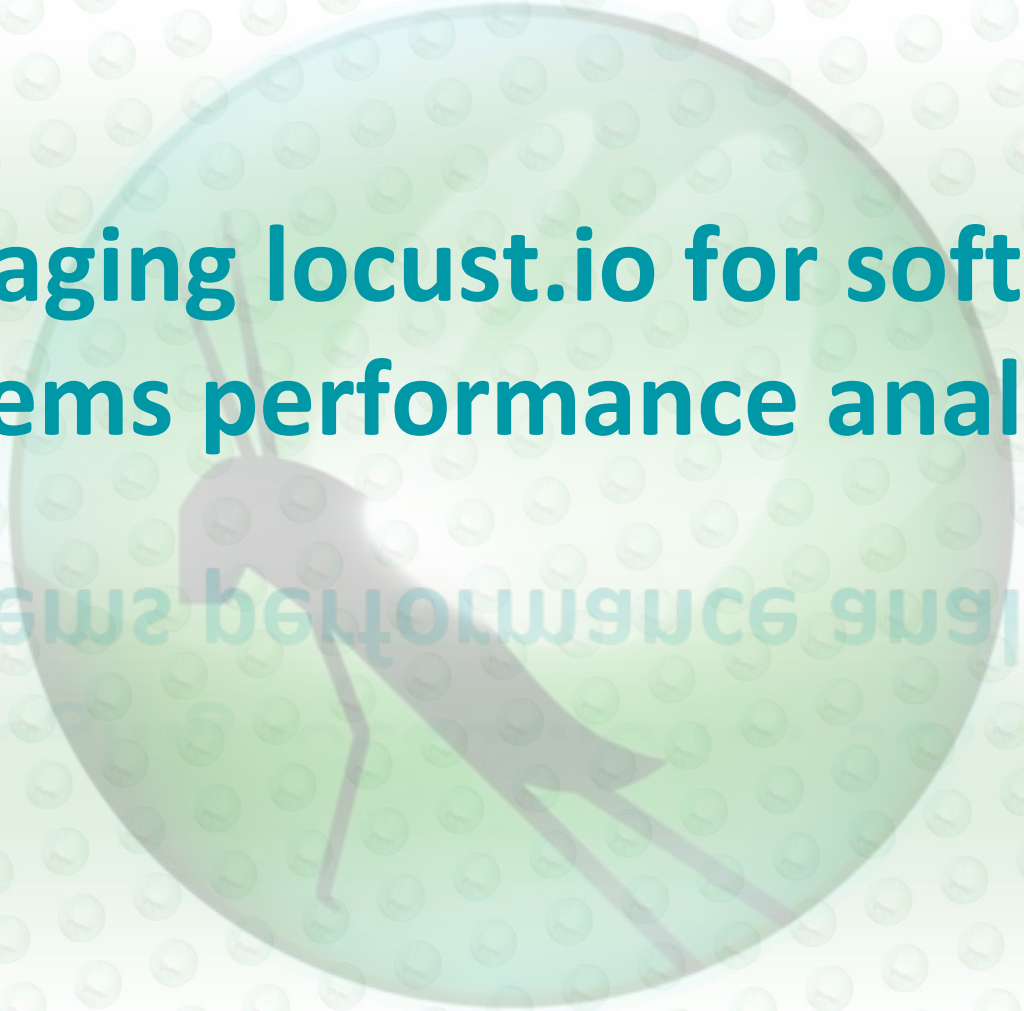
CODE SPACE

Leveraging locust.io for software systems performance analysis



runonautomation@gmail.com

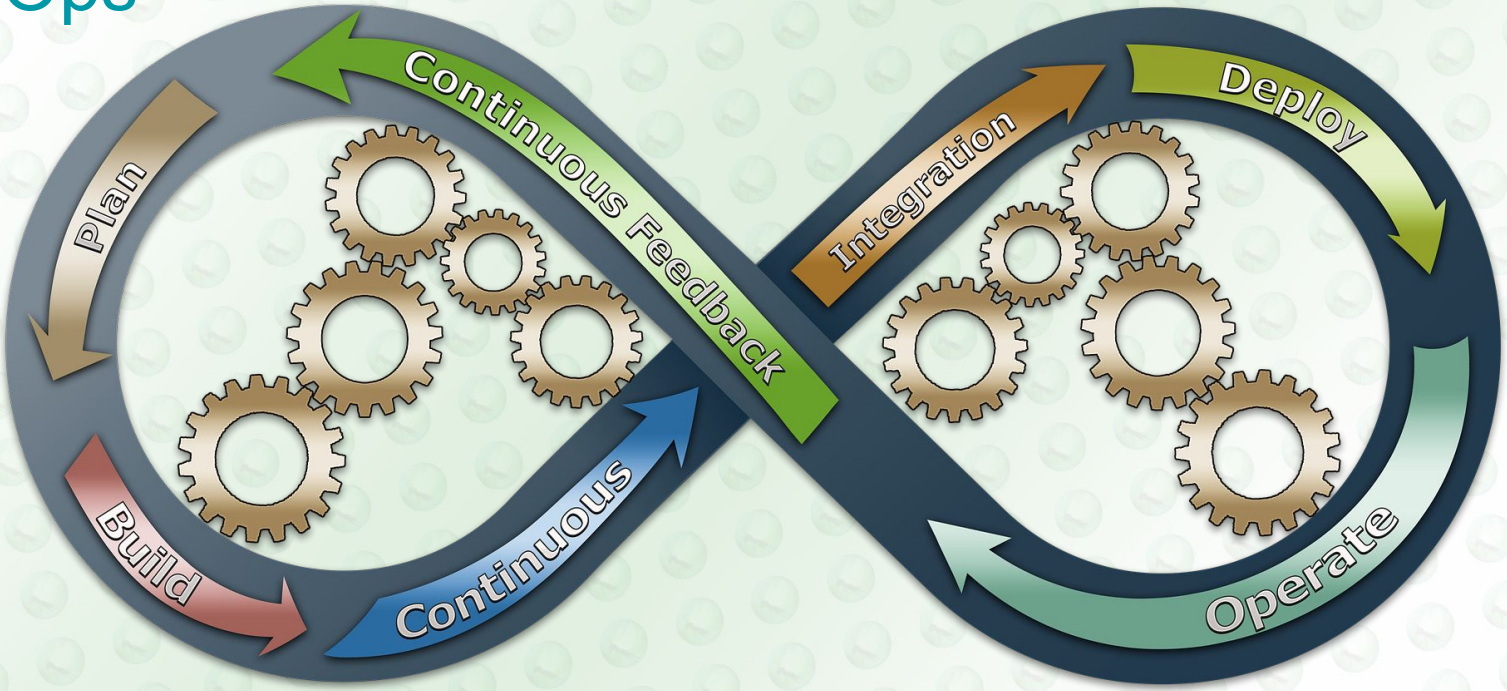
CODE SPACE

Performance Quality Operations

Culture. Infrastructure. Tools.

