Hello User, Welcome to Database

Joe DeVivo works for Basho,
tweets at @joedevivo, and
put some relevant files at github.com/joedevivo/ricon

Go ahead, solve CAP. If someone downloads your amazing database and can’t configure it, then it doesn’t matter.

If people can’t set your product up, they secretly hate you

I found an amazing database back in 2010.

It wasn’t too hard to configure back then

turns out, that was the only part that didn’t scale

```
%% -*- tab-width: 4;erlang-indent-level: 4;indent-tabs-mode: nil -*-
%% ex: ts=4 sw=4 et
[
 %% Riak Core config
 {riak_core, [
    %% Default location of ringstate
    {ring_state_dir, "data/ring"},

    %% riak_web_ip is the IP address that Riak's HTTP interface will
    %%  bind to.  If this is undefined, the HTTP interface will not run.
    {web_ip, "127.0.0.1"},

    %% riak_web_port is the TCP port that Riak's HTTP interface will
    %% bind to.
     {web_port, 8098}
    ]},

 %% Riak KV config
 {riak_kv, [
    %% Storage_backend specifies the Erlang module defining the storage
    %% mechanism that will be used on this node.
    {storage_backend, riak_kv_dets_backend},

    %% Different storage backends can use other configuration variables.
    %%  For instance, riak_dets_backend_root determines the directory
    %%  under which dets files will be placed.
    {riak_kv_dets_backend_root, "data/dets"},
%% -*- tab-width: 4;erlang-indent-level: 4;indent-tabs-mode: nil -*-
%% ex: ts=4 sw=4 et
[
 %% Riak Core config
 {riak_core, [
              %% Default location of ringstate
              {ring_state_dir, "data/ring"},

              %% riak_web_ip is the IP address that Riak's HTTP interface will
              %%  bind to.  If this is undefined, the HTTP interface will not run.
              {web_ip, "127.0.0.1"},

              %% riak_web_port is the TCP port that Riak's HTTP interface will
              %% bind to.
              {web_port, 8098}
             ]},

 %% Riak KV config
 {riak_kv, [
            %% Storage_backend specifies the Erlang module defining the storage
            %% mechanism that will be used on this node.
            {storage_backend, riak_kv_dets_backend},

            %% Different storage backends can use other configuration variables.
            %%  For instance, riak_dets_backend_root determines the directory
            %%  under which dets files will be placed.
            {riak_kv_dets_backend_root, "data/dets"},

            %% riak_handoff_port is the TCP port that Riak uses for
            %% intra-cluster data handoff.
            {handoff_port, 8099},

            %% pb_ip is the IP address that Riak's Protocol Buffers interface
            %% will bid to.  If this is undefined, the interface will not run.
            {pb_ip,   "0.0.0.0"},

            %% pb_port is the TCP port that Riak's Protocol Buffers interface
            %% will bind to
            {pb_port, 8087},

            %% raw_name is the first part of all URLS used by Riak's raw HTTP
            %% interface.  See riak_web.erl and raw_http_resource.erl for
            %% details.
            %{raw_name, "riak"},

            %% mapred_name is URL used to submit map/reduce requests to Riak.
            {mapred_name, "mapred"},

            %% js_vm_count is the number of Javascript VMs to start per Riak
            %% node.  8 is a good default for smaller installations. A larger
            %% number like 12 or 16 is appropriate for installations handling
            %% lots of map/reduce processing.
            {js_vm_count, 8},

            %% js_source_dir should point to a directory containing Javascript
            %% source files which will be loaded by Riak when it initializes
            %% Javascript VMs.
            %{js_source_dir, "/tmp/js_source"}

            %% riak_stat enables the use of the "riak-admin status" command to
            %% retrieve information the Riak node for performance and debugging needs
            {riak_kv_stat, true}
           ]},

 %% SASL config
 {sasl, [
         {sasl_error_logger, {file, "log/sasl-error.log"}},
         {errlog_type, error},
         {error_logger_mf_dir, "log/sasl"},      % Log directory
         {error_logger_mf_maxbytes, 10485760},   % 10 MB max file size
         {error_logger_mf_maxfiles, 5}           % 5 files max
         ]}
].
```

```
## Name of the riak node
-name dev1@127.0.0.1

## Cookie for distributed erlang
-setcookie riak

## Heartbeat management; auto-restarts VM if it dies or becomes unresponsive
## (Disabled by default..use with caution!)
##-heart

## Enable kernel poll and a few async threads
+K true
+A 5

## Increase number of concurrent ports/sockets
-env ERL_MAX_PORTS 4096

## Tweak GC to run more often
-env ERL_FULLSWEEP_AFTER 10
```

Two configuration files with different syntaxes!
That’s sure to win us database of the year!

#sarcasm

Guess how many times I broke this?

A lot

Forget a comma?

Your configuration broke

Didn't close that bracket?

Too bad

Not an Erlang developer?

Well, you should be

Erlang

I’ve got hard distributed
databass problems to solve

~~Erlang~~

LOL, Syntax

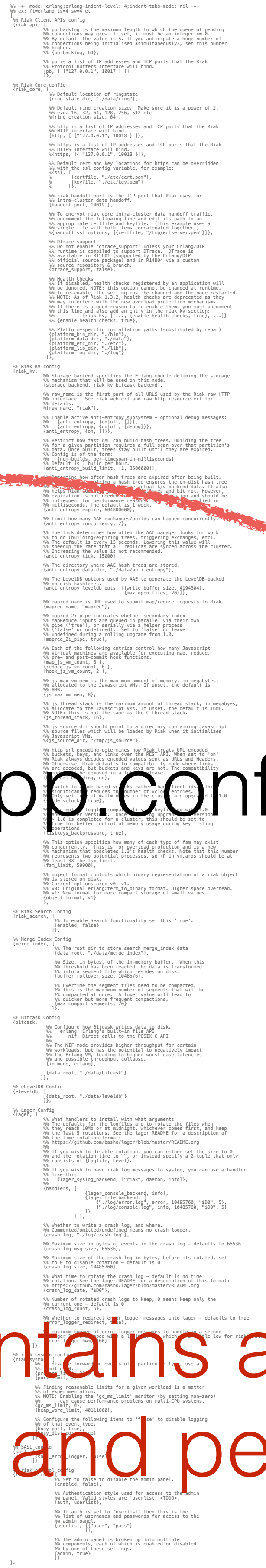
Erlang developers write Erlang friendly configuration settings

