

SWEN438 - DevOps

Lecture 11 - Telemetry

Dr David J. Pearce

*School of Engineering and Computer Science
Victoria University of Wellington*

Telemetry

*“Two hours into the outage, we discovered it was caused by one of the our networking vendors who **accidentally** made a change to our production phone system instead of the hot spare.*

*The outage will impact our **quarterly revenue**, but we don't know how much yet. In order to prevent this from happening again, we're putting together a project to monitor our critical systems for unauthorized changes.”*

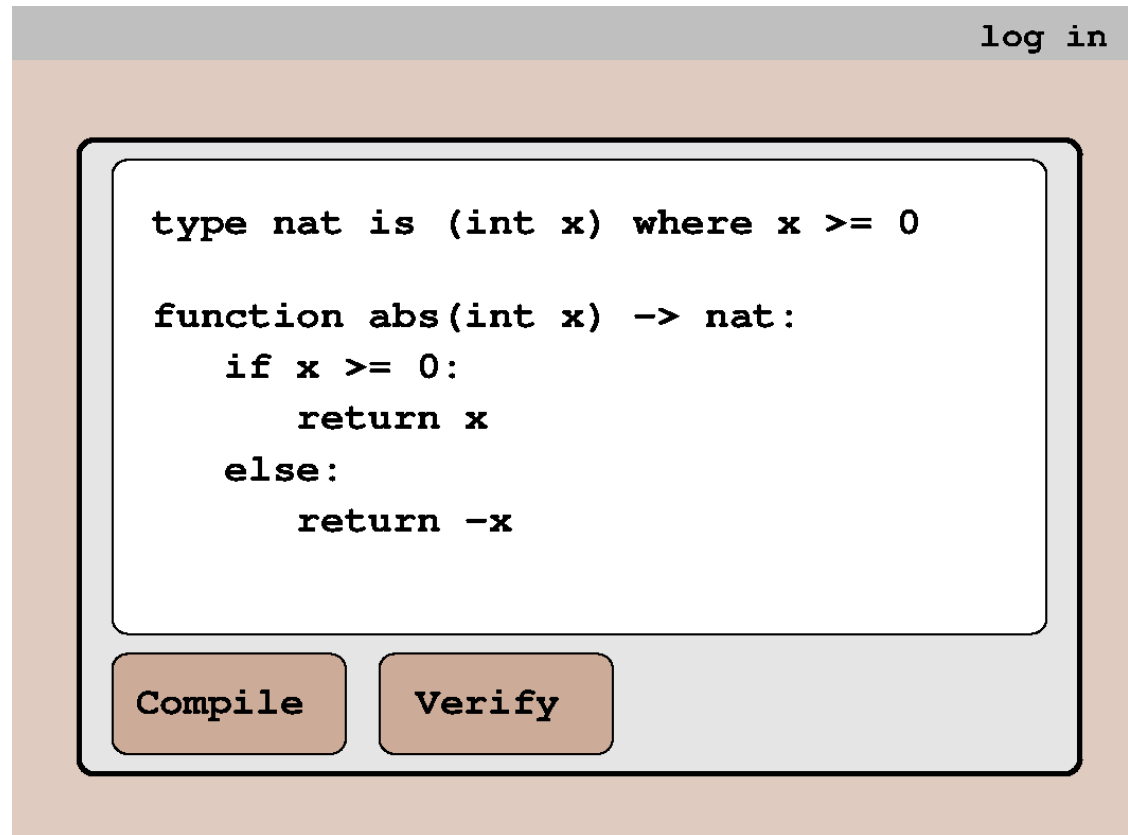
–The Phoenix Project

Telemetry

“an automated communications process by which measurements and other data are collected at remote points and are subsequently transmitted to receiving equipment for monitoring”

–The DevOps Handbook

Telemetry



- What **key metrics** can we record?

Levels of Metric

The diagram illustrates the relationship between code development and business levels. On the left, a code editor window titled "log in" contains the following code:

```
type nat is (int x) where x >= 0

function abs(int x) -> nat:
  if x >= 0:
    return x
  else:
    return -x
```

Below the code are two buttons: "Compile" and "Verify".

On the right, a vertical stack of five levels is shown, representing different layers of abstraction or organization:

- Business Level
- Application Level
- Infrastructure Level
- Client Level
- Deployment Level

- How do our metrics fit with **business organisation**?

