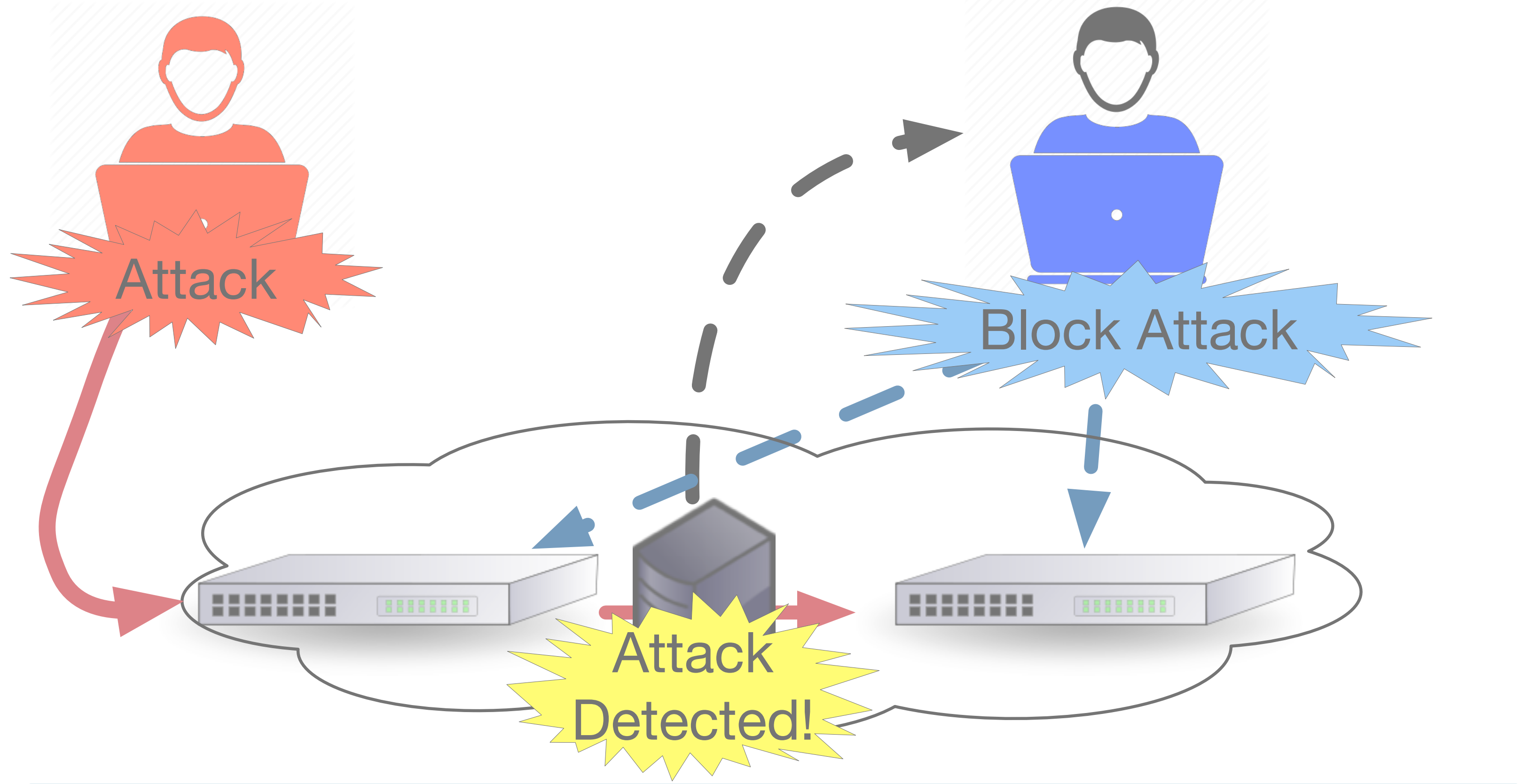
# OFX: Enabling OpenFlow Extensions for Switch-Level Security Applications