- `{on, []}`
- `{off, []}`
- `{on, [debug]}`

one setting, lots of lines

```
%% cool syntax, bro
[{riak_kv, [
    {anti_entropy,
      {on, [debug]}}
    ]}
].
```

actual number!



```
%% -*- mode: erlang;erlang-indent-level: 4;indent-tabs-mode: nil -*-
%% ex: ft=erlang ts=4 sw=4 et
[
 %% Riak Client APIs config
 {riak_api, [
            %% pb_backlog is the maximum length to which the queue of pending
            %% connections may grow. If set, it must be an integer >= 0.
            %% By default the value is 5. If you anticipate a huge number of
            %% connections being initialised *simultaneously*, set this number
            %% higher.
            %% {pb_backlog, 64},

            %% pb is a list of IP addresses and TCP ports that the Riak
            %% Protocol Buffers interface will bind.
            {pb, [ {"127.0.0.1", 10017 } ]}
            ]},

 %% Riak Core config
 {riak_core, [
              %% Default location of ringstate
              {ring_state_dir, "./data/ring"},

              %% Default ring creation size.  Make sure it is a power of 2,
              %% e.g. 16, 32, 64, 128, 256, 512 etc
              %{ring_creation_size, 64},

              %% http is a list of IP addresses and TCP ports that the Riak
              %% HTTP interface will bind.
              {http, [ {"127.0.0.1", 10018 } ]},

              %% https is a list of IP addresses and TCP ports that the Riak
              %% HTTPS interface will bind.
              %{https, [{ "127.0.0.1", 10018 }]},

              %% Default cert and key locations for https can be overridden
              %% with the ssl config variable, for example:
              %{ssl, [
              %       {certfile, "./etc/cert.pem"},
              %       {keyfile, "./etc/key.pem"}
              %      ]},

              %% riak_handoff_port is the TCP port that Riak uses for
              %% intra-cluster data handoff.
              {handoff_port, 10019 },

              %% To encrypt riak_core intra-cluster data handoff traffic,
              %% uncomment the following line and edit its path to an
              %% appropriate certfile and keyfile.  (This example uses a
              %% single file with both items concatenated together.)
              %{handoff_ssl_options, [{certfile, "/tmp/erlserver.pem"}]},

              %% DTrace support
              %% Do not enable 'dtrace_support' unless your Erlang/OTP
              %% runtime is compiled to support DTrace.  DTrace is
              %% available in R15B01 (supported by the Erlang/OTP
              %% official source package) and in R14B04 via a custom
              %% source repository & branch.
              {dtrace_support, false},

              %% Health Checks
              %% If disabled, health checks registered by an application will
              %% be ignored. NOTE: this option cannot be changed at runtime.
              %% To re-enable, the setting must be changed and the node restarted.
              %% NOTE: As of Riak 1.3.2, health checks are deprecated as they
              %% may interfere with the new overload protection mechanisms.
              %% If there is a good reason to re-enable them, you must uncomment
              %% this line and also add an entry in the riak_kv section:
              %%          {riak_kv, [ ..., {enable_health_checks, true}, ...]}
              %% {enable_health_checks, true},

              %% Platform-specific installation paths (substituted by rebar)
              {platform_bin_dir, "./bin"},
              {platform_data_dir, "./data"},
              {platform_etc_dir, "./etc"},
              {platform_lib_dir, "./lib"},
              {platform_log_dir, "./log"}
             ]},

 %% Riak KV config
 {riak_kv, [
            %% Storage_backend specifies the Erlang module defining the storage
            %% mechanism that will be used on this node.
            {storage_backend, riak_kv_bitcask_backend},

            %% raw_name is the first part of all URLS used by the Riak raw HTTP
            %% interface.  See riak_web.erl and raw_http_resource.erl for
            %% details.
            %{raw_name, "riak"},

            %% Enable active anti-entropy subsystem + optional debug messages:
            %%   {anti_entropy, {on|off, []}},
            %%   {anti_entropy, {on|off, [debug]}},
            {anti_entropy, {on, []}},

            %% Restrict how fast AAE can build hash trees. Building the tree
            %% for a given partition requires a full scan over that partition's
            %% data. Once built, trees stay built until they are expired.
            %% Config is of the form:
            %%   {num-builds, per-timespan-in-milliseconds}
            %% Default is 1 build per hour.
            {anti_entropy_build_limit, {1, 3600000}},

            %% Determine how often hash trees are expired after being built.
            %% Periodically expiring a hash tree ensures the on-disk hash tree
            %% data stays consistent with the actual k/v backend data. It also
            %% helps Riak identify silent disk failures and bit rot. However,
            %% expiration is not needed for normal AAE operation and should be
            %% infrequent for performance reasons. The time is specified in
            %% milliseconds. The default is 1 week.
            {anti_entropy_expire, 604800000},

            %% Limit how many AAE exchanges/builds can happen concurrently.
            {anti_entropy_concurrency, 2},

            %% The tick determines how often the AAE manager looks for work
            %% to do (building/expiring trees, triggering exchanges, etc).
            %% The default is every 15 seconds. Lowering this value will
            %% speedup the rate that all replicas are synced across the cluster.
            %% Increasing the value is not recommended.
            {anti_entropy_tick, 15000},

            %% The directory where AAE hash trees are stored.
            {anti_entropy_data_dir, "./data/anti_entropy"},

            %% The LevelDB options used by AAE to generate the LevelDB-backed
            %% on-disk hashtrees.
            {anti_entropy_leveldb_opts, [{write_buffer_size, 4194304},
                                         {max_open_files, 20}]},

            %% mapred_name is URL used to submit map/reduce requests to Riak.
            {mapred_name, "mapred"},

            %% mapred_2i_pipe indicates whether secondary-index
            %% MapReduce inputs are queued in parallel via their own
            %% pipe ('true'), or serially via a helper process
            %% ('false' or undefined).  Set to 'false' or leave
            %% undefined during a rolling upgrade from 1.0.
            {mapred_2i_pipe, true},

            %% Each of the following entries control how many Javascript
            %% virtual machines are available for executing map, reduce,
            %% pre- and post-commit hook functions.
            {map_js_vm_count, 8 },
            {reduce_js_vm_count, 6 },
            {hook_js_vm_count, 2 },

            %% js_max_vm_mem is the maximum amount of memory, in megabytes,
            %% allocated to the Javascript VMs. If unset, the default is
            %% 8MB.
            {js_max_vm_mem, 8},

            %% js_thread_stack is the maximum amount of thread stack, in megabyes,
            %% allocate to the Javascript VMs. If unset, the default is 16MB.
            %% NOTE: This is not the same as the C thread stack.
            {js_thread_stack, 16},

            %% js_source_dir should point to a directory containing Javascript
