What could possibly go wrong?

What orb2db tries to do

- Read packet, append data to end of accumulating waveform
- Start new waveform every day

What can go wrong?

- Missing packet: gap
  - Fill gap
- Duplicate packet
  - Ignore packet
- Out of order packet
  - Reorder packets

More problems

- Meta data changes: calib, calper
- Packets are not contiguous in time
- Packet time drifts from computed time

What happens?

- Orb2db starts a new wfdisc record
  - Lots of database records slow everything
- Also starts a new mini seed record
  - Data compression may be expansion
  - Disk fills up

A solution: cdorb2db

- Reads packet
- Inserts data into waveform file wherever it belongs
  - Time is truncated to fixed tick
  - Overlaps and gaps are not detected
- Only one wfdisc record per day
- Data is uncompressed but has fixed size
  - Can be compressed later by db2msd
- Experience at BRTT is that these issues don't affect event location

copying saved data into ring buffer

- copy into ring buffer:
  - sun 1300 pkts/second
  - anfexport: 4300 pkts/second
  - mac-mini 6900 pkts/second
  - xserve: 7100 pkts/second
  - xserve, ssd drive: 8000 pkts/second
    - cp of same packet file: 10 second vs 27 seconds, e.g. 22 kpkts/second
-

copying from ring buffer: faster

- cdorb2db: first time: 4.8 kpkts/sec
  - second time: 21.5 kpkts/sec
- orb2db: first time: 15.3 kpkts/sec
  - second time: 10.75 kpkts/sec

practical example

- copy packets into orb with miniseed2orb
- copy packets from orb with orbmsd2days
- 17 Mbyte of miniseed
- 8 Mbyte orbserver

```
#!/bin/bash

cat <<EOF
Use miniseed2orb and orbmsd2days to transfer data
from one place to another.
EOF

ORB=:dq
DB=db/db
ORBBUF=/tmp/orbrt/
DATA=/opt/antelope/testdata/seed/XM_CACO_HHZ.msd
COPY=2005/005/XM_CACO_HHZ_.msd

rtmanage -lv <<EOF
orbserver -t -r -P $ORBBUF -s 4M -p $ORB orbserver
@2
orbmsd2days -S state/orbmsd2days -vv $ORB
@miniseed2orb  -vv $DATA $ORB
@is_idle -v $COPY
cmp $DATA $COPY
@msdd $DATA
@msdd $COPY
@2
@miniseed2db -v 2005 db
@rm -rf $ORBBUF
@EXIT
EOF
```

```
  0.013322 orbserver: Will throttle incoming streams if reap streams fall behind

  0.013738 orbserver: orbserver orbserver   Antelope Unreleased dev-64 Mac OS X 10.6.6
2011-03-19  0:10 <#>
  0.013759 orbserver:    f31af1aa84d9deef5c7f71adcafe92847cf0973c (+8 files changed) <#>
  0.014513 orbserver:     Sat Mar 19 13:12:31 2011 <#>

--> orbserver -t -r -P /tmp/orbrt/ -s 8M -p :dq orbserver <#>

  0.017743 orbserver: resetting ring buffer at open
  0.516849 orbserver: orb last initialized  3/19/2011 (078) 19:14:08.299
  0.516889 orbserver:      8.000 Mbytes packet buffer
  0.516903 orbserver:          0 of      22867 maximum packets
  0.516958 orbserver:   0 srcnames used of 10000 maximum
  2.332099 orbserver: 1 reaping clients(0 stalled), Lag is 0.179 => delay is 0.004
seconds, #pkts threshold is 22
  2.422653 orbserver: 1 reaping clients(0 stalled), Lag is 0.230 => delay is 0.006
seconds, #pkts threshold is 22
  2.615900 orbserver: 1 reaping clients(0 stalled), Lag is 0.052 => delay is 0.001
seconds, #pkts threshold is 22

 16.215799 orbserver: received signal #15=SIGTERM: Terminated -- Shutting down
 17.219643 orbserver: halted by signal #15
```

- What happens depends on details, but
- in case I'll outline now,
  - expected 4280 packets, got 4027
  - what can happen is reader gets stuck at trailing edge, starts missing packets
  - when it starts doing that, orb2db gets slower.

- Might think the lesson is:
  - Make your orbservers large!
- but for a real-time system, you want everything (including the ring buffers) to fit in memory at once.
- if that's not true, system may start swapping, speed may be reduced by factors of 100 or more