```java
import java.util.concurrent.locks.ReentrantLock;

class BetterSafeState implements State {
    private byte[] value;
    private byte maxval;
    private ReentrantLock lock;

    BetterSafeState(byte[] v, byte m) {
        lock = new ReentrantLock();
        value = v;
        maxval = m;
    }
    BetterSafeState(byte[] v) {
        this(v, (byte) 127);
    }

    public int size() {
        return value.length;
    }

    public byte[] current() {
        return value;
    }

    public boolean swap(int i, int j) {
        // Lock before doing any reading or setting of the array
        lock.lock();

        if (value[i] <= 0 || value[j] >= maxval) {
            // unlock before leaving thread - avoiding deadlock
            lock.unlock();
            return false;
        }

        value[i]--;
        value[j]++;

        lock.unlock();
        return true;
    }
}
```