            %% source files which will be loaded by Riak when it initializes
            %% Javascript VMs.
            %{js_source_dir, "/tmp/js_source"},

            %% http_url_encoding determines how Riak treats URL encoded
            %% buckets, keys, and links over the REST API. When set to 'on'
            %% Riak always decodes encoded values sent as URLs and Headers.
            %% Otherwise, Riak defaults to compatibility mode where links
            %% are decoded, but buckets and keys are not. The compatibility
            %% mode will be removed in a future release.
            {http_url_encoding, on},

            %% Switch to vnode-based vclocks rather than client ids.  This
            %% significantly reduces the number of vclock entries.
            %% Only set true if *all* nodes in the cluster are upgraded to 1.0
            {vnode_vclocks, true},

            %% This option toggles compatibility of keylisting with 1.0
            %% and earlier versions.  Once a rolling upgrade to a version
            %% > 1.0 is completed for a cluster, this should be set to
            %% true for better control of memory usage during key listing
            %% operations
            {listkeys_backpressure, true},

            %% This option specifies how many of each type of fsm may exist
            %% concurrently.  This is for overload protection and is a new
            %% mechanism that obsoletes 1.3's health checks. Note that this number
            %% represents two potential processes, so +P in vm.args should be at
            %% least 3X the fsm_limit.
            {fsm_limit, 50000},

            %% object_format controls which binary representation of a riak_object
            %% is stored on disk.
            %% Current options are: v0, v1.
            %% v0: Original erlang:term_to_binary format. Higher space overhead.
            %% v1: New format for more compact storage of small values.
            {object_format, v1}
           ]},

 %% Riak Search Config
 {riak_search, [
                %% To enable Search functionality set this 'true'.
                {enabled, false}
               ]},

 %% Merge Index Config
 {merge_index, [
                %% The root dir to store search merge_index data
                {data_root, "./data/merge_index"},

                %% Size, in bytes, of the in-memory buffer.  When this
                %% threshold has been reached the data is transformed
                %% into a segment file which resides on disk.
                {buffer_rollover_size, 1048576},

                %% Overtime the segment files need to be compacted.
                %% This is the maximum number of segments that will be
                %% compacted at once.  A lower value will lead to
                %% quicker but more frequent compactions.
                {max_compact_segments, 20}
               ]},

 %% Bitcask Config
 {bitcask, [
             %% Configure how Bitcask writes data to disk.
             %%   erlang: Erlang's built-in file API
             %%      nif: Direct calls to the POSIX C API
             %%
             %% The NIF mode provides higher throughput for certain
             %% workloads, but has the potential to negatively impact
             %% the Erlang VM, leading to higher worst-case latencies
             %% and possible throughput collapse.
             {io_mode, erlang},

             {data_root, "./data/bitcask"}
           ]},

 %% eLevelDB Config
 {eleveldb, [
             {data_root, "./data/leveldb"}
            ]},

 %% Lager Config
 {lager, [
            %% What handlers to install with what arguments
            %% The defaults for the logfiles are to rotate the files when
            %% they reach 10Mb or at midnight, whichever comes first, and keep
            %% the last 5 rotations. See the lager README for a description of
            %% the time rotation format:
            %% https://github.com/basho/lager/blob/master/README.org
            %%
            %% If you wish to disable rotation, you can either set the size to 0
            %% and the rotation time to "", or instead specify a 2-tuple that only
            %% consists of {Logfile, Level}.
            %%
            %% If you wish to have riak log messages to syslog, you can use a handler
            %% like this:
            %%   {lager_syslog_backend, ["riak", daemon, info]},
            %%
            {handlers, [
                           {lager_console_backend, info},
                           {lager_file_backend, [
                               {"./log/error.log", error, 10485760, "$D0", 5},
                               {"./log/console.log", info, 10485760, "$D0", 5}
                           ]}
                       ] },

            %% Whether to write a crash log, and where.
            %% Commented/omitted/undefined means no crash logger.
            {crash_log, "./log/crash.log"},

            %% Maximum size in bytes of events in the crash log - defaults to 65536
            {crash_log_msg_size, 65536},

            %% Maximum size of the crash log in bytes, before its rotated, set
            %% to 0 to disable rotation - default is 0
            {crash_log_size, 10485760},

            %% What time to rotate the crash log - default is no time
            %% rotation. See the lager README for a description of this format:
            %% https://github.com/basho/lager/blob/master/README.org
            {crash_log_date, "$D0"},

            %% Number of rotated crash logs to keep, 0 means keep only the
            %% current one - default is 0
            {crash_log_count, 5},

            %% Whether to redirect error_logger messages into lager - defaults to true
            {error_logger_redirect, true},

            %% maximum number of error_logger messages to handle in a second
            %% lager 2.0.0 shipped with a limit of 50, which is a little low for riak's startup
            {error_logger_hwm, 100}
        ]},

 %% riak_sysmon config
 {riak_sysmon, [
         %% To disable forwarding events of a particular type, use a
         %% limit of 0.
         {process_limit, 30},
         {port_limit, 2},

         %% Finding reasonable limits for a given workload is a matter
         %% of experimentation.
         %% NOTE: Enabling the 'gc_ms_limit' monitor (by setting non-zero)
         %%       can cause performance problems on multi-CPU systems.
         {gc_ms_limit, 0},
         {heap_word_limit, 40111000},

         %% Configure the following items to 'false' to disable logging
         %% of that event type.
         {busy_port, true},
         {busy_dist_port, true}
        ]},

 %% SASL config
 {sasl, [
         {sasl_error_logger, false}
        ]},

 %% riak_control config
 {riak_control, [
                %% Set to false to disable the admin panel.
                {enabled, false},

                %% Authentication style used for access to the admin
                %% panel. Valid styles are 'userlist' <TODO>.
                {auth, userlist},

                %% If auth is set to 'userlist' then this is the
                %% list of usernames and passwords for access to the
                %% admin panel.
                {userlist, [{"user", "pass"}
                           ]},

                %% The admin panel is broken up into multiple
                %% components, each of which is enabled or disabled
                %% by one of these settings.
                {admin, true}
                ]}
].
```

