Manageability, availability and performance in Porcupine:
  a highly scalable, cluster-based mail service
Saito, Bershad, Levy
SOSP 1999

figure 3: is porcupine faster than a sendmail cluster? why?

does porcupine scale better than a sendmail cluster? why?

why 800 msgs/sec with 30 servers?

how much slower is replication? why?

what is the main bottleneck?
  cpu? disk? network?

could they have got same performance w/ slower cpu? disk? net?

figure 6: what is skew and why do they care?

why does performance for some systems go down with increased skew?

why does performance not go down for dynamic lines?

why doesn't their sophisticated load balance beat random?

why doesn't random have terrible performance due to no affinity?

why is spread > 1 helpful for porcupine? for static?

why does performance go down at 300 in Fig 10?

what happens to incoming mail in the middle of Figure 10?
  what about deletes of msgs in failed fragment?

what happens when users read mail in the middle of Figure 10?

how does porcupine find a user's mail after repair?

why isn't the system slower after 600 than before 300?
  we'd expect each user's mail to be more spread out...

what happens to incoming mail in the middle of Figure 11?
  what about deletes?

why the slow rise after 600 in Figure 11?