John Sonchack, Adam Aviv, Eric Keller, and Jonathan M. Smith

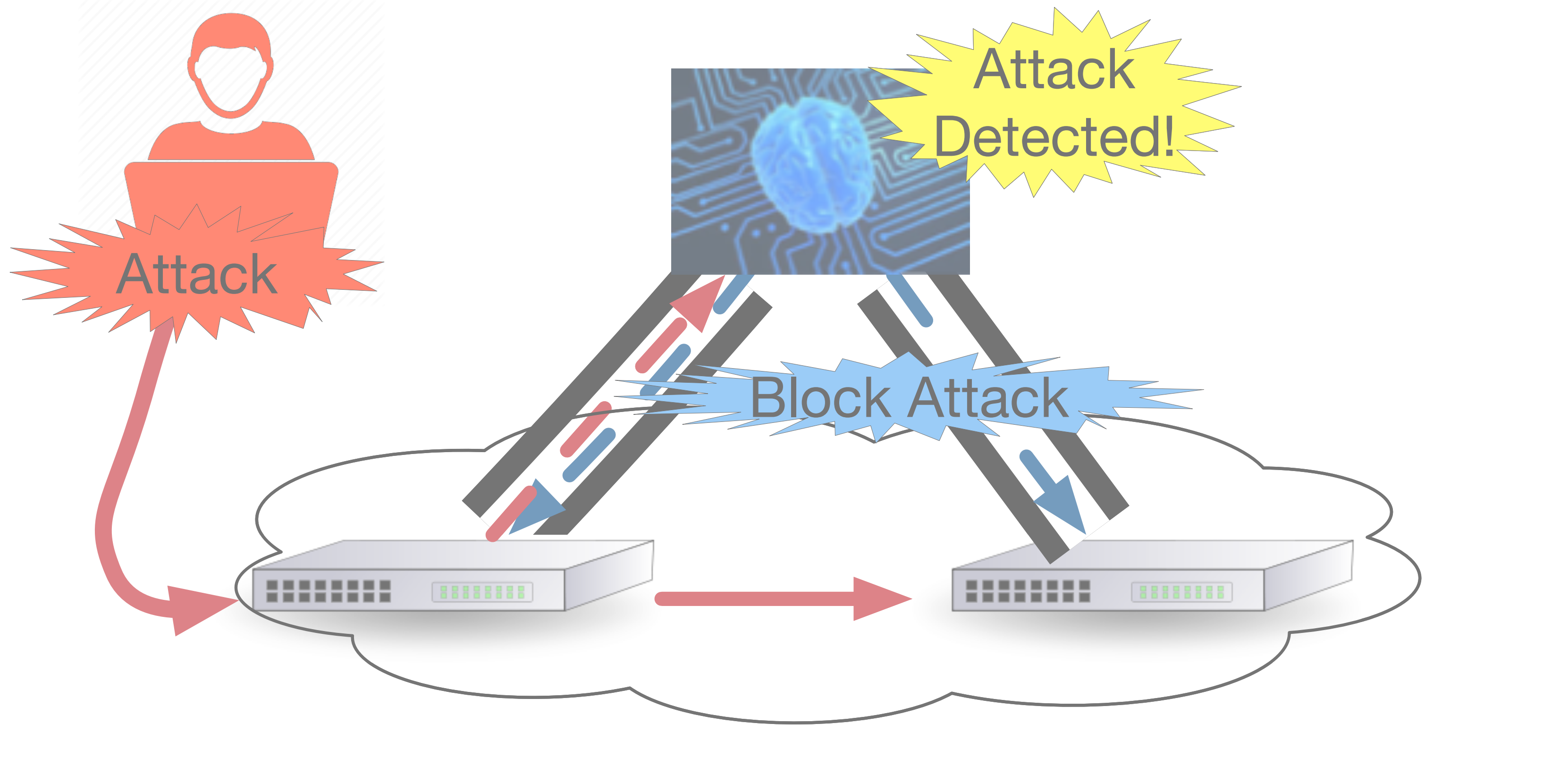
University of Pennsylvania

University of Colorado Boulder

## Existing Network Security Platforms

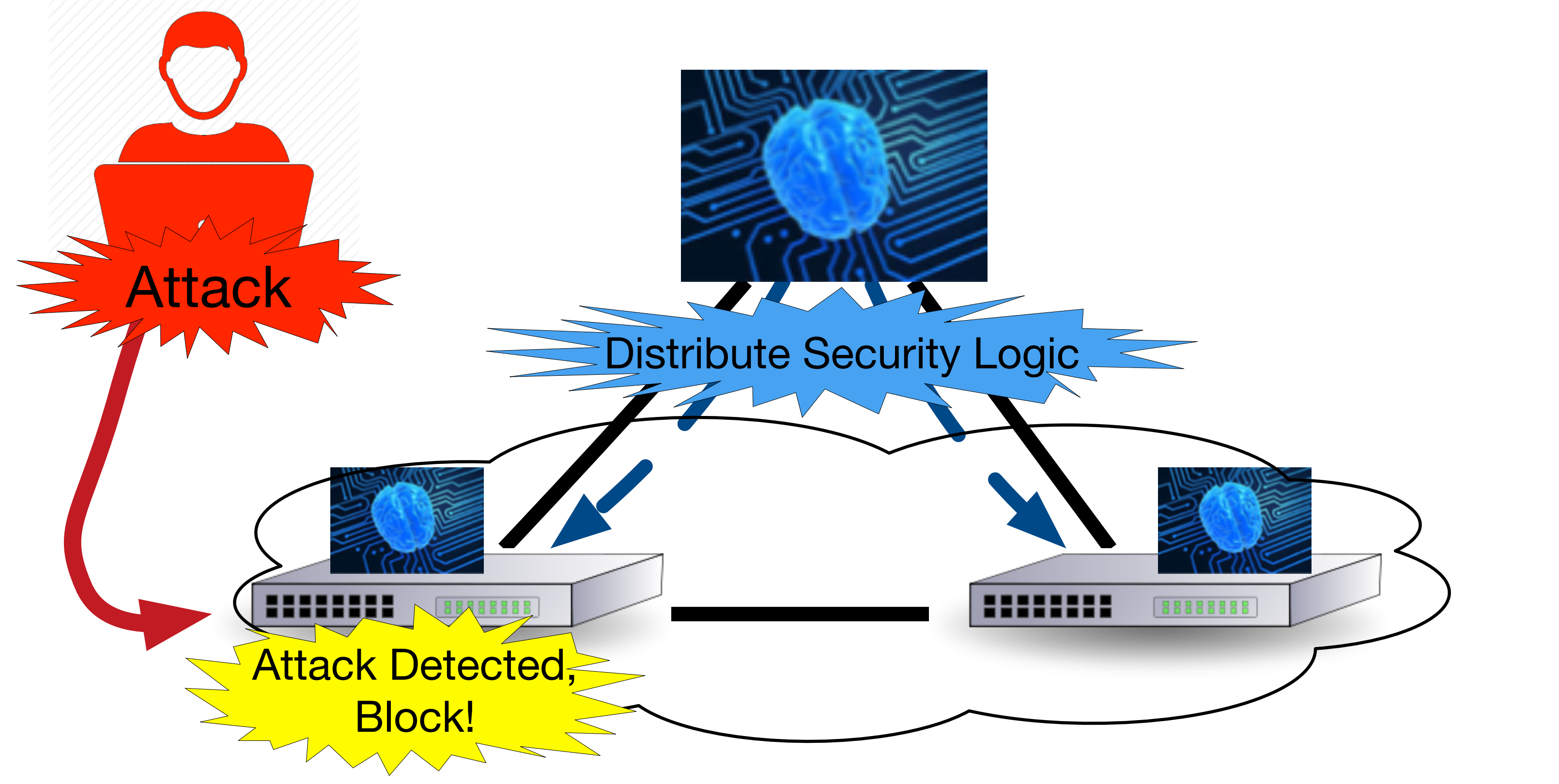


**Traditional network security applications** are deployed onto middlebox servers and have limited control over traffic or the network.



**OpenFlow security applications** can program switches but must do advanced processing and flow set up at the centralized OpenFlow controller, which limits performance and scalability.

