Ceph - Bug #682

higher pgp_bits setting leads to strange radostool errors

01/05/2011 04:47 PM - Colin McCabe

Status:	Resolved	**% Done:**	0%
Priority:	Normal	**Spent time:**	0.00 hour
Assignee:			
Category:			
Target version:	v0.25		
Source:		**Reviewed:**	
Tags:		**Affected Versions:**	
Backport:		**ceph-qa-suite:**	
Regression:	No	**Pull request ID:**	
Severity:	3 - minor	**Crash signature:**	

Description

git-bisect shows that 8f104243f823c3b5c827a4c7e59d637d38846e3f broke test_unfound.sh, test_lost.sh, and probably some other unit tests.

The test output looks like this:

```
+ for i in '`seq -w 1 $num_objs`'
 + ./rados -p data put obj01 /tmp/tmp.dSNCR8Px0o/ver1
 + for i in '`seq -w 1 $num_objs`'
 + ./rados -p data put obj02 /tmp/tmp.dSNCR8Px0o/ver1
 + for i in '`seq -w 1 $num_objs`'
 + ./rados -p data put obj03 /tmp/tmp.dSNCR8Px0o/ver1
 + for i in '`seq -w 1 $num_objs`'
 + ./rados -p data put obj04 /tmp/tmp.dSNCR8Px0o/ver1
 + for i in '`seq -w 1 $num_objs`'
 + ./rados -p data put obj05 /tmp/tmp.dSNCR8Px0o/ver1
 error writing data/obj05: No such device or address
 + die 'radostool failed'
 + echo radostool failed
 radostool failed
 + exit 1
 + cleanup
```

After the test runs, I can replace obj01, obj02, obj03, or obj04 with new objects, or read old objects, but any attempt to create a new object fails.

Some interesting snippets from osd.0's log:

```
2011-01-05 15:48:38.554165 7fcd23fff710 osd0 6 request for pool=1 (metadata) owner=0 perm=7 may_re
ad=0 may_write=1 may_exec=0 require_exec_caps=0
 2011-01-05 15:48:38.554175 7fcd23fff710 osd0 6 pg[1.3( empty n=0 ec=2 les=5 3/3/3) [1,0] r=1 acti
ve] misdirected op in 3
 2011-01-05 15:48:38.558943 7fcd23fff710 log [WRN] : mds0 10.3.14.10:6807/2947 misdirected mds0.1:
19 1.3 to osd0 not [1,0]
 2011-01-05 15:48:38.558981 7fcd23fff710 -- 10.3.14.10:6800/2857 --> 10.3.14.10:6807/2947 -- osd_o
p_reply(19 mds0_sessionmap [writefull 0~17] ack = ~~6 (No such device or address)) v1~~ ?+0 0x7fcd18
032050 con=0x29c8450
 2011-01-05 15:48:38.559053 7fcd237fe710 osd0 6 _dispatch 0x29d0f80 osd_sub_op(mds0.1:4 1.7 1.0000
0000/head [] v 6'3 snapset=0=[]:[] snapc=0=[]) v3
 2011-01-05 15:48:38.559088 7fcd23fff710 -- 10.3.14.10:6800/2857 <== mds0 10.3.14.10:6807/2947 9 =
=== osd_op(mds0.1:20 mds_anchortable  [writefull 0~29] 1.f6a7) v1 ==== 123+0+29 (918054842 0 21879
140
 5) 0x29cdfe0
```

There are a bunch of other places in the code where we see this from osd_op_reply.

Log file for osd.0 is attached.

History

#1 - 01/05/2011 05:01 PM - Colin McCabe

by the way, it should be 100% reproducible, just pull the latest unstable and run "./test/test_unfound.sh run"

#2 - 01/05/2011 10:10 PM - Sage Weil

try this?

```
diff --git a/src/osdc/Objecter.cc b/src/osdc/Objecter.cc
index 032377c..f96b7ab 100644
--- a/src/osdc/Objecter.cc
+++ b/src/osdc/Objecter.cc
@@ -549,7 +549,7 @@ bool Objecter::recalc_op_target(Op *op)
   pg_t pgid = op->pgid;
   if (op->oid.name.length())
     pgid = osdmap->object_locator_to_pg(op->oid, op->oloc);
-  osdmap->pg_to_acting_osds(pgid, acting);
+  osdmap->pg_to_acting_osds(osdmap->raw_pg_to_pg(pgid), acting);

   if (op->pgid != pgid || is_pg_changed(op->acting, acting)) {
     op->pgid = pgid;
@@ -577,7 +577,7 @@ bool Objecter::recalc_linger_op_target(LingerOp *linger_op)
  {
   vector<int> acting;
   pg_t pgid = osdmap->object_locator_to_pg(linger_op->oid, linger_op->oloc);
-  osdmap->pg_to_acting_osds(pgid, acting);
+  osdmap->pg_to_acting_osds(osdmap->raw_pg_to_pg(pgid), acting);

   if (pgid != linger_op->pgid || is_pg_changed(linger_op->acting, acting)) {
     linger_op->pgid = pgid;
```

#3 - 01/06/2011 10:35 AM - Colin McCabe

- Status changed from New to Resolved

Looks like we have a winner...

Submitted the patch as cfd87ceefb46358adaa1751975c8d3a6b063bdf9

C.