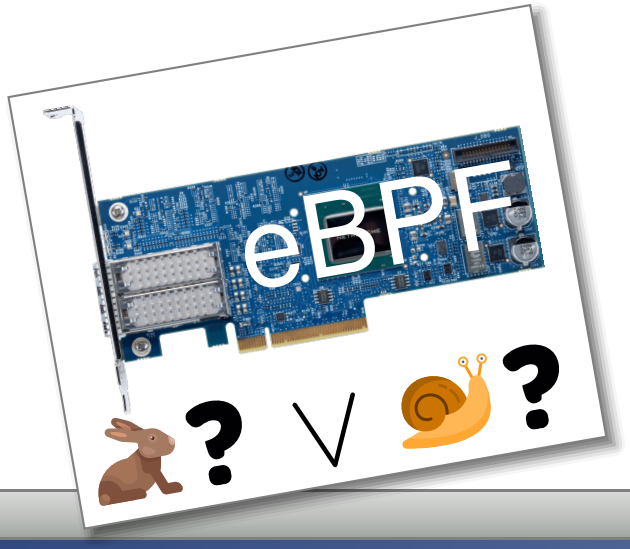


Demystifying the Performance of XDP BPF

Oliver Hohlfeld[2],
Johannes Krude[1], Jens Helge Reelfs[1],
Jan Rüth[1], Klaus Wehrle[1]

[1]RWTH Aachen University, [2]Brandenburg University of Technology

DFG Collaborative Research Centre 1053 – MAKI
Multi Mechanism Adaptation for the Future Internet

https://comsys.rwth-aachen.de/

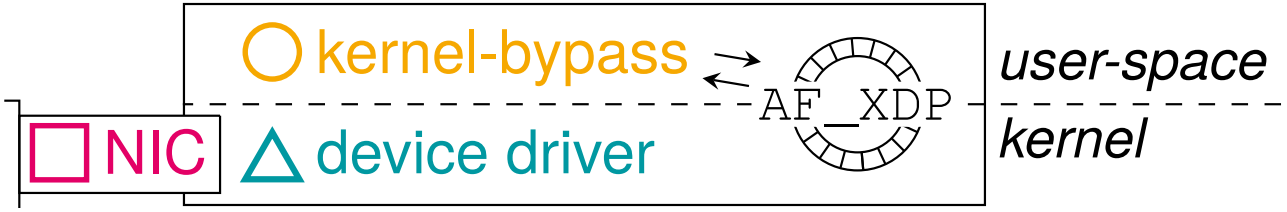
NetSoft 2019, 2019-06-25

Offloading Packet Processing

- $\geq 10\,\mathrm{GBit/s}$ **challenge network stacks**
- **Common solution: Offloading from user-space to**
  - CPU based SmartNIC
  - Device driver
  - Host OS in case of virtualization

Offloading Packet Processing

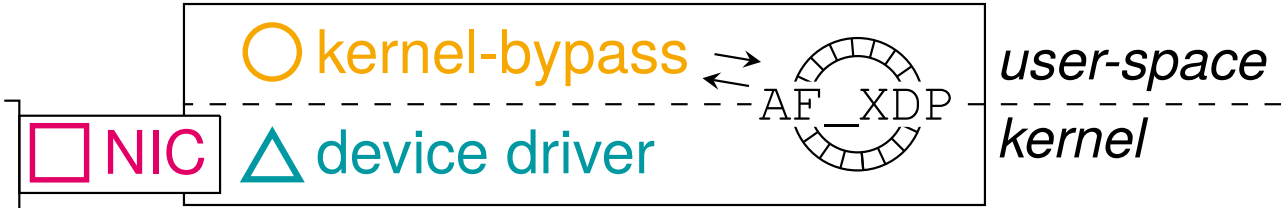
- $\geq 10\,\mathrm{GBit/s}$ **challenge network stacks**
- **Common solution: Offloading from user-space to**
  - CPU based SmartNIC
  - Device driver
  - Host OS in case of virtualization



- **XDP & eBPF**
  - Safe code execution with native performance
  - Executed within the device-driver or on the NIC
  - Can steer packets directly to user-space, bypassing most of the stack

Offloading Packet Processing

- $\geq 10\,\mathrm{GBit/s}$ **challenge network stacks**
- **Common solution: Offloading from user-space to**
  - CPU based SmartNIC
  - Device driver
  - Host OS in case of virtualization



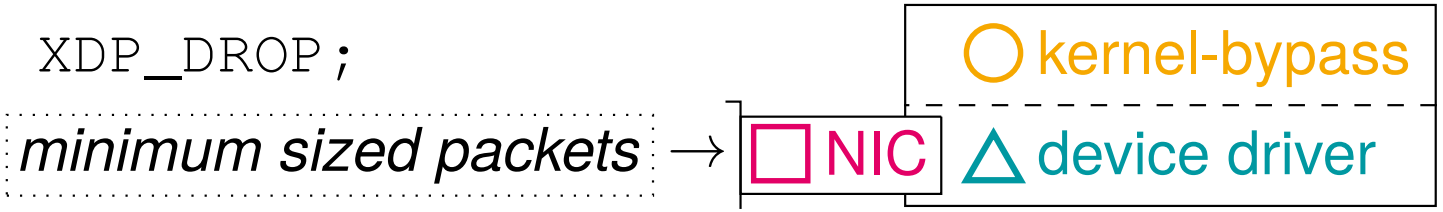
- **XDP & eBPF**
  - Safe code execution with native performance
  - Executed within the device-driver or on the NIC
  - Can steer packets directly to user-space, bypassing most of the stack

