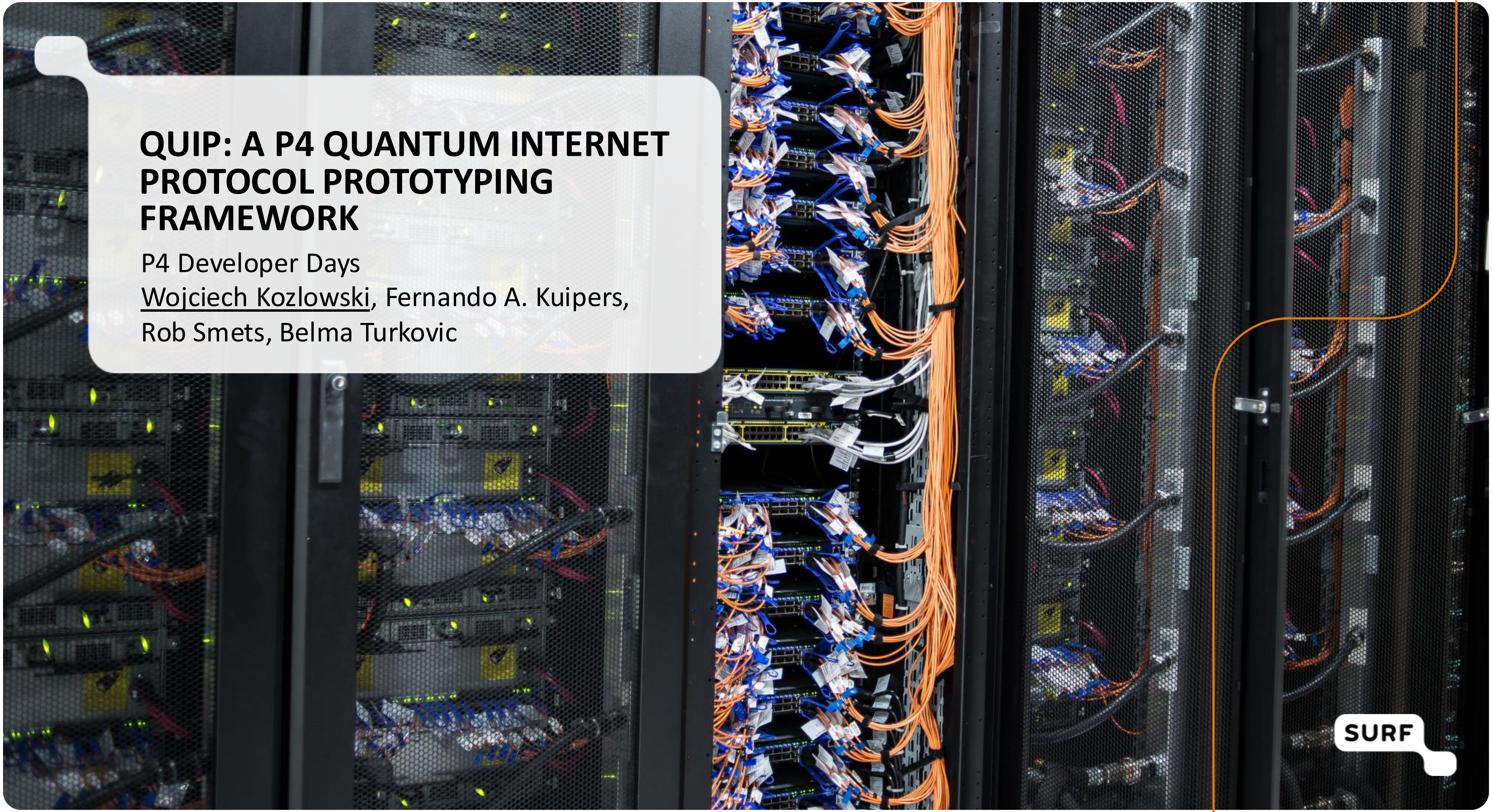


QUIP: A P4 QUANTUM INTERNET PROTOCOL PROTOTYPING FRAMEWORK

P4 Developer Days

Wojciech Kozlowski, Fernando A. Kuipers,
Rob Smets, Belma Turkovic

SURF

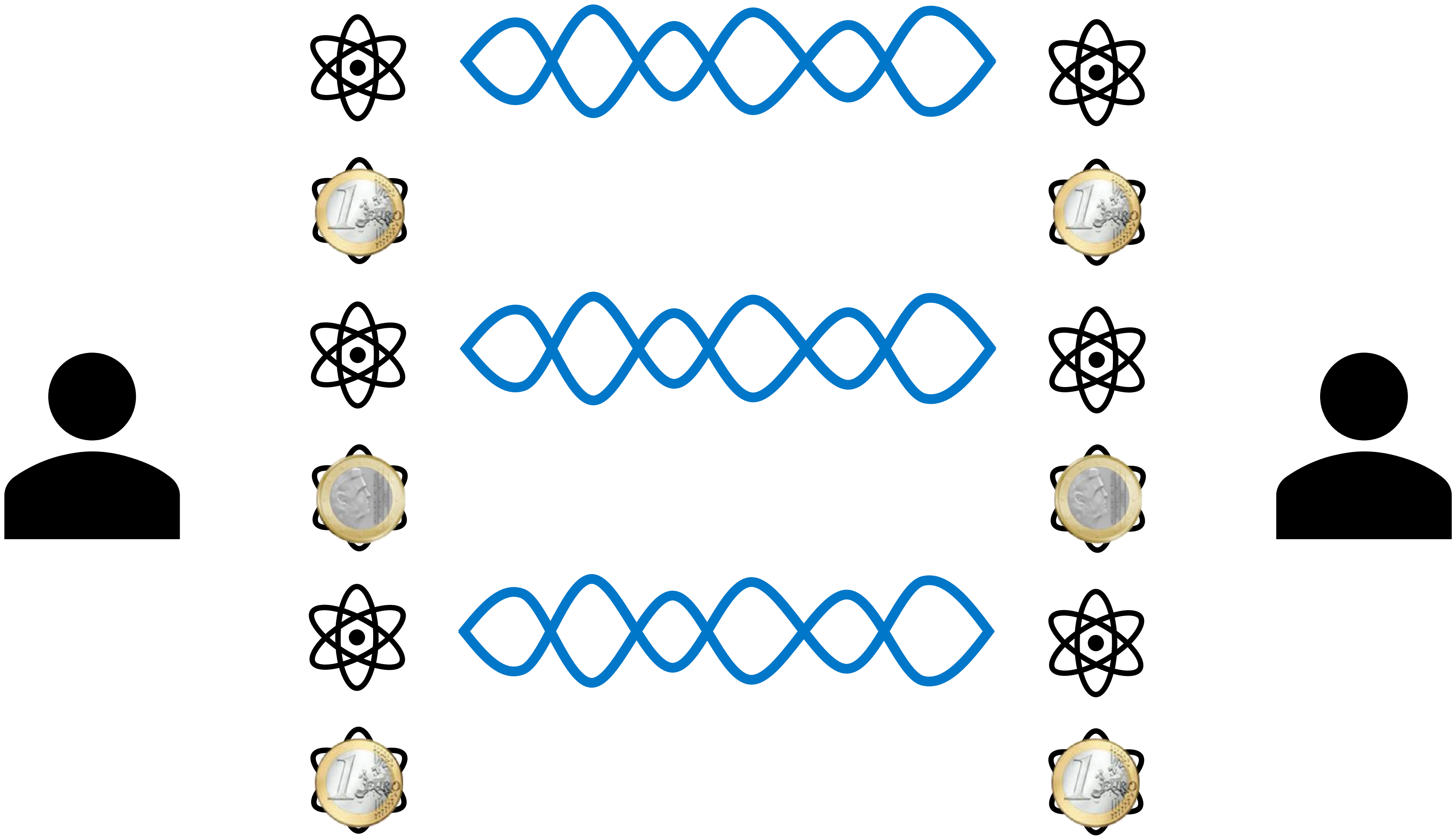
1936 IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 42, NO. 7, JULY 2024

QuIP: A P4 Quantum Internet Protocol Prototyping Framework

Wojciech Kozlowski, Fernando A. Kuipers, *Senior Member, IEEE*, Rob Smets, and Belma Turkovic

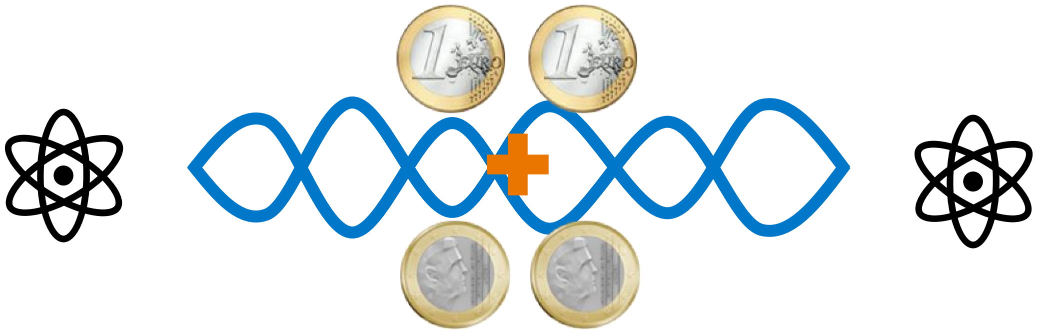
Wojciech Kozlowski, *QuTech, Delft University of Technology (now at SURF)*
Fernando A. Kuipers, *Delft University of Technology*
Rob Smets, *SURF (now at TNO)*
Belma Turkovic, *TNO*

Quantum Entanglement

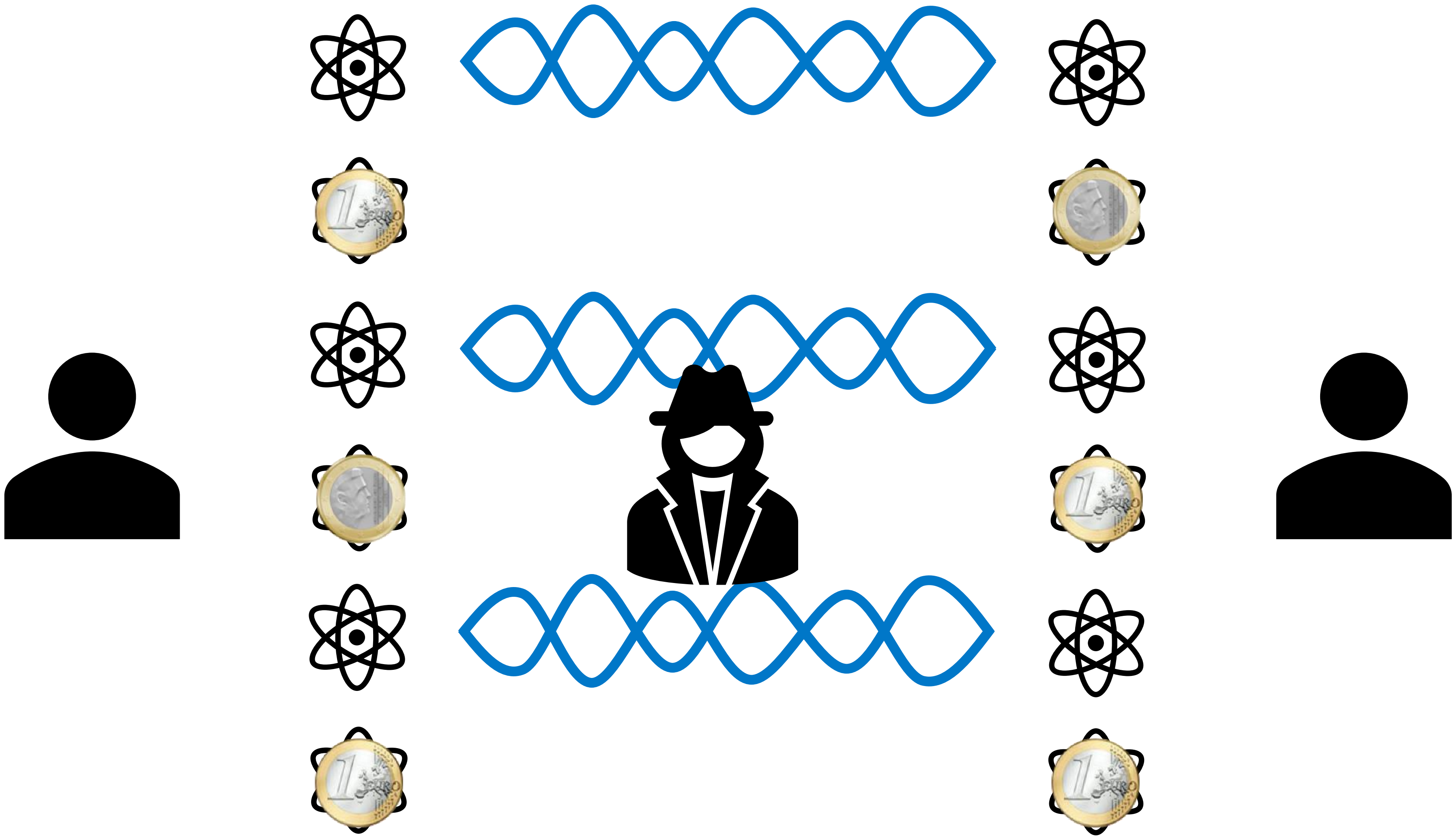


Quantum Entanglement

- Measuring a pair of quantum entangled particles always yields a random outcome, but the outcome is **always correlated** (i.e., always the same or always opposite).
  - I will use *always the same* for my examples but *always opposite* is also valid.
- It is like flipping two independent coins **that always land on the same side**.
- This is not possible “classically” (i.e., using non-quantum physics).
- Quantum entanglement also guarantees that the randomness is also secure.
- Until the measurement, the result is **undefined, not unknown** and a measurement will perturb the system.



