Reliability Patterns for Large-scale Selenium Tests

Waseem Hamshawi

Senior Automation Engineer, LivePerson

waseemh@liveperson.com
github.com/waseemh
linkedin.com/in/waseem-hamshawi

Liveperson transforms the connection between brands and consumers

3BN
Visits
Per month

200BN
API calls
per month

7
Data
Centers
(worldwide)

6000
Virtual
Servers

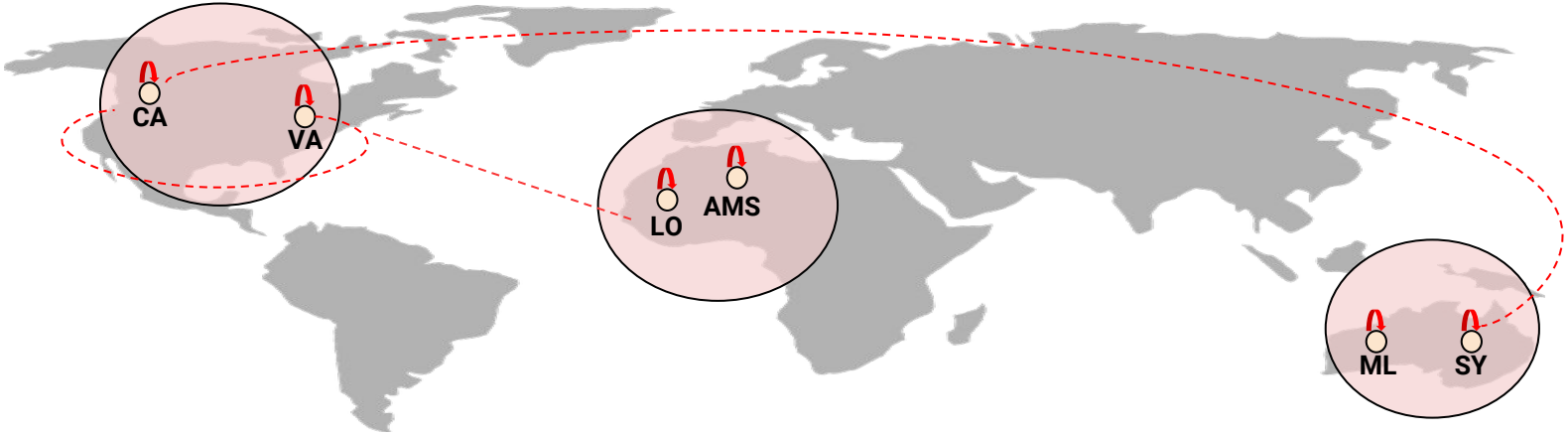