This Paper: Comparison of same eBPF Programs at □, △ & ○

- **Netronome Agilio CX 2x10GbE & Linux 4.18.10 on Core i7-7700**

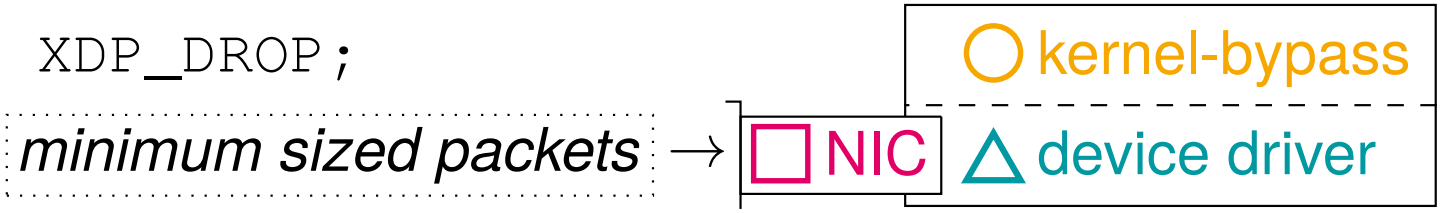
Throughput Baseline: Drop All

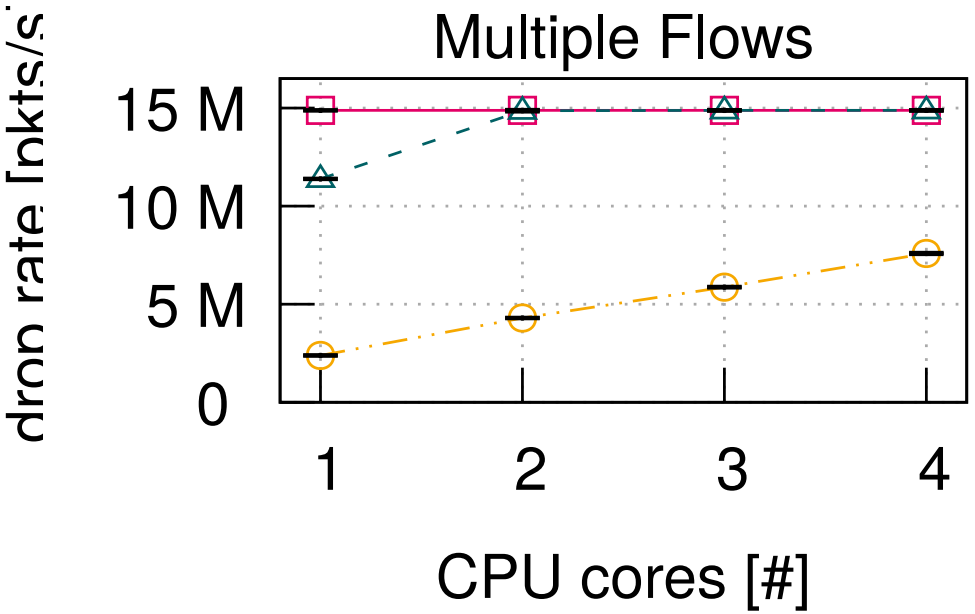
Program: `return XDP_DROP;`



Throughput Baseline: Drop All

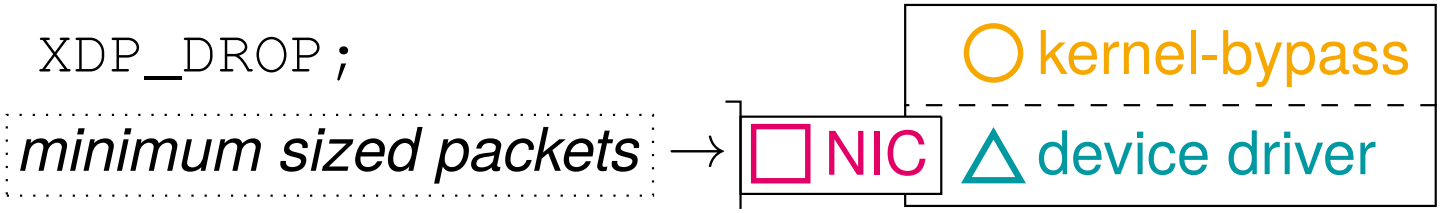
Program: `return XDP_DROP;`

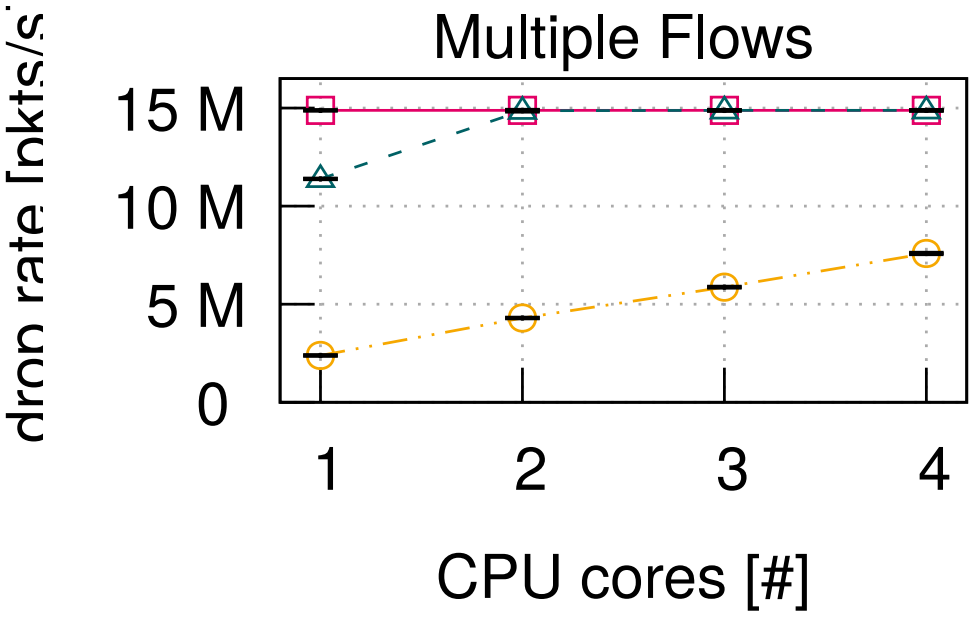




Throughput Baseline: Drop All

Program: `return XDP_DROP;`

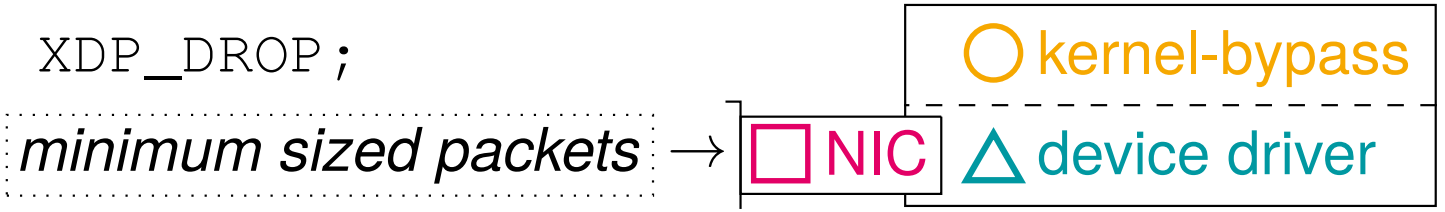


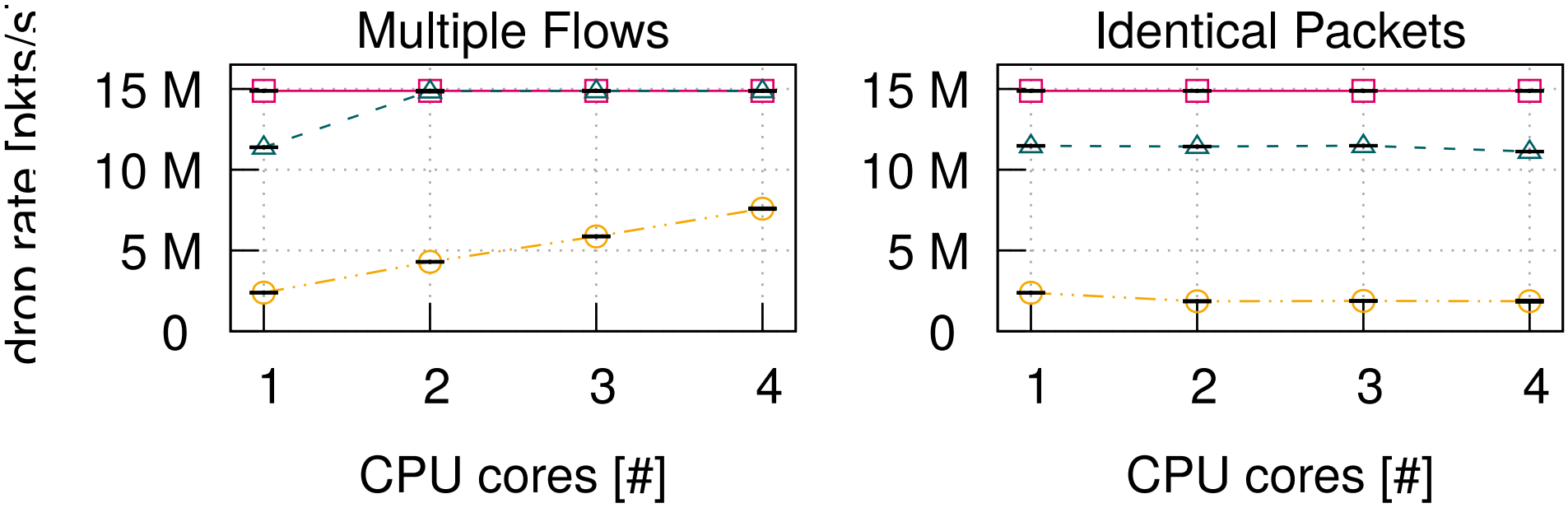


- Offloading improves performance

Throughput Baseline: Drop All