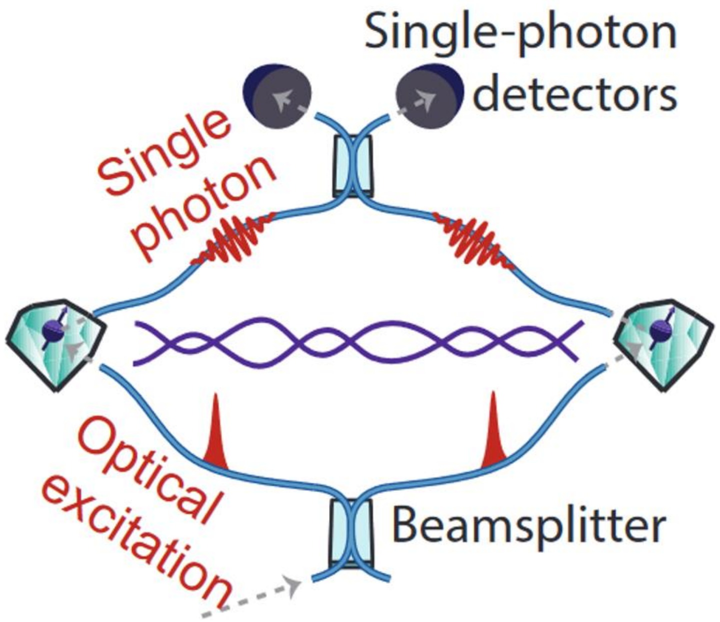
Quantum Entanglement



Quantum Entanglement

- An eavesdropper perturbs the system such that the outcomes are no longer fully correlated – some of the coin flips will no longer match.
- If the sender and receiver compare a subset of their outcomes, then they can detect an eavesdropper while keeping the unshared outcomes secret:
  - If all compared outcomes match => there is no eavesdropper,
  - If some outcomes do not match => there is an eavesdropper.
- This is how **Quantum Key Distribution (QKD)** works!
- In reality, the presence of noise and hardware limitations make the actual protocol more complicated, but the principles remain the same.

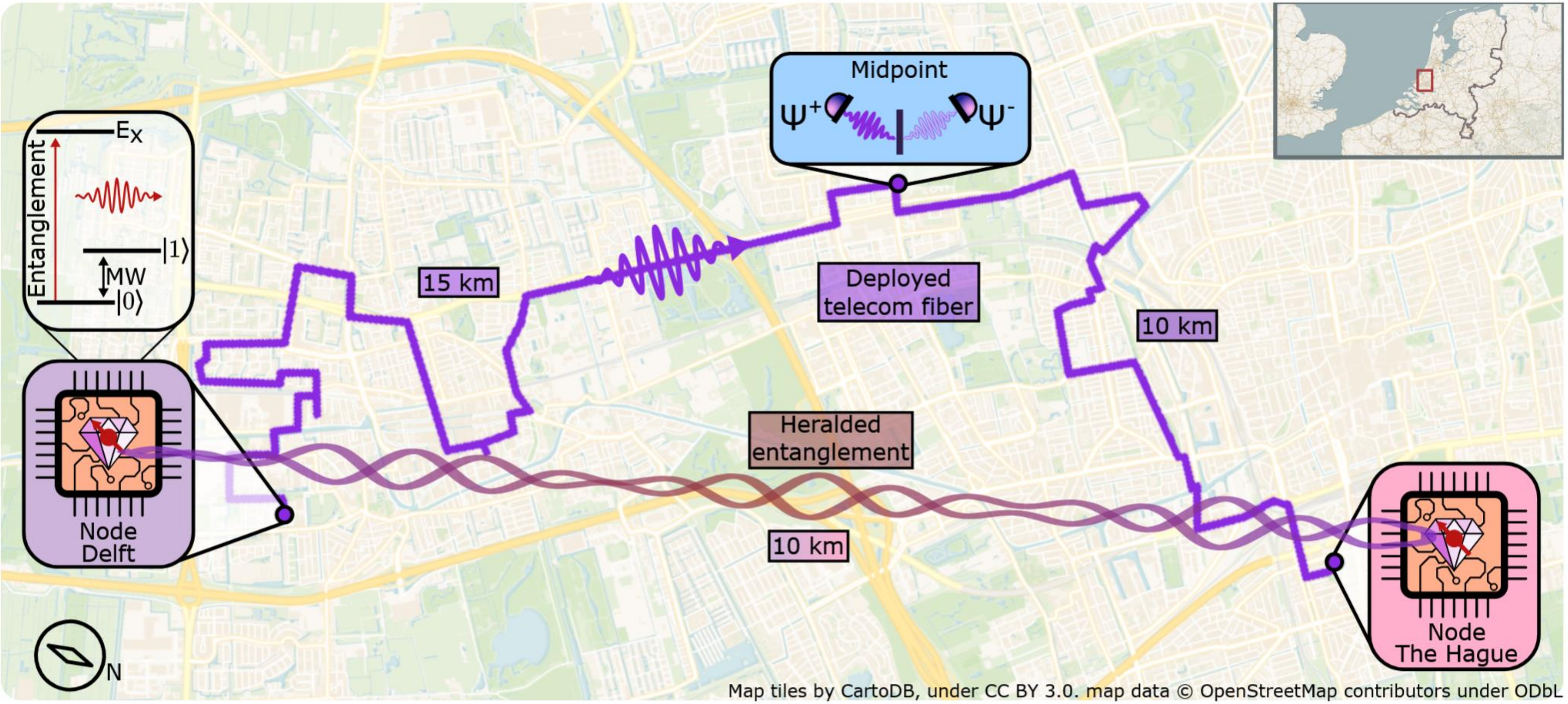
Quantum Entanglement



Deterministic delivery of remote entanglement on a quantum network
P. C. Humphreys, N. Kalb, et al.
Nature 558, pages 268-273 (2018)
arXiv:1712.07567

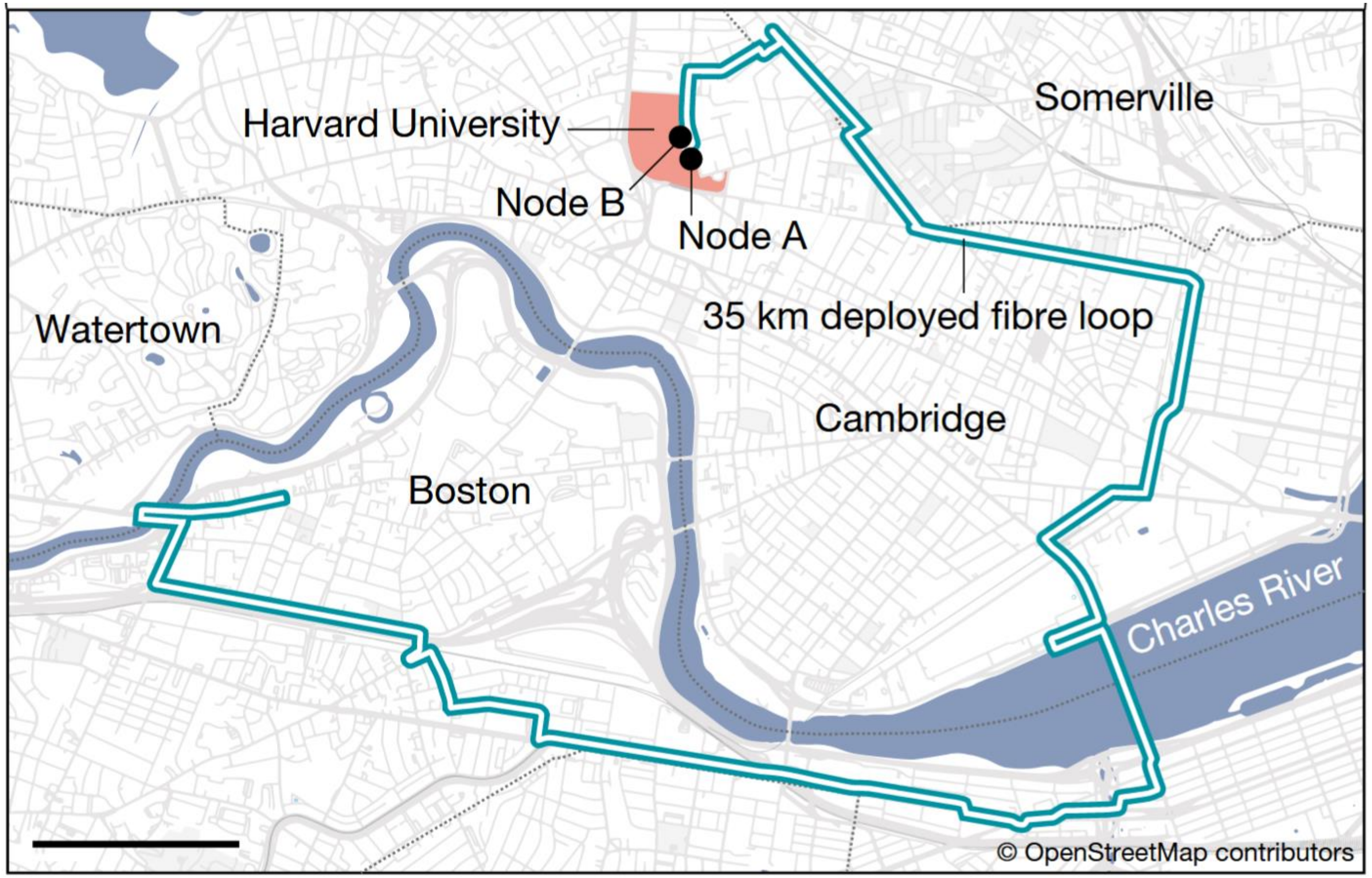
- Entanglement is always first generated locally, followed by a movement of one or both entangled qubits across the link through quantum channels.
- In this scheme, each node emits a photon entangled with a local qubit.
- The photons **must** arrive at the midpoint at exactly the same time.
- If one of the detectors clicks, then the photons have not been lost and the qubits at the nodes are now entangled.

Quantum Entanglement



Metropolitan-scale heralded entanglement of solid-state qubits
A. J. Stolk, K. L. van der Enden, et al.
arXiv:2404.03723

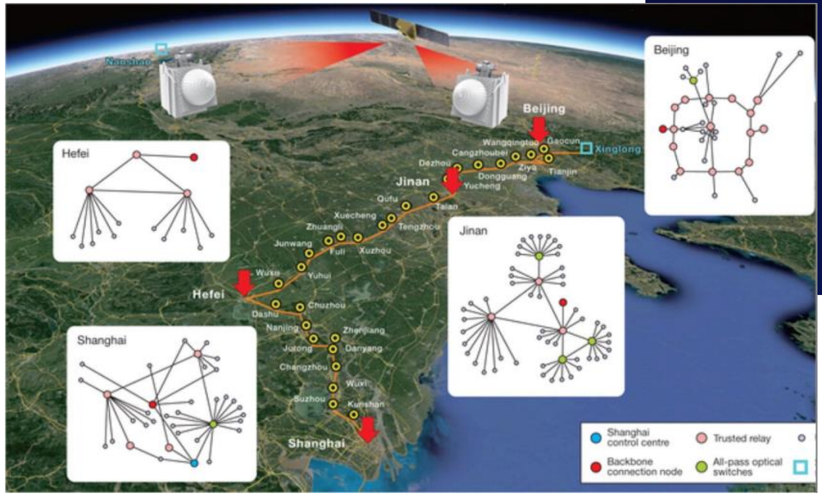
Quantum Entanglement



Entanglement of nanophotonic quantum memory nodes in a telecom network
C. M. Knaut, A. Suleymanzade, Y.-C. Wei, D. R. Assumpcao, P.-J. Stas, et al.
Nature 629, pages 573-578 (2024)
arXiv:2310.01316

Quantum Entanglement: Quantum Key Distribution

- QKD networks are being deployed in testbeds around the world.
- In Europe alone, the EuroQCI (European Quantum Communication Infrastructure) consists of 26 such testbeds.
- Several vendors are available: LuxQuanta, Q*Bird, Toshiba, ID Quantique, …
- China boasts longest chain: 2000 km combined with satellite effort.



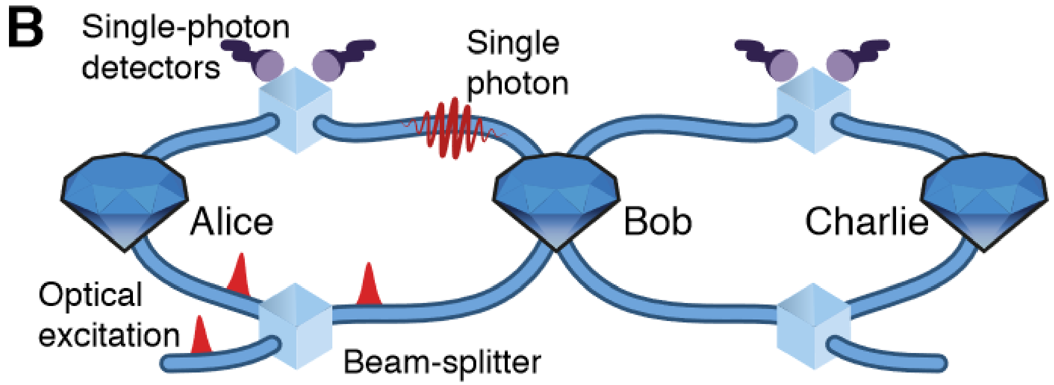
Quantum Entanglement

- However, few of the current QKD deployments utilize quantum entanglement.
- QKD can be implemented by using the quantum states of single photons which are sent down the fiber from sender to receiver.
- The **no-cloning theorem** states that it is impossible to copy an unknown quantum state and thus also to amplify the quantum signal.
- This can be overcome with quantum error correction, but the technological demands are enormous and not possible in the foreseeable future.
- This imposes a distance limitation for the quantum signal (~100-200 km) – the secret key is exposed at intermediate nodes, also called **Trusted Repeater Nodes**.
- The solution to this is **quantum entanglement**.