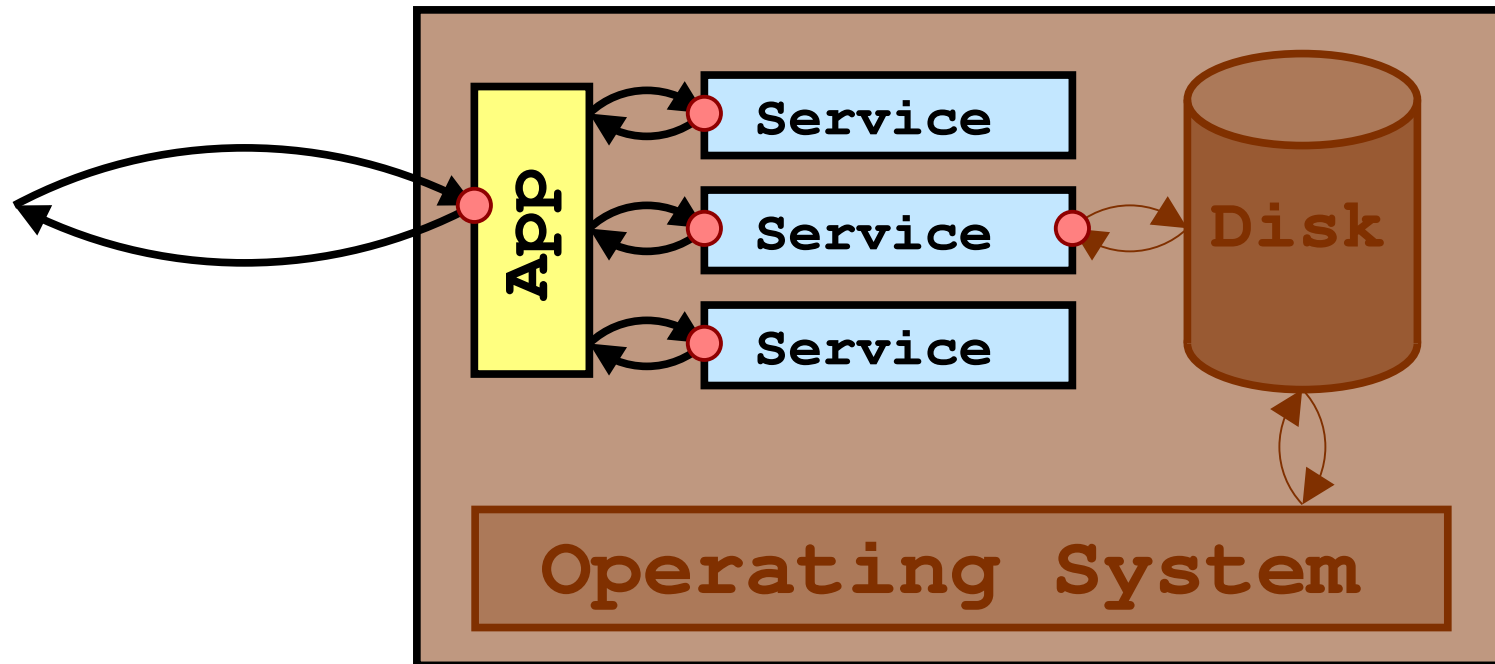
Levels of Metric (Cont'd)

- **Business Level.** E.g. account creations, sales revenue, etc.
- **Application Level.** E.g. time to process request, fault rate, average program size, rate of compile failures, rate of verification failures, etc.
- **Infrastructure Level.** E.g. server traffic, number of concurrent connections, CPU load, memory usage, etc.
- **Client Level** E.g. frequency of interactions, types of interactions, etc.
- **Deployment Level.** E.g. deployment frequencies, lead time for changes, etc.