Distributed Testing

25
Scenarios

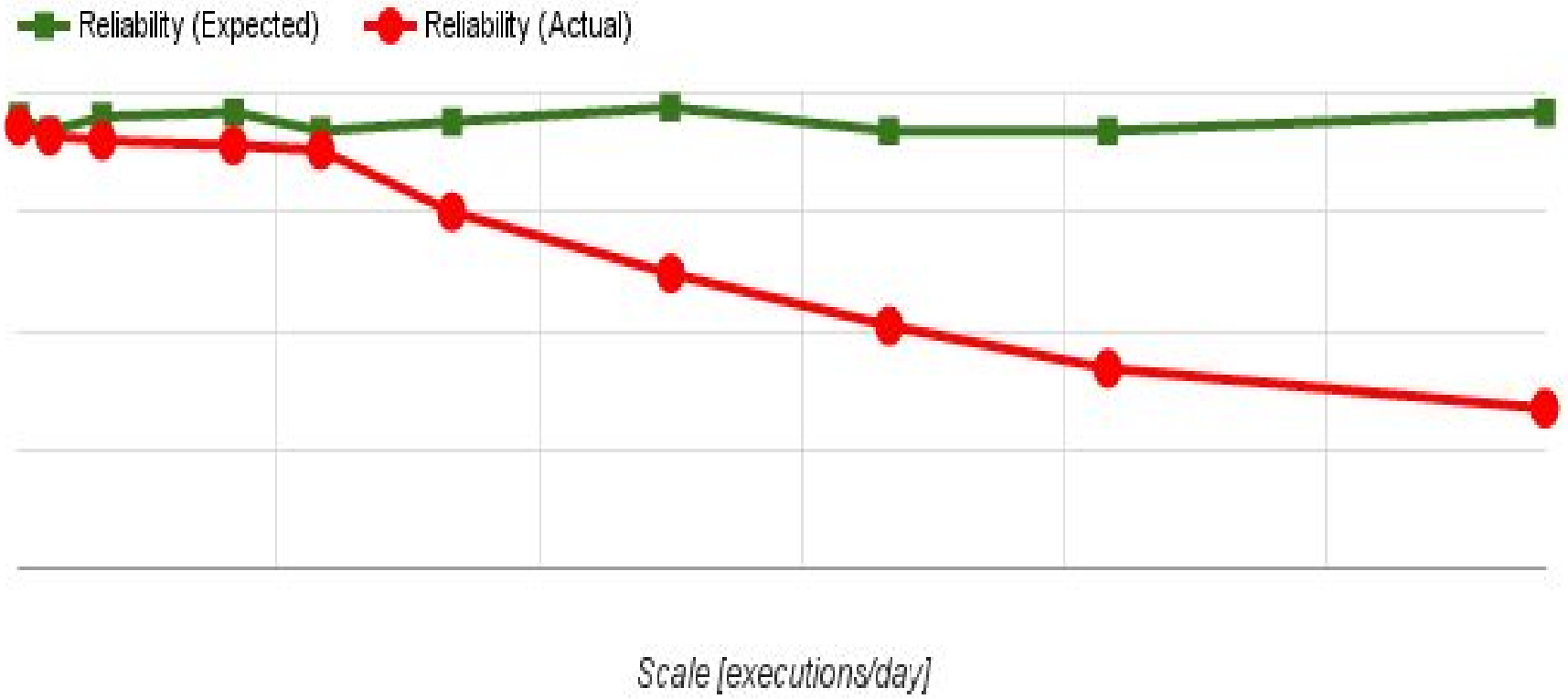
350,000
Executions per day

7
Data Centers
(Geo-distributed)

70
Virtual Machines

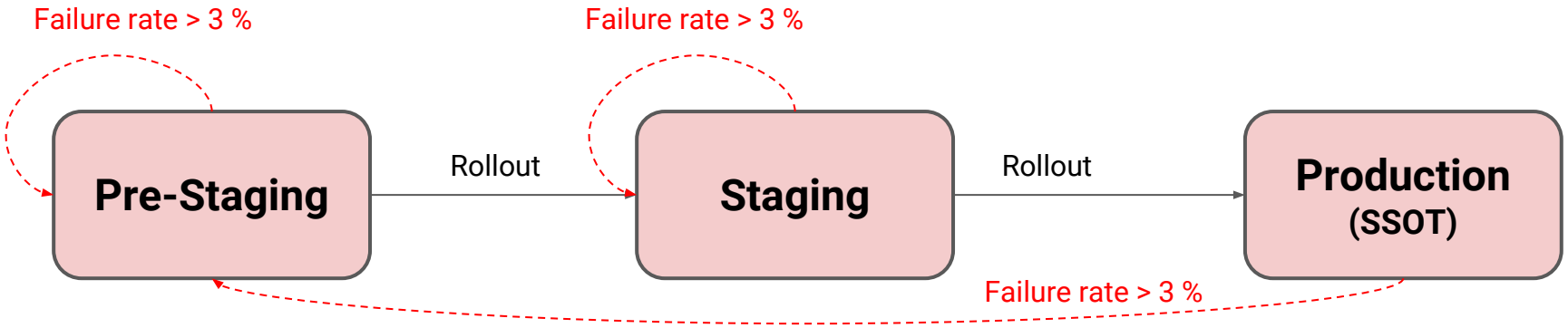


Scalability vs Reliability



Reliable Deployments

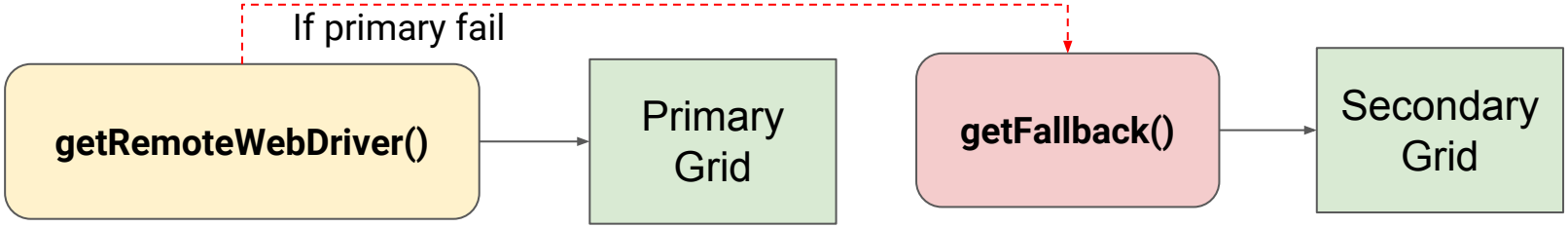
- Scale goes up → More deployments
- Minimize risk when modifying tests codebase
- Apply software deployment principals to test automation
- Multi-tier phased deployment
- Rollout criteria: failure rate < x% for [0,t] time range
- Feature Flags



Single Point of Failures

- Rule of Thumb: External access points and APIs will fail
- Test code should be tolerance to external failures
- Provide Fallbacks to every access point
- Selenium Grid - Single point of failure