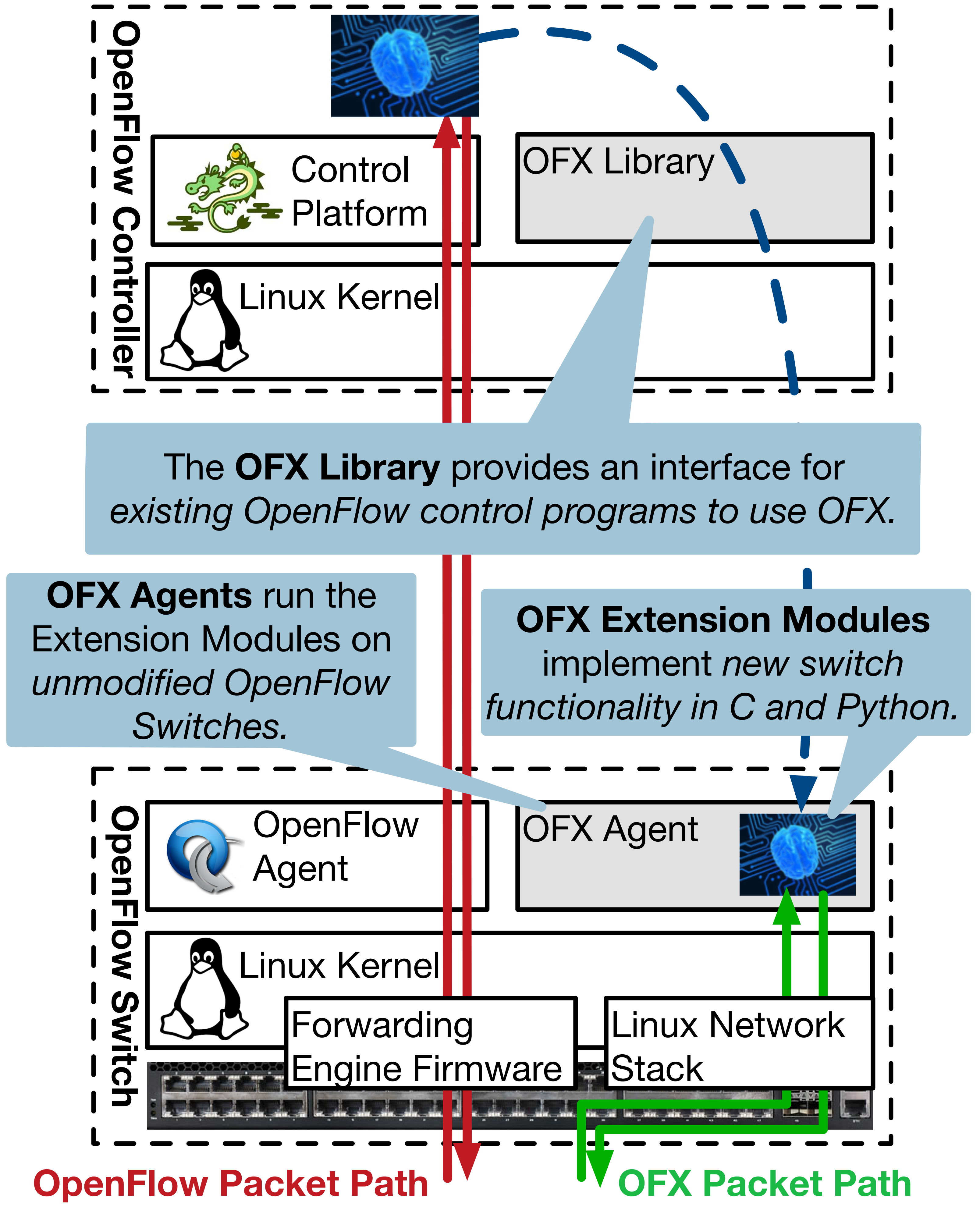
## The OFX Platform



**OFX improves OpenFlow security application performance and scalability** by allowing them to install custom software modules to process packets and set up flows ***at the switch.***