Quantum Entanglement

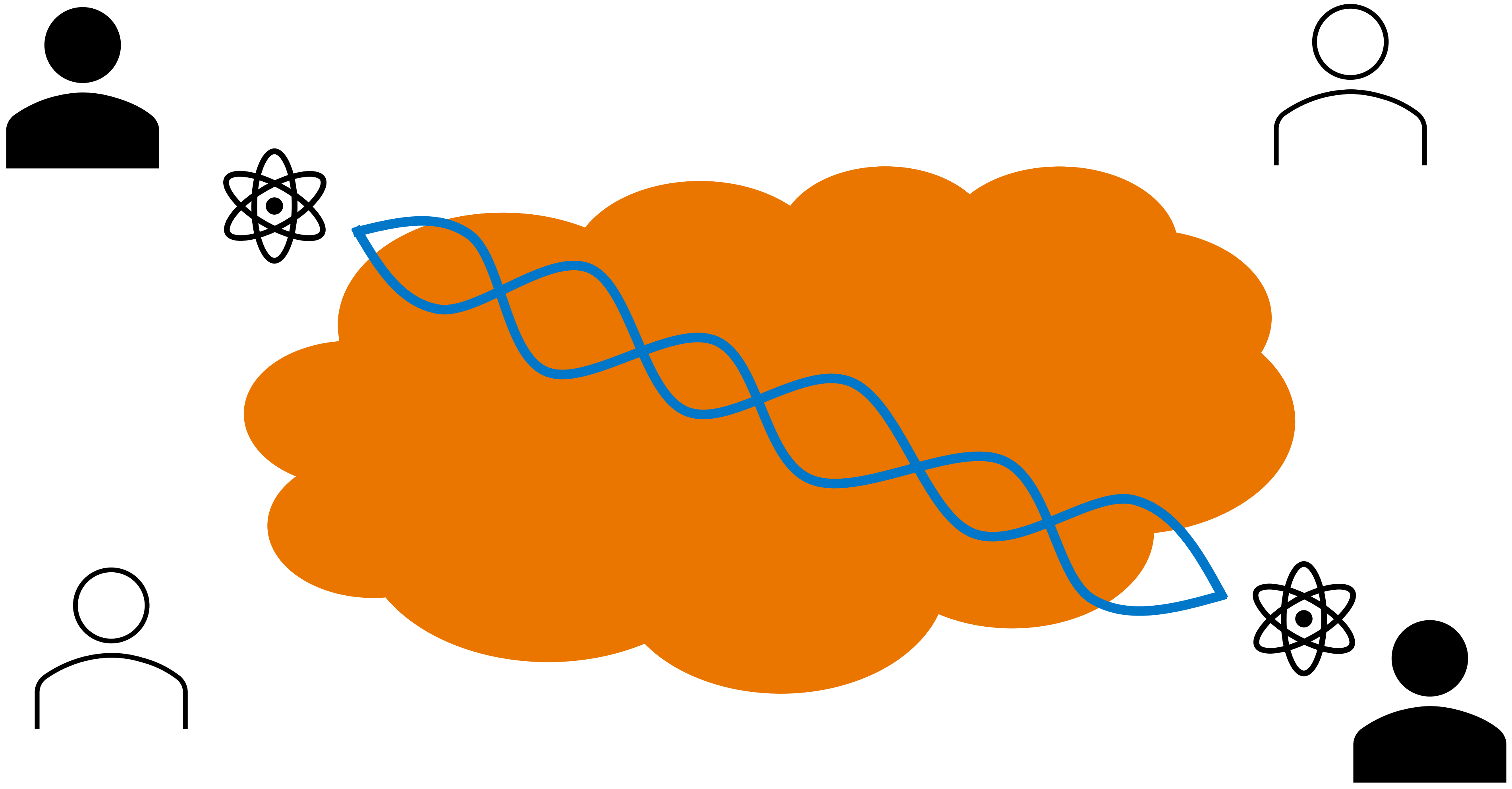
2-bit measurement outcome

2-bit measurement outcome

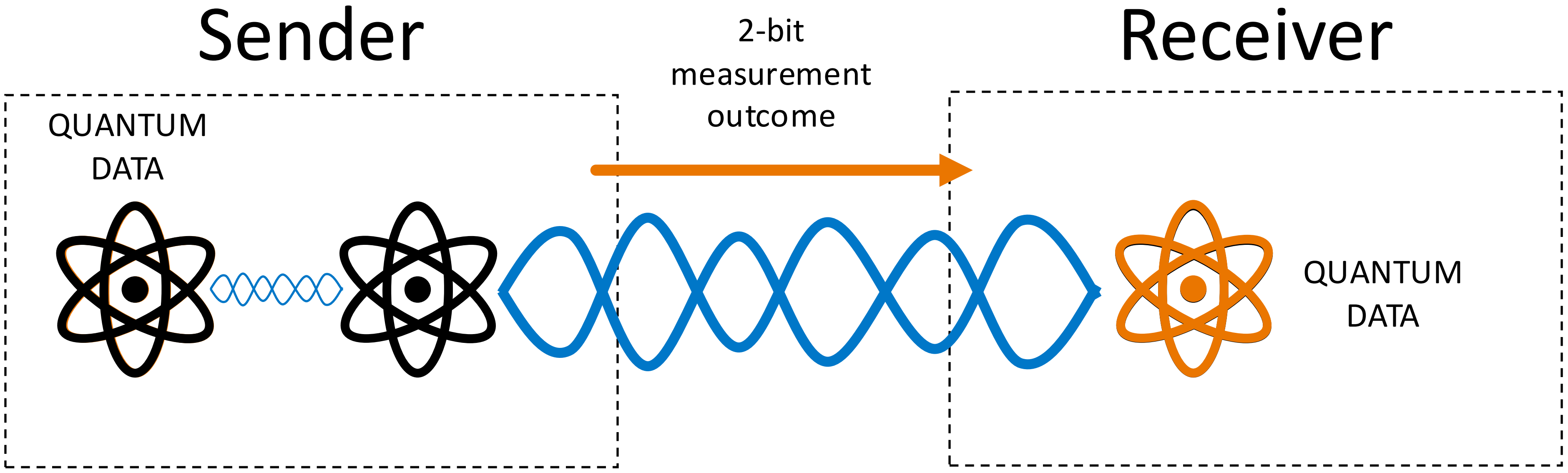


Realization of a multi-node quantum network of remote solid-state qubits
M. Pompili, S. L. N. Hermans, S. Baier, et al.
Science, 372, 259-264 (2021)
arXiv:2102.04471

Quantum Entanglement



Quantum Entanglement: Teleportation



Quantum Entanglement

Entanglement is the fundamental building block of quantum networks

SURF

Quantum Entanglement

Entanglement is the fundamental building block of quantum networks

- Either of the qubits can be sent to another device which, in principle, can be anywhere in the universe.
- Provided negligible noise has been introduced, the two qubits will forever remain in the entangled state until a measurement is performed.

The physical distance does not matter at all for entanglement

- It is possible to leverage the non-classical correlations provided by entanglement in order to design completely new types of application protocols that are not possible to achieve with just classical communication.

Problem Statement

- The research community has developed several approaches to quantum network architecture and protocol design, as well as quantum simulators.
- Due to the low technological level of the hardware, simulators are crucial in quantum network protocol research.
- However, they do not share any code, tooling, or even node architecture.
- This results in two key challenges:

  1. **Quantum network protocol research happens in silos based on simulators as it is easier to build upon prior work than porting code between simulators.**
  2. **Yet it can still be difficult to apply prior results and extend simulations due to tight coupling between protocol and simulation code.**

- What we propose with QuIP: decouple protocol implementations from simulators.

The $P4_{16}$ Network Programming Language

- To decouple quantum network protocols implementations from simulations we need a Domain-Specific Language (DSL).
- But we cannot assume a node architecture – quantum networks are not ready for stable device architectures.
- Exploring device architectures must be part of quantum network protocol research.

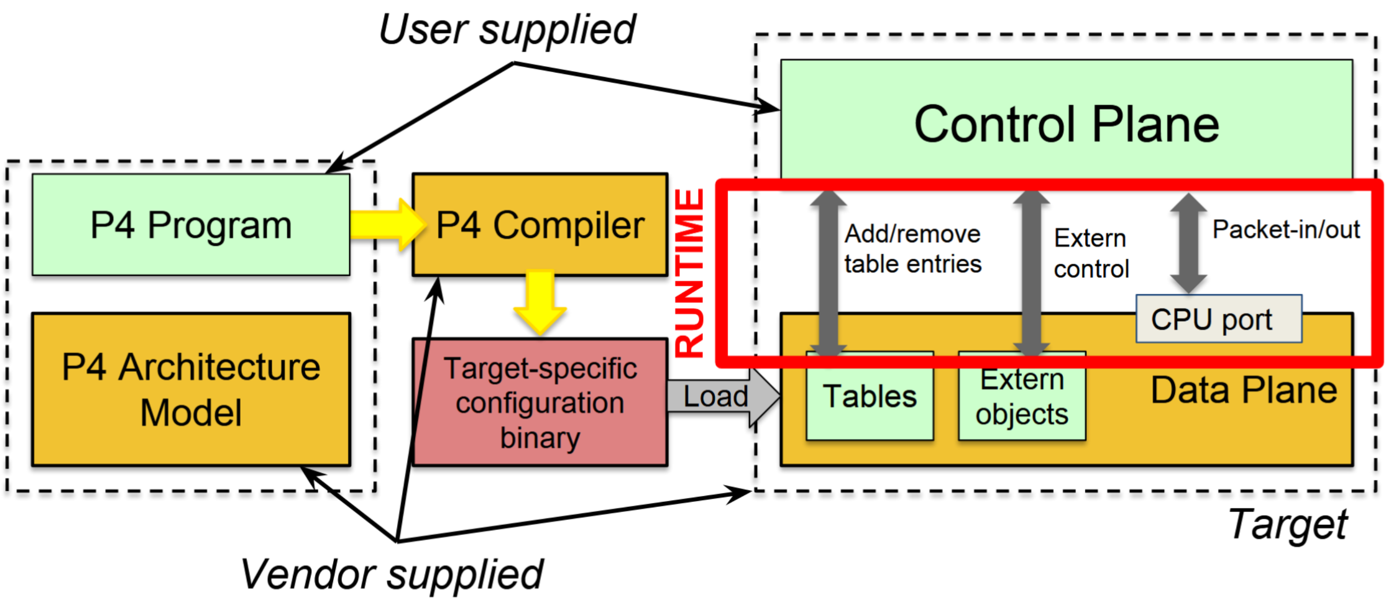


Image source: p4.org

The $P4_{16}$ Network Programming Language

- The P4 language is an existing network DSL and despite being designed for packet networks we found many of its concepts suitable for quantum entanglement.
  - E.g., viewing the device as a fixed-function device with programmable blocks.
- The $P4_{16}$ language explicitly discards the notion of a single generic architecture.
- The P4 language has an ecosystem of open-source tooling, such as the compiler.

