```python
"""
Assignment 1: RIP protocol
Team: Bach Vu (25082165), Charlie Hunter (27380476)
Timer main program/timer.py
"""
import random, time

class RTimer:
    PRINT_TIMEOUT = 0
    PERIODIC_TIMEOUT = 1
    ENTRY_TIMEOUT = 2
    GARBAGE_TIMEOUT = 3
    ENTRIES_TIMEOUT = 4
    GARBAGES_TIMEOUT = 5
    def __init__(self, base):
        self._timeout = base
        self._time_logs = [-1.0, -1.0, -1.0, -1.0, -1.0, -1.0]

    def get_print_timeout(self):
        """ How often to print routing table """
        return self._timeout * 1/2

    def get_periodic_timeout(self):
        """ From config, Ripv2 value 30 +- (0,5) """
        return self._timeout * (1-random.uniform(-1/5, 1/5))

    def get_entry_timeout(self):
        """ Expiry of a routing entry. Ripv2 value 180 """
        return self._timeout * 6

    def get_garbage_timeout(self):
        """ Delete expired entry delay. Ripv2 value 120 """
        return self._timeout * 4

    def get_entry_check_timeout(self):
        """ Router periodic check expired entries """
        return 1 #self._timeout / 5

    def get_garbage_check_timeout(self):
        """ Router periodic check expired garbage """
        return 1 # self._timeout / 5

    def reset_timer(self, mode):
        self._time_logs[mode] = time.time()

    def is_expired(self, mode, curr_time, ttl=None):
        """ Check time log """
        curr_time = curr_time.timestamp()
        if ttl is not None:
            self._time_logs[mode] = ttl
        if self._time_logs[mode] == -1:
            return True

        timeout_value = [self.get_print_timeout, self.get_periodic_timeout,
            self.get_entry_timeout, self.get_garbage_timeout,
            self.get_entry_check_timeout, self.get_garbage_check_timeout]
        time_elapsed = curr_time - self._time_logs[mode]
        return time_elapsed >= timeout_value[mode]()
```