Program: `return XDP_DROP;`



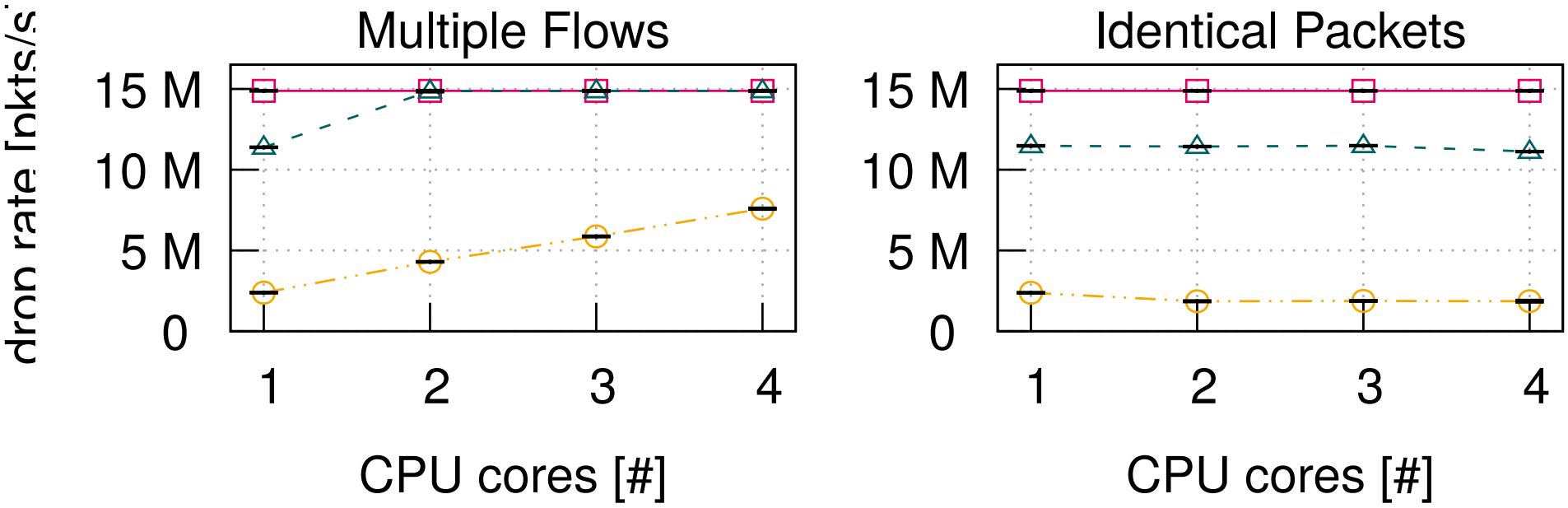


- Offloading improves performance

Throughput Baseline: Drop All

Program: `return XDP_DROP;`

minimum sized packets → NIC | device driver | kernel-bypass

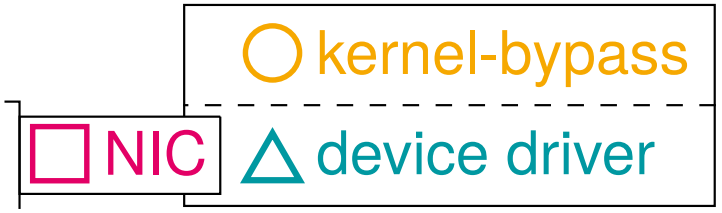


- Offloading improves performance
- Traffic distribution must be taken into account

Processing Complexity

Program:

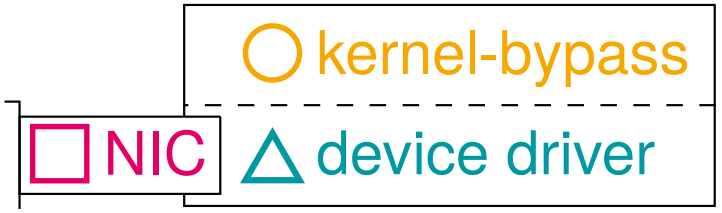
```
for (i = 1; i < n; i++)
    do_something();
return XDP_DROP;
```



Processing Complexity

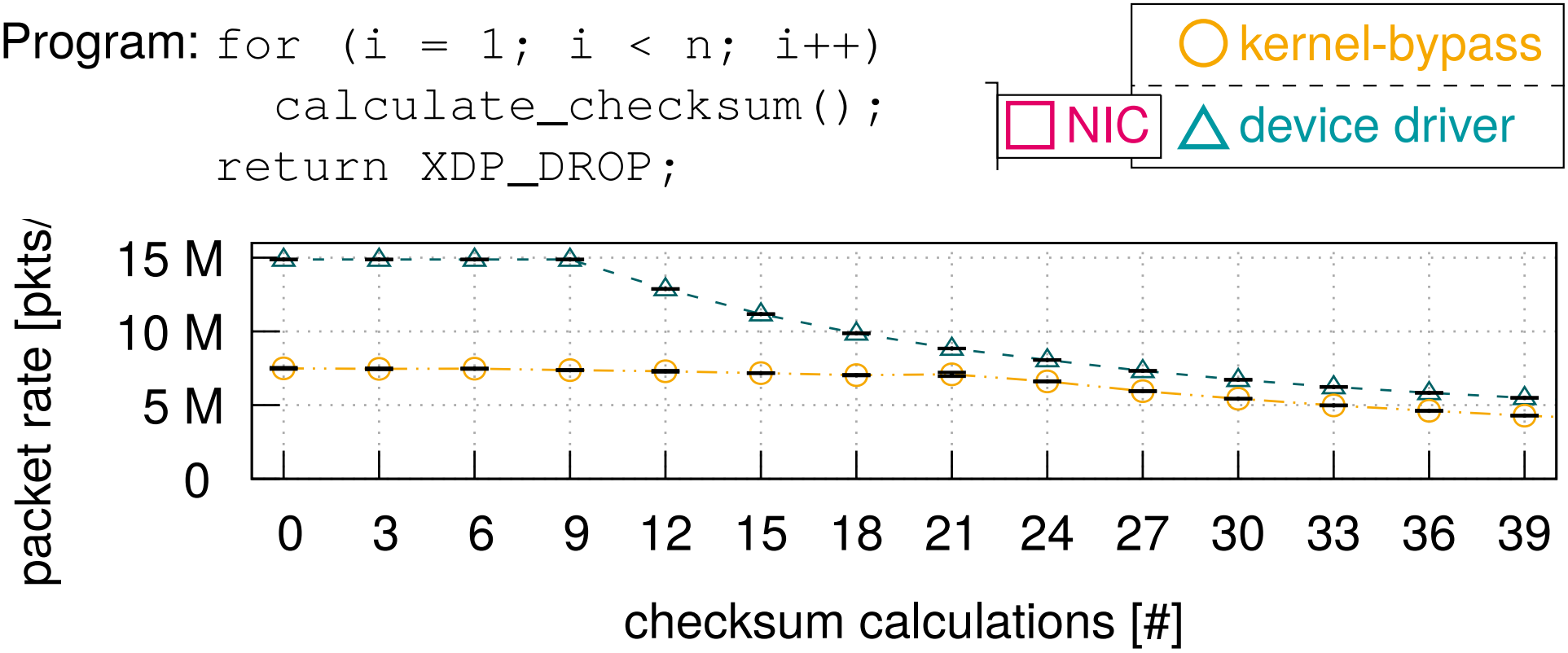
Program:

```
for (i = 1; i < n; i++)
  calculate_checksum();
return XDP_DROP;
```