## Design

**Goals: Performance, Programmability, Deployability**



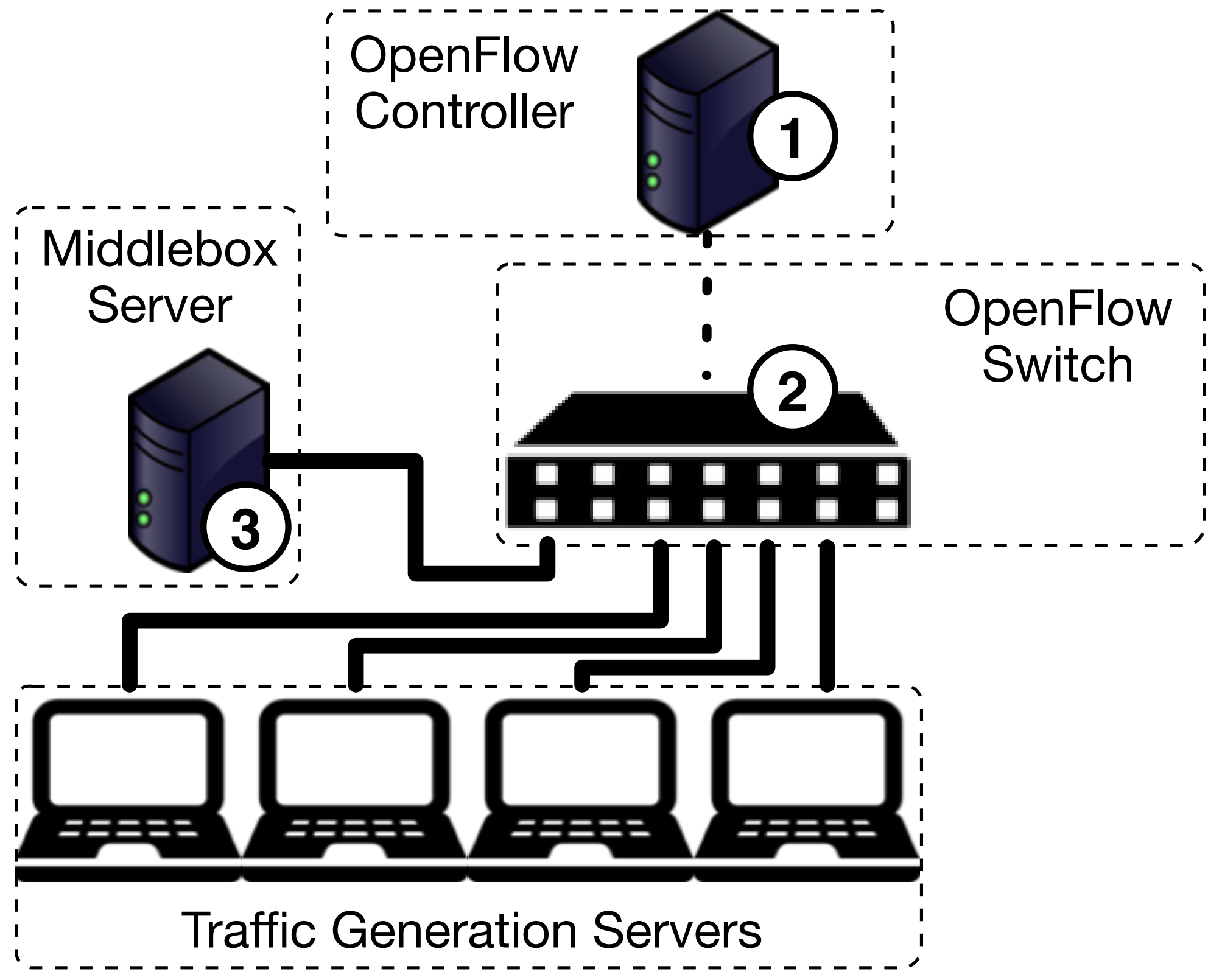
The **OFX Library** provides an interface for *existing OpenFlow control programs to use OFX.*

