netQuil

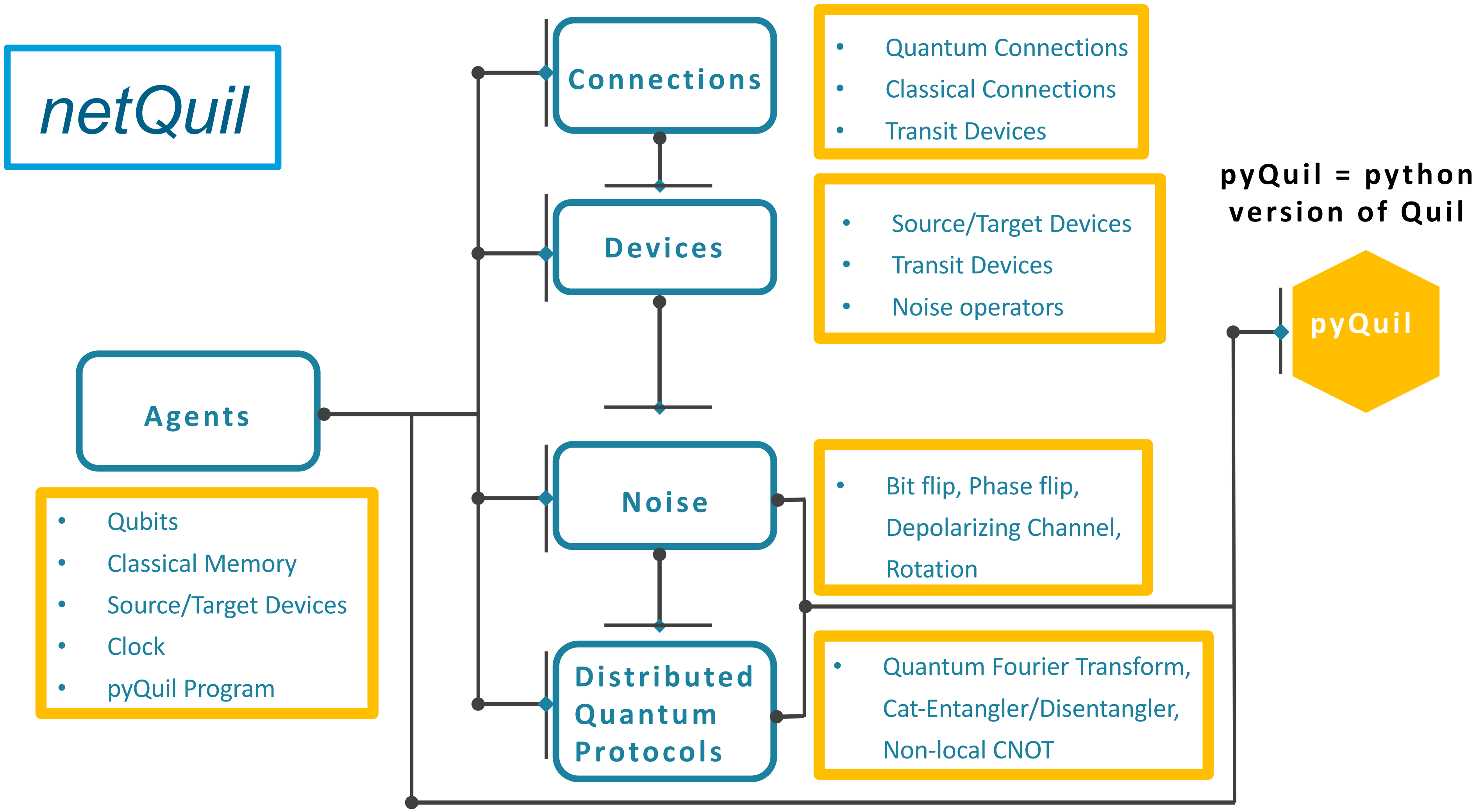
A Distributed Quantum Network Simulator

Matthew Radzihovsky and Zac Espinosa
Stanford University
INQNET – Palo Alto Foundry