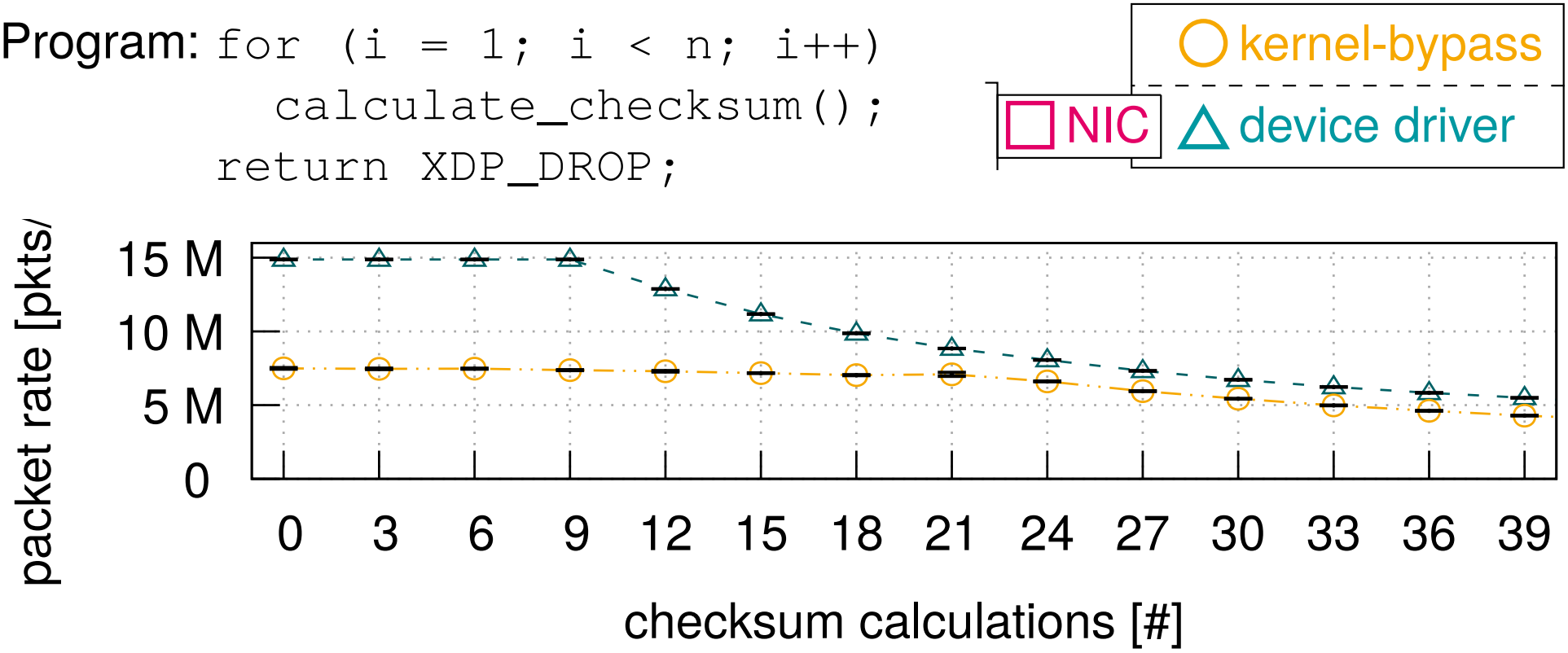
Processing Complexity

Program:
```
for (i = 1; i < n; i++)
    calculate_checksum();
return XDP_DROP;
```



Processing Complexity

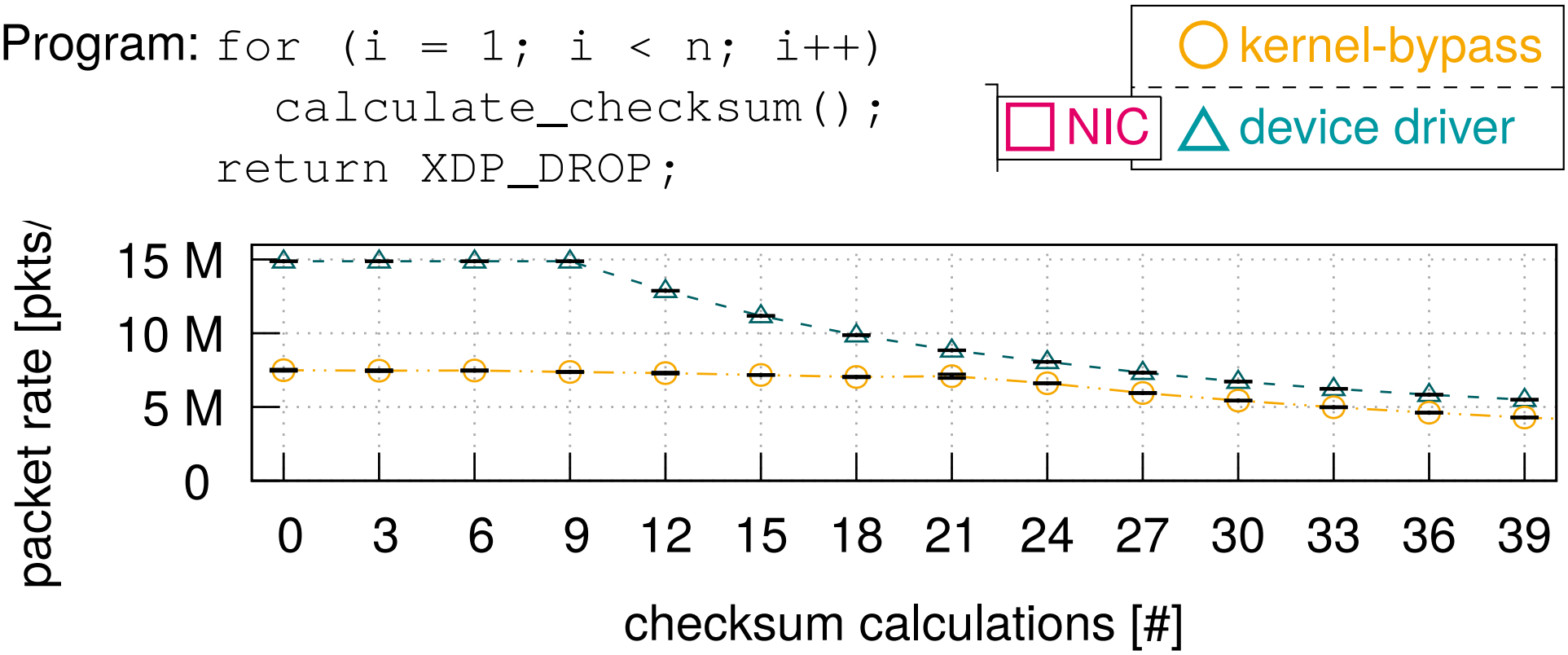
Program:
```
for (i = 1; i < n; i++)
    calculate_checksum();
return XDP_DROP;
```