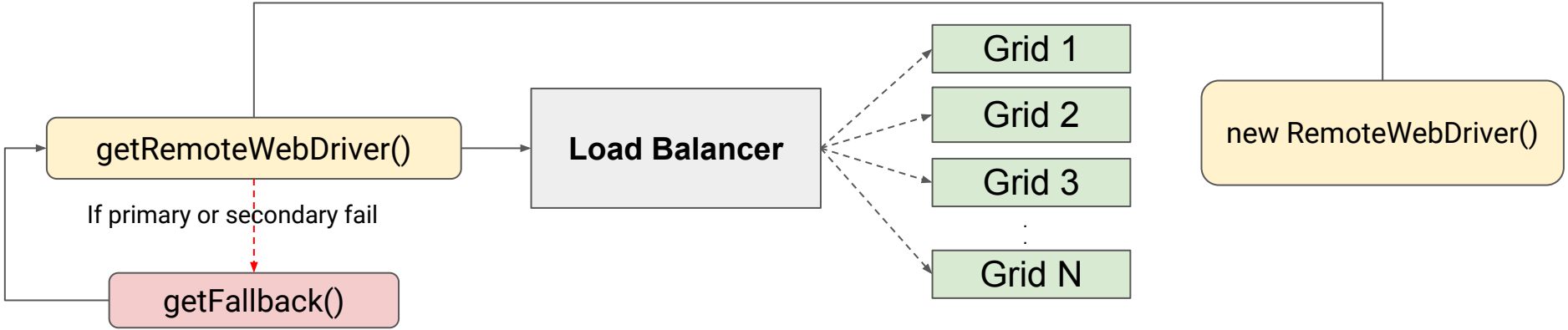
Secondary Hub/Grid failover



Single Point of Failures

Grid Load Balancing

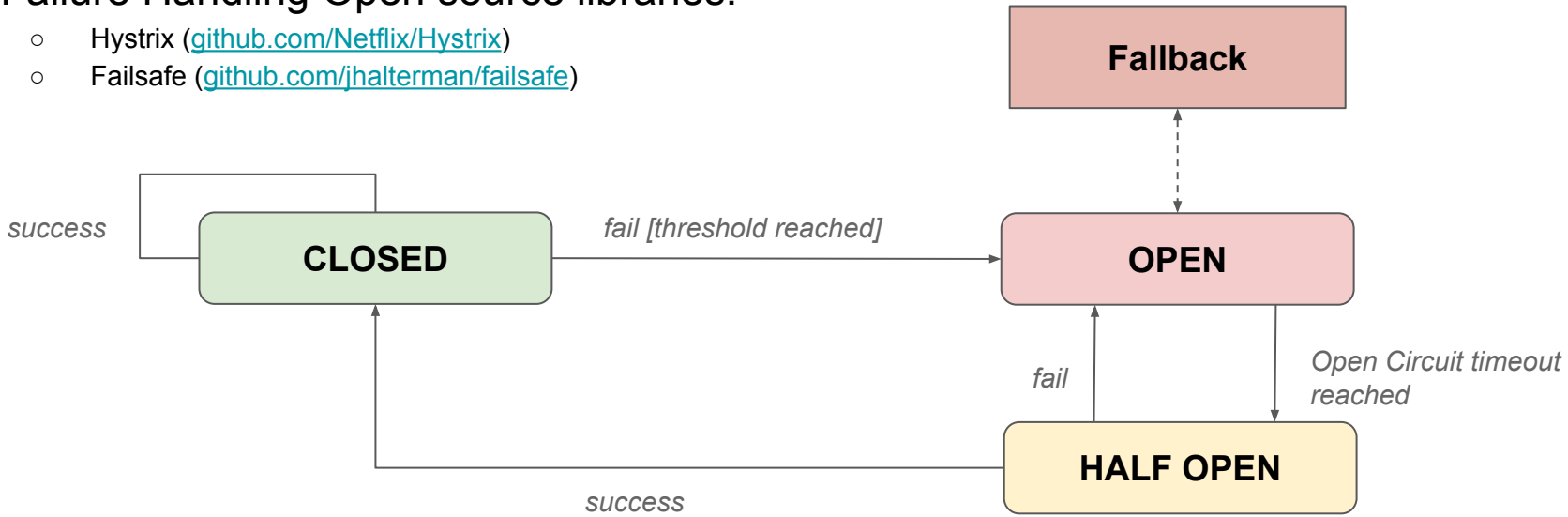
- Breakdown large Grids into smaller ones
- Redundant, highly available Grid setup
- Monitor and load balance
- Open source LB: Ribbon (github.com/Netflix/ribbon)



