2.0 Warmup From EP Engine Perspective

Added by Jin Lim, last edited by Jin Lim on Nov 29, 2012

Multi-Phase Warmup

Starting up Couchbase Server's Eventually Persistent Engine (EP Engine) requires loading user data on media disk into memory.

Considering the inefficiency of sequential data loading in realtime, it instead divides the data loading into multi-phases and establishes onliness of data gradually in order.

The basic idea is to introduce intermediate level, called "run level", between the initial and done states to allow Couchbase Server to start servicing user data faster and safer.

Phase 1

In this phase, Couchbase Server does not have any data service yet (offline).

1. EP Engine populates the list of all persisted vbuckets by loading initial state of each vbucket. It labels this state as "**Initialize**".
2. EP Engine starts prefetching keys only based on the populated vbucket list. Label this state as "**KeyDump**".

Phase 2

In this phase, Couchbase Server can first transitions to the "**run level**" to start data service before reaching to the final active state (online).

1. If the **access log is enabled**:
   a. EP Engine scans a cached access file. Label this state as "**CheckForAccessLog**".
   b. EP Engine starts loading each key and data in a sequence based on the access file. Label this state as "**LoadingAccessLog**".
2. If the **access log is disabled** or **being empty**:
   a. EP Engine starts loading data of each key that was found during the "KeyDump" state in Phase 1. Label this state as "**LoadingData**".
3. EP Engine completes loading data and Couchbase Server reaches to the final active state. Labe this state as "**Done**".

Data Availability

EP Engine reaches to the "**run level**" based on one of following conditions.

1. If loading data of every key supplied from the access log completes **OR**
2. If loading data of every key persisted on vbuckets completes **OR**
3. If total number of currently loaded keys >= ep_warmup_min_items_threshold **OR**
4. If total % of memory filled by resident keys >= ep_warmup_min_memory_threshold **OR**
5. If total memory use by Couchbase Server >= mem_low_wat (memory low water mark)

Upon entering the "**run level**", EP Engine transitions the warmup state to "**Done**" and immediately stops data loading of remaining keys. Couchbase Server is now fully capable of data service, taking new writes and responding reads. Thereafter any subsequent data request on non-resident keys will read from on-media database files (EP Engine's background data fetch).

Access Log

In Couchbase Server 2.0, the access log is enabled by default in order to supplement previously known "**hot key**" during warmup. A background task of EP Engine periodically scans every active key in memory and compiles them into a cached access file called **access.log**. EP Engine also keeps a backup of this cached file, **access.old**, in case the current access log is found to be corrupted during warmup.

Users must note that EP Engine stops warmup and begins data service immediately after loading every cached key on the access log. It does not continue to load data from persistent media in order to raise total number of resident keys in memory.

The current EP Engine schedules the scanning and compilation of new access log every 24 hours. Users can dynamically configure the interval by using Couchbase Server's **cbconfig** tool. Refer to **"cbconfig" Example** for more details.

Warmup State Table

State	String Value	Warmup Phase	Description
Initial	**initialize**	**Phase 1**	
LoadingMutationLog	**loading mutation log**	**Phase 1**	**disabled by default**
EstimateDatabaseItemCount	**estimating database item count**	**Phase 1**	
KeyDump	**loading keys**	**Phase 1**	
CheckForAccessLog	**determine access log availability**	**Phase 2**	
LoadingAccessLog	**loading access log**	**Phase 2**	
LoadingData	**loading data**	**Phase 2**	**if not LoadingAccessLog**
Done	**done**	**Phase 2**	

Error Code

To CouchBaser Server clients, ENGINE_TMPFAIL (0x0d) gets generated during warmup. After the completion of warmup or EP Engine reaching to the "run level", ENGINE_ERROR_CODE enums defined in Memcached Protocol's <types.h> get generated.

Obtaining Warmup Status

Refer section **CouchBase Server Warmup Statistics** below for details of warmup related statics.

The general method for obtaining EP Engine's warmup status is to query them using **cbstats**.

As the following examples shows, one can use cbstats to collect and determine various warmup status at a point of time.

