2f343db31

Principles of Choas Engineering. (May, 2018). *Principles of Chaos Engineering*. Retrieved from principlesofchaos.org: https://principlesofchaos.org/