app.config

hyperbole!

warning: contains about a billion []'s, {}'s commas and periods to misplace

NOTHING!

Everything you need to know about a single configuration setting should be on a single line

That line can be anywhere in the file

If you set something more than once, the last one wins

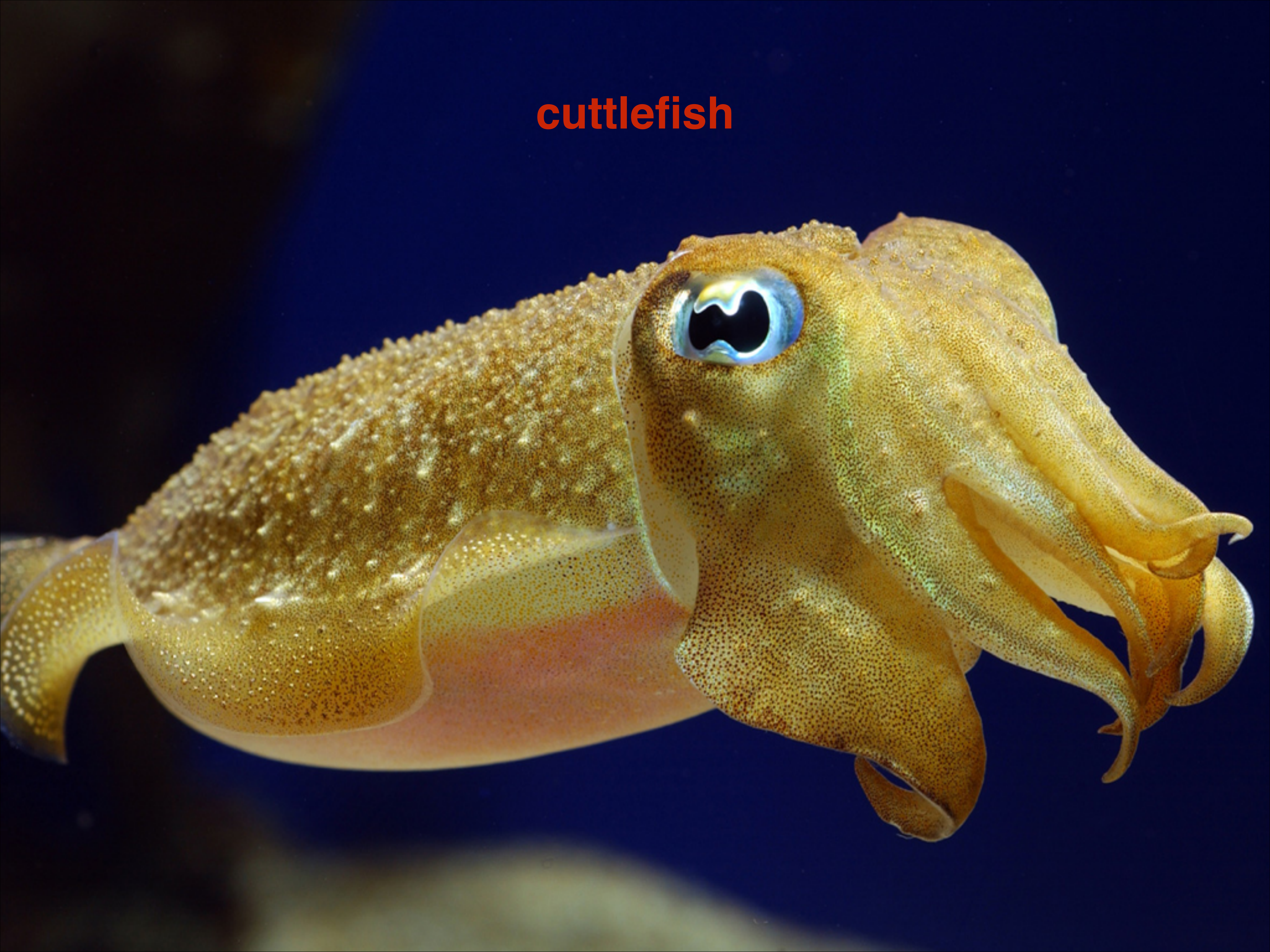
Shoutout to
Last Write Wins!

Key/Value is a pretty good idea

What did I want to give the user?

```
## enable active anti-entropy subsystem
## possible values: on, off, debug
anti_entropy = on
```

cuttlefish



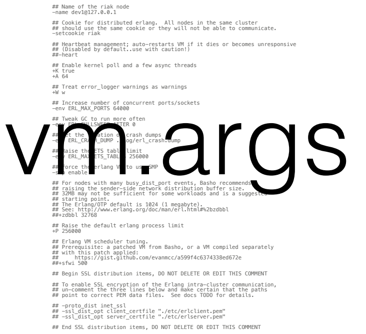
```
## Restrict how fast AAE can build hash trees. Building the tree
## for a given partition requires a full scan over that partition's
## data. Once built, trees stay built until they are expired.
## Config is of the form:
## {num-builds, per-timespan}
## Default is 1 build per hour.
anti_entropy.build_limit.number = 1

anti_entropy.build_limit.per_timespan = 1h

## Determine how often hash trees are expired after being built.
## Periodically expiring a hash tree ensures the on-disk hash tree
## data stays consistent with the actual k/v backend data. It also
## helps Riak identify silent disk failures and bit rot. However,
## expiration is not needed for normal AAE operation and should be
## infrequent for performance reasons. The time is specified in
## milliseconds. The default is 1 week.
anti_entropy.expire = 1w

## Limit how many AAE exchanges/builds can happen concurrently.
anti_entropy.concurrency = 2

## The tick determines how often the AAE manager looks for work
## to do (building/expiring trees, triggering exchanges, etc).
## The default is every 15 seconds. Lowering this value will
## speedup the rate that all replicas are synced across the cluster.
## Increasing the value is not recommended.
anti_entropy.tick = 15s
```

```
➜  rel git:(develop) ✗ cat riak.conf | grep anti_entropy
anti_entropy = on
anti_entropy.build_limit.number = 1
anti_entropy.build_limit.per_timespan = 1h
anti_entropy.expire = 1w
anti_entropy.concurrency = 2
anti_entropy.tick = 15s
anti_entropy.data_dir = ./data/anti_entropy
anti_entropy.write_buffer_size = 4MB
anti_entropy.max_open_files = 20
```

```
## Default ring creation size.  Make sure it is a power of 2,
## e.g. 16, 32, 64, 128, 256, 512 etc
ring_size = 42
```

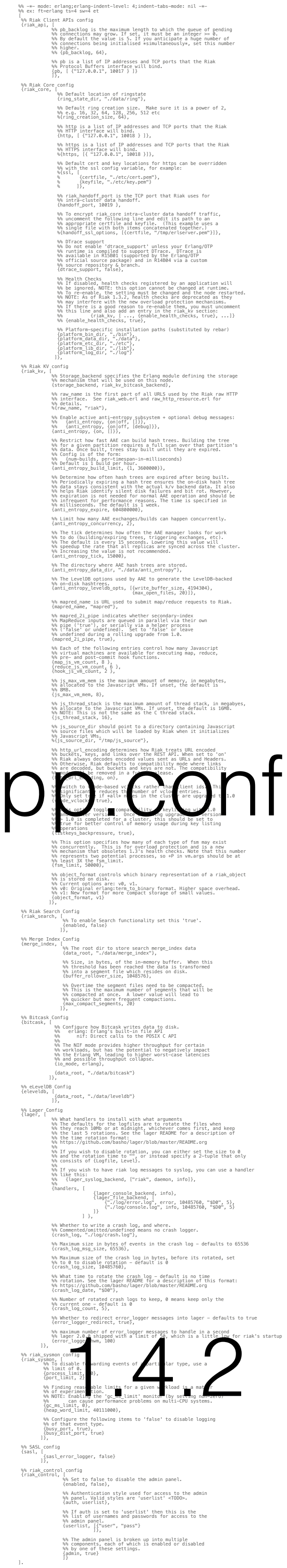
```
➜  rel git:(develop) ✗ ./riak/bin/riak console
10:11:27.284 [info] Application lager started on node nonode@nohost
10:11:27.284 [info] Checking /Users/joe/dev/basho/riak/rel/riak/bin/../etc/
app.config exists... false
10:11:27.284 [info] Checking /Users/joe/dev/basho/riak/rel/riak/bin/../etc/
vm.args exists... false
10:11:27.284 [info] No app.config or vm.args detected in /Users/joe/dev/basho/
riak/rel/riak/bin/../etc, activating cuttlefish
10:11:27.717 [info] Adding Defaults
10:11:27.723 [info] Applying Datatypes
10:11:27.751 [info] Validation
10:11:27.751 [error] ring_size invalid, not a power of 2 greater than 1
10:11:27.751 [error] Some validator failed, aborting
```