Agile && DevOps

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan



CODE SPACE

...in performance engineering

- Define service quality metrics including performance based on Business requirements
- Capacity planning
- Architecture with horizontal scalability in mind

- Continuous performance testing
- Automated stress, load testing
- Static code analysis and vulnerabilities management
- Soak testing

- Dynamic scaling
- Monitoring for services and infrastructure
- Monitoring of the end user experience
- Root cause diagnostics of service disruptions

Availability

Reliability

Performance

Accessibility

Integrity

Regulatory

Security

Direct and indirect impact of performance characteristics on Service Quality

Cloud Systems Performance Short Version

Throughput - How much RPS of specific workfload? :)

Latency - Will the response time fit in XX milliseconds?

Resources utilization - Is it ok to spawn a VM per X clients?

Scalability - X2 Throughput with X2 capacity?

Availability - Will users notice on X RPS when N nodes go down?

And… Performance testing...

Performance regression

Implement continuous performance testing for system and components

Load

Determine the interpolated load that your production the system can handle

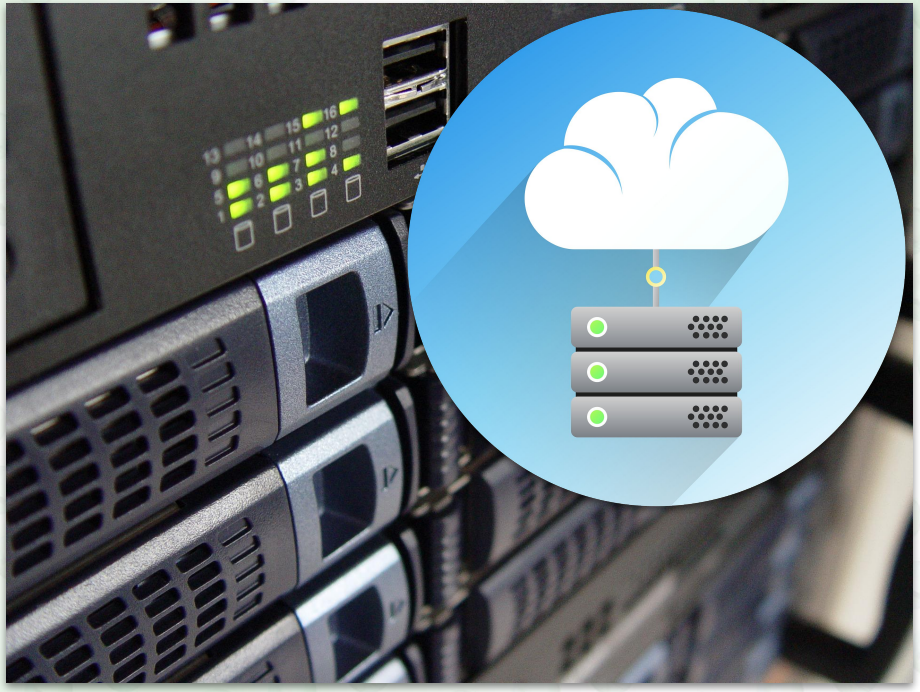
Stress

Verify components durability and recovery time

Infrastructure

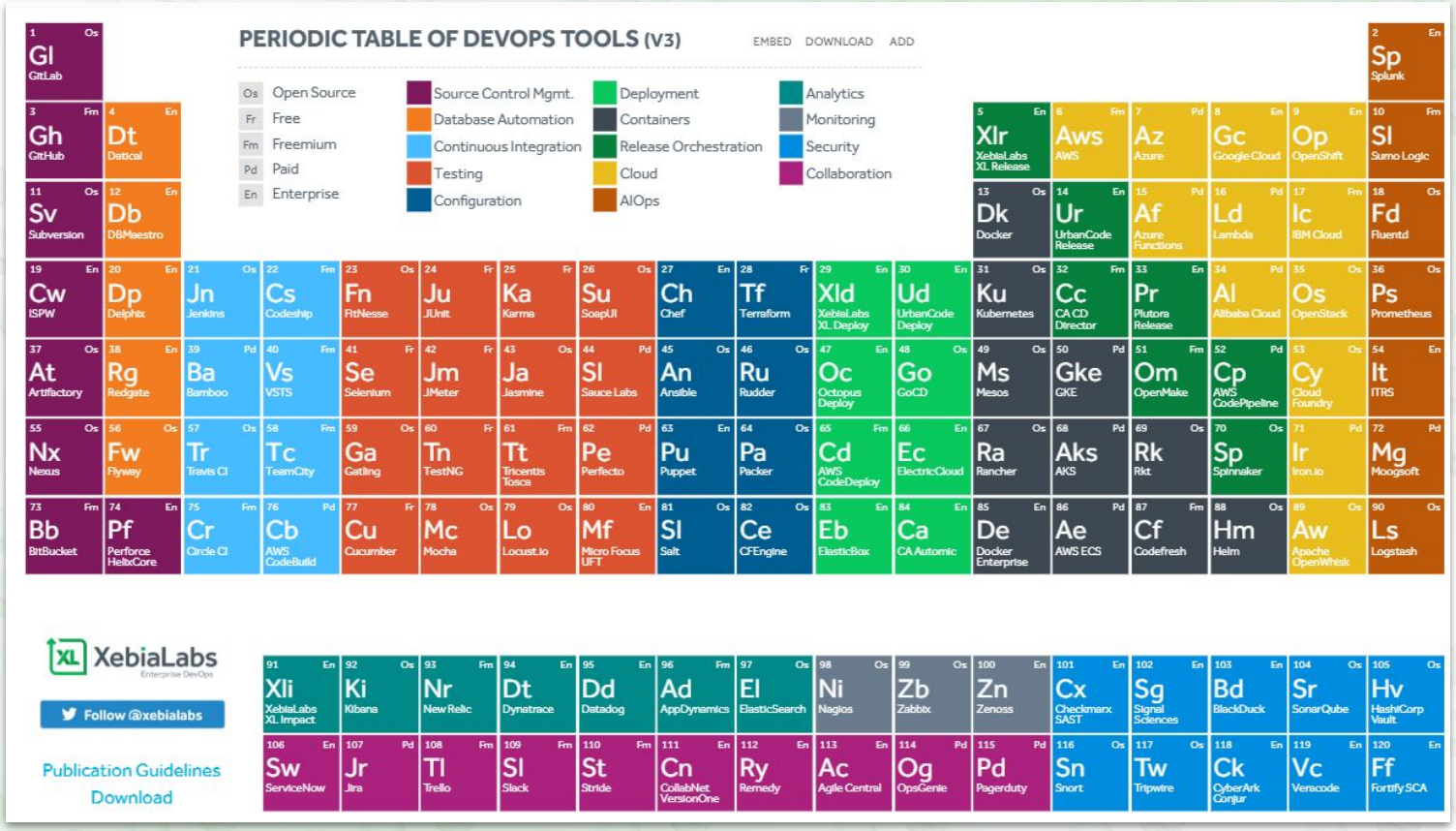
- **Energy efficient computing and storage**
- High availability
- Agile workloads scheduling
- Automated operations
- Maintenance costs