- Performance depends on the processing complexity

Processing Complexity

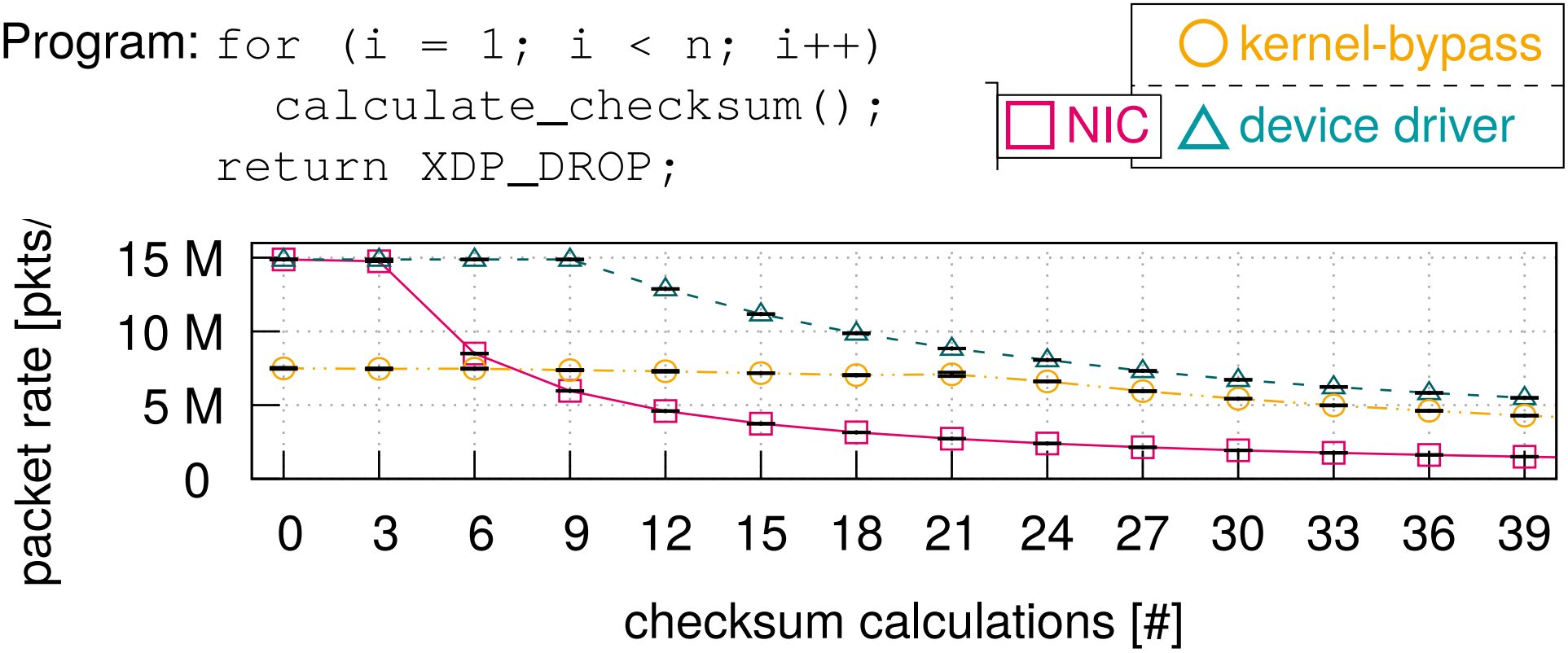
Program:
```
for (i = 1; i < n; i++)
    calculate_checksum();
return XDP_DROP;
```



- Performance depends on the processing complexity
- Most beneficial if task is smaller than the removed overhead

Processing Complexity

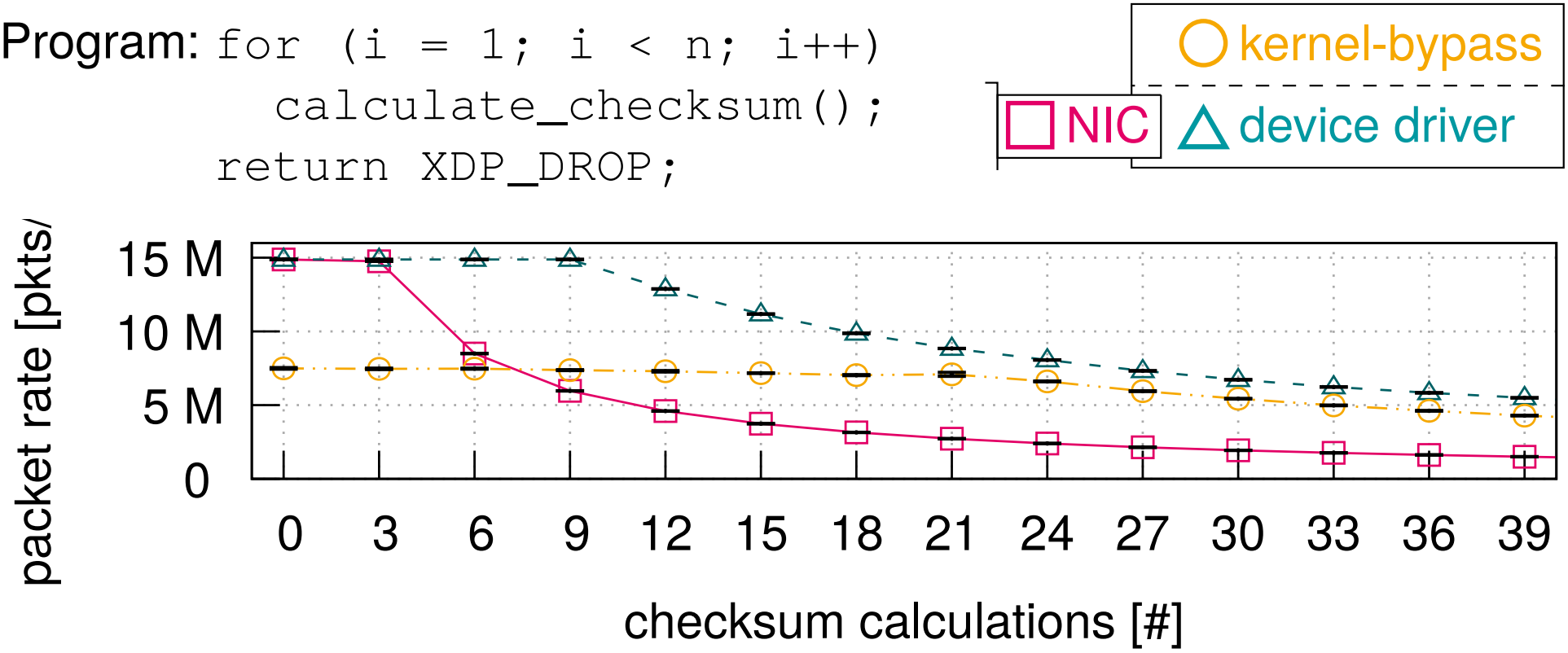
Program:
```
for (i = 1; i < n; i++)
    calculate_checksum();
return XDP_DROP;
```



- Performance depends on the processing complexity
- Most beneficial if task is smaller than the removed overhead

Processing Complexity

Program:
```
for (i = 1; i < n; i++)
    calculate_checksum();
return XDP_DROP;
```

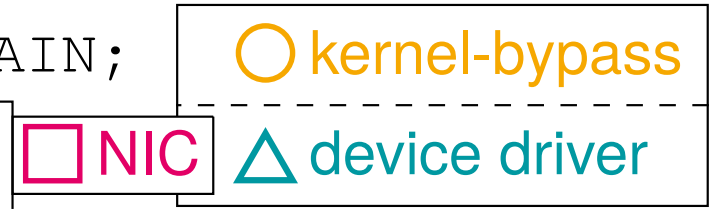


- Performance depends on the processing complexity
- Most beneficial if task is smaller than the removed overhead
- Slow NIC CPU gets easily overloaded