vm.args



```
## Name of the riak node
-name dev1@127.0.0.1

## Cookie for distributed erlang.  All nodes in the same cluster
## should use the same cookie or they will not be able to communicate.
-setcookie riak

## Heartbeat management; auto-restarts VM if it dies or becomes unresponsive
## (Disabled by default..use with caution!)
##-heart

## Enable kernel poll and a few async threads
+K true
+A 64

## Treat error_logger warnings as warnings
+W w

## Increase number of concurrent ports/sockets
-env ERL_MAX_PORTS 64000

## Tweak GC to run more often
-env ERL_FULLSWEEP_AFTER 0

## Set the location of crash dumps
-env ERL_CRASH_DUMP ./log/erl_crash.dump

## Raise the ETS table limit
-env ERL_MAX_ETS_TABLES 256000

## Force the erlang VM to use SMP
-smp enable

## For nodes with many busy_dist_port events, Basho recommends
## raising the sender-side network distribution buffer size.
## 32MB may not be sufficient for some workloads and is a suggested
## starting point.
## The Erlang/OTP default is 1024 (1 megabyte).
## See: http://www.erlang.org/doc/man/erl.html#%2bzdbbl
##+zdbbl 32768

## Raise the default erlang process limit
+P 256000

## Erlang VM scheduler tuning.
## Prerequisite: a patched VM from Basho, or a VM compiled separately
## with this patch applied:
##     https://gist.github.com/evanmcc/a599f4c6374338ed672e
##+sfwi 500

## Begin SSL distribution items, DO NOT DELETE OR EDIT THIS COMMENT

## To enable SSL encryption of the Erlang intra-cluster communication,
## un-comment the three lines below and make certain that the paths
## point to correct PEM data files.  See docs TODO for details.

## -proto_dist inet_ssl
## -ssl_dist_opt client_certfile "./etc/erlclient.pem"
## -ssl_dist_opt server_certfile "./etc/erlserver.pem"

## End SSL distribution items, DO NOT DELETE OR EDIT THIS COMMENT
```