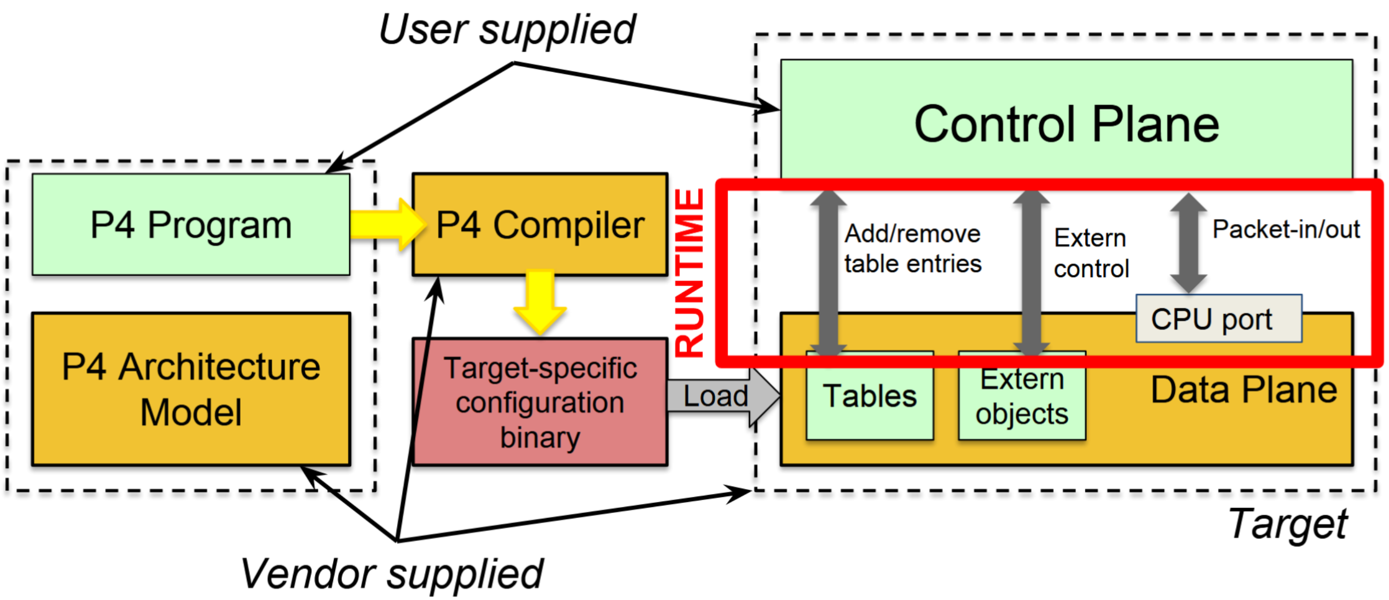
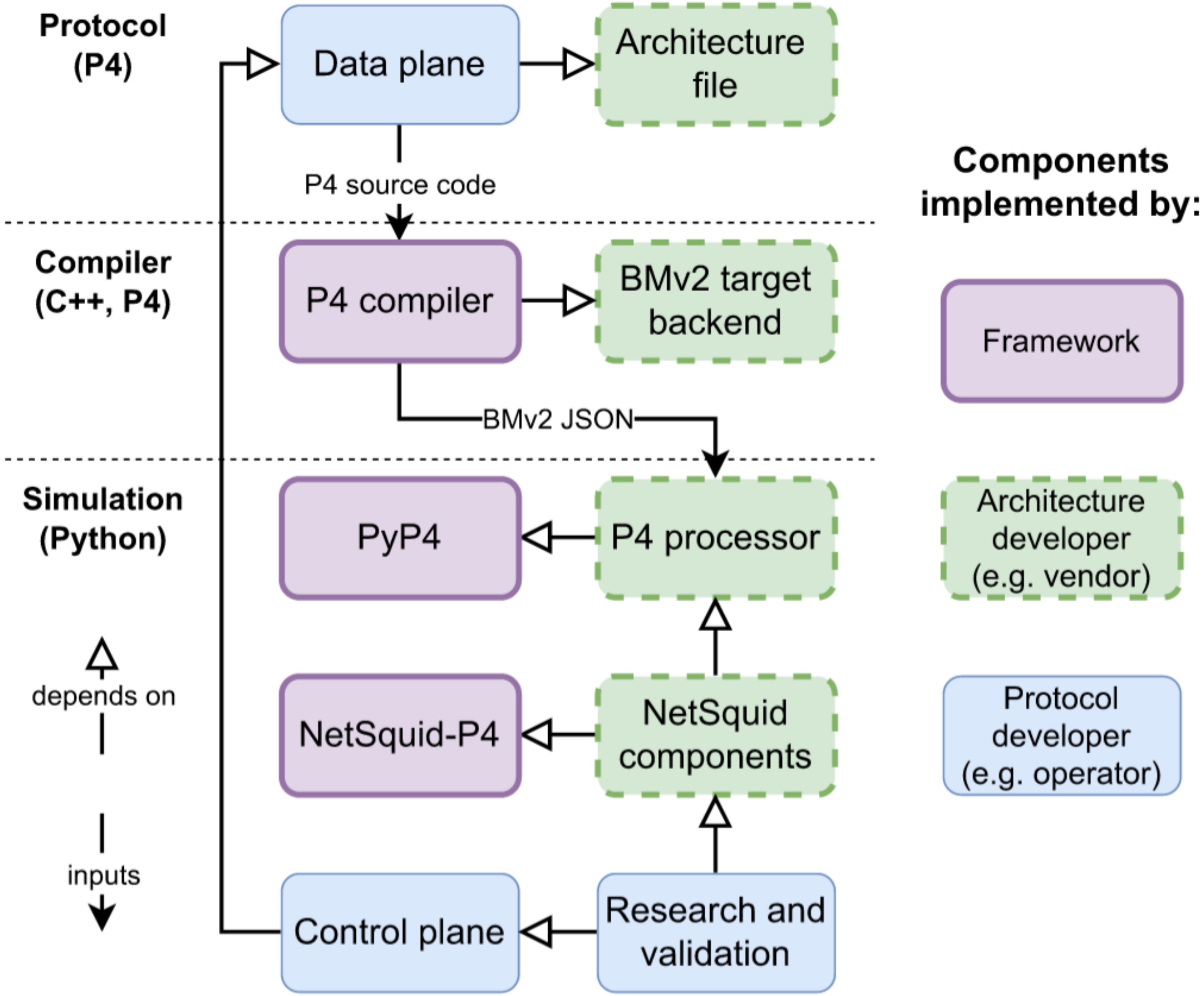


Image source: p4.org

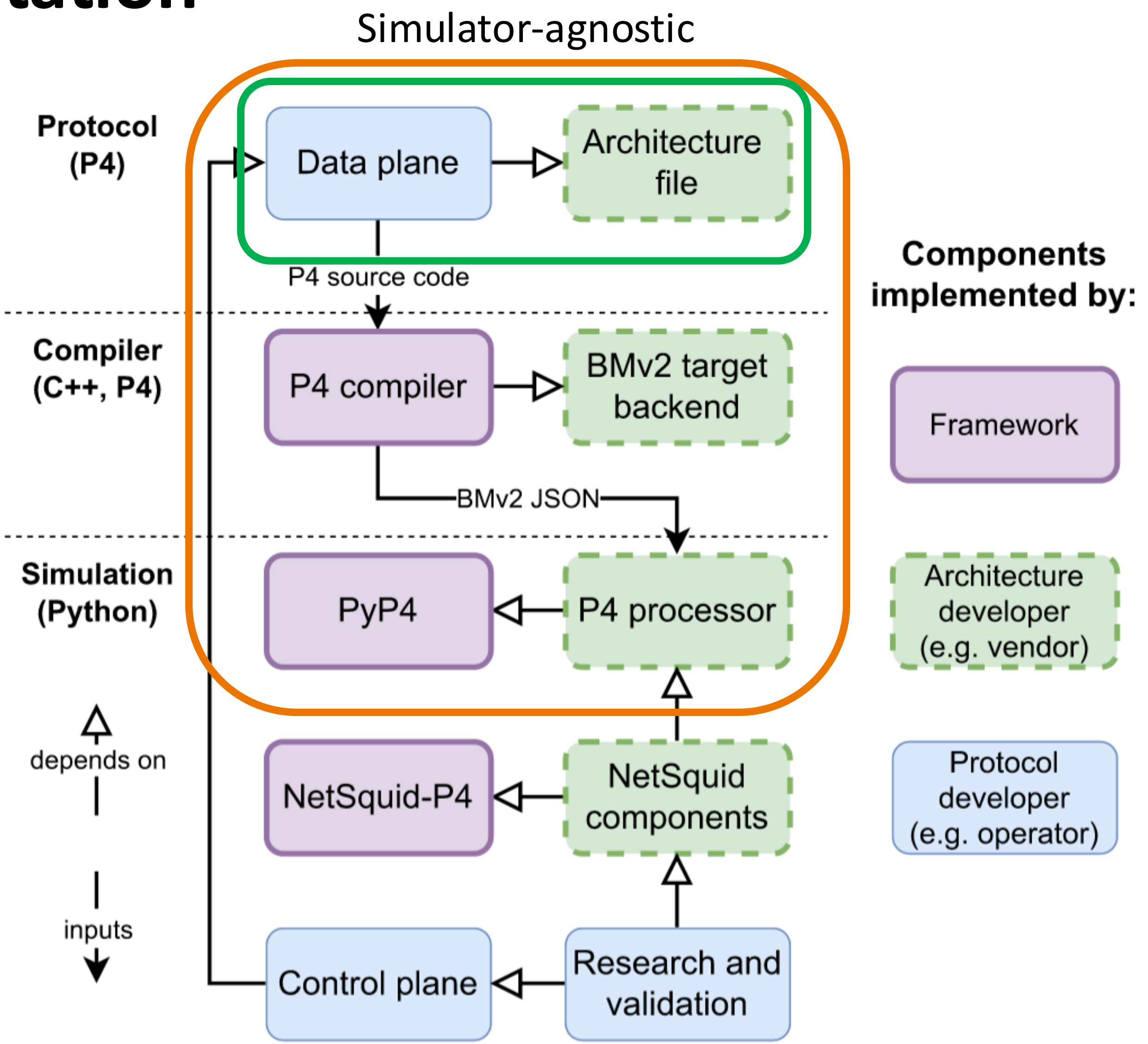
Architecture and Implementation

- Key challenge: $P4_{16}$ allows for custom architectures but it does not envision adjusting them often or that researchers would propose their own.
- We implemented QuIP, a collection of software packages, which allow flexible design-space exploration of quantum device architectures.
- An “architecture developer” can now implement a new architecture in this framework and a “protocol developer” can use it for protocol research.



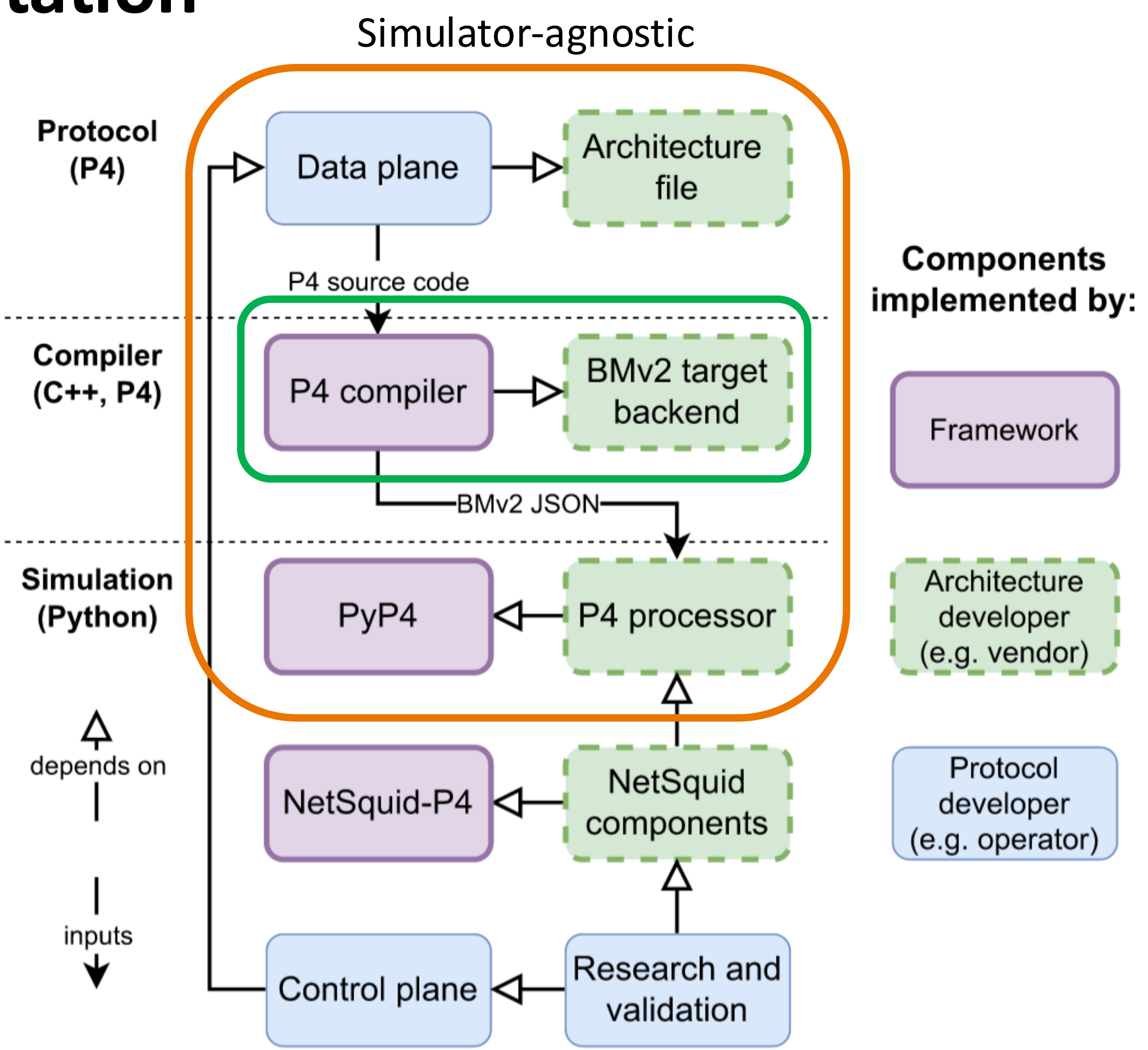
Architecture and Implementation

1. Data plane / Architecture
   - An architecture file defines the fixed-function device, its programmable blocks, and the connections between the blocks.
   - A protocol developer implements their protocol against the device specification in the architecture.
   - $P4_{16}$ leaves interface definition to the architectures enabling quantum device architectures without any extensions to the language itself.