Single Point of Failures

Circuit Breakers

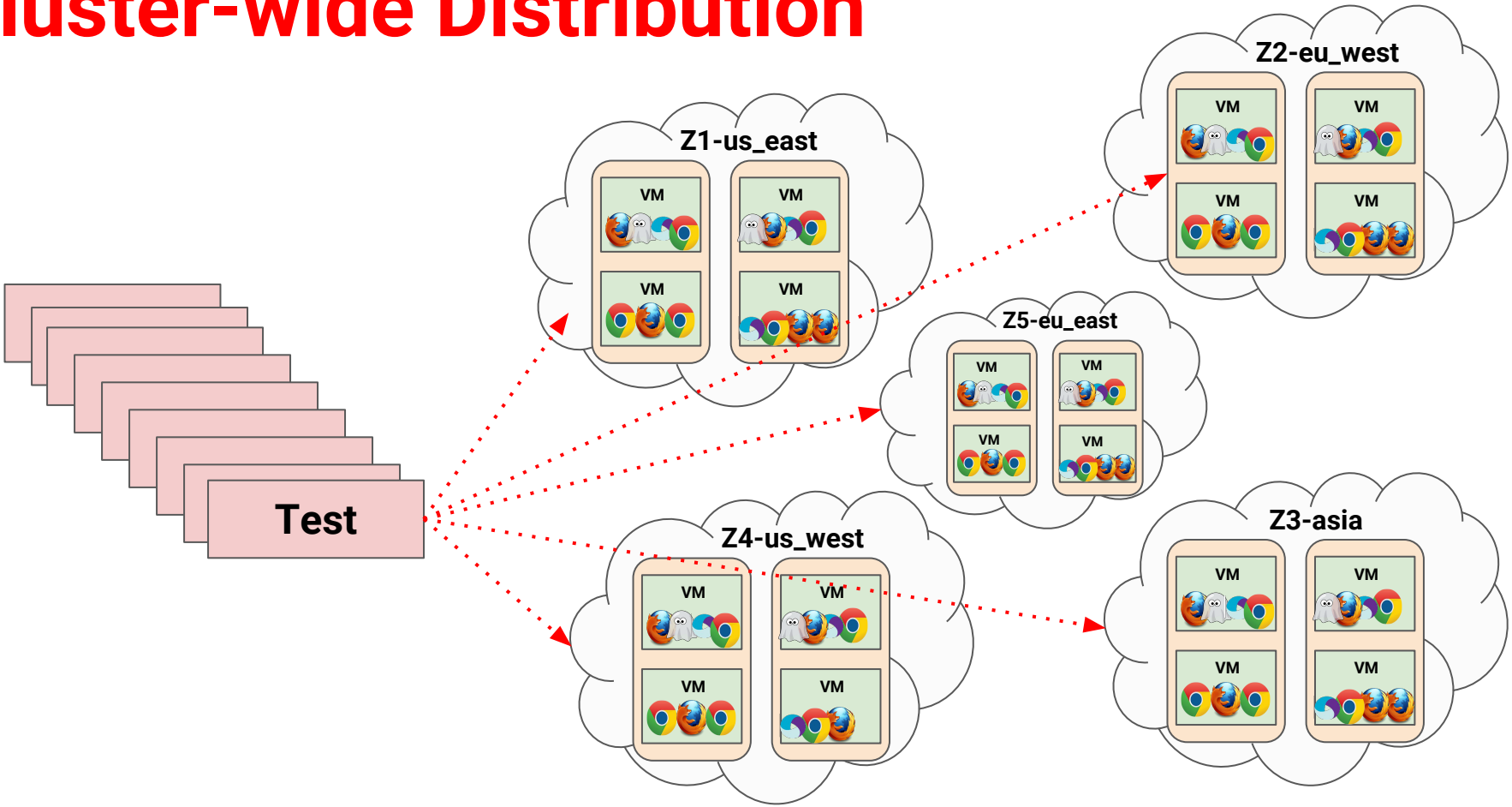
- Don’t beat a dead horse
- Prevent unnecessary timed-out requests
- Shed load on external access point
- Failure Handling Open source libraries:
  - Hystrix (github.com/Netflix/Hystrix)
  - Failsafe (github.com/jhalterman/failsafe)