Reducing Latency

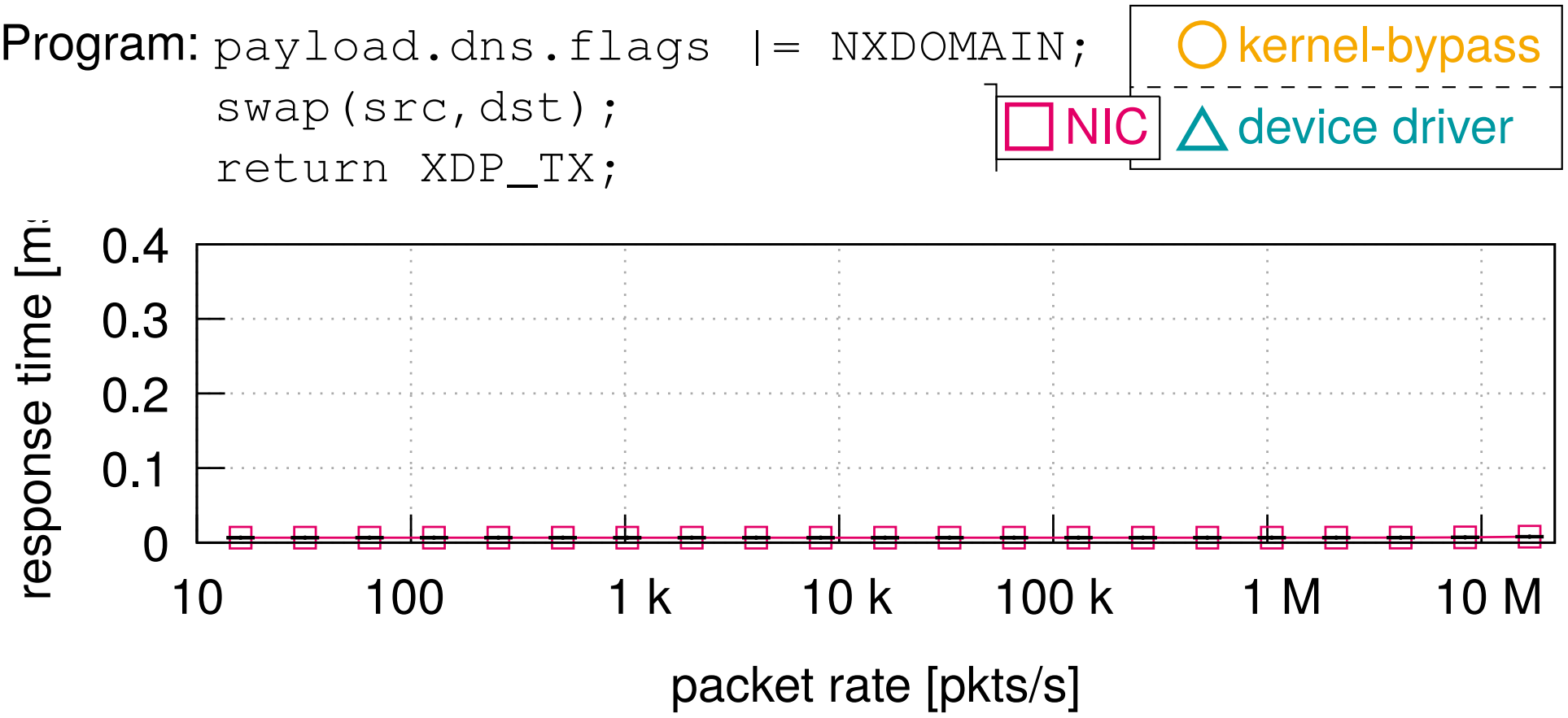
Program:

```
payload.dns.flags |= NXDOMAIN;
swap(src,dst);
return XDP_TX;
```