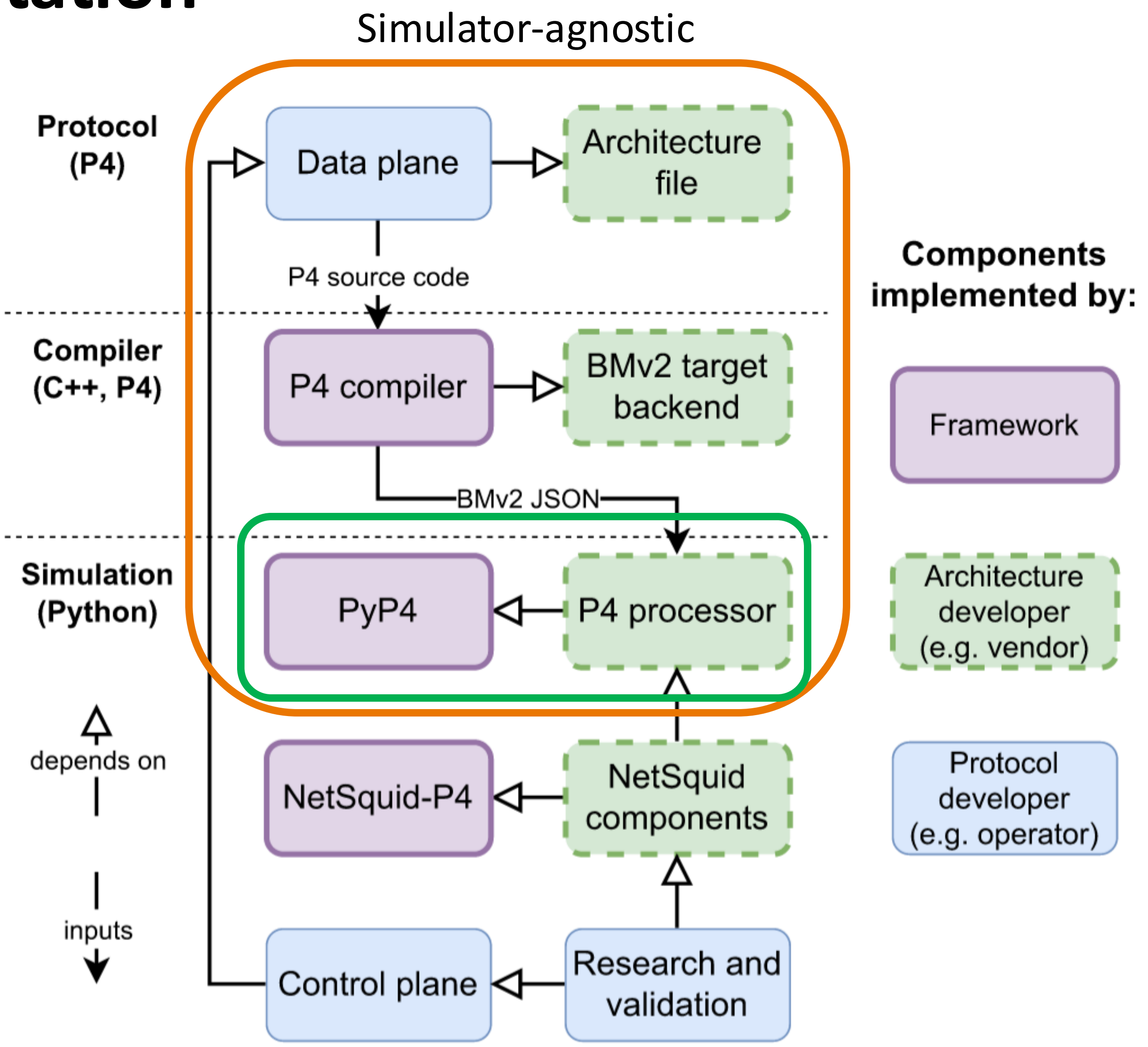
Architecture and Implementation

2. P4 Compiler / BMv2 Backend
   - We can use p4c, the open-source reference compiler.
   - A new backend must be created for a new device architecture.
   - Implementing the backend using BMv2 JSON is most suitable for software targets.
   - Adding a BMv2 backend is simple since the compiler only needs to know what blocks are available and their interfaces.



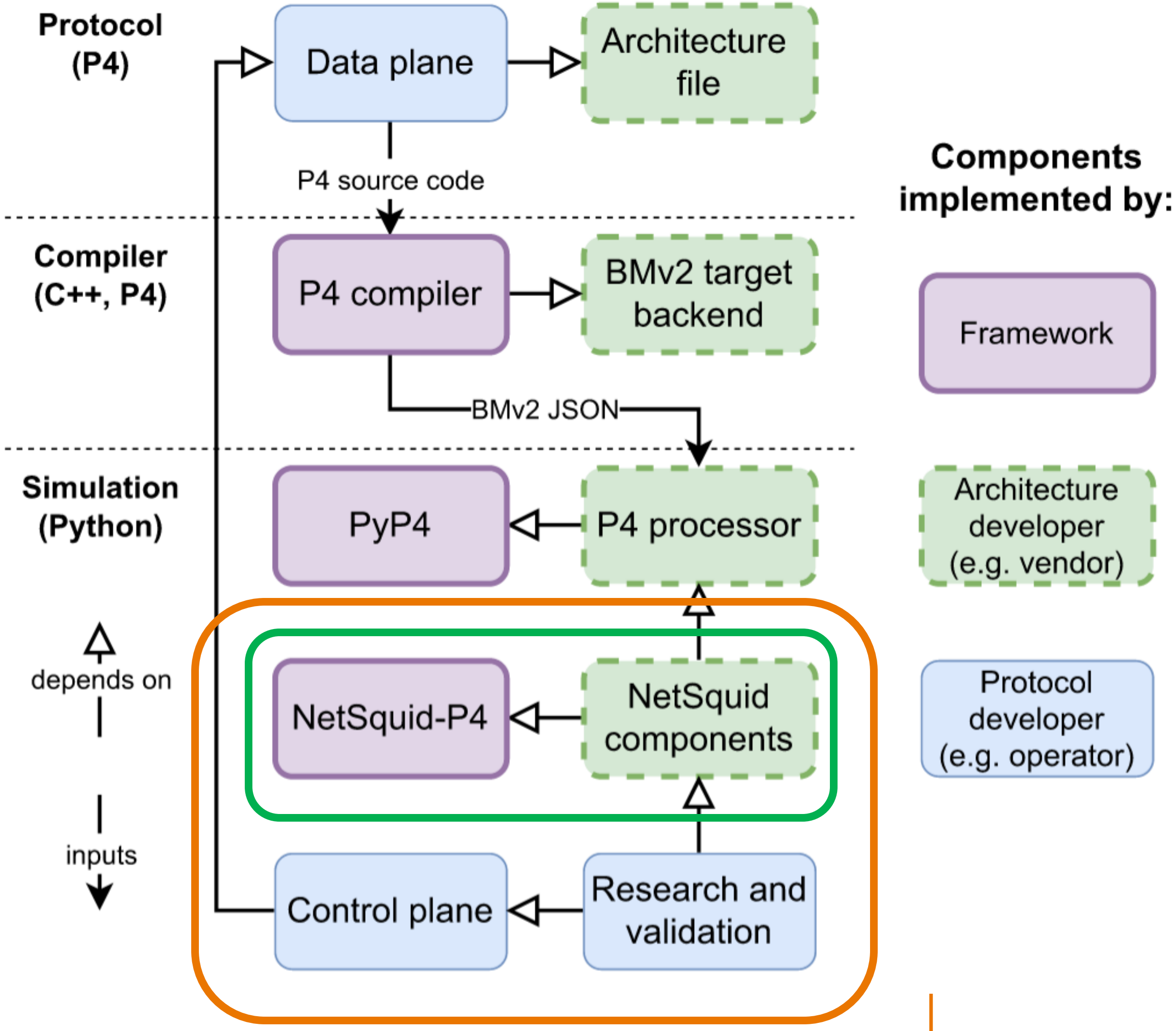
Architecture and Implementation

3. PyP4 / P4 Processor
   - Python is the de facto standard programming language in the quantum information community.
   - Many simulators expose (or are implemented in) Python interfaces, such as NetSquid or SeQUeNCe.
   - PyP4 provides a framework to implement P4 processors in Python.
   - The architecture developer must implement the fixed-function logic and extern function calls.



Architecture and Implementation

4. NetSquid-P4 / NetSquid Components
   - NetSquid-P4 provides a framework for providing NetSquid-native interfaces for PyP4 processors.
   - It is not necessary, but it reflects our expectation that it would help quantum device experts with little expertise in P4 to develop compatible simulations.