Hystrix Command - Example

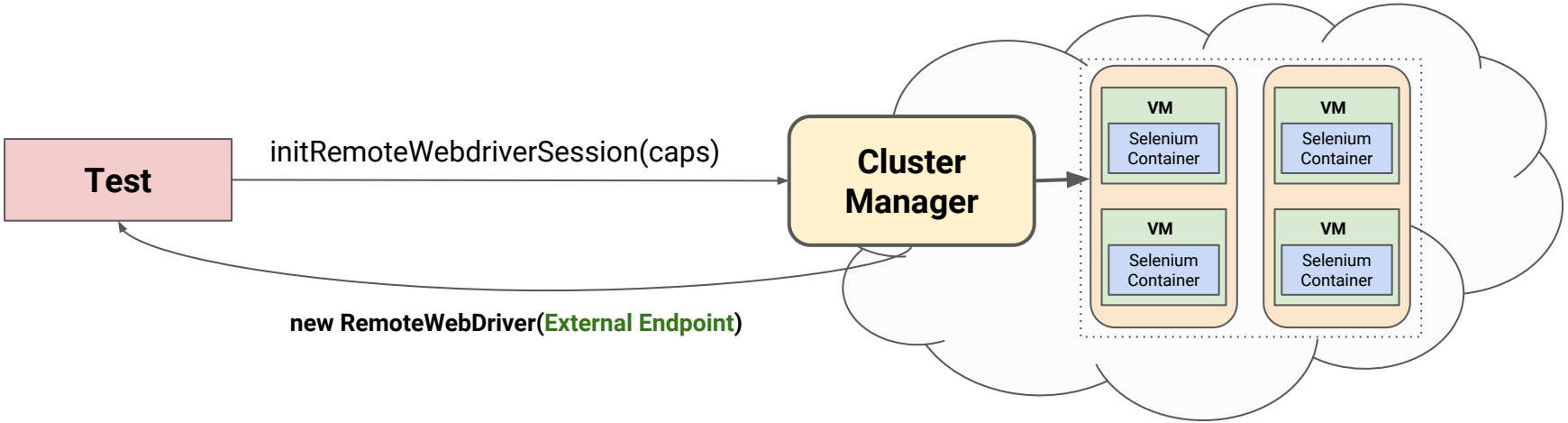
```
public class RemoteWebDriverHystrixCommand extends HystrixCommand<WebDriver> {

    DesiredCapabilities desiredCapabilities;

    protected RemoteWebDriverHystrixCommand(HystrixCommandGroupKey group, DesiredCapabilities desiredCapabilities) {
        super(group);
        this.desiredCapabilities = desiredCapabilities;
    }

    @Override
    protected WebDriver run() throws Exception {
        return new RemoteWebDriver(PRIMARY_GRID, desiredCapabilities);
    }

    @Override
    protected WebDriver getFallback() {
        return new RemoteWebDriver(SECONDARY_GRID, desiredCapabilities);
    }

}
```

Cluster-wide Distribution



Cluster-wide Distribution

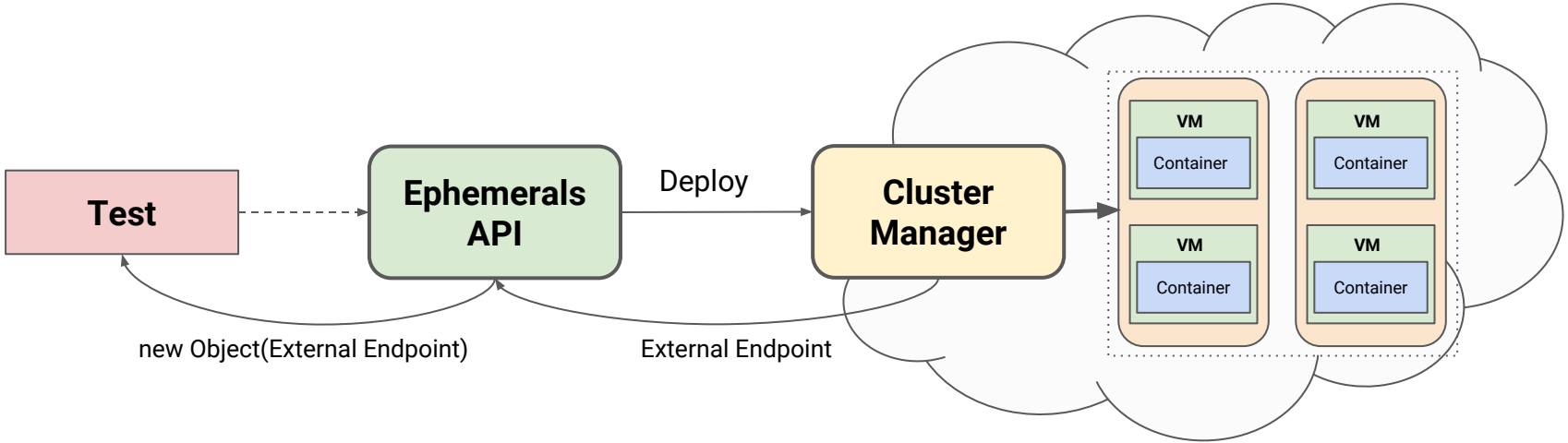
- Launch **short-lived** Selenium **Standalone** Server containers
- Deploy Selenium containers **on-demand**
- Let Cluster Manager do the distribution
- Scale is dependent **only** on cluster's capacity
- Initialize *RemoteWebDriver* based on deployment endpoint (IP:Port)