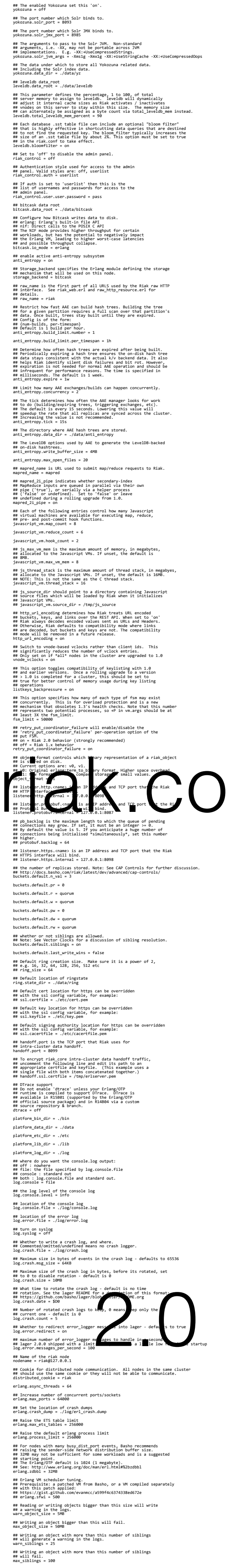
app.config



```
%% -*- mode: erlang;erlang-indent-level: 4;indent-tabs-mode: nil -*-
%% ex: ft=erlang ts=4 sw=4 et
[
 %% Riak Client APIs config
 {riak_api, [
            %% pb_backlog is the maximum length to which the queue of pending
            %% connections may grow. If set, it must be an integer >= 0.
            %% By default the value is 5. If you anticipate a huge number of
            %% connections being initialised *simultaneously*, set this number
            %% higher.
            %% {pb_backlog, 64},

            %% pb is a list of IP addresses and TCP ports that the Riak
            %% Protocol Buffers interface will bind.
            {pb, [ {"127.0.0.1", 10017 } ]}
            ]},

 %% Riak Core config
 {riak_core, [
              %% Default location of ringstate
              {ring_state_dir, "./data/ring"},

              %% Default ring creation size.  Make sure it is a power of 2,
              %% e.g. 16, 32, 64, 128, 256, 512 etc
              %{ring_creation_size, 64},

              %% http is a list of IP addresses and TCP ports that the Riak
              %% HTTP interface will bind.
              {http, [ {"127.0.0.1", 10018 } ]},

              %% https is a list of IP addresses and TCP ports that the Riak
              %% HTTPS interface will bind.
              %{https, [{ "127.0.0.1", 10018 }]},

              %% Default cert and key locations for https can be overridden
              %% with the ssl config variable, for example:
              %{ssl, [
              %       {certfile, "./etc/cert.pem"},
              %       {keyfile, "./etc/key.pem"}
              %      ]},

              %% riak_handoff_port is the TCP port that Riak uses for
              %% intra-cluster data handoff.
              {handoff_port, 10019 },

              %% To encrypt riak_core intra-cluster data handoff traffic,
              %% uncomment the following line and edit its path to an
              %% appropriate certfile and keyfile.  (This example uses a
              %% single file with both items concatenated together.)
              %{handoff_ssl_options, [{certfile, "/tmp/erlserver.pem"}]},

              %% DTrace support
              %% Do not enable 'dtrace_support' unless your Erlang/OTP
              %% runtime is compiled to support DTrace.  DTrace is
              %% available in R15B01 (supported by the Erlang/OTP
              %% official source package) and in R14B04 via a custom
              %% source repository & branch.
              {dtrace_support, false},

              %% Health Checks
              %% If disabled, health checks registered by an application will
              %% be ignored. NOTE: this option cannot be changed at runtime.
              %% To re-enable, the setting must be changed and the node restarted.
              %% NOTE: As of Riak 1.3.2, health checks are deprecated as they
              %% may interfere with the new overload protection mechanisms.
              %% If there is a good reason to re-enable them, you must uncomment
              %% this line and also add an entry in the riak_kv section:
              %%          {riak_kv, [ ..., {enable_health_checks, true}, ...]}
              %% {enable_health_checks, true},

              %% Platform-specific installation paths (substituted by rebar)
              {platform_bin_dir, "./bin"},
              {platform_data_dir, "./data"},
              {platform_etc_dir, "./etc"},
              {platform_lib_dir, "./lib"},
              {platform_log_dir, "./log"}
             ]},

 %% Riak KV config
 {riak_kv, [
            %% Storage_backend specifies the Erlang module defining the storage
            %% mechanism that will be used on this node.
            {storage_backend, riak_kv_bitcask_backend},

            %% raw_name is the first part of all URLS used by the Riak raw HTTP
            %% interface.  See riak_web.erl and raw_http_resource.erl for
            %% details.
            %{raw_name, "riak"},

            %% Enable active anti-entropy subsystem + optional debug messages:
            %%   {anti_entropy, {on|off, []}},
            %%   {anti_entropy, {on|off, [debug]}},
            {anti_entropy, {on, []}},

            %% Restrict how fast AAE can build hash trees. Building the tree
            %% for a given partition requires a full scan over that partition's
            %% data. Once built, trees stay built until they are expired.
            %% Config is of the form:
            %%   {num-builds, per-timespan-in-milliseconds}
            %% Default is 1 build per hour.
            {anti_entropy_build_limit, {1, 3600000}},

            %% Determine how often hash trees are expired after being built.
            %% Periodically expiring a hash tree ensures the on-disk hash tree
            %% data stays consistent with the actual k/v backend data. It also
            %% helps Riak identify silent disk failures and bit rot. However,
            %% expiration is not needed for normal AAE operation and should be
            %% infrequent for performance reasons. The time is specified in
            %% milliseconds. The default is 1 week.
            {anti_entropy_expire, 604800000},

            %% Limit how many AAE exchanges/builds can happen concurrently.
            {anti_entropy_concurrency, 2},

            %% The tick determines how often the AAE manager looks for work
            %% to do (building/expiring trees, triggering exchanges, etc).
            %% The default is every 15 seconds. Lowering this value will
            %% speedup the rate that all replicas are synced across the cluster.
            %% Increasing the value is not recommended.
            {anti_entropy_tick, 15000},

            %% The directory where AAE hash trees are stored.
            {anti_entropy_data_dir, "./data/anti_entropy"},

            %% The LevelDB options used by AAE to generate the LevelDB-backed
            %% on-disk hashtrees.
            {anti_entropy_leveldb_opts, [{write_buffer_size, 4194304},
                                         {max_open_files, 20}]},

            %% mapred_name is URL used to submit map/reduce requests to Riak.
            {mapred_name, "mapred"},

            %% mapred_2i_pipe indicates whether secondary-index
            %% MapReduce inputs are queued in parallel via their own
            %% pipe ('true'), or serially via a helper process
            %% ('false' or undefined).  Set to 'false' or leave
            %% undefined during a rolling upgrade from 1.0.
            {mapred_2i_pipe, true},

            %% Each of the following entries control how many Javascript
            %% virtual machines are available for executing map, reduce,
            %% pre- and post-commit hook functions.
            {map_js_vm_count, 8 },
            {reduce_js_vm_count, 6 },
            {hook_js_vm_count, 2 },

            %% js_max_vm_mem is the maximum amount of memory, in megabytes,
            %% allocated to the Javascript VMs. If unset, the default is
            %% 8MB.
            {js_max_vm_mem, 8},

            %% js_thread_stack is the maximum amount of thread stack, in megabyes,
            %% allocate to the Javascript VMs. If unset, the default is 16MB.
            %% NOTE: This is not the same as the C thread stack.
            {js_thread_stack, 16},

            %% js_source_dir should point to a directory containing Javascript