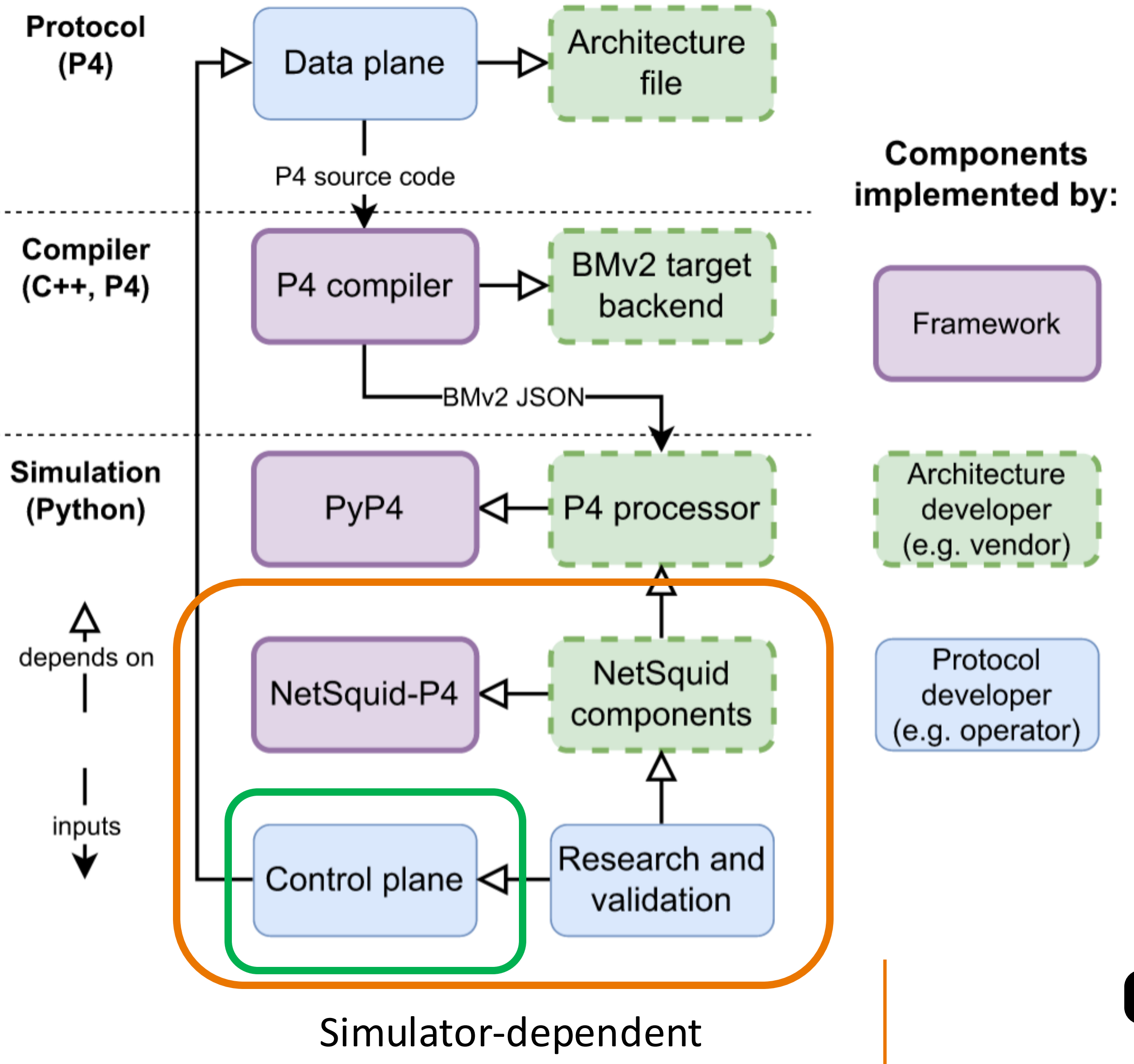


Simulator-dependent

Architecture and Implementation

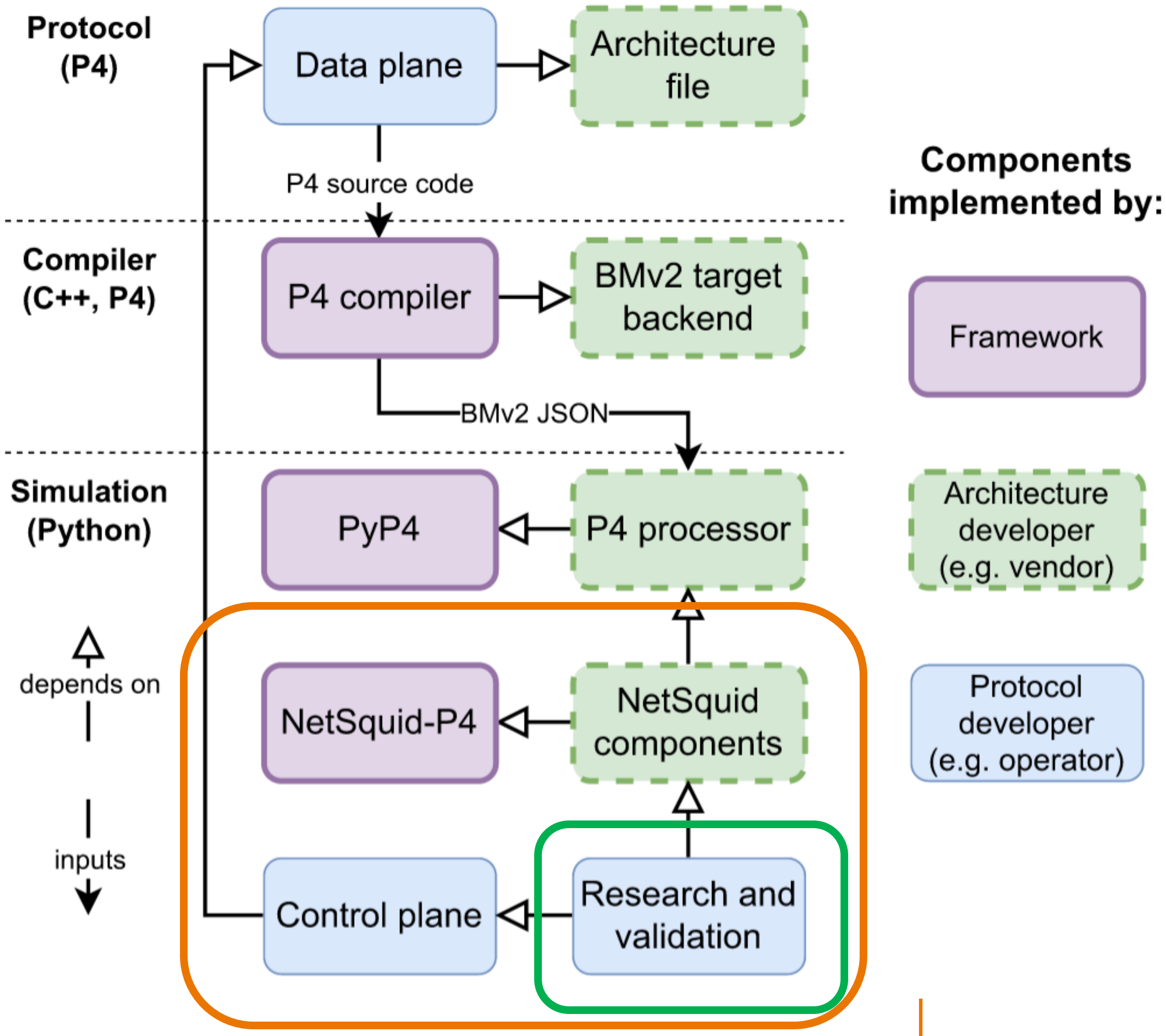
5. Control Plane

- P4 allowed us to decouple the data plane protocols from the simulator.
- The control plane could also be decoupled (using P4Runtime) but it was out of scope for our work.
- The control plane must be implemented in the simulator, but it can make use of interfaces provided by through PyP4 processor.



Architecture and Implementation

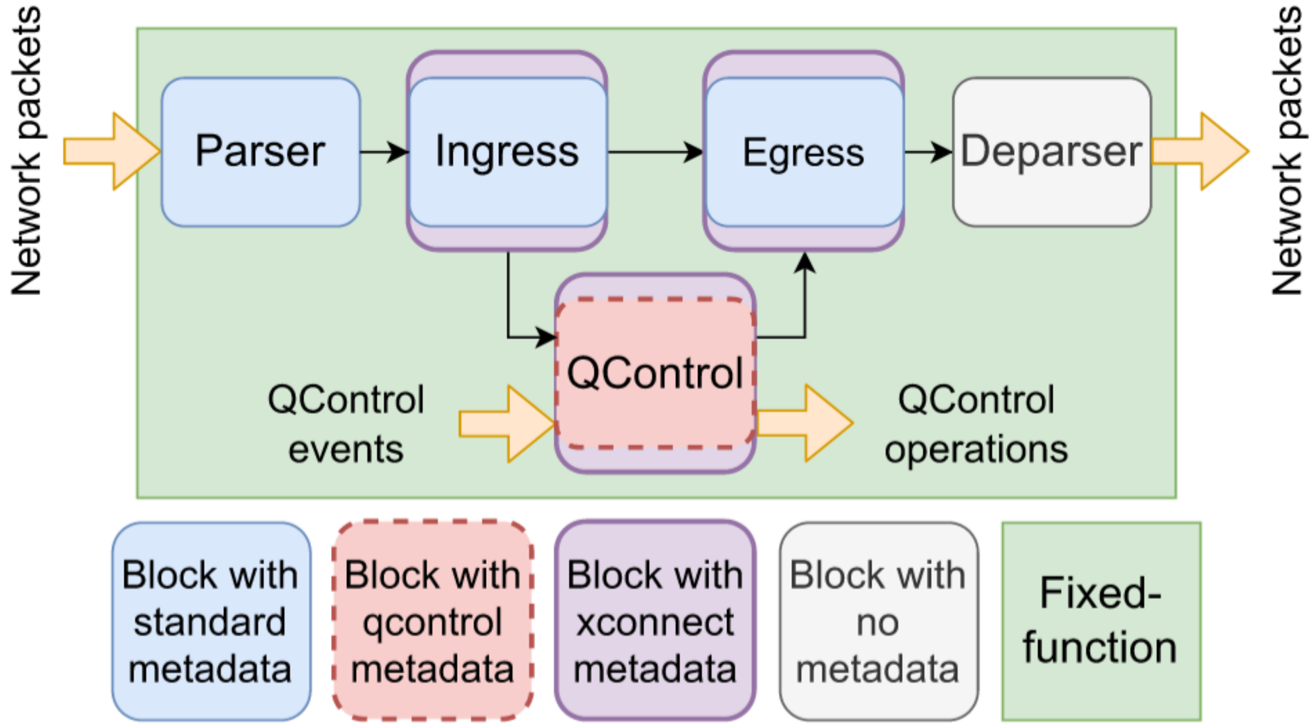
5. Research and Validation
   - The simulated model of the device must be implemented by the simulation developer.
   - The data plane protocols must be implemented by the protocol researcher.
   - While this is still a lot of work there is now a clear decoupling between the protocol and the simulation.



Simulator-dependent

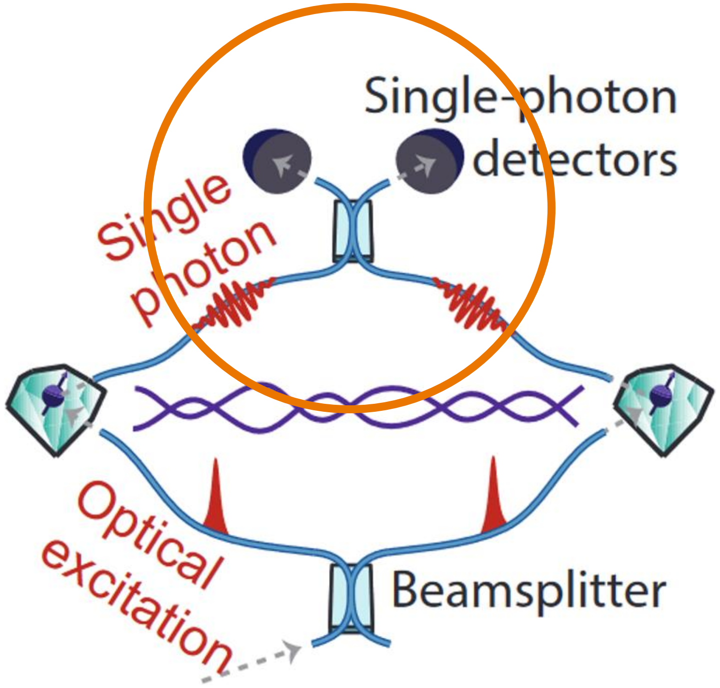
Case Study: V1Quantum

- V1Quantum is the architecture we created to address our original research problem: control planes for networks with heralding hubs.
- Basically, V1Model with extras:
  - QControl block
  - QControl events and operations
  - Cross-connect (xconnect) metadata
- We implemented V1Quantum in QuIP, implemented quantum link and network layer protocols, and studied a heralding hub in NetSquid.



V1Quantum: Heralding Hubs

- A heralding station is required to connect any pair of routers and/or end nodes.
- There is no fundamental barrier preventing us from connecting more than two nodes via reconfigurable optical switches and creating “hubs”.
- But at the time of our work, this was not considered. To the best of our knowledge all simulations were hard coded to pairs of nodes.



Deterministic delivery of remote entanglement on a quantum network
P. C. Humphreys, N. Kalb, et al.
Nature 558, pages 268-273 (2018)
arXiv:1712.07567

V1Quantum: Heralding Hubs

- A single “heralding hub” connected to multiple end nodes is easier to deploy than a quantum router connected to each end node via a dedicated heralding station.
- A “heralding hub” does not need a quantum memory making it much cheaper than a quantum router.
- A “heralding hub” is basically the architecture of MDI-QKD, a technology available today, and thus is a potential upgradeability path for entanglement.

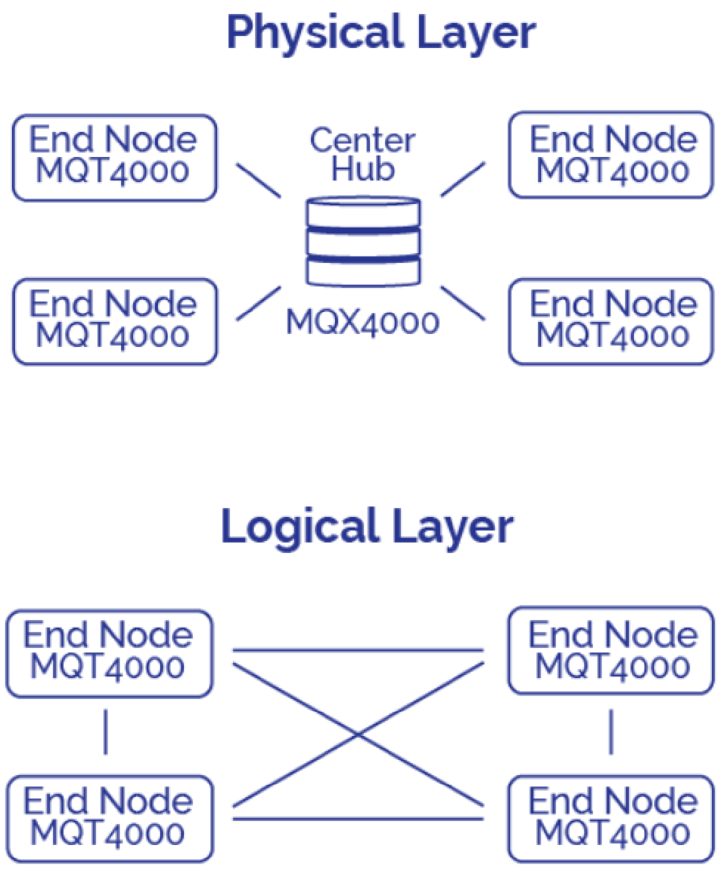


Image source: q-bird.com

V1Quantum: Quantum Network Layers

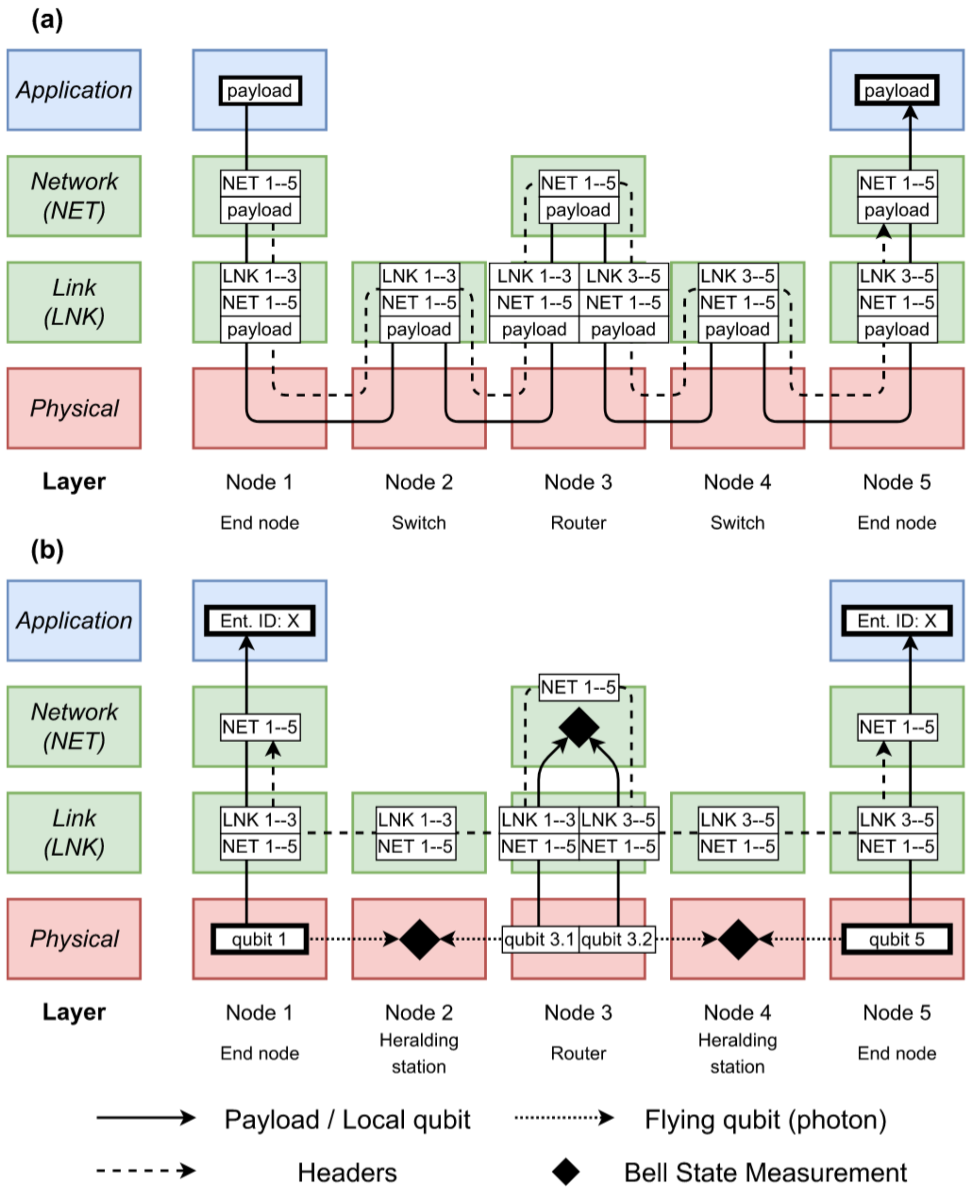
- Early lesson learned: do not try to implement all layers in P4.
- In our first attempt, we tried to implement the physical layer protocol using V1Quantum as well.
- The complexity exploded rapidly, and it is at this point where we learned to value the fixed-function aspect of P4 architectures.
- In the end, only the link and network layer protocols were done in P4.

