Page fragments

A page fragment is an arbitrary-length arbitrary-offset area of memory which resides within a 0 or higher order compound page. Multiple fragments within that page are individually refcounted, in the page's reference counter.

The page_frag functions, page_frag_alloc and page_frag_free, provide a simple allocation framework for page fragments. This is used by the network stack and network device drivers to provide a backing region of memory for use as either an sk_buff->head, or to be used in the "frags" portion of skb_shared_info.

In order to make use of the page fragment APIs a backing page fragment cache is needed. This provides a central point for the fragment allocation and tracks allows multiple calls to make use of a cached page. The advantage to doing this is that multiple calls to get_page can be avoided which can be expensive at allocation time. However due to the nature of this caching it is required that any calls to the cache be protected by either a per-cpu limitation, or a per-cpu limitation and forcing interrupts to be disabled when executing the fragment allocation.

The network stack uses two separate caches per CPU to handle fragment allocation. The netdev_alloc_cache is used by callers making use of the netdev_alloc_frag and __netdev_alloc_skb calls. The napi_alloc_cache is used by callers of the __napi_alloc_frag and __napi_alloc_skb calls. The main difference between these two calls is the context in which they may be called. The "netdev" prefixed functions are usable in any context as these functions will disable interrupts, while the "napi" prefixed functions are only usable within the softirq context.

Many network device drivers use a similar methodology for allocating page fragments, but the page fragments are cached at the ring or descriptor level. In order to enable these cases it is necessary to provide a generic way of tearing down a page cache. For this reason __page_frag_cache_drain was implemented. It allows for freeing multiple references from a single page via a single call. The advantage to doing this is that it allows for cleaning up the multiple references that were added to a page in order to avoid calling get_page per allocation.

Alexander Duyck, Nov 29, 2016.