```
  }
  else
  {
    update_flow(flow, packet);
  }

  mrapi_mutex_unlock(flow_table[bucket].lock, &lock_key, &status);
    if (status != MRAPI_SUCCESS)
      die("Lock release failure.\n");
}
```

Again, the key MRAPI primitives are dynamic shared memory allocation and a mutex primitive. Statically allocated shared memory objects are used in the example code, but are not strictly required.

6.6 Metadata use cases

6.6.1 dynamic attribute example

Below is an example of monitoring a resource (L3 cache hits) and registering a callback event for when the counter rolls over.

```
mca_status_t mrapi_status;

#define WRONG wrong(__LINE__);
void wrong(unsigned line) {
  fprintf(stderr,"WRONG: line=%u status=%s\n",
line,mrapi_display_status(mrapi_status));
  fflush(stdout);
  exit(1);
}

/* Callbacks for handling when the counters rollover */
mrapi_boolean_t rollover = MRAPI_FALSE;
void l3cache_hits_rollover(void) {
  rollover = MRAPI_TRUE;
}

int main () {
  mrapi_parameters_t parms;
  mrapi_info_t version;
  mrapi_resource_t     *root;
  mrapi_rsrc_filter_t   filter;
  mrapi_resource_t     *l3cache;

  /* initialize */
  mrapi_initialize(DOMAIN,NODE,parms,&version,&mrapi_status);
```