Layer	Function	Implemention
Application	Quantum network applications	Software (NetSquid)
Transport	Qubit teleportation	Not implemented
Network (QNP)	End-to-end entanglement generation	P4 program
Link (QEGP)	Robust link entanglement generation	P4 program
Physical	Attempt entanglement generation	Fixed-function (NetSquid)

SURF

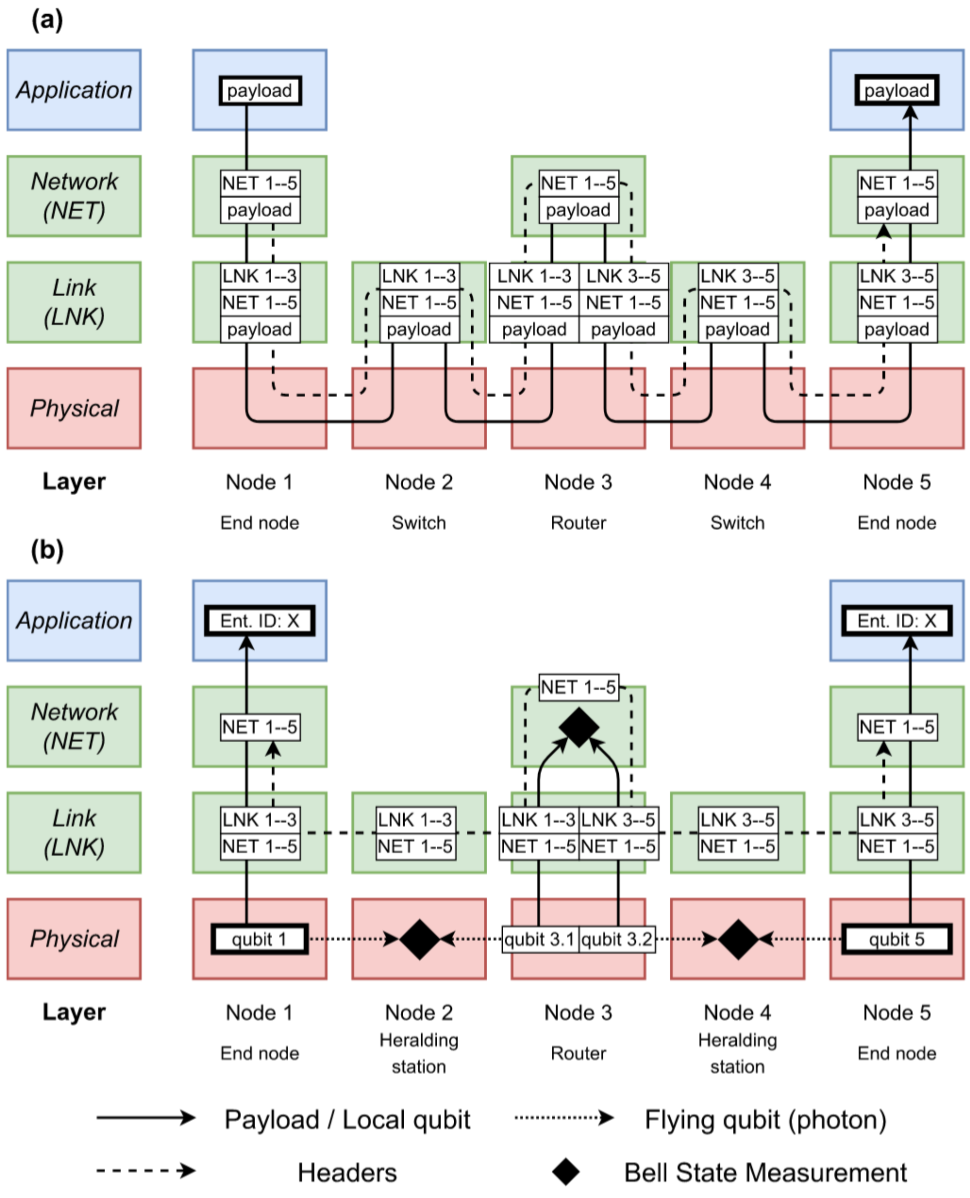
V1Quantum: Quantum Abstractions

- Another early lesson: lack of entanglement-based abstractions made development difficult in P4.
- Quantum networks also do classical signaling so we need packet constructs, but they are not enough.
- The QControl programmable block in the V1Quantum architecture allowed us to elevate entanglement to a first-class abstraction in our architecture.



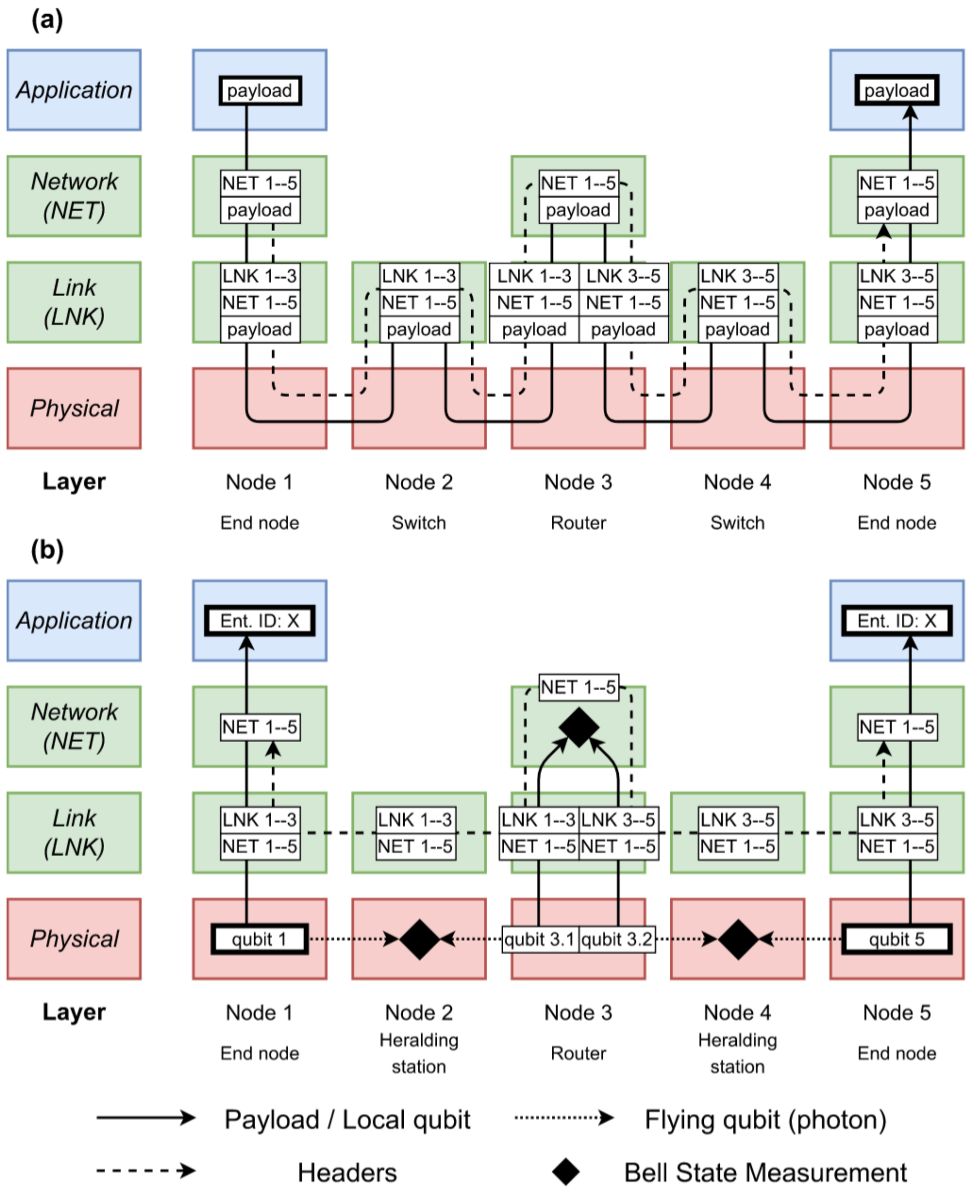
V1Quantum: Quantum Abstractions

1. Entanglement Objects:
   - A classical payload follows a path from source to destination.
   - A quantum "payload" is different – entanglement has two ends.
   - An "entanglement object" is our abstraction that allows the nodes in the network to correlate actions on the same entangled pair.



V1Quantum: Quantum Abstractions

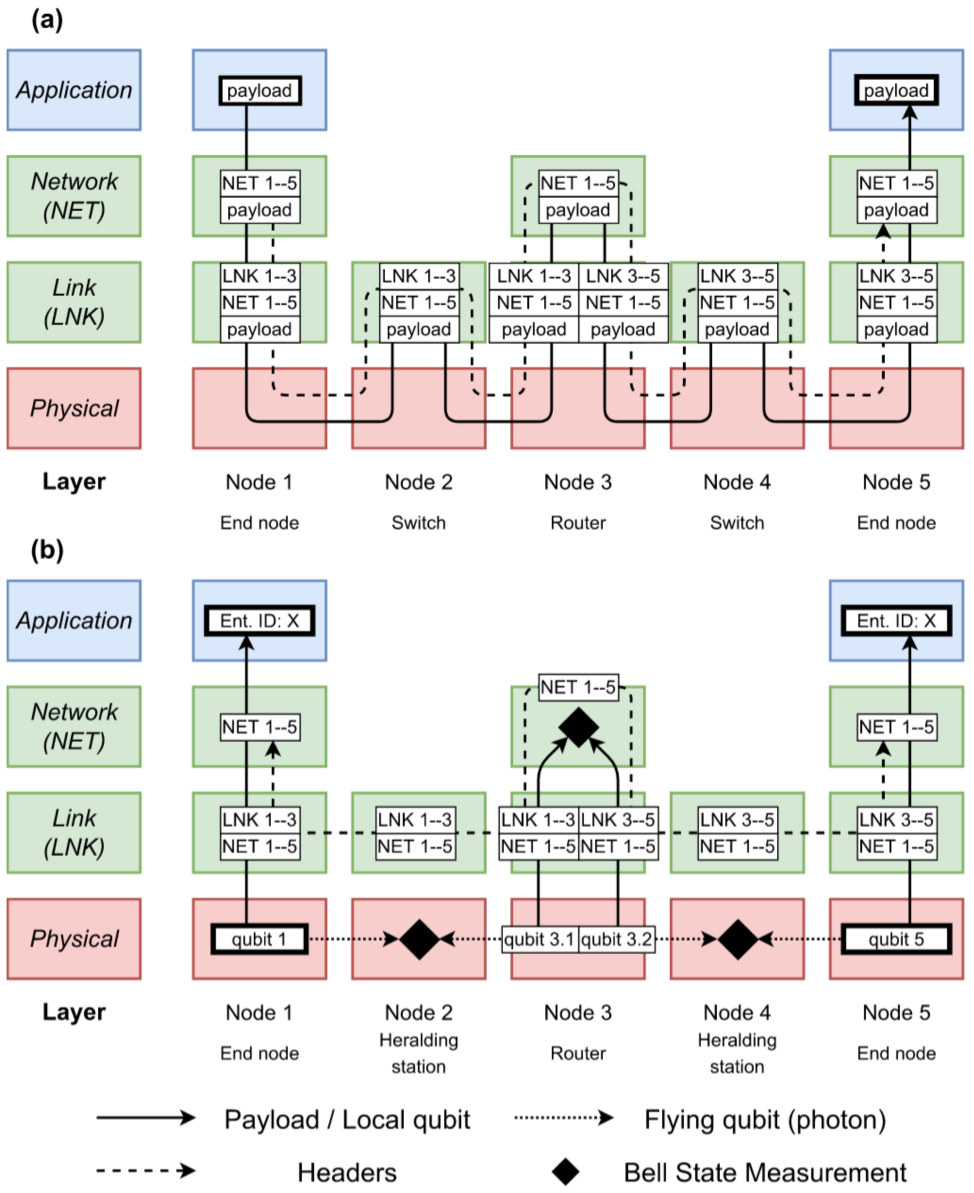
2. Bell State Measurements
   - A classical payload follows a path from source to destination.
   - A quantum "payload" is different – it does not actually travel.
   - An end-to-end entanglement object is created via a series of Bell State Measurements which "stitch" entanglement together.
   - Bell State Measurement is the same as entanglement swapping.