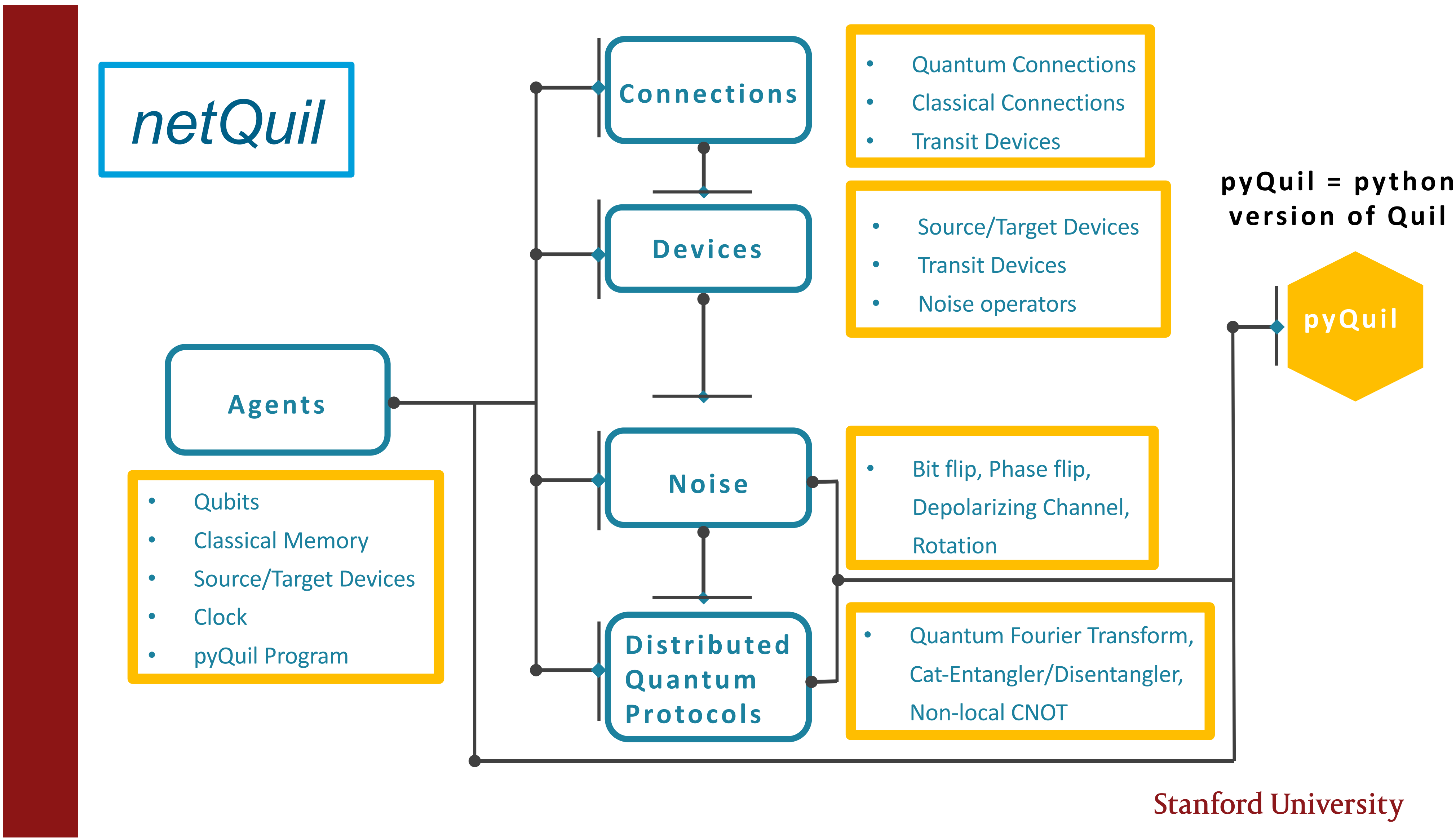
Motivation

Establish quantum network capabilities

- Quantum computing simulation platforms:
  - Quipper, IBM Q, LIQUi|>, QCL, Quil
- Test quantum distributed algorithms before infrastructure is available
- Easily accessible to a large audience
- Built off extensible quantum computing simulator, pyQuil



Stanford University


netQuil
Connections
Devices
Agents
Noise
Distributed Quantum Protocols
Quantum Connections
Classical Connections
Transit Devices
Source/Target Devices
Transit Devices
Noise operators
Bit flip, Phase flip, Depolarizing Channel, Rotation
Quantum Fourier Transform, Cat-Entangler/Disentangler, Non-local CNOT
Qubits
Classical Memory
Source/Target Devices
Clock
pyQuil Program
pyQuil = python version of Quil
pyQuil
Stanford University

Distributed Protocols

Two Primitive Distributed Protocol

A. Cat-entangler

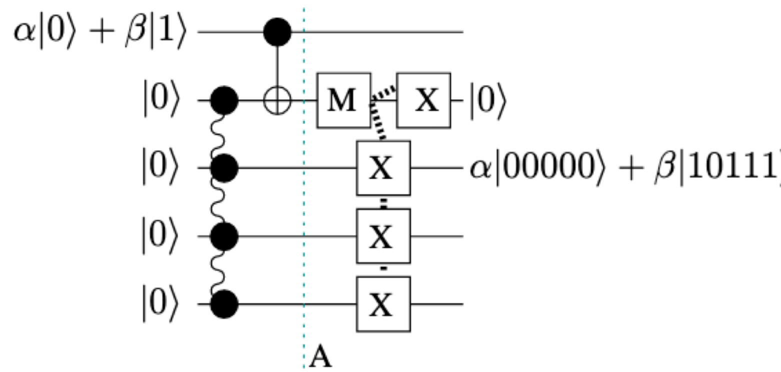
- Projects the state of a local control bit onto a system of entangled qubits.