Ephemerals

A library for creating **short-lived** testing endpoints over container clusters

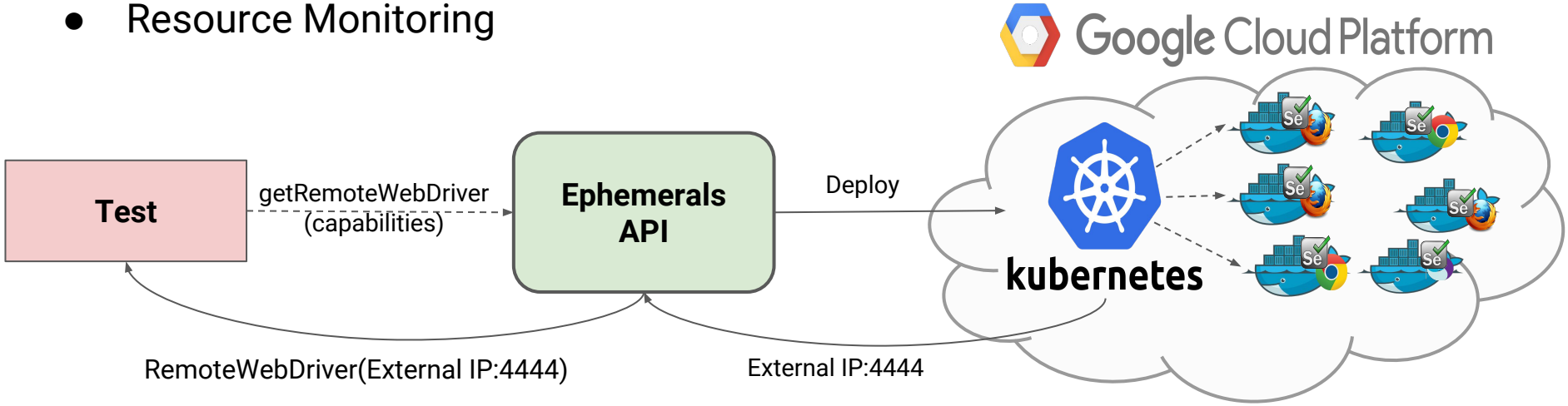
- Tests Integration - Programmatically launch test environment
- Plugable Cluster Management Systems
- Cluster-based Scaling



Use Case: GCP, Kubernetes and Docker

Container Orchestration using Google Container Engine (GKE) and Kubernetes

- Healthchecks (Liveness/Readiness Probes)
- Self-healing mechanisms
- Autoscaling
- Resource Monitoring



Ephemerals - JUnit Integration

```
@Rule
public EphemeralResource<RemoteWebDriver> seleniumResource =
    new  EphemeralResource(
           new SeleniumEphemeral.Builder(deploymentContext)
                  .withDesiredCapabilities(FIREFOX_42)
                  .build());

@Test
public void test() {
   RemoteWebDriver remoteWebDriver = seleniumResource.get();
    .....
}
```

DEMO

github.com/LivePersonInc/ephemerals

Takeaways

- Deploy safely, Just like your software
- No Assumptions - Failure Awareness
- Avoid or protect SPOFs
- Containerization - Short lived, throwaway instances
- Cluster-based Orchestration
- Cloud is the Future - also for Testing