Chaos Engineering – Approach, Tools & Case study.

Pavan Kumar Yarragunta, MD Ejaz Uddin, Bhushan Jain,

Abstract -- When application, Infrastructure and Network crashes on Production, it can have huge impact on the business, Chaos testing and engineering helps to identified the vulnerabilities in the system. Chaos experiment/attack is a team effort. This paper will provide overview of what is Chaos testing, Approach for Chaos testing, different tools and attacks that can be considered for Chaos experiments along with well established case study

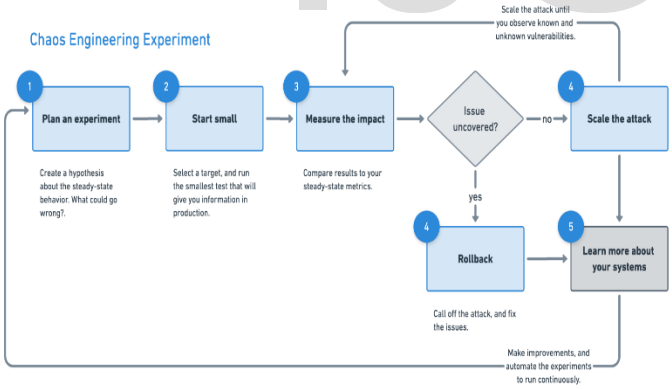
Index Terms— Chaos Engineering, Telecom, Attacks, Gremlin, Vulnerabilities

INTRODUCTION

Chaos Engineering is a disciplined approach to identify vulnerabilities in systems in the production environment. It is implemented to check the system's reliability, stability, and capability of surviving against unstable and unexpected conditions by inducing artificial failures to build confidence in the systems capability.

APPROACH

Following is the recommended approach model to be followed for Chaos experiments



TOOLS

Comparison of the available Chaos engineering tools available in the market (commercial and Open source)

Features	Gremlin	Chaos Monkey	Pumba	PowerfulSeal	Litmus
Platforms Supported	Almost Any Platform	Any platform the Spinnaker supports	Docker Containers	Kubernetes Cluster, OpenStack, AWS, AZURE, GCP	Kubernetes Cluster
Failure Mode	Injects Wide variety of failures	Injects one type of failure i.e. Randomly terminate instances	Crashing Containerized applications, emulating n/w failures and stress testing container resources	Kills targeted pods and takes VMs Up and Down	Kills pod and container, pod CPU hog, pod network latency
Recovery Capabilities	Quickly halt and revert attacks	Not capable	Not capable	Not capable	Not capable
User Control	UI, CLI and API	CLI, Scripts & Config Files	CLI	UI	CLI and Scripts
Other Features	Security Auditing, Attack Monitoring	No auditing, outage checking or termination tracking	None	Stdout, Prometheus and datadog metrics collection	Pod & systems log collection

ATTACKS

Below are the different chaos attacks that can be performed at different layers.

- At Resource Level
- At State Level
- At Network Level

Attack at Resource level

a. **CPU** - Generates high load for one or more CPU cores. CPU attacks can be performed on the following parameters
- Cores - (The number of cores to try to utilize)
- Percent - (The percent of each core to utilize)
- All Cores - (consume all available cores)
- Length - (The length of the attack (seconds)

b. **Memory** - Allocates a specific amount of RAM. Parameters for Memory attack:
- MB - (The number of megabytes to allocate)
- GB - (The number of gigabytes to allocate)
- Percentage - (The percentage of total memory to allocate)
- Length - (The length of the attack (seconds))

IJSER

c. **IO** - Puts read/write or both pressure on I/O devices such as hard disks. IO attacks can be performed on below parameters
   - Dir - (The root directory for the IO attack)
   - *Workers* - (The number of IO workers to run concurrently)
   - Mode - (Do only reads, only writes, or both)
   - Block Size - (Number of Kilobytes (KB) that are read/written at a time)
   - Block Count - (The number of blocks read/written by workers)
   - Length - (The length of the attack (seconds))

d. **Disk** - Writes files to disk to fill it to a specific percentage. Below is the parameter on which attacks can be performed.
   - Dir - The root directory for the IO attack.
   - Workers - The number of disk write workers to run concurrently.
   - Block Size - Number of Kilobytes (KB) that are read/written at a time.
   - Volume Percentage - Percent of Volume to fill (0100).
   - Length - The length of the attack (seconds).

Attack at State Level

Shutdown - Performs a shutdown (and an optional reboot) on the host operating system to test how your system behaves when losing one or more cluster machines.

Time Travel - Changes the host's system time, which can be used to simulate adjusting to daylight saving time and other time-related events. Below are the parameters on which attacks can be performed

- NTP - Disable NTP from correcting system time.
- Offset - The offset to the current time (seconds).
- Length - The length of the attack (seconds).

Process Killer- Kills the specified process, which can be used to simulate application or dependency crashes. (Note: does not work for PID 1, consider a Shutdown attack instead), below are the parameter on which attack can be performed

- Signal - The signal to send to target processes. Values: [HUP,INT,QUIT,ILL,TRAP,ABRT,FPE,KILL,SEGV, PIPE,ALRM,TERM,USR1,USR2]
- Interval - The number of seconds to delay before kills.
- Process - The process name to match (allows regex) or the process ID.
- Group - The group name or ID to match against (name matches only).
- User - The user name or ID to match against (name matches only).
- Newest - If set the newest matching process will be killed (name matches only, cannot be used with -o).
- Oldest - If set the oldest matching process will be killed (name matches only, cannot be used with -n).
- Exact - If set the match must be exact and not just a substring match (name matches only).
- Kill Children - If set the processes children will also be killed.
- Full Match - If set the processes name match will occur against the full command line string that the process was launched with.
- Length - The length of the attack (seconds).