1. **ep_warmup,** represents if the warmup is being enabled. Returning value is 1 or "enabled".
   a. "cbstats localhost:port | grep warmup"
   b. "cbstats localhost:port raw warmup"
2. **ep_warmup_thread**, represents if the warmup has completed. Returning value is "running" or "complete".
   a. "cbstats localhost:port | grep warmup"
   b. "cbstats localhost:port raw warmup"
3. **ep_warmup_state**, represent the current progress of the warmup. Refer **Warmup State Table** above for returning values.

   a. "cbstats localhost:port | grep warmup"
   b. "cbstats localhost:port raw wamup"
4. **ep_warmup_time**, represent the total time spent for Couchbase Server to reach to the full active state. Returning value in milliseconds.
   a. "cbstats localhost:port | grep warmup"
   b. "cbstats localhost:port raw wamup"
5. **ep_warmup_key_count**, represent the current number of keys being loaded. Returning value will be increasing during the warmup **Phase 1**.
   a. "cbstats localhost:port raw warmup"
6. **ep_warmup_value_count**, represent the current number of key-values being loaded. Returning value will be increasing during the warmup **Phase 2**.
   a. "cbstats localhost:port raw warmup"

Couchbase Server Warmup Statistics

Couchbase Server provides various server warmup statistics at multiple levels.

Each statistics can be queried via a client binary protocol or the EP Engine stats utility, cbstats.

Top Level Warmup Stats

Name	Description	Value Type
ep_failpartialwarmup	Continue server running after failing loading some data?	Bool (FALSE = 0, TRUE = 1)
ep_waitforwarmup	Block server during the warmup?	Bool (FALSE = 0, TRUE = 1)
ep_warmup	Is warmup enabled?	Bool (FALSE = 0, TRUE = 1)
ep_warmup_batch_size	Size of each batch loaded during warmup	Integer (DEFAULT = 1000)
ep_warmup_dups	Number of failures due to duplicate keys	
ep_warmup_min_items_threshold	Enable data traffic after loading this number of key data	Integer (DEFAULT = 100)
ep_warmup_min_memory_threshold	Enable data traffic after filling this % of memory	Integer (DEFAULT = 100)
ep_warmup_oom	Number of out of memory failures during warmup	
ep_warmup_thread	Has warmup completed?	String ("complete", "running")
ep_warmup_time	Total time spent by loading data	Integer (milliseconds)

Low Level Warmup Stats

Below stats can be queried by passing the keyword, "warmup". For example, using the EP Engine utility one can type a command like

"cbstats localhost:11210 raw warmup" to query the warmup specific stats.

Name	Description	Value Type

ep_warmup	Is warmup enabled?	String ("enabled")
ep_warmup_key_count	How many keys have been loaded?	Integer
ep_warmup_value_count	How many key values (data) have been loaded?	Integer
ep_warmup_dups	Number of failures due to duplicate keys	
ep_warmup_estimated_key_count	Estimated number of keys in database	Integer (DEFAULT = "unknown")
ep_warmup_estimated_value_count	Estimated number of key data to read based on the access log	Integer (DEFAULT = "unknown")
ep_warmup_keys_time	Total time spent by loading persisted keys	Integer
ep_warmup_min_item_threshold	Enable data traffic after loading this number of key data	Integer
ep_warmup_min_memory_threshold	Enable data traffic after filling this % of memory	Integer (%)
ep_warmup_oom	Number of out of memory failures during warmup	Integer
ep_warmup_state	What is current warmup state	String, refer to **WarmupStateTable**
ep_warmup_thread	Is warmup running?	String ("running", "complete")
ep_warmup_time	Total time spent by loading data (warmup)	Integer (milliseconds)

"cbstat" Example — Add to - most stats @ disk warmup

watch

```
./cbstats localhost:12000 raw warmup./cbstats localhost:12000 raw warmup
 ep_warmup:                          enabled
 ep_warmup_key_count:                823359
 ep_warmup_value_count:              823359
 ep_warmup_dups:                     0
 ep_warmup_estimated_key_count:      unknown
 ep_warmup_estimated_value_count:    unknown
 ep_warmup_keys_time:                1748036
 ep_warmup_min_item_threshold:       10
 ep_warmup_min_memory_threshold:     10
 ep_warmup_oom:                      0
 ep_warmup_state:                    done
 ep_warmup_thread:                   complete
 ep_warmup_time:                     21454103
 ep_warmup:                          enabled
```

key count = value - cont
all data resident
everything loaded into memory

DGM = Data Greater than Memory
key count > value_count

/or running.

Note: Not global / per node
own separate too!