...



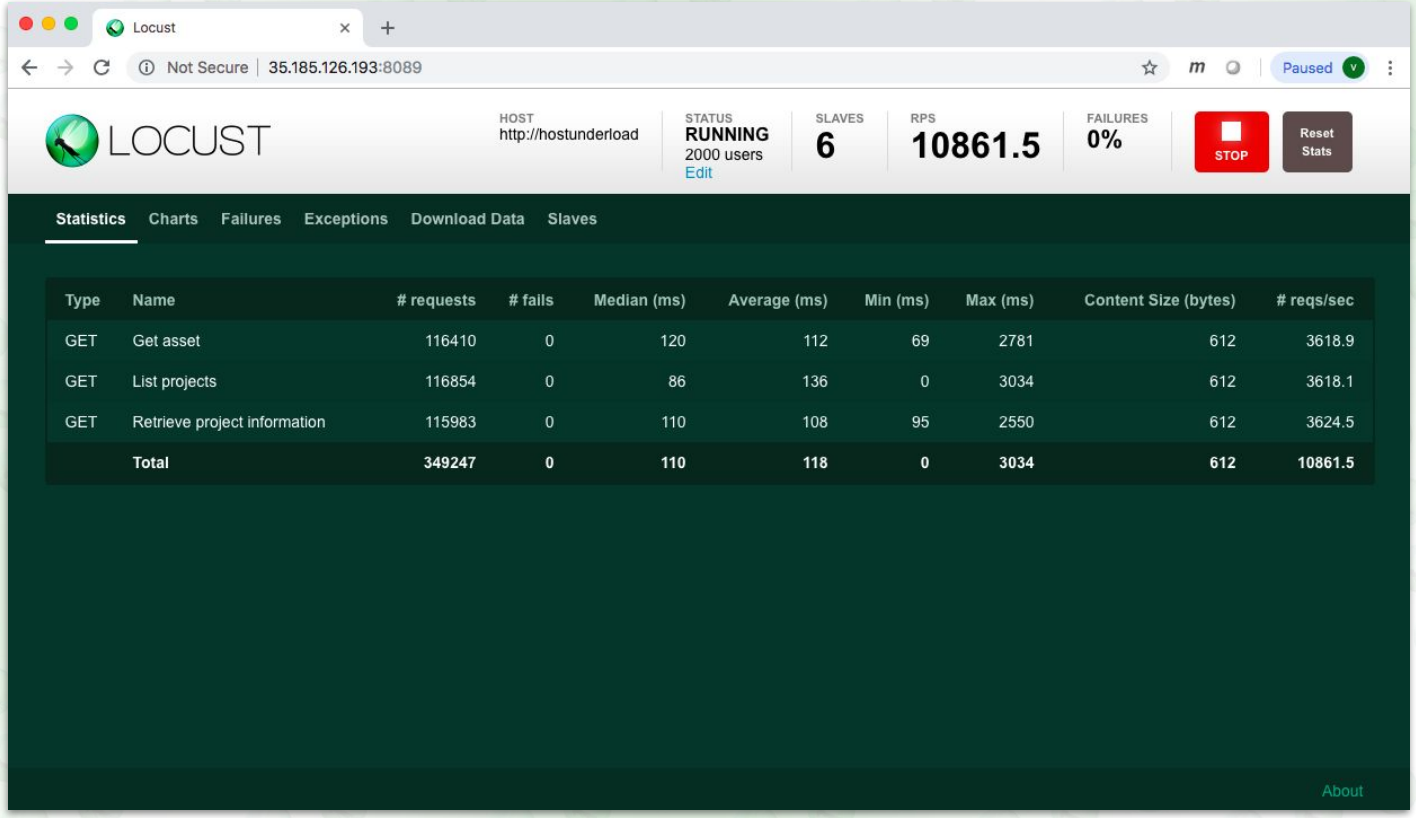
And does not really matter if you are in cloud or on-prem.
What really matters is efficiency.