**OFX Agents** run the Extension Modules on *unmodified OpenFlow Switches.*

**OFX Extension Modules** implement *new switch functionality in C and Python.*

## Evaluation

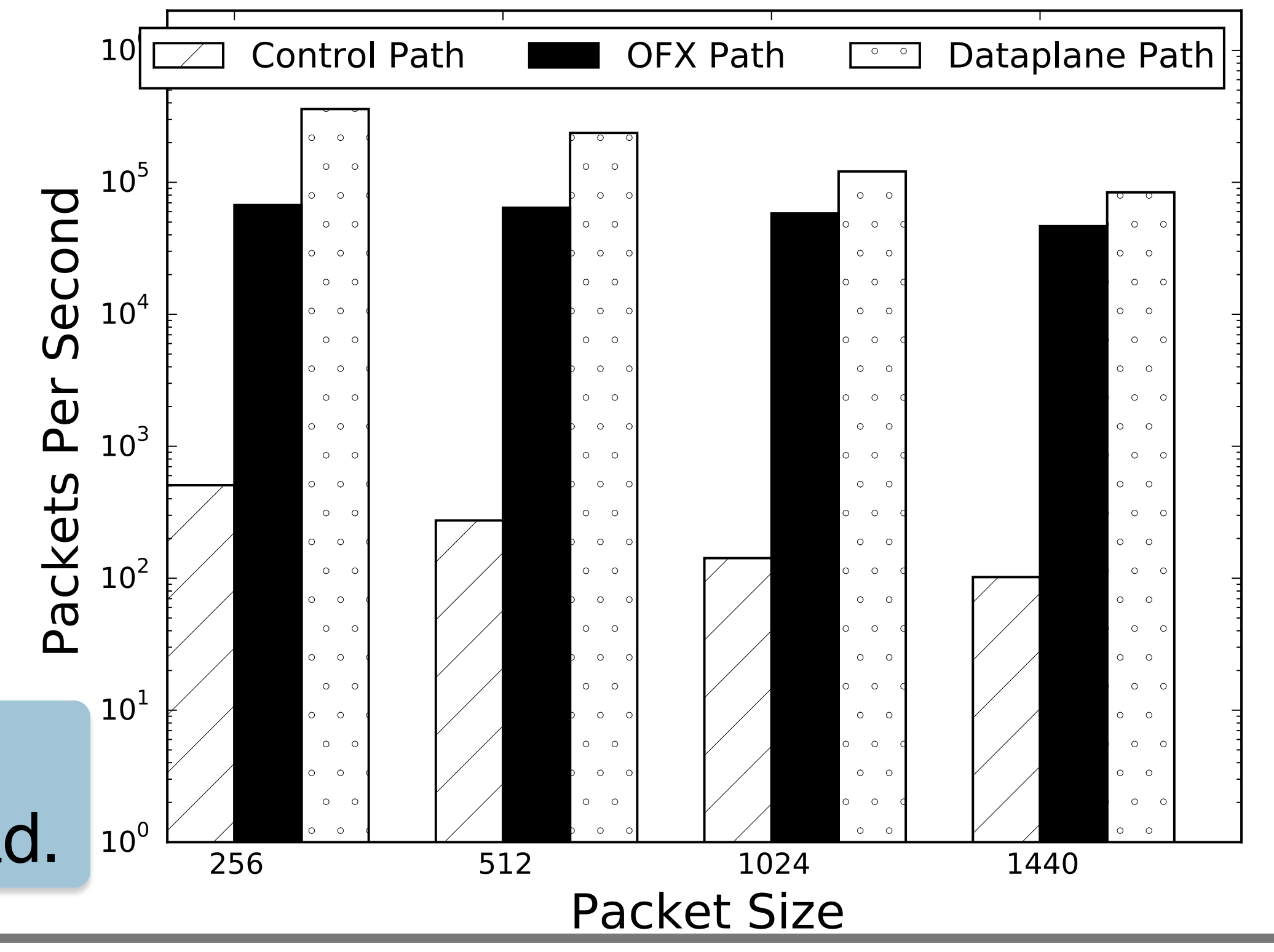
### Testbed



### *What is the raw overhead of processing packets with OFX?*

*This table and graph compare the overhead of processing traffic at an* ***OFX agent*** *and* ***OpenFlow controller****. The OpenFlow forwarding engine (i.e. data path) is a baseline.*

| Statistic | Control Path | OFX Path | Data Path |
|---|---|---|---|
| Min Latency | 3.604 ms | 0.251 ms | 0.169 ms |
| Avg Latency | 4.039 ms | 0.31 ms | 0.232 ms |
| Max latency | 8.08 ms | 0.405 ms | 0.292 ms |
| Max TCP Throughput | 1.2 Mbps | 584 Mbps | 847 Mbps |
| UDP Drop % @ 5MBPS | 72 % | 0 % | 0% |
| UDP Drop % @ 50MBPS | - | 0.13 % | 0% |
| UDP Drop % @ 500MBPS | - | 3.6% | 0% |

