Google Cloud

Blog Menu

CONTAINERS & KUBERNETES

How Migrate for Anthos streamlines legacy Java app modernization

Adv Deganv

Find an article...

Latest stories

Products

Topics

About

RSS Feed

and usage information is incompatible with standard-issue Kubernetes, and requires some complicated workarounds.

To help with this, the most recent release of Migrate for Anthos has a new feature to help streamline and simplify legacy application migration, automatically augmenting container resource visibility for legacy Linux-based applications, such as those that use Oracle Java SE 7 and 8 (prior to update 191). This is crucial if you want to successfully convert your legacy Java applications into containers without having to upgrade or refactor them.

Migrate for Anthos helps you successfully move Java applications into containers by transparently and automatically implementing a userspace filesystem that addresses the limitations of the Linux filesystem. As you probably know, Linux uses cgroups to enforce container resource allocations. However, a known issue when running in Kubernetes, is that the Kubernetes node's procfs /proc file system is mounted by default in the container, and reflects host resources rather than those allocated to the container itself. And because some legacy applications still acquire resource configuration and usage information from files like meminfo and cpuinfo in the /proc directory, rather than from cgroups files, running those applications in a container can result in errors and instability. For example, older Java versions may use the information from meminfo and cpuinfo to determine how much memory to allocate to its JVM heap, how many threads to run in parallel for garbage collection (GC), etc. Running an older Java application in a container that hasn't been properly configured can result in processes being killed due to

Find an article...

Latest stories

Products

Topics

About

RSS Feed

For this test, we're using a JBOSS 8.2.1 server using an older version of Oracle Java SE 7 update 80. You can download this version from Oracle's Java SE 7 Archives. We package it in two ways: as a regular Docker container image, and as a server VM from which we have migrated the application to a container using Migrate for Anthos.

For the application, we use a sample JBoss node-info application with some additional lines of code to simulate memory pressure for each request served. The following modifications were applied:

src/main/java/pl/goldmann/work/helloworld/NodeInfoServlet.java

```
...
int MiB = 1024*1024;
PrintWriter writer = resp.getWriter();
writer.println("Hostname: " + System.getProperty("jboss.host.name"));
writer.println("OS: " + System.getProperty("os.name") + " " + System.getProp
writer.println("Java Runtime: " + System.getProperty("java.runtime.name") +
writer.println("Java sees: ");
writer.println("       Number of processors: " +  Runtime.getRuntime().avail
writer.println("       Max Memory: " + Runtime.getRuntime().maxMemory()/MiB
writer.println("--> grabbing 20 MiB of memory...");
try {
  byte b[] = new byte[20*MiB];
  writer.println("SUCCESS");
} catch (OutOfMemoryError error) {
  writer.println("FAILED: heap full!");
  resp.setStatus(202);
}
```

Find an article...

Latest stories

Products

Topics

About

RSS Feed

Here's what happens on the standard container:

```
Hostname: wildfly-deployment-5c5cd88676-lhzmq
OS: Linux amd64 5.3.0-1012-gke
Java Runtime: Java(TM) SE Runtime Environment 1.7.0_80-b15
Java sees:
        Number of processors: 4
        Max Memory: 3335 MiB
--> grabbing 20 MiB of memory...
SUCCESS
```

But here's what happens on the Migrate for Anthos migrated container:

```
Hostname: app-wildfly-test-878d69f4-dw5wr
OS: Linux amd64 5.3.0-1012-gke
Java Runtime: Java(TM) SE Runtime Environment 1.7.0_80-b15
Java sees:
        Number of processors: 1
        Max Memory: 247 MiB
--> grabbing 20 MiB of memory...
SUCCESS
```

You can immediately see a difference between the results. In the standard container, as already reported in many such tests, Java reports resource values from the host node,

Find an article...

Latest stories

Products

Topics

About

RSS Feed

Now let's see the impact of these differences under load. We generate application load using Hey. For example, the following command generates application load for two minutes, with a request concurrency of 50:

```
./hey_linux_amd64 -z 2m http://##.###.###.###:8080/node-info/
```

Here are the test results with the standard container:

```
Status code distribution:
  [200] 332 responses
  [404] 8343 responses
Error distribution:
  [29]  Get http://##.###.###.###:8080/node-info/: EOF
  [10116]       Get http://##.###.###.###:8080/node-info/: dial
tcp ##.###.###.###:8080: connect: connection refused
  [91]  Get http://##.###.###.###:8080/node-info/: net/http:
request canceled while waiting for connection (Client.Timeout
exceeded while awaiting headers
```

This is a clear indication that the service is not handling the load correctly, and indeed when inspecting the container logs, we see multiple occurrences of

```
*** JBossAS process (79) received KILL signal ***
```

Find an article...

Latest stories

Products

Topics

About

RSS Feed

We showed how Migrate for Anthos automatically augments a known container resource visibility issue in Kubernetes. This helps ensure that legacy applications that run on older Java versions behave correctly after being migrated, without having to manually tune or reconfigure them to fit dynamic constraints applied through the Kubernetes Pod specs. We also demonstrated how the legacy application remains stable and responsive under memory load, without experiencing errors or restarts.

With this feature, Migrate for Anthos can help you harness the benefits of containerization and container orchestration with Kubernetes, to modernize your operations and management of legacy applications. You'll be able to leverage the power of CI/CD with image-based management, non-disruptive rolling updates, and unified policy and application performance management across cloud native and legacy applications, without requiring access to source code or application rewrite.

For more information, see our original release blog that outlines support for day-two operations and more or fill out this form for more info (please mention 'Migrate for Anthos' in the comment box).

POSTED IN:
CONTAINERS & KUBERNETES—CLOUD MIGRATION—ANTHOS—
GOOGLE CLOUD PLATFORM

Find an article...

Latest stories

Products

Topics

About

RSS Feed