Toolset


PERIODIC TABLE OF DEVOPS TOOLS (V3)
EMBED
DOWNLOAD
ADD
Os Open Source
Fr Free
Fm Freemium
Pd Paid
En Enterprise
Source Control Mgmt.
Database Automation
Continuous Integration
Testing
Configuration
Deployment
Containers
Release Orchestration
Cloud
AIOps
Analytics
Monitoring
Security
Collaboration
1 Os Gl GitLab
2 En Sp Splunk
3 Fm Gh GitHub
4 En Dt Datical
5 En Xlr XebiaLabs XL Release
6 Fm Aws AWS
7 Pd Az Azure
8 En Gc Google Cloud
9 En Op OpenShift
10 Fm Sl Sumo Logic
11 Os Sv Subversion
12 En Db DBMaestro
13 Os Dk Docker
14 En Ur UrbanCode Release
15 Pd Af Azure Functions
16 Pd Ld Lambda
17 Fm Ic IBM Cloud
18 Os Fd Fluentd
19 En Cw ISPW
20 En Dp Delphix
21 Os Jn Jenkins
22 Fm Cs Codeship
23 Os Fn FitNesse
24 Fr Ju JUnit
25 Fr Ka Karma
26 Os Su SoapUI
27 En Ch Chef
28 Fr Tf Terraform
29 En Xld XebiaLabs XL Deploy
30 En Ud UrbanCode Deploy
31 Os Ku Kubernetes
32 Fm Cc CA CD Director
33 En Pr Plutora Release
34 Pd Al Alibaba Cloud
35 Os Os OpenStack
36 Os Ps Prometheus
37 Os At Artifactory
38 En Rg Redgate
39 Pd Ba Bamboo
40 Fm Vs VSTS
41 Fr Se Selenium
42 Fr Jm JMeter
43 Os Ja Jasmine
44 Pd Sl Sauce Labs
45 Os An Ansible
46 Os Ru Rudder
47 En Oc Octopus Deploy
48 Os Go GoCD
49 Os Ms Mesos
50 Pd Gke GKE
51 Fm Om OpenMake
52 Pd Cp AWS CodePipeline
53 Os Cy Cloud Foundry
54 En It ITRS
55 Os Nx Nexus
56 Os Fw Flyway
57 Os Tr Travis CI
58 Fm Tc TeamCity
59 Os Ga Gatling
60 Fr Tn TestNG
61 Fm Tt Tricentis Tosca
62 Pd Pe Perfecto
63 En Pu Puppet
64 Os Pa Packer
65 Fm Cd AWS CodeDeploy
66 En Ec ElectricCloud
67 Os Ra Rancher
68 Pd Aks AKS
69 Os Rk Rkt
70 Os Sp Spinnaker
71 Pd Ir Iron.io
72 Pd Mg Moogsoft
73 Fm Bb BitBucket
74 En Pf Perforce HelixCore
75 Fm Cr Circle CI
76 Pd Cb AWS CodeBuild
77 Fr Cu Cucumber
78 Os Mc Mocha
79 Os Lo Locust.io
80 En Mf Micro Focus UFT
81 Os Sl Salt
82 Os Ce CFEngine
83 En Eb ElasticBox
84 En Ca CA Automic
85 En De Docker Enterprise
86 Pd Ae AWS ECS
87 Fm Cf Codefresh
88 Os Hm Helm
89 Os Aw Apache OpenWhisk
90 Os Ls Logstash
XebiaLabs
Enterprise DevOps
Follow @xebialabs
Publication Guidelines
Download
91 En Xli XebiaLabs XL Impact
92 Os Ki Kibana
93 Fm Nr New Relic
94 En Dt Dynatrace
95 En Dd Datadog
96 Fm Ad AppDynamics
97 Os El ElasticSearch
98 Os Ni Nagios
99 Os Zb Zabbix
100 En Zn Zenoss
101 En Cx Checkmarx SAST
102 En Sg Signal Sciences
103 En Bd BlackDuck
104 Os Sr SonarQube
105 Os Hv HashiCorp Vault
106 En Sw ServiceNow
107 Pd Jr Jira
108 Fm Tl Trello
109 Fm Sl Slack
110 Fm St Stride
111 En Cn CollabNet VersionOne
112 En Ry Remedy
113 En Ac Agile Central
114 Pd Og OpsGenie
115 Pd Pd Pagerduty
116 Os Sn Snort
117 Os Tw Tripwire
118 En Ck CyberArk Conjur
119 En Vc Veracode
120 En Ff Fortify SCA

Locust.io

Web UI


Locust
Not Secure | 35.185.126.193:8089
Paused
LOCUST
HOST
http://hostunderload
STATUS
RUNNING
2000 users
Edit
SLAVES
6
RPS
10861.5
FAILURES
0%
STOP
Reset Stats
Statistics
Charts
Failures
Exceptions
Download Data
Slaves
Type | Name | # requests | # fails | Median (ms) | Average (ms) | Min (ms) | Max (ms) | Content Size (bytes) | # reqs/sec
GET | Get asset | 116410 | 0 | 120 | 112 | 69 | 2781 | 612 | 3618.9
GET | List projects | 116854 | 0 | 86 | 136 | 0 | 3034 | 612 | 3618.1
GET | Retrieve project information | 115983 | 0 | 110 | 108 | 95 | 2550 | 612 | 3624.5
Total | 349247 | 0 | 110 | 118 | 0 | 3034 | 612 | 10861.5
About

Command line execution

```
Name                                                          # reqs      # fails     Avg     Min     Max  |  Median   req/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET Get asset                                                 176466     0(0.00%)       1       0      20  |       1 4356.30
GET List projects                                             176476     0(0.00%)       1       0      27  |       1 4356.70
GET Retrieve project information                              176454     0(0.00%)       1       0      19  |       1 4355.50
--------------------------------------------------------------------------------------------------------------------------------------------
Total                                                         529396     0(0.00%)                                        13068.50

Name                                                          # reqs      # fails     Avg     Min     Max  |  Median   req/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET Get asset                                                 189343     0(0.00%)       1       0      20  |       1 4349.00
GET List projects                                             189355     0(0.00%)       1       0      27  |       1 4349.40
GET Retrieve project information                              189331     0(0.00%)       1       0      19  |       1 4348.50
--------------------------------------------------------------------------------------------------------------------------------------------
Total                                                         568029     0(0.00%)                                        13046.90

Name                                                          # reqs      # fails     Avg     Min     Max  |  Median   req/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET Get asset                                                 200988     0(0.00%)       1       0      21  |       1 4310.00
GET List projects                                             201003     0(0.00%)       1       0      27  |       1 4310.60
GET Retrieve project information                              200979     0(0.00%)       1       0      19  |       1 4309.90
--------------------------------------------------------------------------------------------------------------------------------------------
Total                                                         602970     0(0.00%)                                        12930.50

Name                                                          # reqs      # fails     Avg     Min     Max  |  Median   req/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET Get asset                                                 200988     0(0.00%)       1       0      21  |       1 4310.00
GET List projects                                             201003     0(0.00%)       1       0      27  |       1 4310.60
GET Retrieve project information                              200979     0(0.00%)       1       0      19  |       1 4309.90
--------------------------------------------------------------------------------------------------------------------------------------------
Total                                                         602970     0(0.00%)                                        12930.50

Name                                                          # reqs      # fails     Avg     Min     Max  |  Median   req/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET Get asset                                                 212632     0(0.00%)       1       0      21  |       1 4170.40
GET List projects                                             212645     0(0.00%)       1       0      27  |       1 4170.70
GET Retrieve project information                              212613     0(0.00%)       1       0      19  |       1 4170.20
--------------------------------------------------------------------------------------------------------------------------------------------
Total                                                         637890     0(0.00%)                                        12511.30
```