V1Quantum: Quantum Abstractions

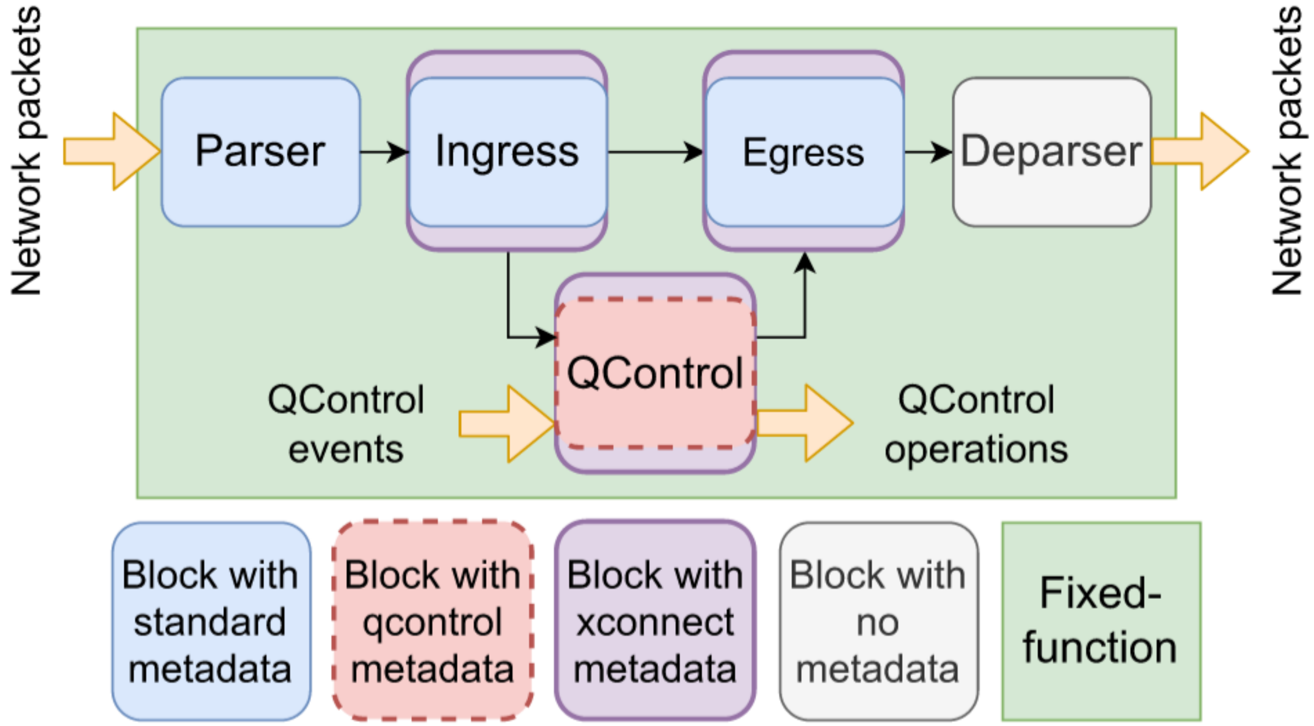
3. Classical Communication

- Classical communication is necessary to achieve coordination.
- The link layer sends “heralding” signals to “herald” successful entanglement at the physical layer.
- The network layer must trace a chain of BSMs across the network to ensure the end nodes can identify the same entanglement object.



V1Quantum: Implementation

1. QControl Programmable Block
   - QControl events: for device events such as BSM output
   - QControl operations: for program to issue instructions to the device, e.g., BSM for an entanglement swap.
   - Cross-connect path: connects classical communication with the quantum events and operations.

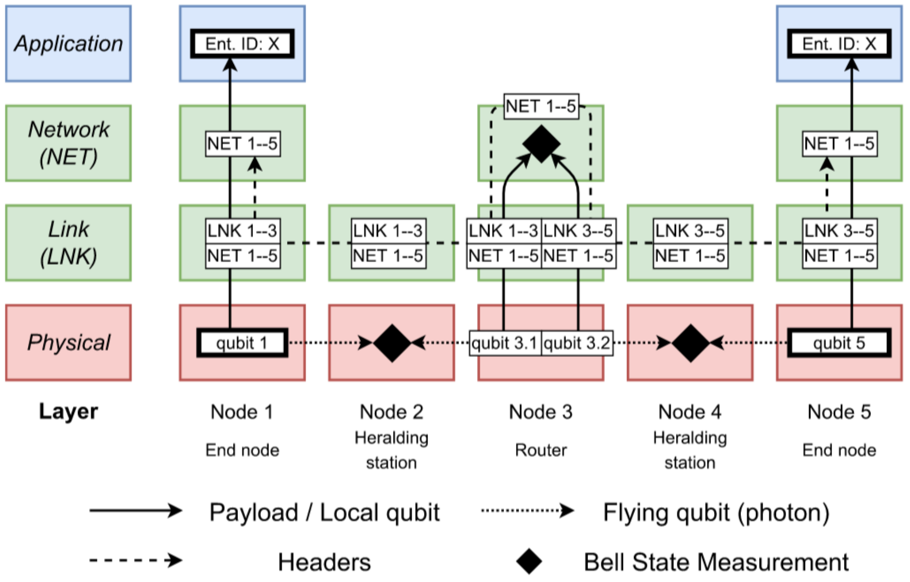


V1Quantum: Implementation

2. Entanglement Headers
   - Headers identify the two ends of entanglement at their layer:
     - Link: (link_label, pair_seq)
     - Network: (circuit_id, pair_id)
   - Headers travel independently of the "payload":
     - Classical communication
     - Travel in both directions
   - Headers collect information along the way (from entanglement swaps).

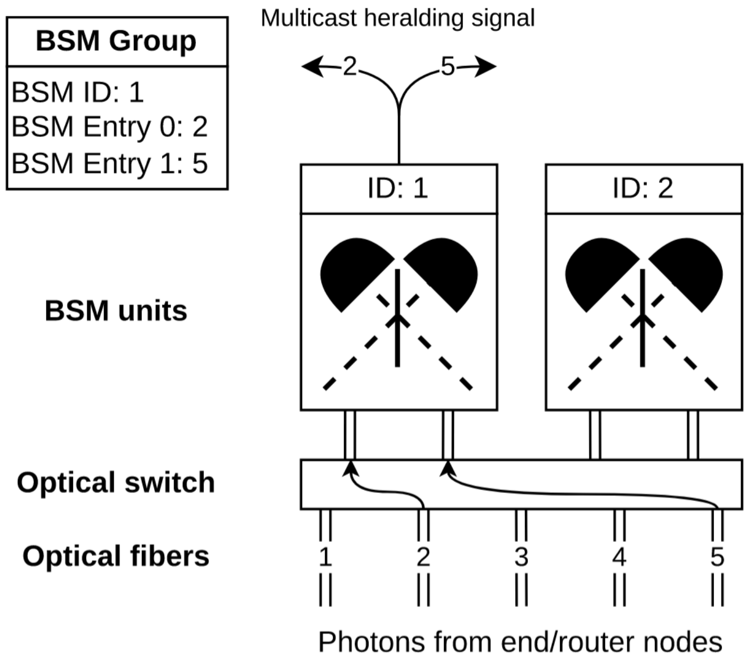
```
header egp_t {
    bit<16> link_label;
    bit<16> pair_seq;
    bit<2> bell_index;
    bit<30> _reserved;
}
```

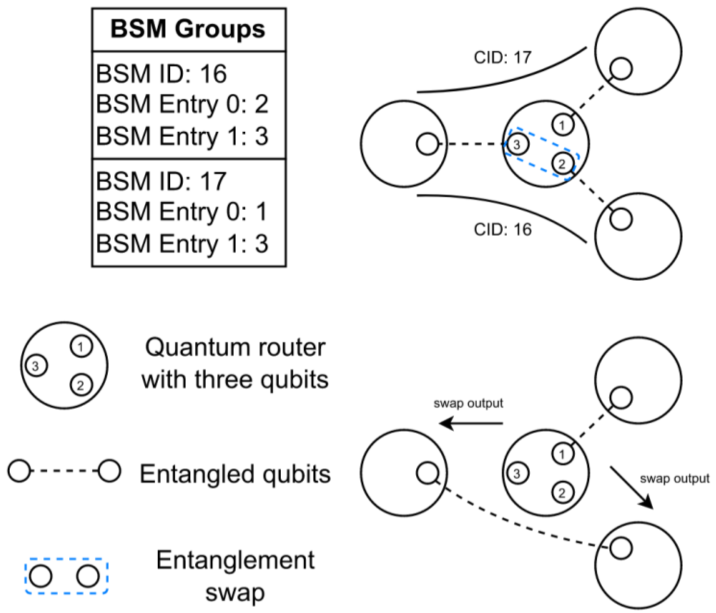
```
header qnp_t {
    bit<16> circuit_id;
    bit<16> pair_id;
    bit<2> bell_index;
    bit<30> _reserved;
}
```



V1Quantum: Implementation

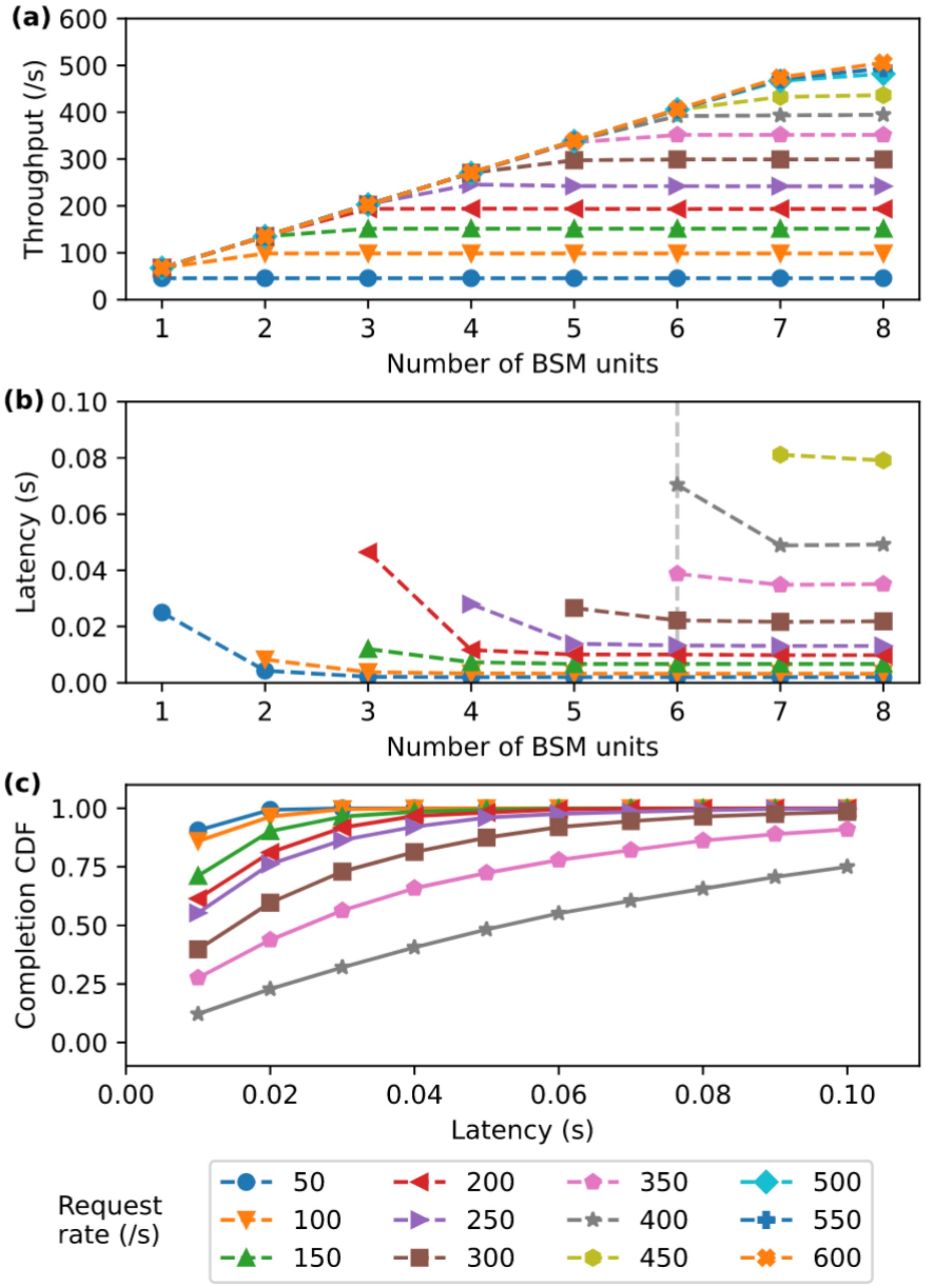
3. BSM Groups
   - The “forwarding table” of quantum entanglement networks.
   - Works for BSMs at the physical layer (heralding stations) and for entanglement swaps at routers.
   - Connects a physical event (a BSM) with a multicast group to identify where to send the headers.
   - Programmed by the control plane, BSM executed by the fixed-function, multicast group available in P4.





V1Quantum: Demo

- Hub-and-spoke topology, 16 end nodes, each 5 km of away.
- Control plane is simple FIFO queue of requests, and it will skip requests if fibres are busy and another request can be served.
- Physical parameters are based on current realistic experimental parameters.
- Demand inspired by QKD, request fixed-size blocks of entanglement at varying rates.



Conclusions

- Succeeded in showing that P4 can be used to decouple protocol implementation from simulation for quantum networks.
- Opens up design-space exploration of quantum device architectures, such as V1Quantum.
- In hindsight:
  - The framework is complex and P4 is difficult to learn by quantum experts.
  - A different, simpler DSL for quantum might help, but this would require significant dev effort, e.g., someone must implement a compiler.