```c
#include <stdbool.h>

bool search(int* nums, int numsSize, int target) {
    if (nums == NULL || numsSize == 0) {
        return false;
    }

    int low = 0;
    int high = numsSize - 1;

    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (nums[mid] == target) {
            return true;
        }

        // Handle the case of duplicates: nums[low] == nums[mid] == nums[high]
        // This makes it impossible to know which side is sorted.
        if (nums[low] == nums[mid] && nums[mid] == nums[high]) {
            low++;
            // We can also add: high--; to potentially speed it up slightly,
            // but just low++ is sufficient to resolve the ambiguity.
            continue;
        }

        // Check if the left half is sorted
        if (nums[low] <= nums[mid]) {
            // Left half is sorted: [low, mid]

            // Check if the target is within the sorted left half
            if (target >= nums[low] && target < nums[mid]) {
                high = mid - 1; // Target is in the left sorted part
            } else {
                low = mid + 1; // Target is in the right rotated part
            }
        }
        // Otherwise, the right half must be sorted
        else {
            // Right half is sorted: [mid, high]

            // Check if the target is within the sorted right half
            if (target > nums[mid] && target <= nums[high]) {
                low = mid + 1; // Target is in the right sorted part
            } else {
                high = mid - 1; // Target is in the left rotated part
            }
        }
    }

    return false;
}
```