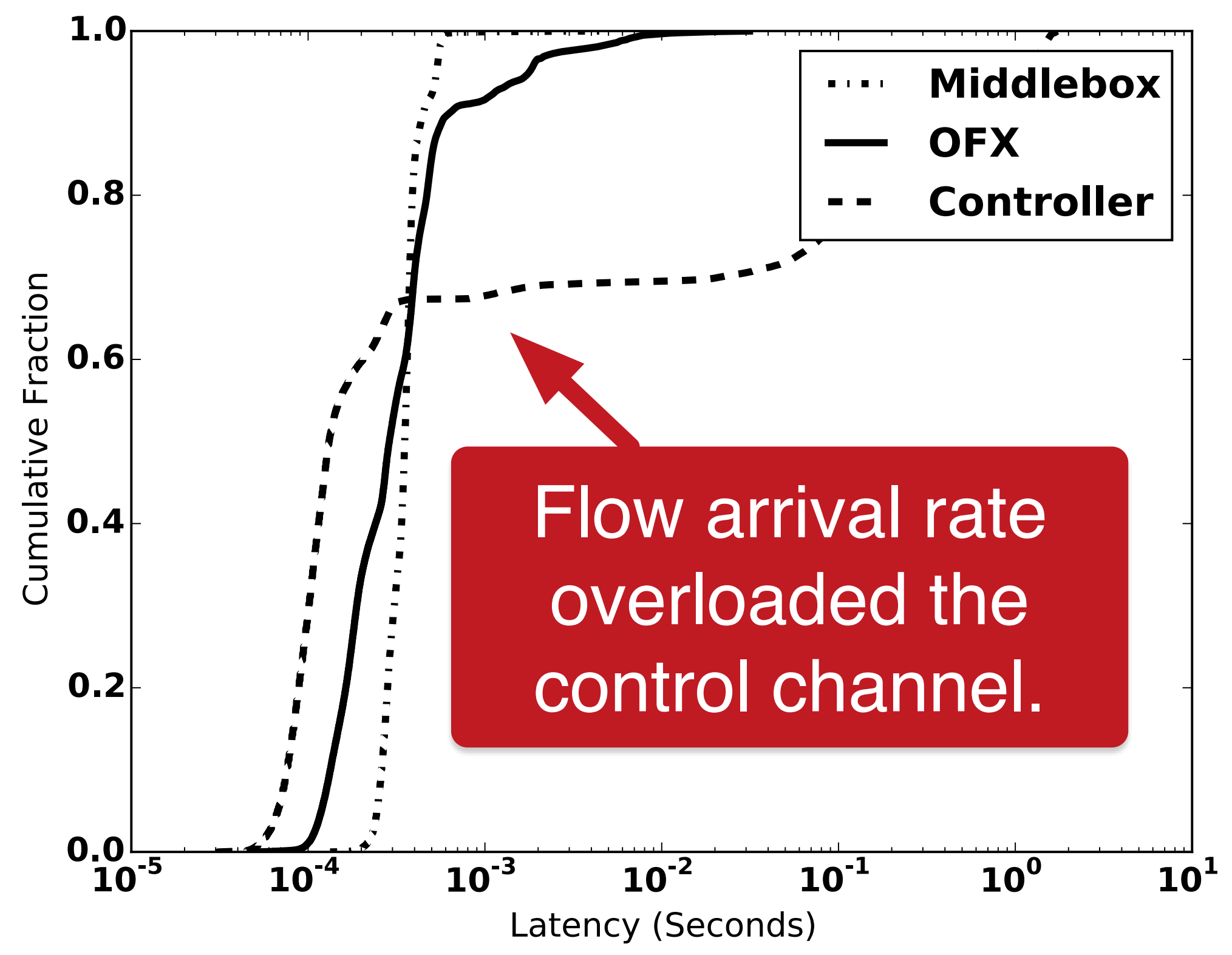


OFX overhead was 15-500x lower than controller processing overhead.