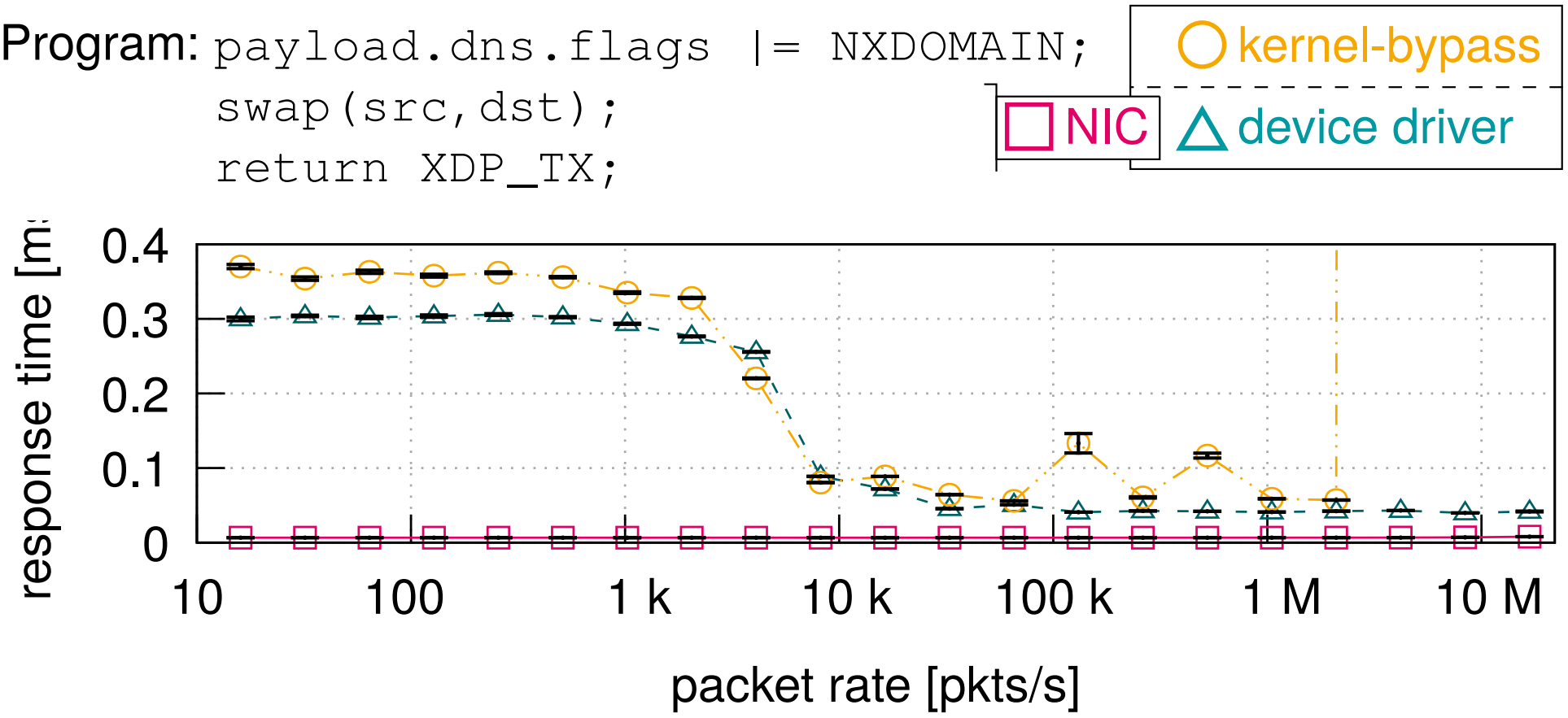
Reducing Latency

Program:
```
payload.dns.flags |= NXDOMAIN;
swap(src,dst);
return XDP_TX;
```



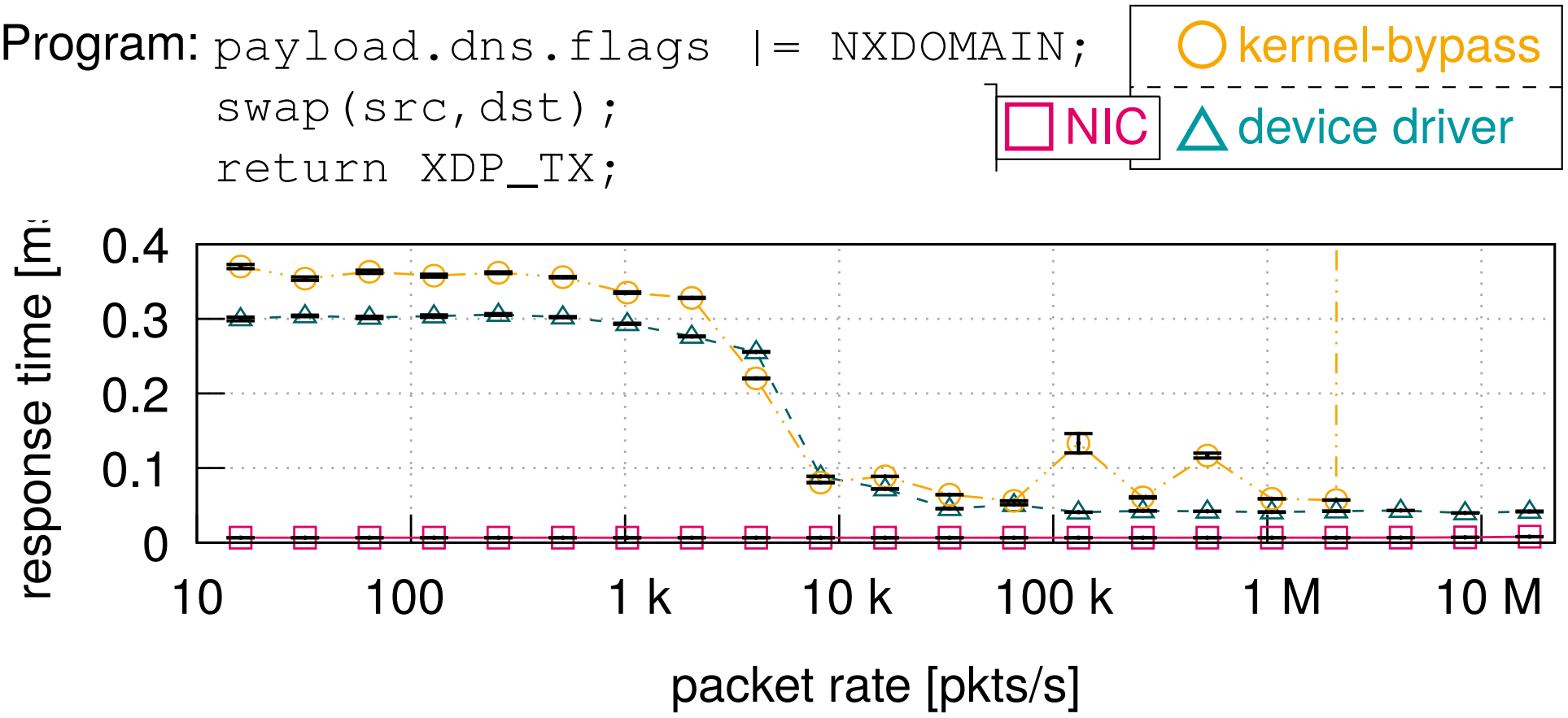
Reducing Latency

Program:
```
payload.dns.flags |= NXDOMAIN;
swap(src,dst);
return XDP_TX;
```



Reducing Latency

Program:
```
payload.dns.flags |= NXDOMAIN;
swap(src,dst);
return XDP_TX;
```

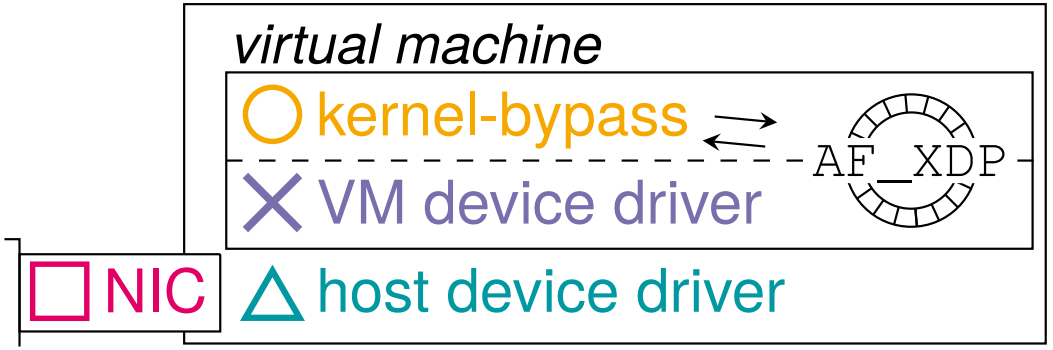


- Latency is highly influenced by the packet rate
- Most latency is introduced before the device driver

Offloading From Virtual Machines

- **Virtualization adds layers**
  - Hypervisor: Xen 4.9.2
  - Open vSwitch
  - vNIC driver

⇒ **More offloading potential**

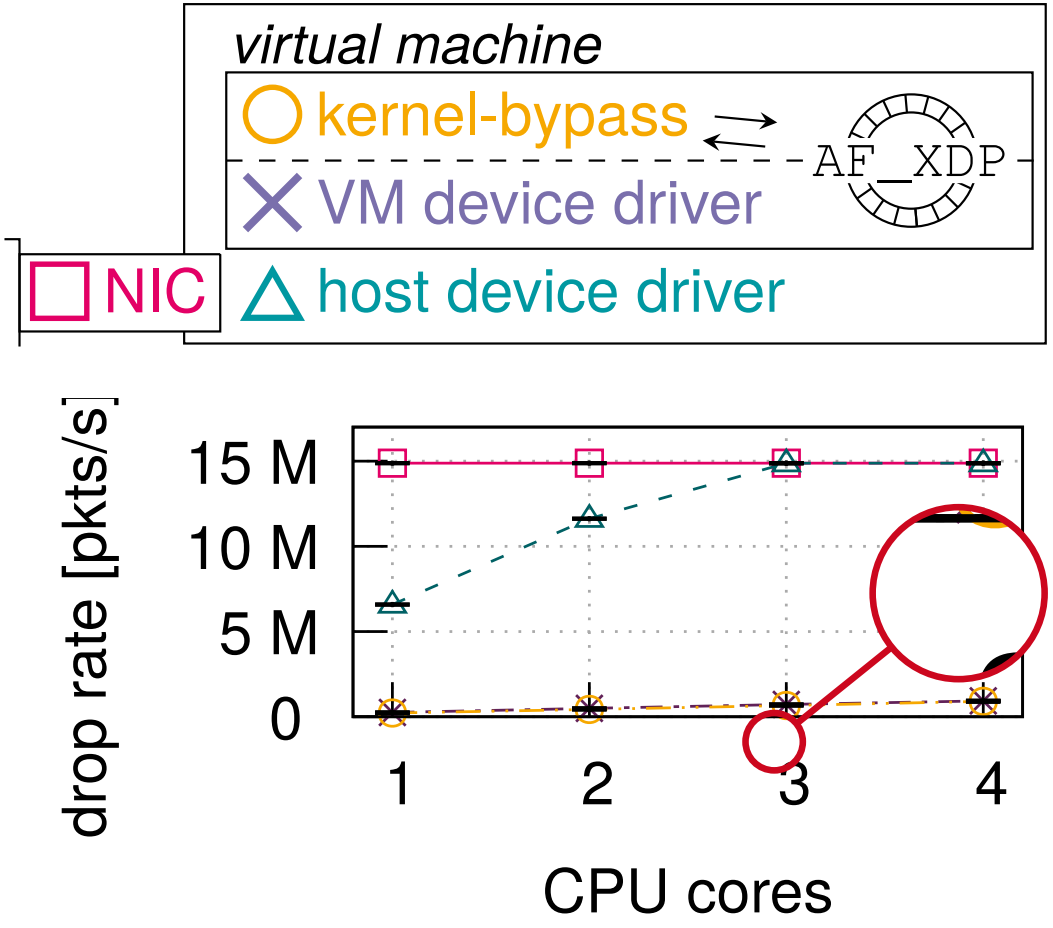


Offloading From Virtual Machines

- **Virtualization adds layers**
  - Hypervisor: Xen 4.9.2
  - Open vSwitch
  - vNIC driver