Attack at Network Level

Blackhole- Drops all matching network traffic. Below are the parameters on which attack can be performed

- IP Addresses - Only impact traffic to these IP addresses. Also accepts CIDR values (i.e. 10.0.0.0/24).
- Device - Impact traffic over this network interface.
- Hostnames - Only impact traffic to these hostnames.
- Egress Ports - Only impact egress traffic to these destination ports. Also accepts port ranges (e.g. 8080-8085).
- Ingress Ports - Only impact ingress traffic to these destination ports. Also accepts port ranges (e.g. 8080-8085).
- Protocol - Only impact a specific protocol.
- Providers - External service providers to affect.
- Tags (WebUI and API Only) - Only impact traffic to hosts running Gremlin clients associated with these tags.
- Length - The length of the attack (seconds).

Latency - Injects latency into all matching egress network traffic. *Packet Loss* - Induces packet loss into all matching egress network traffic. Below are the parameters on which attacks can be performed

- IP Addresses - Only impact traffic to these IP addresses. Also accepts CIDR values (i.e. 10.0.0.0/24).
- Device - Impact traffic over this network interface.

IJSER

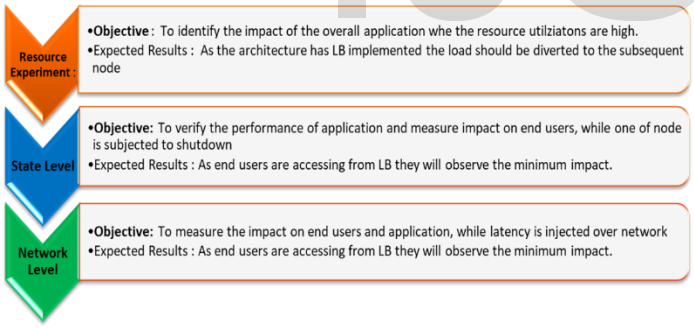
- Hostnames - Only impact traffic to these hostnames.
- Egress Ports - Only impact egress traffic to these destination ports. Also accepts port ranges (e.g. 8080-8085).
- Source Ports - Only impact egress traffic from these source ports. Also accepts port ranges (e.g. 8080-8085).
- MS - How long to delay egress packets (ms).
- Protocol - Only impact a specific protocol.
- Providers(WebUI and API Only) - External service providers to affect.
- Tags (WebUI and API Only) - Only impact traffic to hosts running Gremlin clients associated with these tags.
- Length - The length of the attack (seconds).

DNS - Blocks access to DNS servers, attacks can be performed on the below parameters

- IP Addresses - Only impact traffic to these IP addresses. Also accepts CIDR values (i.e. 10.0.0.0/24).
- Device - Impact traffic over this network interface.
- Protocol - Only impact a specific protocol.
- Providers (WebUI and API Only) - External service providers to affect.
- Tags (WebUI and API Only) - Only impact traffic to hosts running Gremlin clients associated with these tags.
- Length - The length of the attack (seconds).

CASE STUDY

Commercial tool Gremlin was implemented for Major telecom service provider at Philippines, Chaos attacked were carried on one of the major revenue generating platform of the client.



Attack	Result
CPU (30% --> 70% --> 100%)	Even at 100% CPU of all Pods, Node total CPU consumption is Low.
Memory (30% --> 70% --> 100%)	Even at 100% utilization of Memory of all Pods, Node total RAM consumption is Low and No major issues observed.
	Auto-scaled as expected and application recovered in less than 5mins after 100% Memory chaos attack.
Network Latency (300ms --> 700ms --> 1,000ms)	Handled as expected and Application recovered in less than 5mins after chaos 100% Network latency attack.
Network Latency to Database (300ms --> 700ms --> 1,000ms --> Blackhole)	Black hole attack Base DB connection recovery = 16.67 seconds average (27seconds / 10seconds / 13seconds) Traffic handled as expected and DB connection recover time within SLA.
Main Database Failure --> Failover to backup Database	Application handles Database Failover well. Failover DB1 Test: 27 seconds Time-to-Recover Failover DB2 Test: 10 seconds Time-to-Recover Failover DB3 Test: 13 seconds Time-to-Recover -No major impact on application if there's Latency,

IJSER

	or even total disconnection, going to Databases.
Disk Utilization (30% --> 70% --> 100%)	When 30%/70%/100% Disk util commenced, memory utilization of Pods goes above normal. Once memory limit is reached, it will then cause High CPU (primarily on
	major components). Once these pods' CPU consumption reaches above threshold, consumer connection gives error
Terminate/Shutdown ALL Pods	Time to recover for All Pods is less than 1minute. Shutdown

CONCLUSION

Chaos experiments should be planned before releasing of any product/project to live productions, as its kind of approach to identify vulnerabilities in systems in the production environment.

ACKNOWLEDGMENT

The author wishes to The author wishes to thank

Dr. Satish Pai – COO (Communication, Media & Entertainment) – Americas, TechMahindra and L. Ravichandran – President & Chief Operating Officer at Tech Mahindra and Dnyanesh Belitkar - IBU Head - CDUT&A CDU CME Testing & Automation, , Dilip Matur - Principal Technical Architect TechMahindra and my team Md Ejaz Uddin and Bhushan Kumar Jain for showcasing TechMahindra capabilities to the Customer .

REFERENCES

[1] https://www.gremlin.com/

IJSER