Distributed

Web UI:

locust -f locustfile.py --slave --master-host=0.0.0.0 &> locust_slave.log

locust -f locustfile.py --master --host=http://hostunderload

Console:

locust -f locustfile.py --slave --master-host=0.0.0.0 &> locust_slave.log

locust -f locustfile.py -c 5000 --master --host=http://hostunderload --expect-slaves 2 --no-web

HttpLocust

CODE SPACE

locustfile_user.py

```python
from locust import Locust, HttpLocust, TaskSet, task

# TaskSet is a collection of tasks a user (Locust) should
class DemoHttpTaskSet(TaskSet):
    @task
    def index(self):
        self.client.get("/") # session-aware client provided by HttpLocust

class DemoTaskSet(TaskSet):
    @task
    def Demo_task(self):
        print("Locust instance (%r) does a specific task" % (self.locust))

class DemoHttpLocust(HttpLocust): # HttpLocust - user with HTTP client (self.client)
    weight = 2 # DemoHttpLocust with be spawned twice more often than DemoLocust
    task_set = DemoHttpTaskSet
    # minimum and maximum time in milliseconds, that a simulated user will wait
    # between executing each task (randomized)
    min_wait = 1000
    max_wait = 3000

    def setup(self):
        print("(%r) Locust setup - first action")

    def teardown(self):
        print("(%r) Locust teardown - last action")

class DemoLocust(Locust): # Locust class rappresents 1 simulated user
    weight = 1 # DemoLocust with be spawned twice less often than DemoHttpLocust
    # Tasks that user needs to execute
    task_set = DemoTaskSet
```

TaskSet

CODE SPACE

locustfile_taskset.py

```python
from locust import HttpLocust, TaskSet, task
import datetime
class DemoHttpTaskSet(TaskSet):
    def setup(self):
        print("(%r) DemoHttpTaskSet setup (once per class)" % datetime.datetime.now())

    def teardown(self):
        print("(%r) DemoHttpTaskSet teardown (once per class)" % datetime.datetime.now())

    def on_start(self):
        print("(%r) DemoHttpTaskSet on_start" % datetime.datetime.now())

    def on_stop(self):
        print("(%r) DemoHttpTaskSet on_stop" % datetime.datetime.now())

    @task(3)
    def demo_task(self):
        self.client.get("/", name="Demo Task")

    @task(1)
    class DemoHttpSubTaskSet(TaskSet):
        # setup teardown and on_start also applicable
        @task
        def demo_sub_task(self):
            self.client.get("/", name="Demo Sub Task")

        @task
        def stop(self):
            self.interrupt() # Required to terminate the TaskSet execution

class DemoHttpLocust(HttpLocust):
    task_set = DemoHttpTaskSet
```

TaskSequence

CODE SPACE

locustfile_sequence.py

```python
from locust import HttpLocust, TaskSequence, seq_task, task

class DemoTaskSequence(TaskSequence):
    # seq_task - Defines the order of task execution
    # login -> get_info -> get_asset
    @seq_task(1)
    def login(self):
        self.client.get("/", name="Login")

    @seq_task(2)
    def get_info(self):
        self.client.get("/", name="Get info")

    @seq_task(3)
    @task(2) # Defines that the get asset runs twice
    def get_asset(self):
        self.client.get("/", name="Get assset")

class DemoHttpLocust(HttpLocust):
    task_set = DemoTaskSequence
```

IDE debugging

```
1   from locust import HttpLocust, TaskSet, task
2
3   class DemoHttpTaskSet(TaskSet):
4       @task
5       def demo_task(self):
6           res = self.client.get("/", name="Debugging Example")
7           print(str(res))
8
9   class DemoHttpLocust(HttpLocust):
10      task_set = DemoHttpTaskSet
11
12  if __name__ == '__main__':
13      from argparse import Namespace
14      from locust import runners as r
15      opts = Namespace()
16      opts.host = "http://google.com"
17      opts.num_clients = 1
18      opts.hatch_rate = opts.num_clients
19      opts.reset_stats=False
20      r.locust_runner = \
21          r.LocalLocustRunner(locust_classes=[DemoHttpLocust],
22                              options=opts)
23      r.locust_runner.start_hatching(wait=True)
24      r.locust_runner.greenlet.join()
```

HTML Parsing

locustfile_html.py

```python
from locust import HttpLocust, TaskSequence, seq_task, task
from pyquery import PyQuery #pip install pyquery
import random

class DemoTaskSequence(TaskSequence):
    @seq_task(1)
    @task(1)
    def index_page(self):
        r = self.client.get("/", name="Get Root")
        pq = PyQuery(r.content)
        self.links = pq("a") # Get links

    @seq_task(2)
    @task(1)
    def load_page(self, url=None):
        url = random.choice(self.links).attrib["href"]
        print(str(url))
        self.client.get(url)

class DemoHttpLocust(HttpLocust):
    task_set = DemoTaskSequence
    host = "https://github.com/"
```