Goals of Telemetry

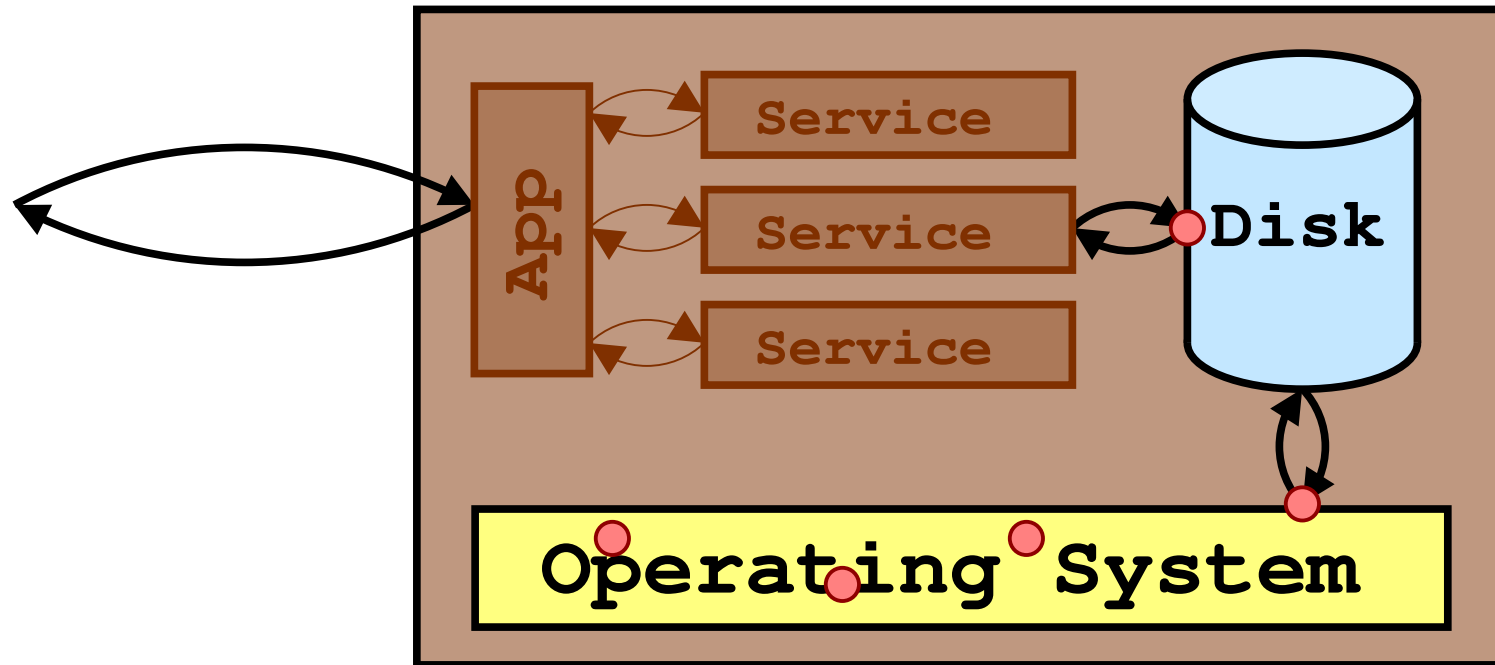
- **Identify** outage or service degradation.
- **Identify** non-functional concerns (e.g. unauthorized activity).
- **Identify** problematic deployments
- **Provide** information for diagnosis and debugging.
- **Provide** information for data analysis (e.g. A/B Testing).

White Box Monitoring



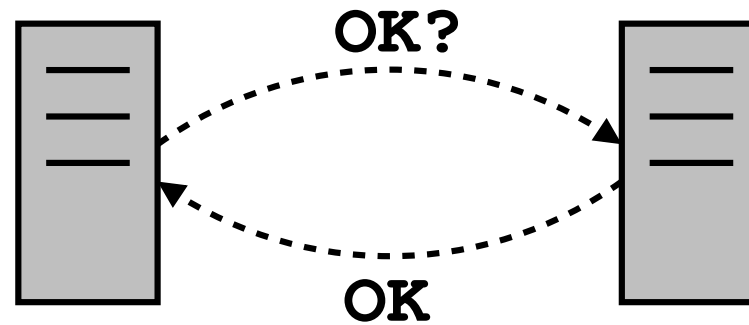
- Monitoring **with** knowledge of application(s) internals
- Can **add** instrumentation to emit telemetry as necessary

Black Box Monitoring



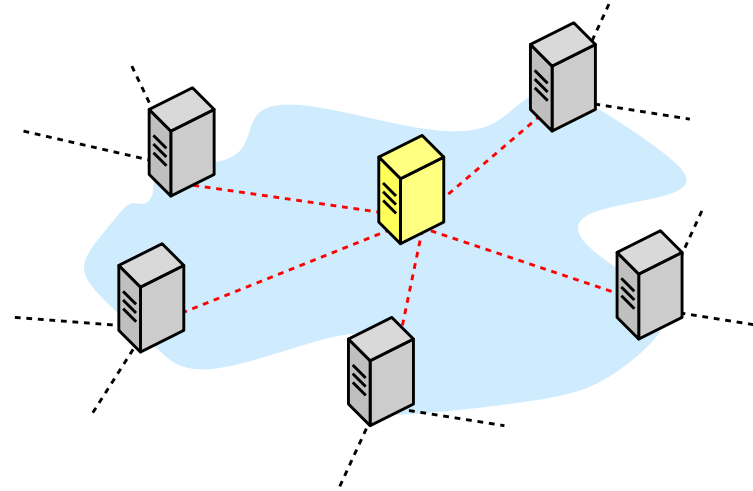
- Monitoring **without** knowledge of application(s).
- May have access to information provided through API
- May have access to server information (e.g. CPU Load)