B. Cat disentangler

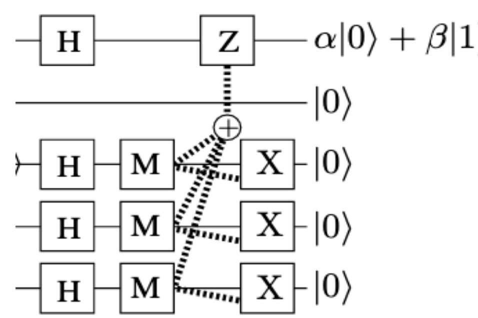
- Reverts quantum system to original state by inverting projection.

C. Universal Set

- Can be used for distributed non-local CNOT, non-local controlled gates, and teleportation



A. Cat-entangler



B. Cat-disentangler

Demo - Quantum Teleportation

Demo - Quantum Teleportation

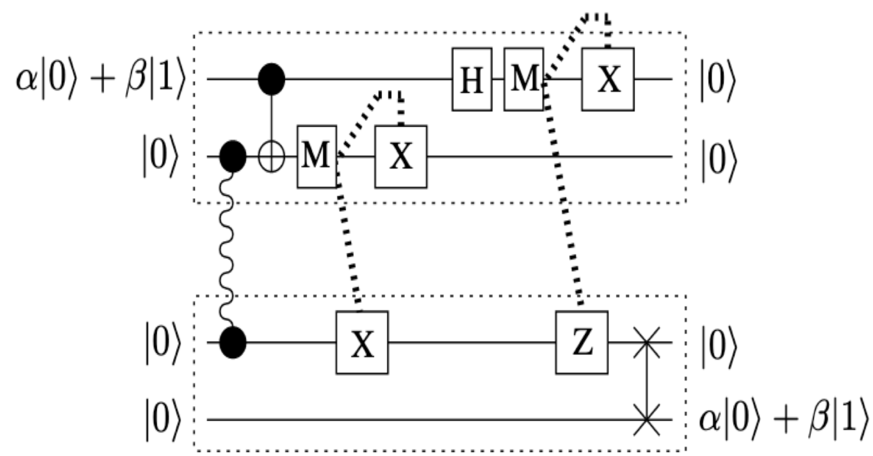
Quantum Teleportation

Agents: Alice, Bob

Premise: Alice wishes to share the arbitrary, unknown state of a qubit, $\alpha|0> \; +\beta|1>,$ with Bob.

Process:

1. Two entangled qubits distributed to Alice and Bob
2. Apply Cat-entangler
3. Apply Cat-disentangler



B. Teleportation Circuit

Quantum Teleportation

Alice

```python
class Alice(Agent):
    '''
    Alice uses cat-entangler and cat-disentangler to teleport psi to Bob
    '''
    def start_teleportation(self, psi, a, b):
        cat_entangler(
            control=(self, psi, a, ro),
            targets=[(bob, b)],
            entangled=False,
            notify=True
        )

    def run(self):
        # Define Qubits
        a, psi = self.qubits
        b = bob.qubits[0]

        # Start Teleport
        self.start_teleportation(psi, a, b)

        # Wait for teleportation to finish
        cbit = self.crecv(bob.name)
```

Bob

```python
class Bob(Agent):
    '''
    Bob waits for cat-entangler to finish and then starts cat-disentangler
    '''
    def finish_teleportation(self, b, psi):
        cat_disentangler(
            control=(self, b, ro),
            targets=[(alice, psi)],
            notify=True
        )

    def run(self):
        # Define Qubits
        b = self.qubits[0]
        _, psi = alice.qubits

        # Receive Measurement from Cat-entangler
        self.crecv(alice.name)
        if self.crecv(alice.name)[0]:
            self.finish_teleportation(b, psi)
```

Program

```python
p = Program()
p += H(2)
p += RZ(math.pi/2, 2)

# Create Classical Memory
ro = p.declare('ro', 'BIT', 3)
alice = Alice(p, qubits=[0,2], name='alice')
bob = Bob(p, qubits=[1], name='bob')

QConnect(alice, bob)
CConnect(alice, bob)

Simulation(alice, bob).run()
qvm = QVMConnection()
qvm.run(p)
```