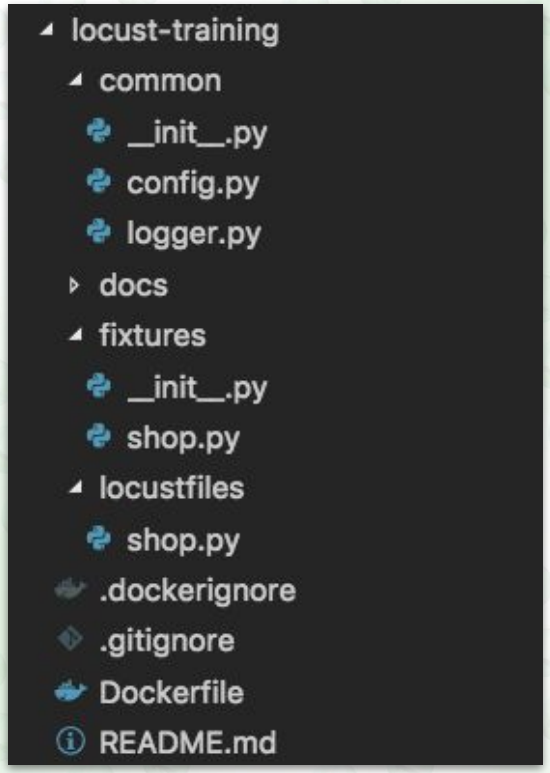
Hacking Options

- **Custom** Clients
- Performance tuning with **FastHTTPLocust**
- **Extending** the API
- Anything… It's just **Python**

Typical implementation

If you need more than just a locustfile.py

Solution structure



- Common utility libraries (format converters, custom clients)
- Fixtures reused in multiple tests to generate test data
- Locustfiles
- Docker and git configuration

Building multi-scenario execution

Codename Taurus

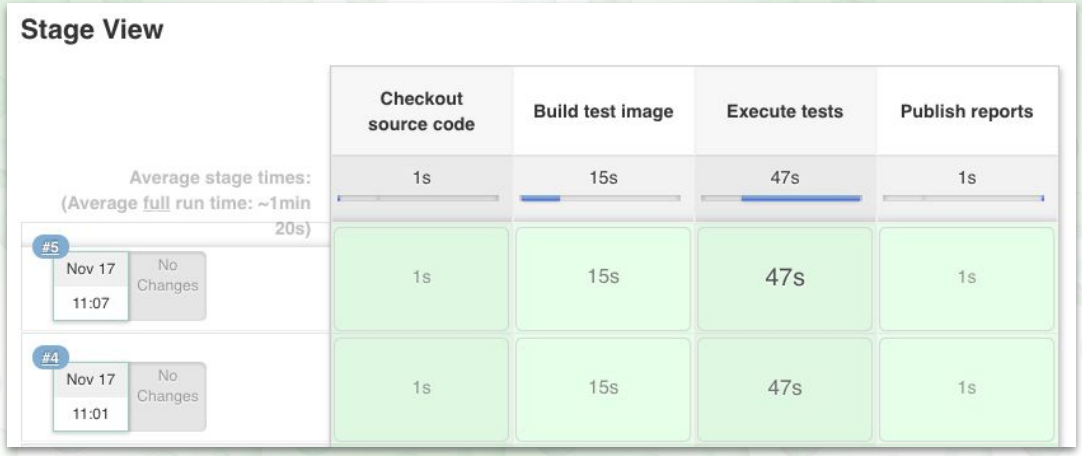
hello.yaml

```yaml
execution:
- executor: locust
  concurrency: 2
  ramp-up: 10s
  iterations: 10
  scenario: hello
  slaves: 2
- executor: locust
  concurrency: 5
  ramp-up: 5s
  iterations: 10
  scenario: hello
  slaves: 4
###
scenarios:
  hello:
    default-address: http://google.com
    script: hello.py
```

pip install bzt

Continuous performance testing

- Use a continuous integration system to run the performance tests
- Track the performance results within CI or with external services