            %% source files which will be loaded by Riak when it initializes
            %% Javascript VMs.
            %{js_source_dir, "/tmp/js_source"},

            %% http_url_encoding determines how Riak treats URL encoded
            %% buckets, keys, and links over the REST API. When set to 'on'
            %% Riak always decodes encoded values sent as URLs and Headers.
            %% Otherwise, Riak defaults to compatibility mode where links
            %% are decoded, but buckets and keys are not. The compatibility
            %% mode will be removed in a future release.
            {http_url_encoding, on},

            %% Switch to vnode-based vclocks rather than client ids.  This
            %% significantly reduces the number of vclock entries.
            %% Only set true if *all* nodes in the cluster are upgraded to 1.0
            {vnode_vclocks, true},

            %% This option toggles compatibility of keylisting with 1.0
            %% and earlier versions.  Once a rolling upgrade to a version
            %% > 1.0 is completed for a cluster, this should be set to
            %% true for better control of memory usage during key listing
            %% operations
            {listkeys_backpressure, true},

            %% This option specifies how many of each type of fsm may exist
            %% concurrently.  This is for overload protection and is a new
            %% mechanism that obsoletes 1.3's health checks. Note that this number
            %% represents two potential processes, so +P in vm.args should be at
            %% least 3X the fsm_limit.
            {fsm_limit, 50000},

            %% object_format controls which binary representation of a riak_object
            %% is stored on disk.
            %% Current options are: v0, v1.
            %% v0: Original erlang:term_to_binary format. Higher space overhead.
            %% v1: New format for more compact storage of small values.
            {object_format, v1}
           ]},

 %% Riak Search Config
 {riak_search, [
                %% To enable Search functionality set this 'true'.
                {enabled, false}
               ]},

 %% Merge Index Config
 {merge_index, [
                %% The root dir to store search merge_index data
                {data_root, "./data/merge_index"},

                %% Size, in bytes, of the in-memory buffer.  When this
                %% threshold has been reached the data is transformed
                %% into a segment file which resides on disk.
                {buffer_rollover_size, 1048576},

                %% Overtime the segment files need to be compacted.
                %% This is the maximum number of segments that will be
                %% compacted at once.  A lower value will lead to
                %% quicker but more frequent compactions.
                {max_compact_segments, 20}
               ]},

 %% Bitcask Config
 {bitcask, [
             %% Configure how Bitcask writes data to disk.
             %%   erlang: Erlang's built-in file API
             %%      nif: Direct calls to the POSIX C API
             %%
             %% The NIF mode provides higher throughput for certain
             %% workloads, but has the potential to negatively impact
             %% the Erlang VM, leading to higher worst-case latencies
             %% and possible throughput collapse.
             {io_mode, erlang},

             {data_root, "./data/bitcask"}
           ]},

 %% eLevelDB Config
 {eleveldb, [
             {data_root, "./data/leveldb"}
            ]},

 %% Lager Config
 {lager, [
            %% What handlers to install with what arguments
            %% The defaults for the logfiles are to rotate the files when
            %% they reach 10Mb or at midnight, whichever comes first, and keep
            %% the last 5 rotations. See the lager README for a description of
            %% the time rotation format:
            %% https://github.com/basho/lager/blob/master/README.org
            %%
            %% If you wish to disable rotation, you can either set the size to 0
            %% and the rotation time to "", or instead specify a 2-tuple that only
            %% consists of {Logfile, Level}.
            %%
            %% If you wish to have riak log messages to syslog, you can use a handler
            %% like this:
            %%   {lager_syslog_backend, ["riak", daemon, info]},
            %%
            {handlers, [
                           {lager_console_backend, info},
                           {lager_file_backend, [
                               {"./log/error.log", error, 10485760, "$D0", 5},
                               {"./log/console.log", info, 10485760, "$D0", 5}
                           ]}
                       ] },

            %% Whether to write a crash log, and where.
            %% Commented/omitted/undefined means no crash logger.
            {crash_log, "./log/crash.log"},

            %% Maximum size in bytes of events in the crash log - defaults to 65536
            {crash_log_msg_size, 65536},

            %% Maximum size of the crash log in bytes, before its rotated, set
            %% to 0 to disable rotation - default is 0
            {crash_log_size, 10485760},

            %% What time to rotate the crash log - default is no time
            %% rotation. See the lager README for a description of this format:
            %% https://github.com/basho/lager/blob/master/README.org
            {crash_log_date, "$D0"},

            %% Number of rotated crash logs to keep, 0 means keep only the
            %% current one - default is 0
            {crash_log_count, 5},

            %% Whether to redirect error_logger messages into lager - defaults to true
            {error_logger_redirect, true},

            %% maximum number of error_logger messages to handle in a second
            %% lager 2.0.0 shipped with a limit of 50, which is a little low for riak's startup
            {error_logger_hwm, 100}
        ]},

 %% riak_sysmon config
 {riak_sysmon, [
         %% To disable forwarding events of a particular type, use a
         %% limit of 0.
         {process_limit, 30},
         {port_limit, 2},

         %% Finding reasonable limits for a given workload is a matter
         %% of experimentation.
         %% NOTE: Enabling the 'gc_ms_limit' monitor (by setting non-zero)
         %%       can cause performance problems on multi-CPU systems.
         {gc_ms_limit, 0},
         {heap_word_limit, 40111000},

         %% Configure the following items to 'false' to disable logging
         %% of that event type.
         {busy_port, true},
         {busy_dist_port, true}
        ]},

 %% SASL config
 {sasl, [
         {sasl_error_logger, false}
        ]},

 %% riak_control config
 {riak_control, [
                %% Set to false to disable the admin panel.
                {enabled, false},

                %% Authentication style used for access to the admin
                %% panel. Valid styles are 'userlist' <TODO>.
                {auth, userlist},

                %% If auth is set to 'userlist' then this is the
                %% list of usernames and passwords for access to the
                %% admin panel.
                {userlist, [{"user", "pass"}
                           ]},

                %% The admin panel is broken up into multiple
                %% components, each of which is enabled or disabled
                %% by one of these settings.
                {admin, true}
                ]}
].
```