Health Checks



- Periodically **probe** to check system alive
- Can use **non-specific** probes (e.g. ping or HTTP)
- Can use **application-specific** probes

Content Delivery Networks (CDN)

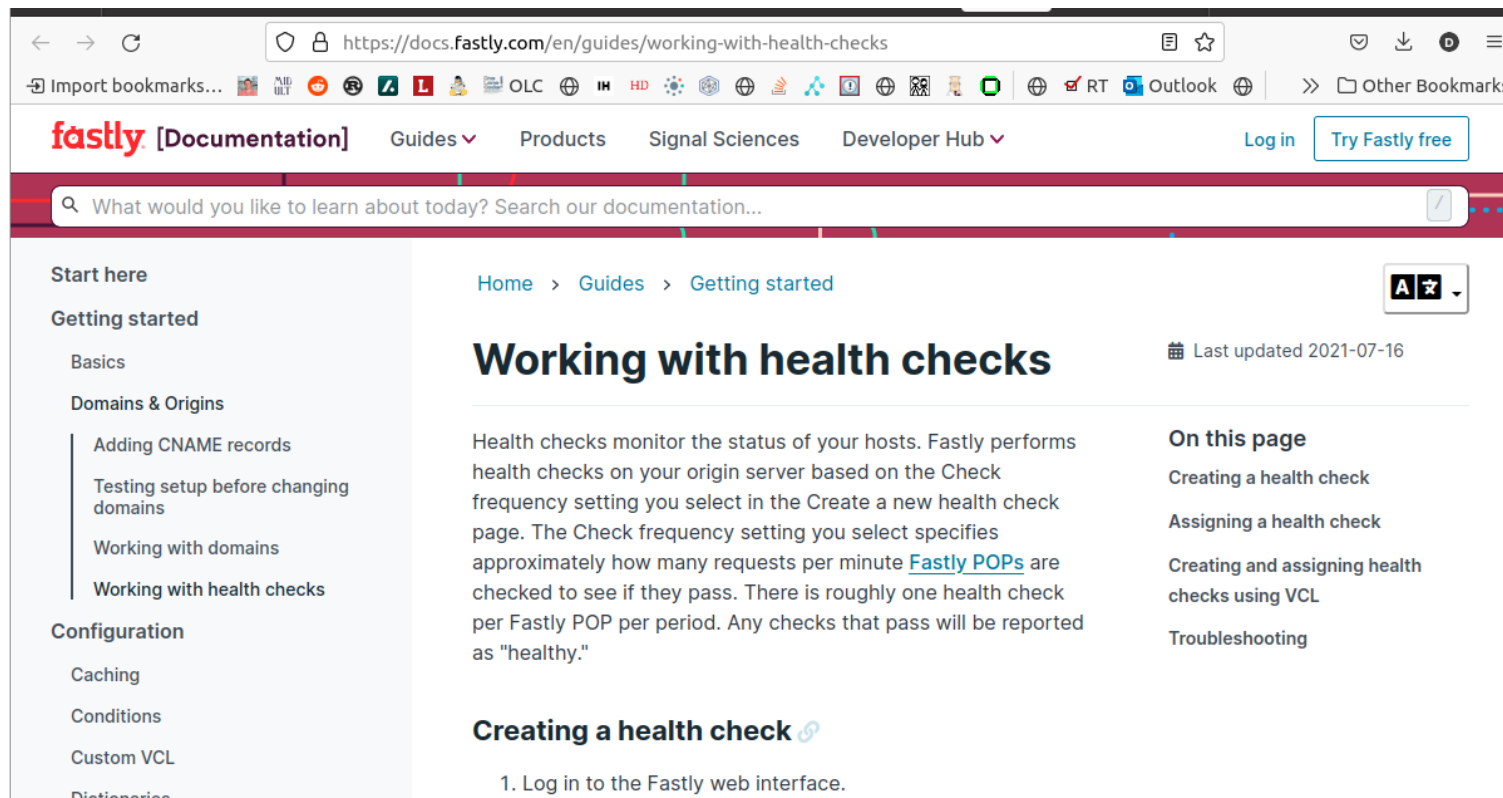


*“A **content delivery network**, or *content distribution network (CDN)*, is a geographically distributed network of proxy servers and their data centers. The goal is to provide high availability and performance by distributing the service spatially relative to end users.”*