Performance issues analysis toolset

CODE SPACE

Monitoring

Prometheus

influxdb

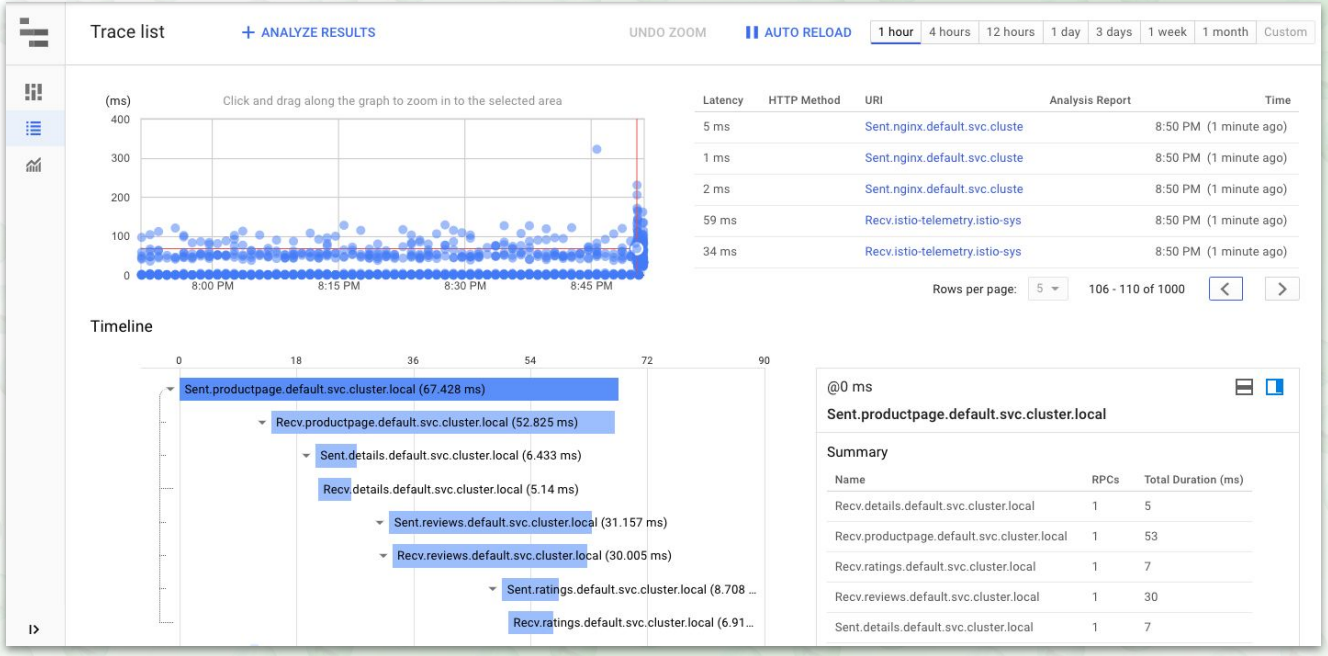
sensu

Nagios®

Stackdriver

DATADOG

Distributed tracing



ZIPKIN

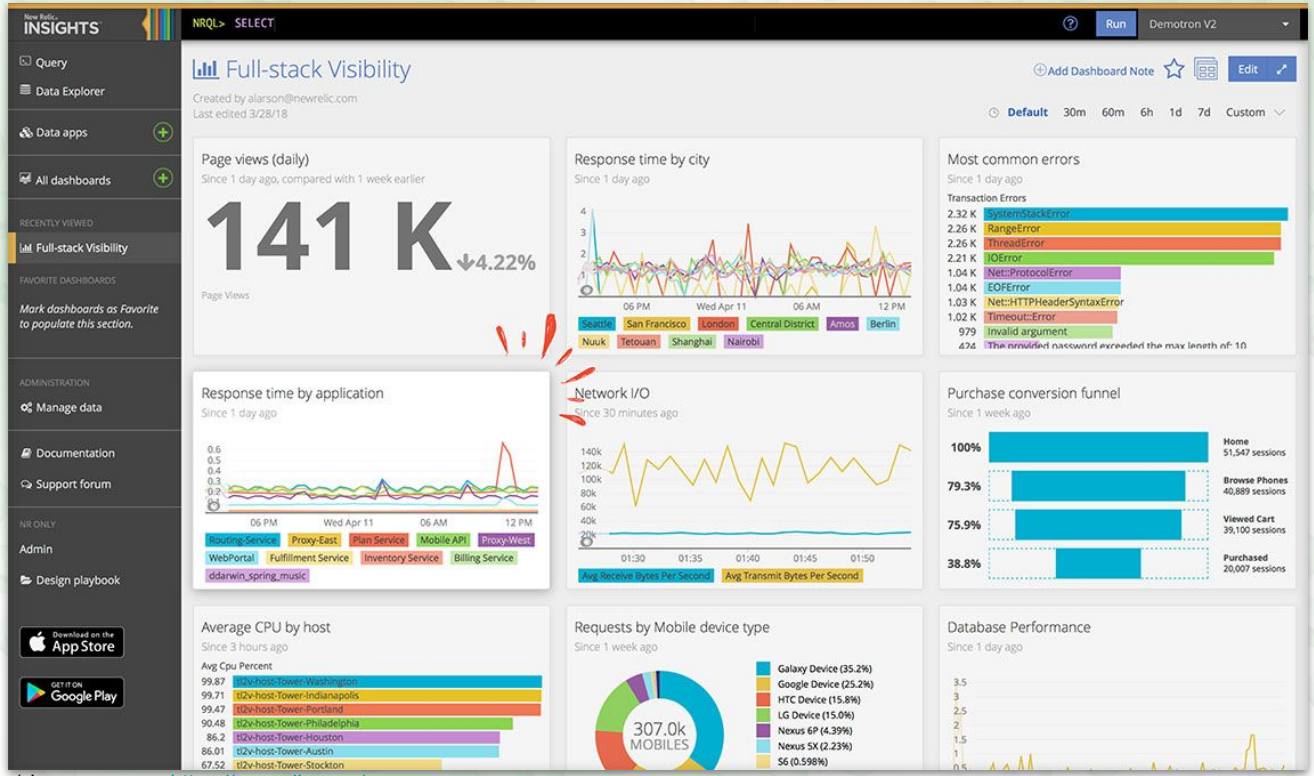
New Relic.

JAEGER

Stackdriver

Applications performance monitoring

- OpenCensus
- Insping
- New Relic APM
- AppDynamics
- Foglight
- ExtraHop
- ...