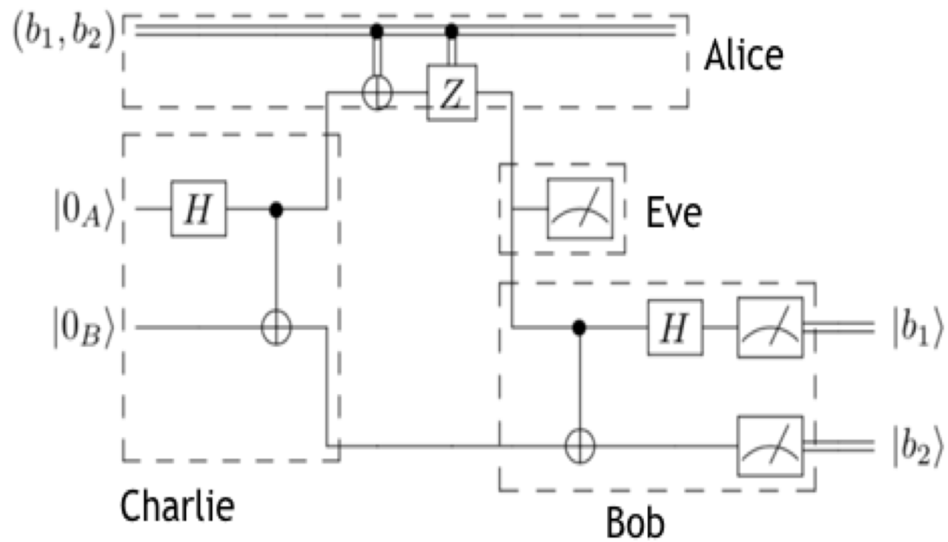
Output

```
Initial State: (0.5-0.5j)|0> + (0.5+0.5j)|1>
----------------------
Final State:  (0.5-0.5j)|000> + (0.5+0.5j)|010>
----------------------
```

Middle-man Attack Demo:
Quantum Networks Resistance to Attacks

Agents: Alice, Bob, Charlie, Eve

Premise: Alice wishes to share classical bits with Bob using superdense coding, but her qubit is intercepted by a third agent Eve

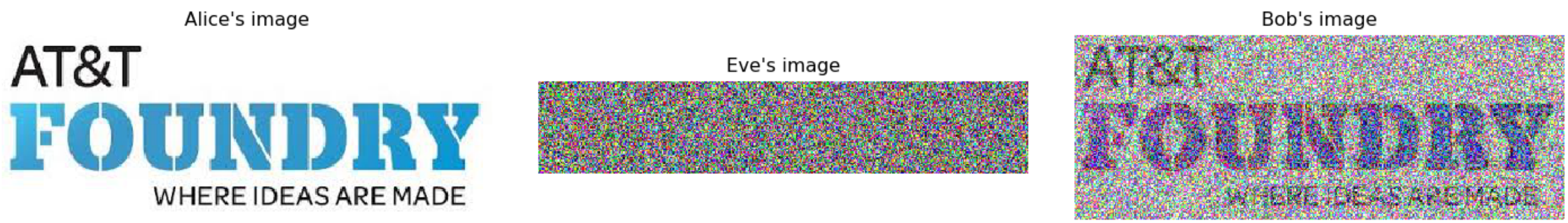


Process:

1. Charlie entangles two qubits and distributes them to Alice and Bob
2. Alice prepares her qubits based on the classical bits she wishes to send
3. Eve intercepts, measures, and resends Alice's qubit as it is sent to Bob
4. Bob receives the qubit, thinking it is directly from Alice. He measures the qubit from Alice and qubit from Charlie, learning that half the information is corrupted and there is an intruder.

Stanford University

Middle-man Attack Demo:



- Eve only has access to the intercepted qubit
  - She recovers random noise

- The qubit sent from Alice to Bob is corrupted by Eve
  - Bob recovers half the image from his entangled qubit and is alerted to an intruder

Middle-man Attack Fun:

Alice's image

Eve's image

Bob's image

To Use and Future Work

- Documentation (https://att-innovate.github.io/netQuil/index.html)
  - Open sourced on GitHub (https://github.com/att-innovate/netQuil)
  - Can directly use pip to download: `pip install netquil`
  - To use, simply include netQuil library: `from netQuil import *`
- Whitepaper
  - DOI:

In progress:

- Develop realistic noise models based on devices used in experiments
- Increase features of system


PALO ALTO
AT&T FOUNDRY