–Wikipedia

- **Examples:** *Fastly, CloudFlare, etc.*

Content Delivery Networks (CDN)



The screenshot shows a web browser displaying the Fastly documentation page for "Working with health checks". The page is titled "Working with health checks" and is part of the "Getting started" guide. The main content explains that health checks monitor the status of hosts and that Fastly performs these checks based on the frequency setting selected in the "Create a new health check" page. It also mentions that Fastly POPs are checked to see if they pass and that checks that pass are reported as "healthy".

Working with health checks Last updated 2021-07-16

Health checks monitor the status of your hosts. Fastly performs health checks on your origin server based on the Check frequency setting you select in the Create a new health check page. The Check frequency setting you select specifies approximately how many requests per minute [Fastly POPs](#) are checked to see if they pass. There is roughly one health check per Fastly POP per period. Any checks that pass will be reported as "healthy."

Creating a health check [🔗](#)

1. Log in to the Fastly web interface.

On this page

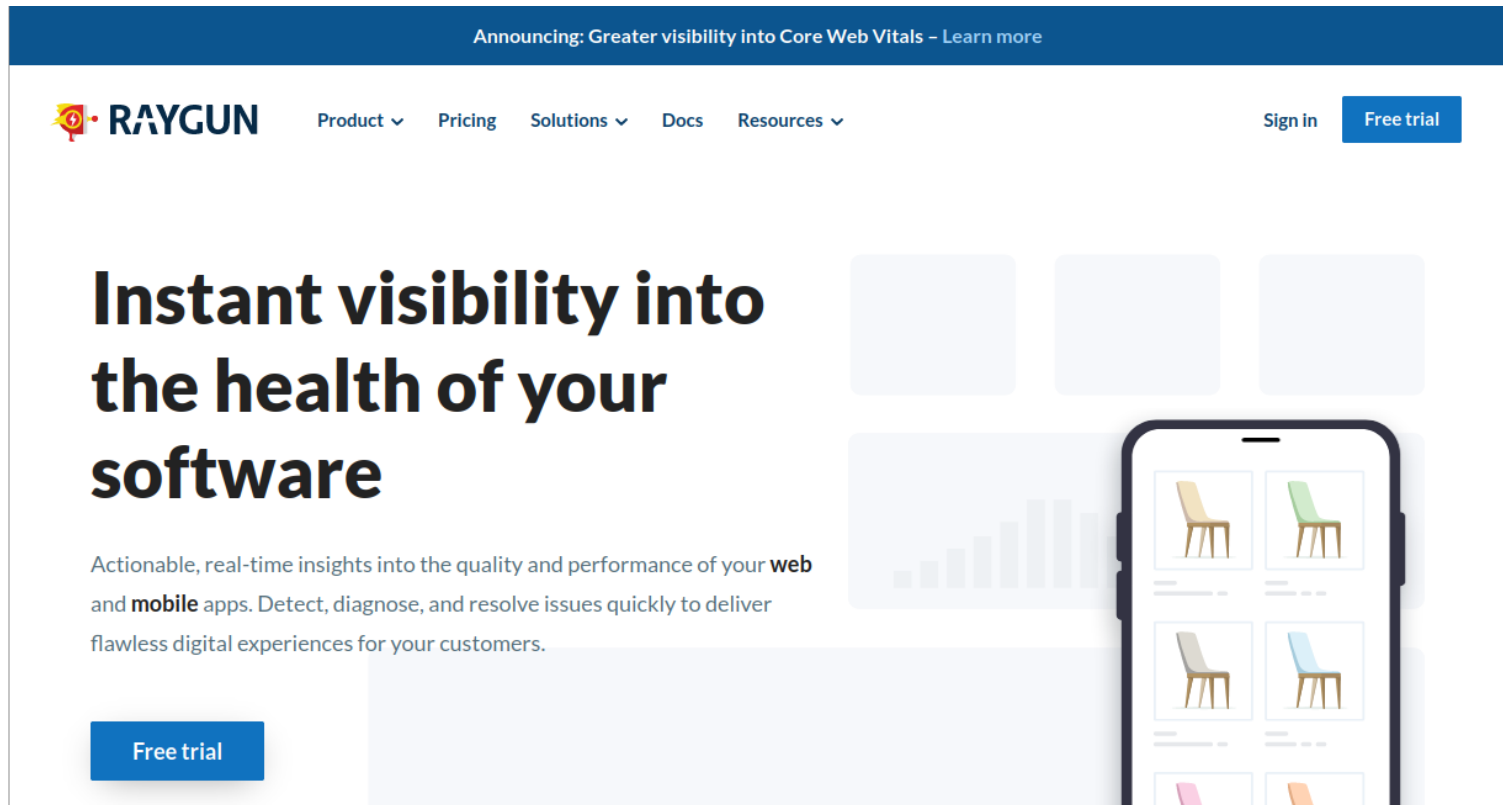
- Creating a health check
- Assigning a health check
- Creating and assigning health checks using VCL
- Troubleshooting