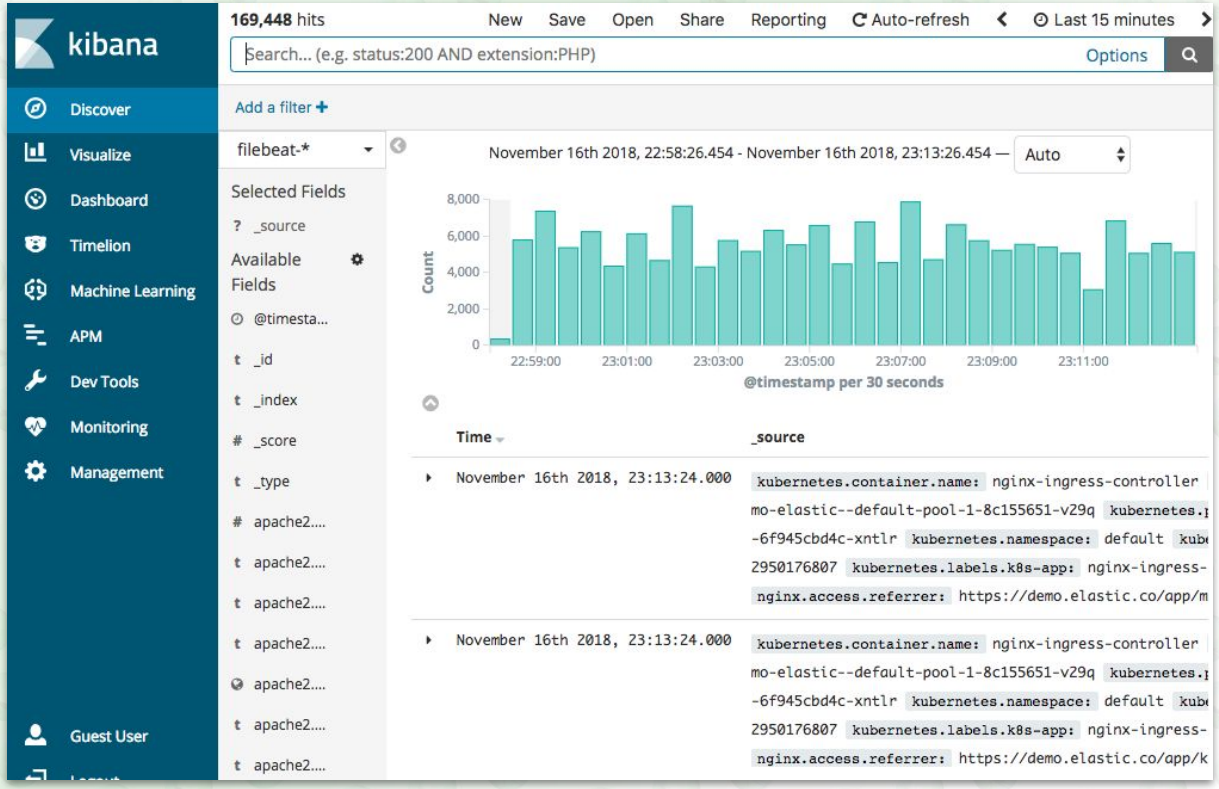
* Image source: https://newrelic.com/

Centralized logging

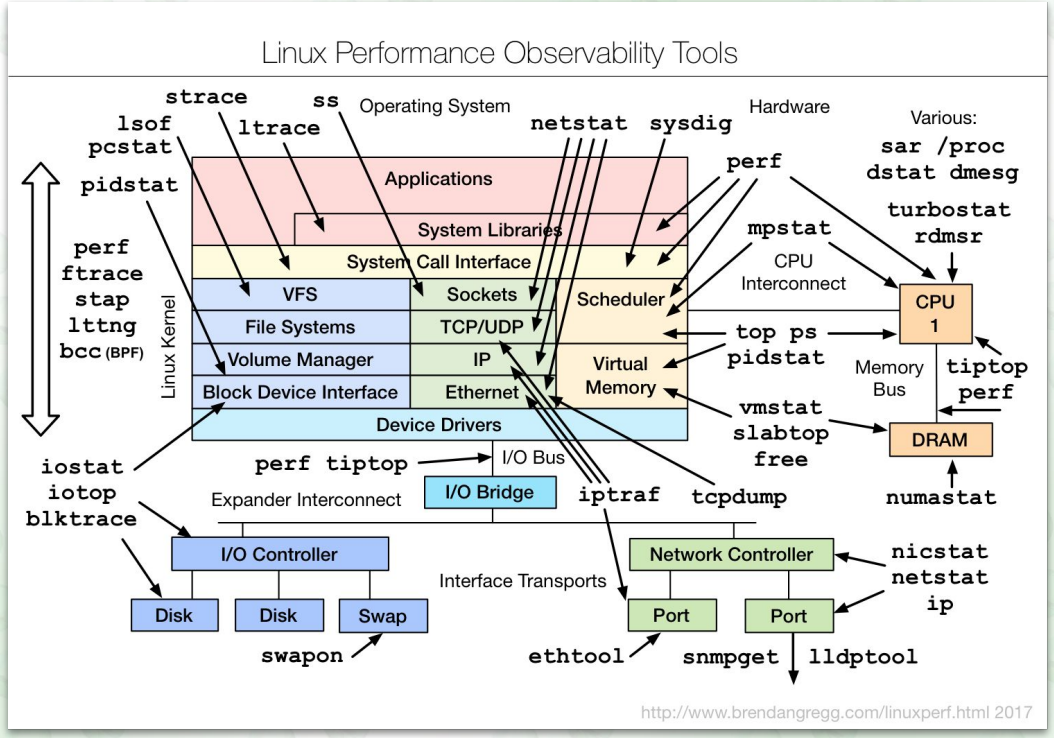
elastic fluentd

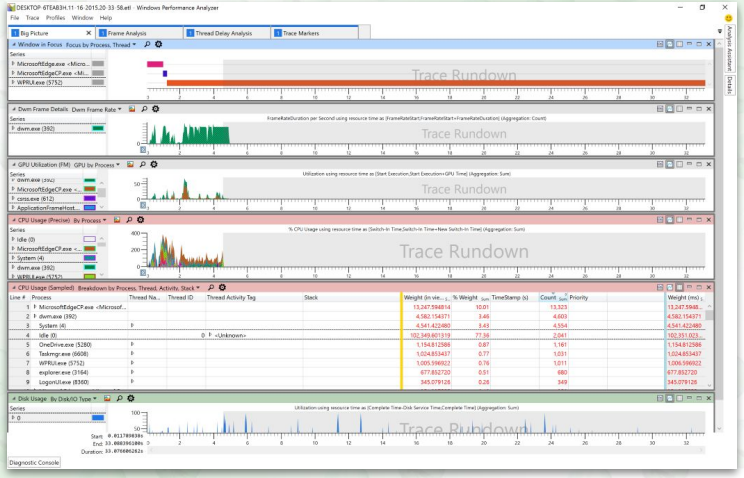
splunk>

graylog



OS Performance analysis





Summary

What’s next?

Performance engineer

Performance engineer job is not to write a script and break the system with many RPS, but it’s to control the software delivery process culture, ensure that core quality gates are in place and are effective.

Review the toolset that is essential in performance engineering:

- Monitoring/APM
- Distributed tracing
- Logging
- Analytics

Locust ...

- IS a good tool to emulate complex user workflows using plain Python scenarios
- IS horizontally scalable and allows simulating 50k+ RPS in distributed mode
- IS customizable
- IS NOT the tool for generating max density RPS for simple GET requests

CODE SPACE

Q&A

CODE SPACE

References

Name	Reference link
The Art of Application Performance Testing, 2nd Edition	http://shop.oreilly.com/product/0636920033233.do
https://github.com/GoogleCloudPlatform/distributed-load-testing-using-kubernetes	https://github.com/GoogleCloudPlatform/distributed-load-testing-using-kubernetes
Linux performance observability	https://medium.com/@chrishantha/linux-performance-observability-tools-19ae2328f87f
Windows performance analyzer	https://docs.microsoft.com/en-us/windows-hardware/test/wpt/windows-performance-analyzer

Name	Reference link
Examples for this training	https://github.com/runonautomation/locust-training
Example solution structure	https://github.com/runonautomation/locust-framework
Locust documentation	https://docs.locust.io/en/stable/
Locust custom clients	https://medium.com/locust-io-lets-get-some-fun/locust-custom-client-23e205f4611f
Extending locust monitoring	https://www.blazemeter.com/blog/locust-monitoring-with-grafana-in-just-fifteen-minutes
Locust experiments	https://medium.com/locust-io-experiments