USC Viterbi School of Engineering

A Secure Computation Framework for SDNs

University of Maryland 1856 — A. James Clark School of Engineering

Nachikethas A.J   Ranjan Pal   Kaushik N
Yan Huang   Elaine Shi   Minlan Yu

What:

- A novel approach for provably secure computation for multi-controller architecture in SDN.
- Techniques from Secure Multi- Party Computation (SMPC) are used to address security and fault-tolerance concerns of SDN applications.
- Provide a secure framework for SDN applications running on multiple controllers.

Why:

- Controllers can become high-value and attractive targets for an adversary.
- Malicious insiders may leak sensitive information or sabotage network operations.
- Compromised controllers can affect the results of the computational task.

How:

- Consider a network managed by two controllers C_1 and C_2. Let x_1 and x_2 be their inputs. Our goal is to compute $y = f(x_1, x_2)$ such that each controller learns only y and is ignorant of the input of the other.
- SMPC provides solution to this problem and when applied to multi-controller architecture in SDN improves security:

✓ When a subset of the controllers are compromised, no sensitive information such as network topology is leaked.

✓ The network's resilience to controller failure is improved.

- Switches send secret shares of sensitive data to the controllers.
- Any coalition of *t* controllers or smaller learns no information about the sensitive data (other than the outcome of the secure computation).
- As a proof of concept, we implemented a secure randomized algorithm with low overhead, for identifying heavy hitters in a network.

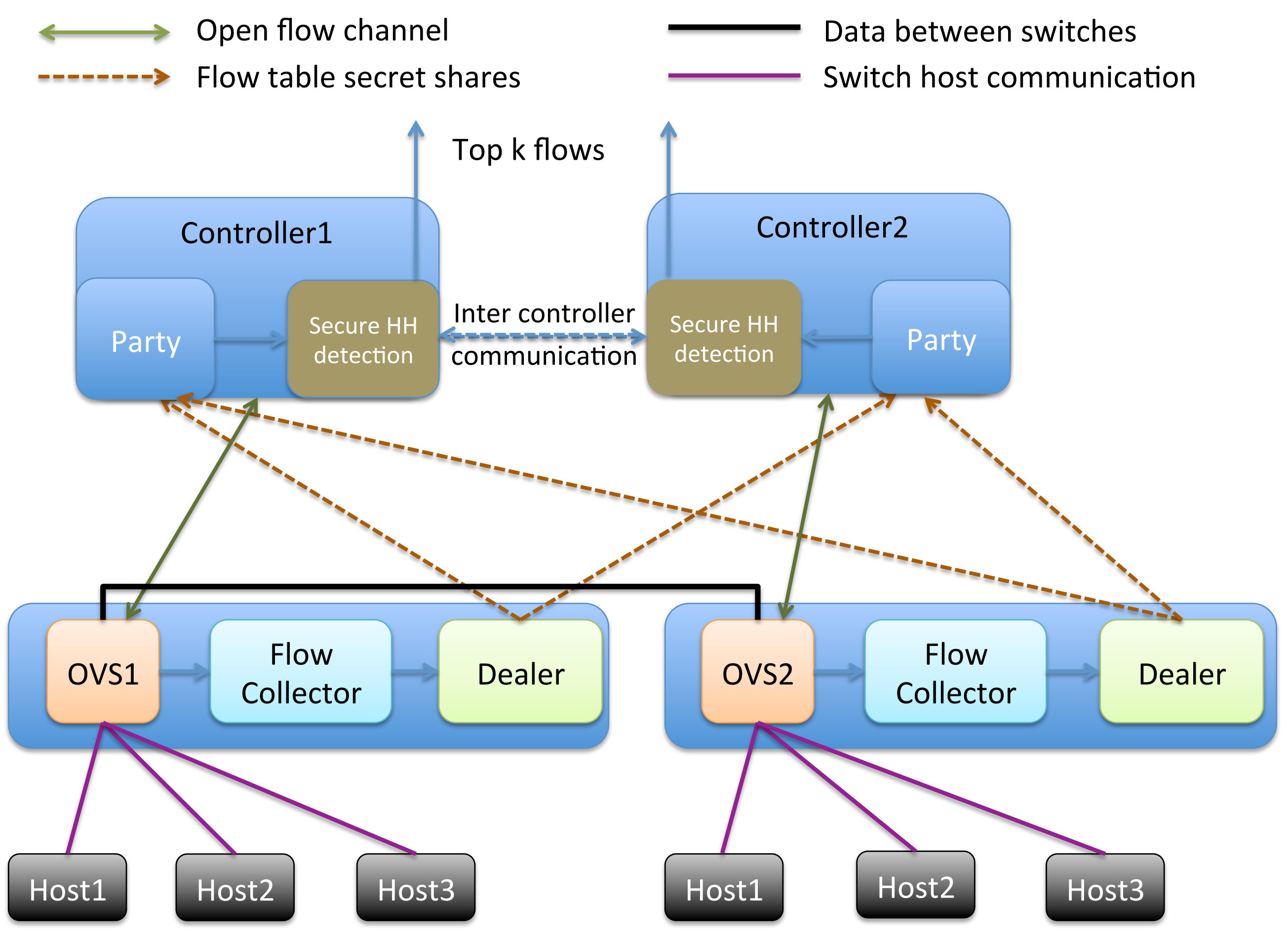
Case Study : Heavy Hitter Detection:

- We define heavy hitters as the top-k sources that send traffic to the network.
- At each switch the dealer splits the flow table entries into secret shares which are distributed among the controllers.
- Using these shares the controllers engage in a SMPC protocol to identify the heavy hitters.
- As a proof of concept we implement this application for a SDN consisting of two controllers.

Future:

- Improve the security *vs.* performance tradeoff.
- Increase support for network operations.

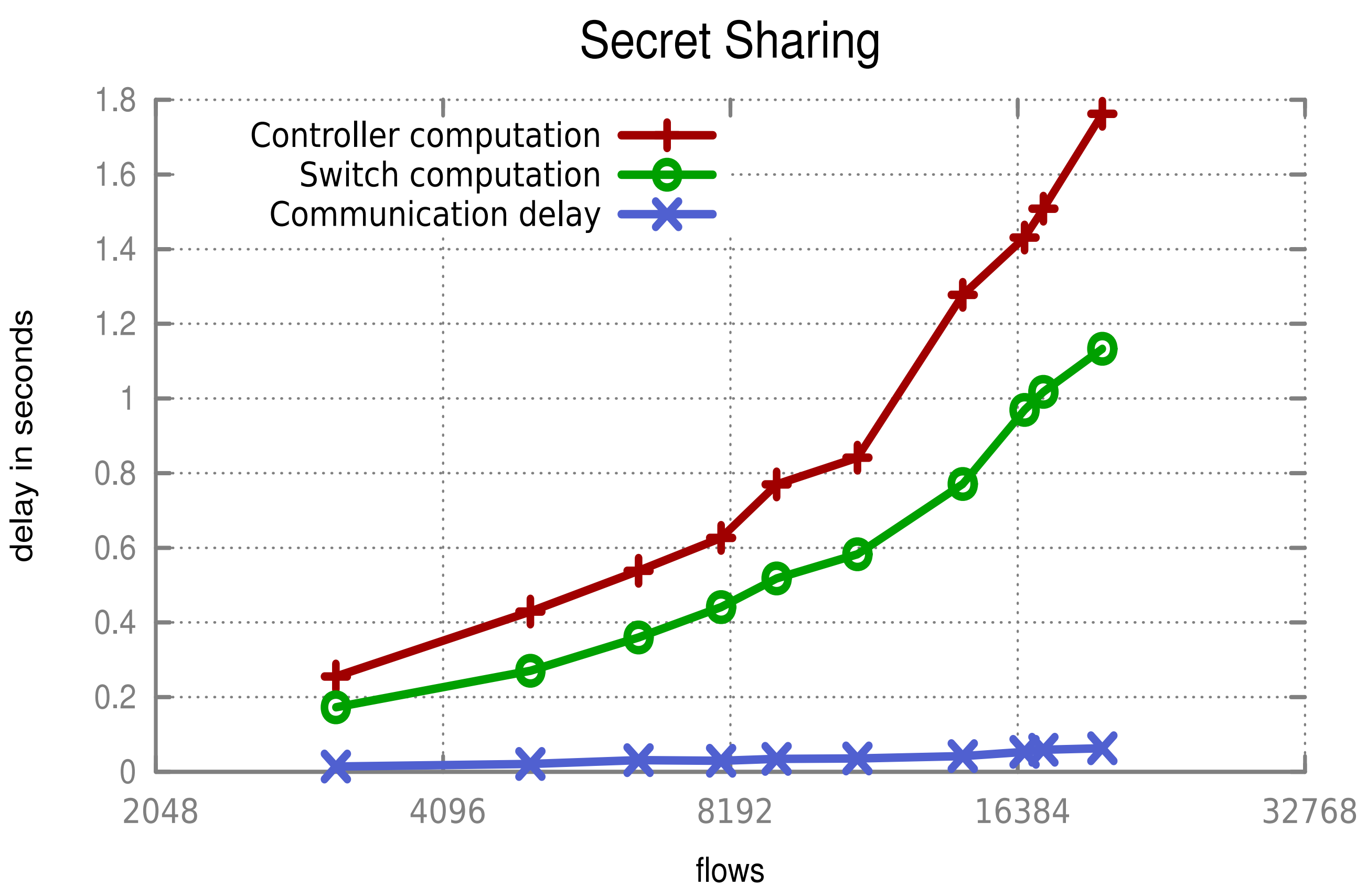
Architecture

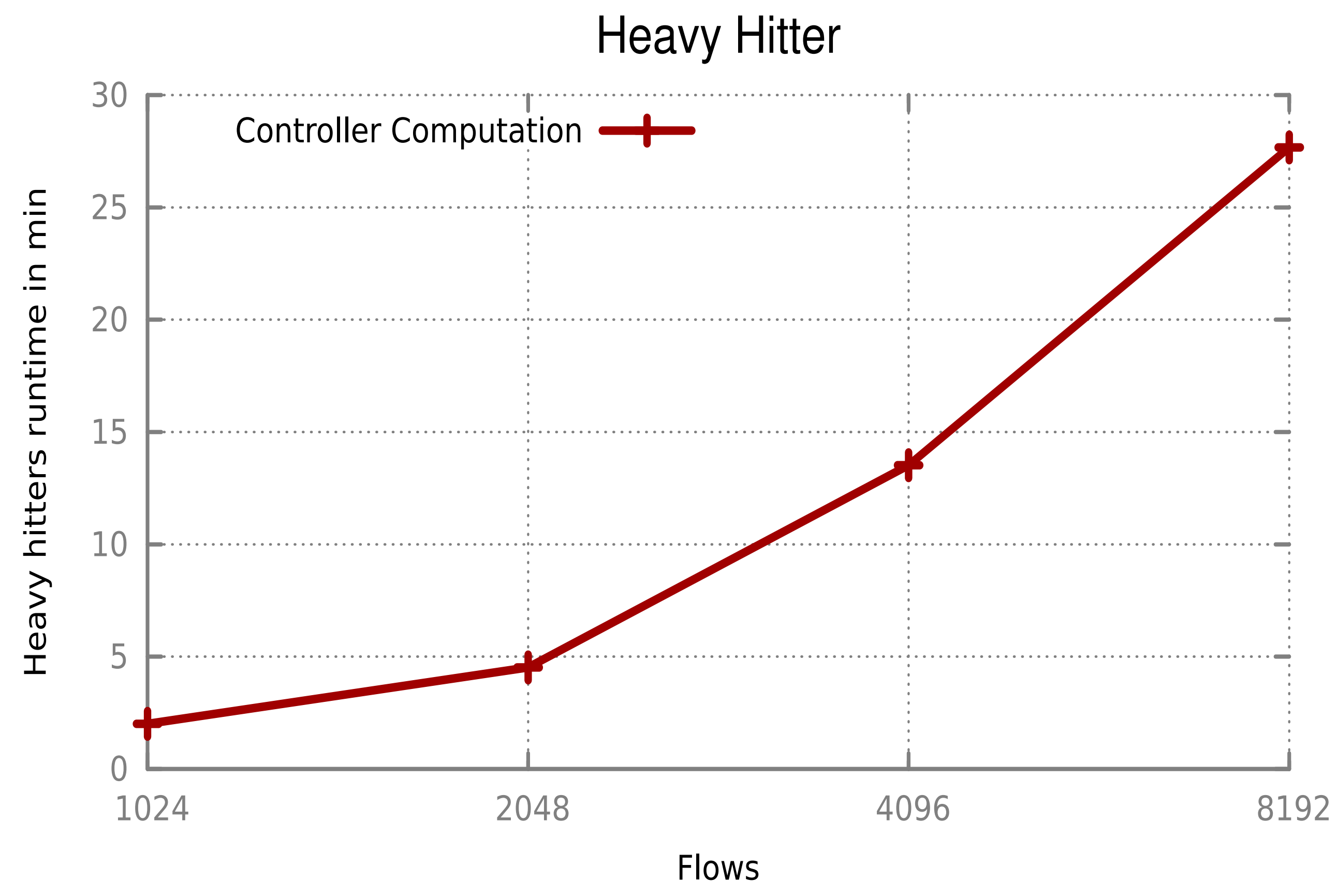


Heavy Hitter Detection Algorithm

```
Data: Stream1, Stream2
Result: The top $k$ flows
Stream = (Stream1,Stream2);
Stream = ObliviousSort(Stream);
for $i \leftarrow 1$ to length(Stream) do
    if Stream[i].IP == Stream[i+1].IP then
        Stream[i+1].nPackets +=
        Stream[i].nPackets;
        Stream[i].nPackets = 0 ;
    end
end
Stream = ObliviousSort(Stream);
Print(Top $k$ records in Stream);
```

Results


Secret Sharing
Controller computation
Switch computation
Communication delay
delay in seconds
flows


Heavy Hitter
Controller Computation
Heavy hitters runtime in min
Flows