1.4.2

riak.conf



```
## The enabled Yokozuna set this 'on'.
yokozuna = off

## The port number which Solr binds to.
yokozuna.solr_port = 8093

## The port number which Solr JMX binds to.
yokozuna.solr_jmx_port = 8985

## The arguments to pass to the Solr JVM.  Non-standard
## arguments, i.e. -XX, may not be portable across JVM
## implementations.  E.g. -XX:+UseCompressedStrings.
yokozuna.solr_jvm_args = -Xms1g -Xmx1g -XX:+UseStringCache -XX:+UseCompressedOops

## The data under which to store all Yokozuna related data.
## Including the Solr index data.
yokozuna.data_dir = ./data/yz

## leveldb data_root
leveldb.data_root = ./data/leveldb

## This parameter defines the percentage, 1 to 100, of total
## server memory to assign to leveldb.  leveldb will dynamically
## adjust it internal cache sizes as Riak activates / inactivates
## vnodes on this server to stay within this size.  The memory size
## can alternately be assigned as a byte count via total_leveldb_mem instead.
leveldb.total_leveldb_mem_percent = 90

## Each database .sst table file can include an optional "bloom filter"
## that is highly effective in shortcutting data queries that are destined
## to not find the requested key. The bloom_filter typically increases the
## size of an .sst table file by about 2%. This option must be set to true
## in the riak.conf to take effect.
leveldb.bloomfilter = on

## Set to 'off' to disable the admin panel.
riak_control = off

## Authentication style used for access to the admin
## panel. Valid styles are: off, userlist
riak_control.auth = userlist

## If auth is set to 'userlist' then this is the
## list of usernames and passwords for access to the
## admin panel.
riak_control.user.user.password = pass

## bitcask data root
bitcask.data_root = ./data/bitcask

## Configure how Bitcask writes data to disk.
## erlang: Erlang's built-in file API
## nif: Direct calls to the POSIX C API
## The NIF mode provides higher throughput for certain
## workloads, but has the potential to negatively impact
## the Erlang VM, leading to higher worst-case latencies
## and possible throughput collapse.
bitcask.io_mode = erlang

## enable active anti-entropy subsystem
anti_entropy = on

## Storage_backend specifies the Erlang module defining the storage
## mechanism that will be used on this node.
storage_backend = bitcask

## raw_name is the first part of all URLS used by the Riak raw HTTP
## interface.  See riak_web.erl and raw_http_resource.erl for
## details.
## raw_name = riak

## Restrict how fast AAE can build hash trees. Building the tree
## for a given partition requires a full scan over that partition's
## data. Once built, trees stay built until they are expired.
## Config is of the form:
## {num-builds, per-timespan}
## Default is 1 build per hour.
anti_entropy.build_limit.number = 1

anti_entropy.build_limit.per_timespan = 1h

## Determine how often hash trees are expired after being built.
## Periodically expiring a hash tree ensures the on-disk hash tree
## data stays consistent with the actual k/v backend data. It also
## helps Riak identify silent disk failures and bit rot. However,
## expiration is not needed for normal AAE operation and should be
## infrequent for performance reasons. The time is specified in
## milliseconds. The default is 1 week.
anti_entropy.expire = 1w

## Limit how many AAE exchanges/builds can happen concurrently.
anti_entropy.concurrency = 2

## The tick determines how often the AAE manager looks for work
## to do (building/expiring trees, triggering exchanges, etc).
## The default is every 15 seconds. Lowering this value will
## speedup the rate that all replicas are synced across the cluster.
## Increasing the value is not recommended.
anti_entropy.tick = 15s

## The directory where AAE hash trees are stored.
anti_entropy.data_dir = ./data/anti_entropy

## The LevelDB options used by AAE to generate the LevelDB-backed
## on-disk hashtrees.
anti_entropy.write_buffer_size = 4MB

anti_entropy.max_open_files = 20

## mapred_name is URL used to submit map/reduce requests to Riak.
mapred_name = mapred

## mapred_2i_pipe indicates whether secondary-index
## MapReduce inputs are queued in parallel via their own
## pipe ('true'), or serially via a helper process
## ('false' or undefined).  Set to 'false' or leave
## undefined during a rolling upgrade from 1.0.
mapred_2i_pipe = on

## Each of the following entries control how many Javascript
## virtual machines are available for executing map, reduce,
## pre- and post-commit hook functions.
javascript_vm.map_count = 8

javascript_vm.reduce_count = 6

javascript_vm.hook_count = 2

## js_max_vm_mem is the maximum amount of memory, in megabytes,
## allocated to the Javascript VMs. If unset, the default is
## 8MB.
javascript_vm.max_vm_mem = 8

## js_thread_stack is the maximum amount of thread stack, in megabyes,
## allocate to the Javascript VMs. If unset, the default is 16MB.
## NOTE: This is not the same as the C thread stack.
javascript_vm.thread_stack = 16

## js_source_dir should point to a directory containing Javascript
## source files which will be loaded by Riak when it initializes
## Javascript VMs.
## javascript_vm.source_dir = /tmp/js_source

## http_url_encoding determines how Riak treats URL encoded
## buckets, keys, and links over the REST API. When set to 'on'
## Riak always decodes encoded values sent as URLs and Headers.
## Otherwise, Riak defaults to compatibility mode where links
## are decoded, but buckets and keys are not. The compatibility
## mode will be removed in a future release.
http_url_encoding = on

## Switch to vnode-based vclocks rather than client ids.  This
## significantly reduces the number of vclock entries.
## Only set on if *all* nodes in the cluster are upgraded to 1.0
vnode_vclocks = on

## This option toggles compatibility of keylisting with 1.0
## and earlier versions.  Once a rolling upgrade to a version
## > 1.0 is completed for a cluster, this should be set to
## true for better control of memory usage during key listing
## operations
listkeys_backpressure = on

## This option specifies how many of each type of fsm may exist
## concurrently.  This is for overload protection and is a new
## mechanism that obsoletes 1.3's health checks. Note that this number
## represents two potential processes, so +P in vm.args should be at
## least 3X the fsm_limit.
fsm_limit = 50000

## retry_put_coordinator_failure will enable/disable the
## 'retry_put_coordinator_failure' per-operation option of the
## put FSM.
## on = Riak 2.0 behavior (strongly recommended)
## off = Riak 1.x behavior
retry_put_coordinator_failure = on

## object_format controls which binary representation of a riak_object
## is stored on disk.
## Current options are: v0, v1.
## v0: Original erlang:term_to_binary format. Higher space overhead.
## v1: New format for more compact storage of small values.
object_format = v1

## listener.http.<name> is an IP address and TCP port that the Riak
## HTTP interface will bind.
listener.http.internal = 127.0.0.1:8098

## listener.protobuf.<name> is an IP address and TCP port that the Riak
## Protocol Buffers interface will bind.
listener.protobuf.internal = 127.0.0.1:8087

## pb_backlog is the maximum length to which the queue of pending
## connections may grow. If set, it must be an integer >= 0.
## By default the value is 5. If you anticipate a huge number of
## connections being initialised *simultaneously*, set this number
## higher.
## protobuf.backlog = 64

## listener.https.<name> is an IP address and TCP port that the Riak
## HTTPS interface will bind.
## listener.https.internal = 127.0.0.1:8098

## the number of replicas stored. Note: See CAP Controls for further discussion.
## http://docs.basho.com/riak/latest/dev/advanced/cap-controls/
buckets.default.n_val = 3

buckets.default.pr = 0

buckets.default.r = quorum

buckets.default.w = quorum

buckets.default.pw = 0

buckets.default.dw = quorum

buckets.default.rw = quorum

## whether or not siblings are allowed.
## Note: See Vector Clocks for a discussion of sibling resolution.
buckets.default.siblings = on

buckets.default.last_write_wins = false

## Default ring creation size.  Make sure it is a power of 2,
## e.g. 16, 32, 64, 128, 256, 512 etc
## ring_size = 64

## Default location of ringstate
ring.state_dir = ./data/ring

## Default cert location for https can be overridden
## with the ssl config variable, for example:
## ssl.certfile = ./etc/cert.pem

## Default key location for https can be overridden
## with the ssl config variable, for example:
## ssl.keyfile = ./etc/key.pem

## Default signing authority location for https can be overridden
## with the ssl config variable, for example:
## ssl.cacertfile = ./etc/cacertfile.pem

## handoff.port is the TCP port that Riak uses for
## intra-cluster data handoff.
handoff.port = 8099

## To encrypt riak_core intra-cluster data handoff traffic,
## uncomment the following line and edit its path to an
## appropriate certfile and keyfile.  (This example uses a
## single file with both items concatenated together.)
## handoff.ssl.certfile = /tmp/erlserver.pem

## DTrace support
## Do not enable 'dtrace' unless your Erlang/OTP
## runtime is compiled to support DTrace.  DTrace is
## available in R15B01 (supported by the Erlang/OTP
## official source package) and in R14B04 via a custom
## source repository & branch.
dtrace = off

platform_bin_dir = ./bin

platform_data_dir = ./data

platform_etc_dir = ./etc

platform_lib_dir = ./lib

platform_log_dir = ./log

## where do you want the console.log output:
## off : nowhere
## file: the file specified by log.console.file
## console : standard out
## both : log.console.file and standard out.
log.console = file

## the log level of the console log
log.console.level = info

## location of the console log
log.console.file = ./log/console.log

## location of the error log
log.error.file = ./log/error.log

## turn on syslog
log.syslog = off

## Whether to write a crash log, and where.
## Commented/omitted/undefined means no crash logger.
log.crash.file = ./log/crash.log

## Maximum size in bytes of events in the crash log - defaults to 65536
log.crash.msg_size = 64KB

## Maximum size of the crash log in bytes, before its rotated, set
## to 0 to disable rotation - default is 0
log.crash.size = 10MB

## What time to rotate the crash log - default is no time
## rotation. See the lager README for a description of this format:
## https://github.com/basho/lager/blob/master/README.org
log.crash.date = $D0

## Number of rotated crash logs to keep, 0 means keep only the
## current one - default is 0
log.crash.count = 5

## Whether to redirect error_logger messages into lager - defaults to true
log.error.redirect = on

## maximum number of error_logger messages to handle in a second
## lager 2.0.0 shipped with a limit of 50, which is a little low for riak's startup
log.error.messages_per_second = 100

## Name of the riak node
nodename = riak@127.0.0.1

## Cookie for distributed node communication.  All nodes in the same cluster
## should use the same cookie or they will not be able to communicate.
distributed_cookie = riak

erlang.async_threads = 64

## Increase number of concurrent ports/sockets
erlang.max_ports = 64000

## Set the location of crash dumps
erlang.crash_dump = ./log/erl_crash.dump

## Raise the ETS table limit
erlang.max_ets_tables = 256000

## Raise the default erlang process limit
erlang.process_limit = 256000

## For nodes with many busy_dist_port events, Basho recommends
## raising the sender-side network distribution buffer size.
## 32MB may not be sufficient for some workloads and is a suggested
## starting point.
## The Erlang/OTP default is 1024 (1 megabyte).
## See: http://www.erlang.org/doc/man/erl.html#%2bzdbbl
erlang.zdbbl = 32MB

## Erlang VM scheduler tuning.
## Prerequisite: a patched VM from Basho, or a VM compiled separately
## with this patch applied:
## https://gist.github.com/evanmcc/a599f4c6374338ed672e
## erlang.sfwi = 500

## Reading or writing objects bigger than this size will write
## a warning in the logs.
warn_object_size = 5MB

## Writing an object bigger than this will fail.
max_object_size = 50MB

## Writing an object with more than this number of siblings
## will generate a warning in the logs.
warn_siblings = 25

## Writing an object with more than this number of siblings
## will fail.
max_siblings = 100
```

2.0

Thanks