### *How do OFX security applications perform?*

*The plots below show the distribution of latency added to packets by OFX, OpenFlow, and middlebox implementations of a traffic declassifier based on SilverLine.*

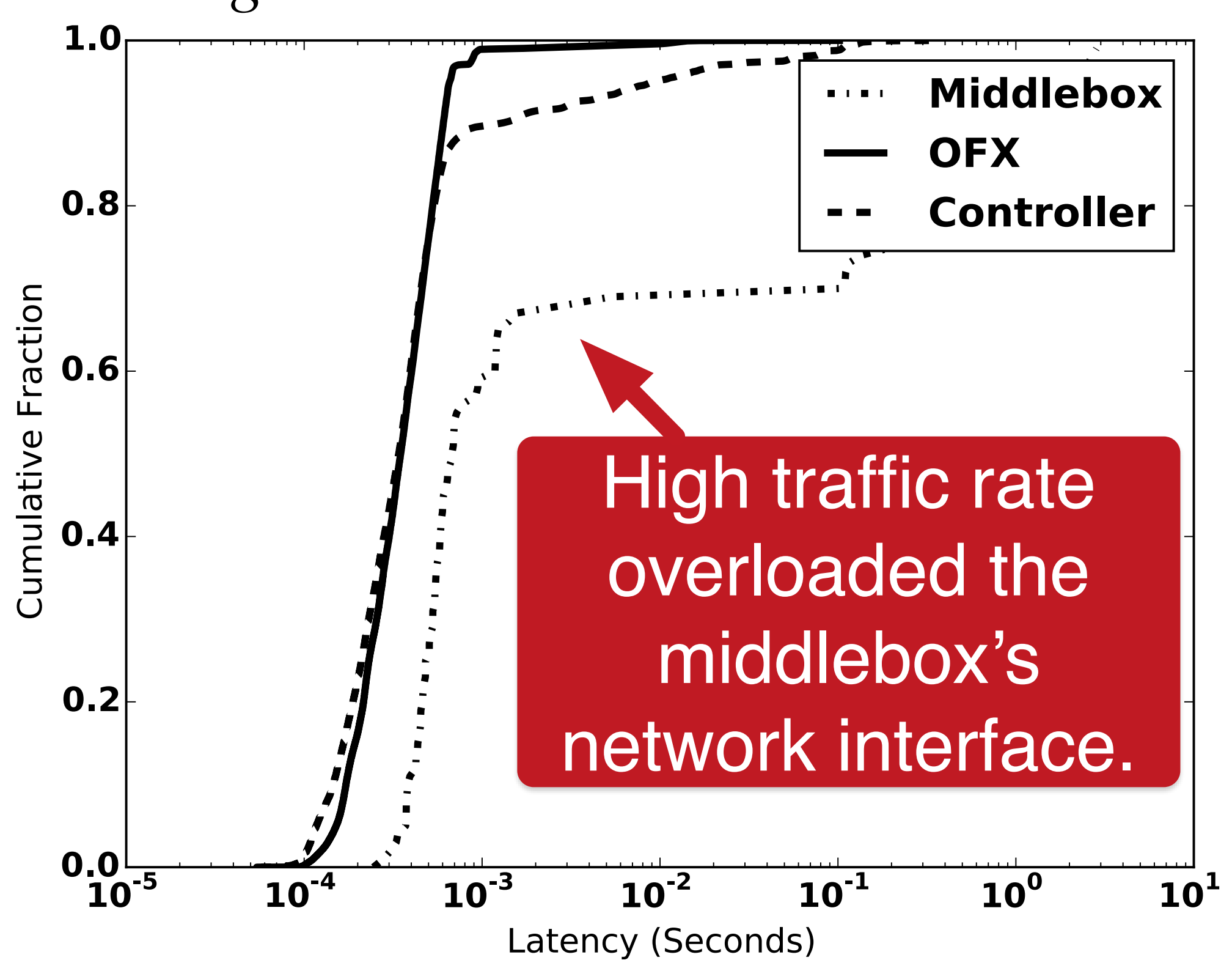
“Median Flow Interarrival” Workload



Flow arrival rate overloaded the control channel.

The OFX implementation of the declassifier added the least amount of average latency and performed most consistently across workloads.

“High Bandwidth Flows” Workload



High traffic rate overloaded the middlebox's network interface.