BayesBenchmark

All of the recent changes to the bayes storage backend where not done lightly. Most updates were run through various benchmarks to make sure they did not slow down the system.

I like to think that this benchmark represents the majority of operation most often occurring with the Bayes subsystem. As always, feedback is appreciated and if you can think of a way to improve the benchmark please speak up.

I would also like to publish BayesBenchmarkResults from various systems, for various setups and backends.

For this benchmark we rely on six email buckets that will be processed in each of the phases. Three of the buckets should be ham, three should be spam. The buckets should be in rough date order, with the oldest messages being in bucket one. In addition there will be two "forget" buckets that consist of half the messages from the first ham and spam buckets.

The benchmark code requires the bucket files to have specific names:

hambucket1.mbox	spambucket1.mbox
hamforget1.mbox	spamforget1.mbox
hambucket2.mbox	spambucket2.mbox
hambucket3.mbox	spambucket3.mbox

I suggest at least 1000 messages per bucket, for sure it should not be less than 200, and maybe even 300 depending on how much autolearning happens in phase 2. Obviously, the more messages you have the more accurate your results will be but your benchmarks will take longer. I tend to use 2000 messages per bucket (1000 in the forget buckets) and my tests take between 40 minutes to 2 hours to finish, depending on the backend and setup.

The benchmark works in several different phases. After each phase is run we might run an sa-learn --dump magic, check the size of the database (assuming we can) and dump the database for reference with sa-learn --backup. The benchmark is generally run with auto expire turned off.

Phase 1:
This is the learning phase, here we run sa-learn on hambucket1.mbox and spambucket1.mbox, getting the timings for each.

NOTE: sa-learn here will automatically sync Phase 2:
This is the spamd scanning phase. We startup a spamd and then startup a forking script that throws all messages in hambucket2.mbox and spambucket2.mbox at the daemon using spamc.

NOTE: If available, this phase will learn/update to the bayes_journal. Once all messages have been scanned a sa-learn --sync is performed to sync the journal file to the database.

Phase 3:
This phase performs an sa-learn --force-expire.

Phase 4:
This is the forget phase. We use sa-learn to forget all the messages in hamforget1.mbox and then do it again for spamforget1.mbox.

Phase 5:
This is the spamassassin scan phase. Here we scan the hambucket3.mbox and then the spambucket3.mbox using the spamassassin script.

I suggest running each benchmark 3 times to make sure your test is not influenced by other system activities.

Here is the code: benchmark.tar.gz It can also be found in the Subversion tree.

You'll need to generate your own ham/spam buckets and place them in the corpus directory and change the username/passwords to match those for your database.

Of course, I'm always glad to hear feedback and suggested changes to the benchmark. It really is something that I came up with on my own and it is possible that I've completely missed the boat. (MichaelParker)