"cbconfig" Example — Most imp & Costly flush_param

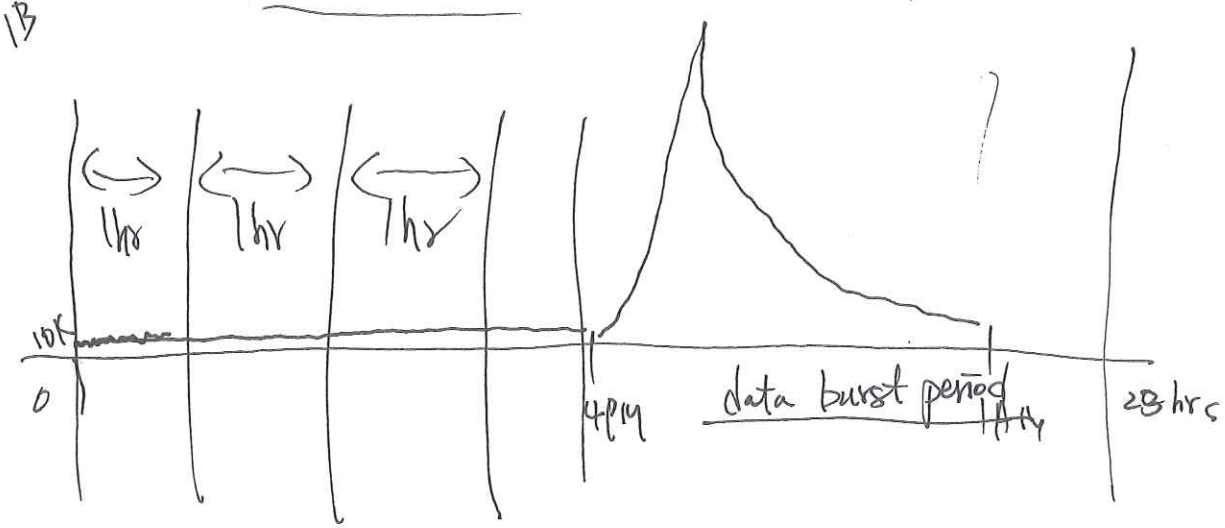
every 3 minutes SQL occurs

```
# set the access log interval to 3 minutes
./cbconfig localhost:12000 set flush_param alog_sleep_time 3
```

(12 hrs × 60)

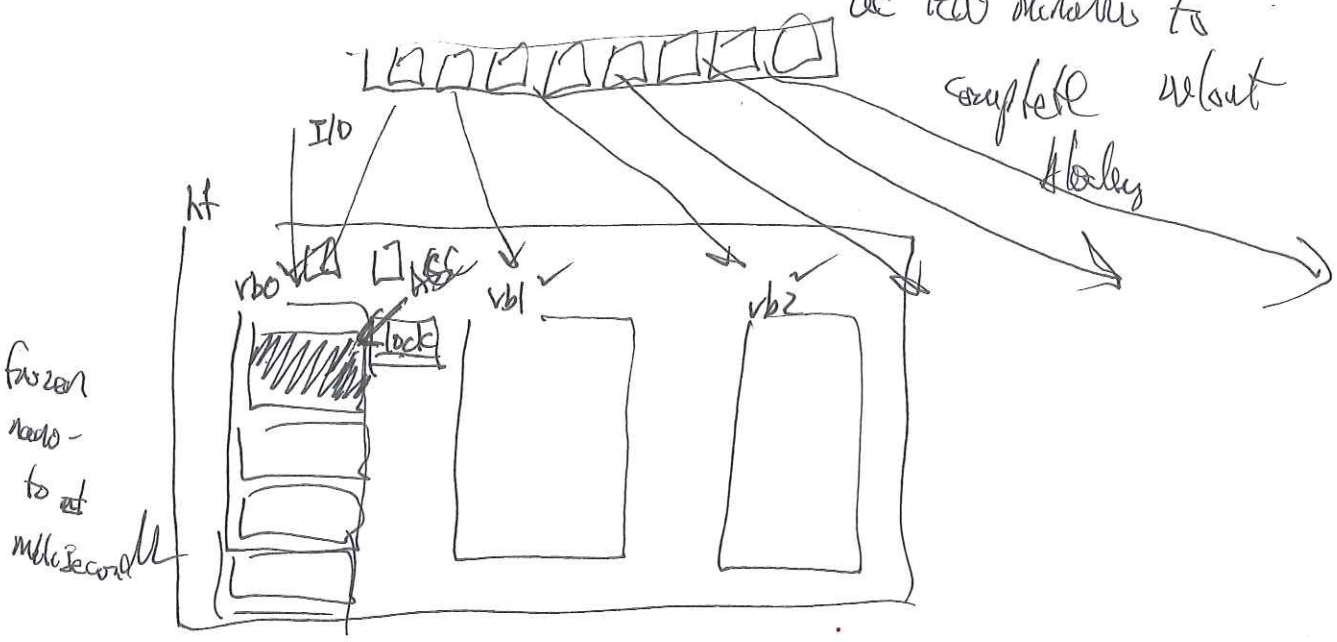
If data access very random time period,
may want to set an interval to lower interval

Interval starts midnight - always



configurable

set w/ milliseconds



as few minutes to complete w/out thrashing