In this work, we present an open-source software platform that calculates key rate for general QKD protocols, building upon the numerical framework proposed by our group that can perform automated security proof of QKD protocols. The software platform is fully modularized with mutually independent modules for descriptions of protocols/channels, solver modules for bounding key rate, and parameter optimization algorithms. It currently supports BB84 and measurement-device-independent QKD (including decoy states), as well as discrete-modulated continuous variable QKD. It also supports finite-size analysis for non-decoy-state protocols. We hope that the open-sourcing can attract theorists to test new protocols and/or contribute to new solvers, as well as appeal to experimentalists who wish to analyze their data or optimize parameters for new experiments.

Architecture

Based on our group’s previous works, we present an open-source platform to calculate the key rate of general QKD protocols. The platform is fully modularized, with three main types of modules, each independent from the rest and is easily swappable between different modules.

1. The user-supplied input data: provides the protocol description and channel model, parameters and solver settings
   - Description file easily caters for various QKD protocols and side-channels
   - Channel model can be from theoretical simulation, can also be from real experimental data

2. The backend solver module: takes in a set of data and calculates its key rate;
   - The solver follows the two-step numerical approach to bound key rate for a given instance of protocol. Both asymptotic and finite-size solvers are included.

3. The main iteration: iterates or optimizes over a range of parameters. It views the solver module as a black box.
   - The optimization of parameter is decoupled from the protocol/solver. Any number and any combination of parameters can be specified as optimizable (or iterable).
   - User can choose between various optimization algorithms, including e.g. efficient local-search algorithms.

Our platform is also structurally such that there are multiple abstraction levels exposed to users with different purposes:

- A casual user can pick up one of the presets to easily perform simulations or optimize parameters for existing protocols.
- A theorist can choose to test key rates of new types of protocols or channels by supplying new description files. An experimentalist can also replace the channel model with real data to calculate key rate.
- An expert user can opt to replace existing solver modules with one of their own, so long as it follows the interface of accepting one set of protocol/channel data and returning a key rate.

References