TRACING CEPH WITH BLKKIN

Ceph can use Blkin, a library created by Marios Kogias and others, which enables tracking a specific request from the time it enters the system at higher levels till it is finally served by RADOS.

In general, Blkin implements the Dapper tracing semantics in order to show the causal relationships between the different processing phases that an IO request may trigger. The goal is an end-to-end visualisation of the request's route in the system, accompanied by information concerning latencies in each processing phase. Thanks to LTTng this can happen with a minimal overhead and in realtime. The LTTng traces can then be visualized with Twitter's Zipkin.

INSTALLING BLKIN

You can install Markos Kogias' upstream Blkin by hand.:

```
cd blkin/
make && make install
```

or build distribution packages using DistroReadyBlkin, which also comes with pkgconfig support. If you choose the latter, then you must generate the configure and make files first.:

```
cd blkin
autoreconf -i
```

CONFIGURING CEPH WITH BLKIN

If you built and installed Blkin by hand, rather than building and installing packages, then set these variables before configuring Ceph.:

```
export BLKIN_CFLAGS=-Iblkin/
export BLKIN_LIBS=-lzipkin-cpp
```

Since there are separate lttng and blkin changes to Ceph, you may want to configure with something like:

```
./configure --with-blkin --without-lttng --with-debug
```

TESTING BLKIN

It's easy to test Ceph's Blkin tracing. Let's assume you don't have Ceph already running, and you compiled Ceph with Blkin support but you did't install it. Then launch Ceph with the `vstart.sh` script in Ceph's src directgory so you can see the possible tracepoints.:

```
cd src
OSD=3 MON=3 RGW=1 ./vstart.sh -n
lttng list --userspace
```

You'll see something like the following::

```
UST events:
-------------
PID: 8987 - Name: ./ceph-osd
      zipkin:timestamp (loglevel: TRACE_WARNING (4)) (type: tracepoint)
      zipkin:keyval (loglevel: TRACE_WARNING (4)) (type: tracepoint)
      ust_baddr_statedump:soinfo (loglevel: TRACE_DEBUG_LINE (13)) (type: tracepoint)

PID: 8407 - Name: ./ceph-mon
      zipkin:timestamp (loglevel: TRACE_WARNING (4)) (type: tracepoint)
```

```
      zipkin:keyval (loglevel: TRACE_WARNING (4)) (type: tracepoint)
      ust_baddr_statedump:soinfo (loglevel: TRACE_DEBUG_LINE (13)) (type: tracepoint)

...
```

Next, stop Ceph so that the tracepoints can be enabled.:

```
./stop.sh
```

Start up an LTTng session and enable the tracepoints.:

```
lttng create blkin-test
lttng enable-event --userspace zipkin:timestamp
lttng enable-event --userspace zipkin:keyval
lttng start
```

Then start up Ceph again.:

```
OSD=3 MON=3 RGW=1 ./vstart.sh -n
```

You may want to check that ceph is up.:

```
./ceph status
```

Now put something in usin rados, check that it made it, get it back, and remove it.:

```
./rados mkpool test-blkin
./rados put test-object-1 ./vstart.sh --pool=test-blkin
./rados -p test-blkin ls
./ceph osd map test-blkin test-object-1
./rados get test-object-1 ./vstart-copy.sh --pool=test-blkin
md5sum vstart*
./rados rm test-object-1 --pool=test-blkin
```

You could also use the example in `examples/librados/` or `rados bench`.

Then stop the LTTng session and see what was collected.:

```
lttng stop
lttng view
```

You'll see something like::

```
[13:09:07.755054973] (+?.?????????) scruffy zipkin:timestamp: { cpu_id = 5 }, { trace_name =
[13:09:07.755071569] (+0.000016596) scruffy zipkin:keyval: { cpu_id = 5 }, { trace_name = "Ma
[13:09:07.755074217] (+0.000002648) scruffy zipkin:keyval: { cpu_id = 5 }, { trace_name = "Ma
...
```

INSTALL ZIPKIN

One of the points of using Blkin is so that you can look at the traces using Zipkin. Users should run Zipkin as a tracepoints collector and also a web service, which means users need to run three services, zipkin-collector, zipkin-query and zipkin-web.

Download Zipkin Package:

```
wget https://github.com/twitter/zipkin/archive/1.1.0.tar.gz
tar zxf 1.1.0.tar.gz
cd zipkin-1.1.0
```

```
bin/collector cassandra &
bin/query cassandra &
bin/web &
```

Check Zipkin:

```
bin/test
Browse http://${zipkin-web-ip}:8080
```

SHOW CEPH'S BLKIN TRACES IN ZIPKIN-WEB

Blkin provides a script which translates lttng result to Zipkin (Dapper) semantics.

Send lttng data to Zipkin:

```
python3 babeltrace_zipkin.py ${lttng-traces-dir}/${blkin-test}/ust/uid/0/64-bit/ -p ${zipkin-
```

Example:

```
python3 babeltrace_zipkin.py ~/lttng-traces-dir/blkin-test-20150225-160222/ust/uid/0/64-bit/
```

Check Ceph traces on webpage:

```
Browse http://${zipkin-web-ip}:8080
Click "Find traces"
```