Start here

- Getting started
 - Basics
 - Domains & Origins
 - Adding CNAME records
 - Testing setup before changing domains
 - Working with domains
 - Working with health checks
- Configuration
 - Caching
 - Conditions
 - Custom VCL
 - Dictionaries

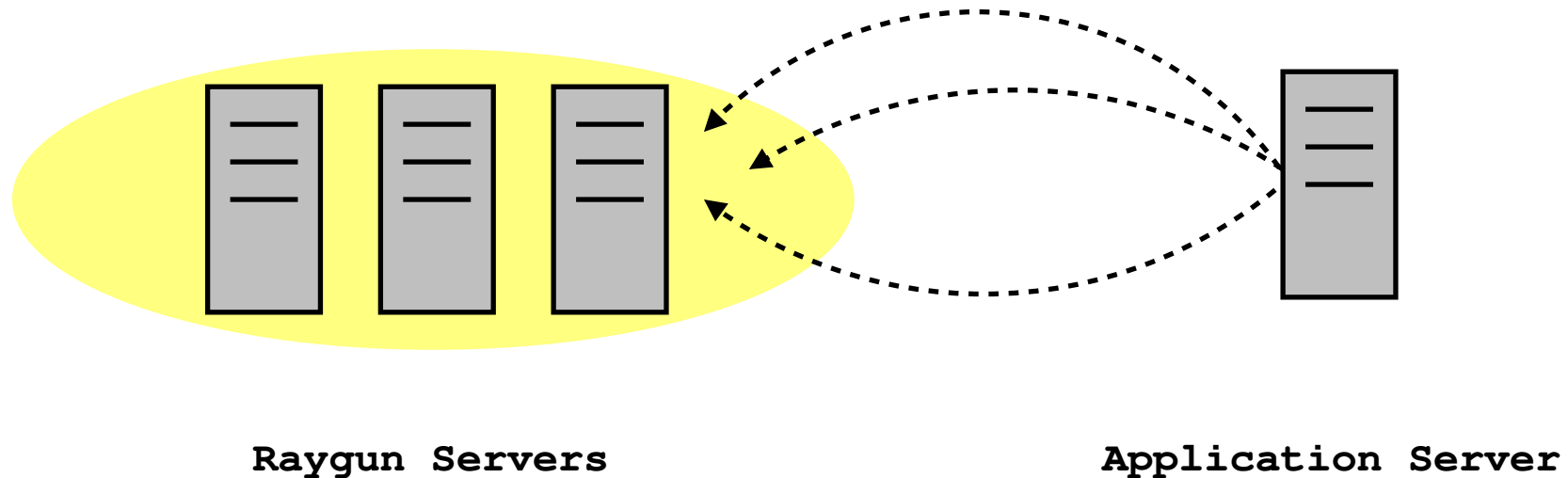
- CDN's are well positioned for **health checks!**

Case Study: Ray Gun



- **Wellington-based Company!**
- **Offers error monitoring & crash reporting tools**

Case Study: Ray Gun



- Code (e.g. Java) **embedded** within application
- Embedded code transmits exceptions in **real time**

Case Study: Ray Gun

```
RaygunClient client = factory.newClient();  
client.setUser(user);  
client.tag("blue").withData("userseg", "dev");  
client.recordBreadcrumb("hello_world");  
...  
client.send(anException);
```

- Straightforward client available for Java applications
- Developers add this to their code base