PrivateLocking

The only way to write thread safe code without losing your mind is to keep your synchronisation simple and small. You cannot test for thread safety. Really you can't. If synchronization is complicated or spread out it's pretty much impossible to know by inspection whether it's correct.

A key technique is to encapsulate synchronization *inside* thread-safe classes. Every public member function should protect *itself* from concurrent access by using private locks or other synchronization objects. You can verify the synchronization of just that class in isolation. It's much easier to build complicated thread-safe code from simple pieces that you know to be individually thread-safe.

It's very dangerous to provide public access to locks because now to establish thread safety for a class you have to inspect *every potential use* of that class. Not to mention every change to or addition of code using the class. Did I mention losing your mind?