⇒ **More offloading potential**

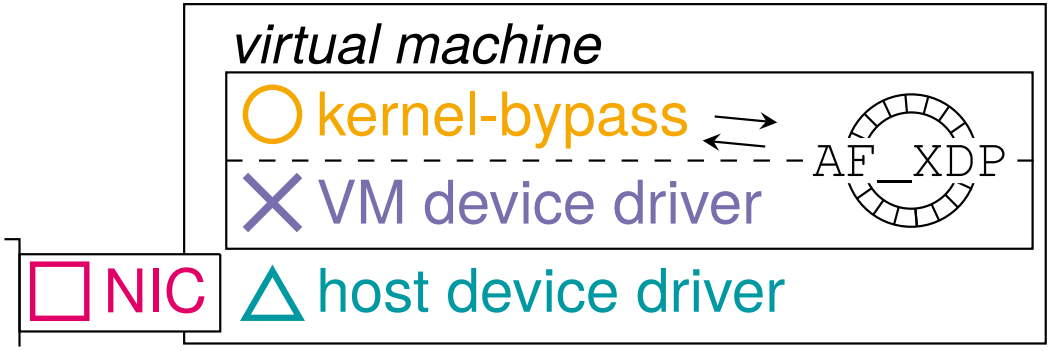
Program: `return XDP_DROP;`



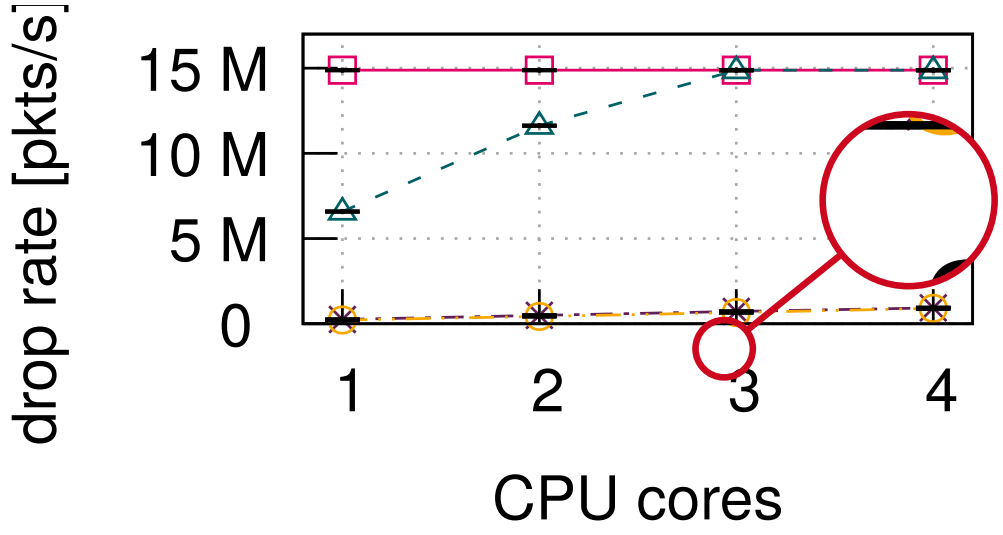
Offloading From Virtual Machines

- **Virtualization adds layers**
  - Hypervisor: Xen 4.9.2
  - Open vSwitch
  - vNIC driver

⇒ **More offloading potential**



Program: `return XDP_DROP;`

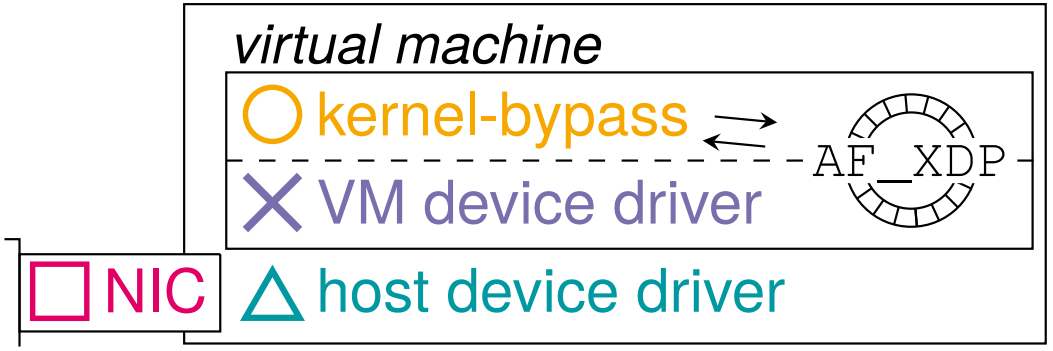


- Only minimal benefit in offloading within a VM

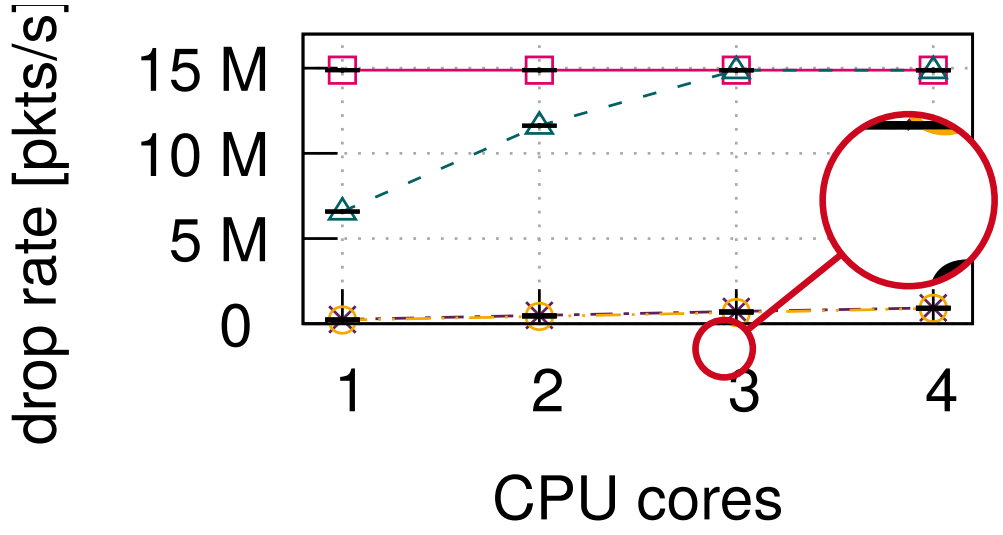
Offloading From Virtual Machines

- **Virtualization adds layers**
  - Hypervisor: Xen 4.9.2
  - Open vSwitch
  - vNIC driver

⇒ **More offloading potential**



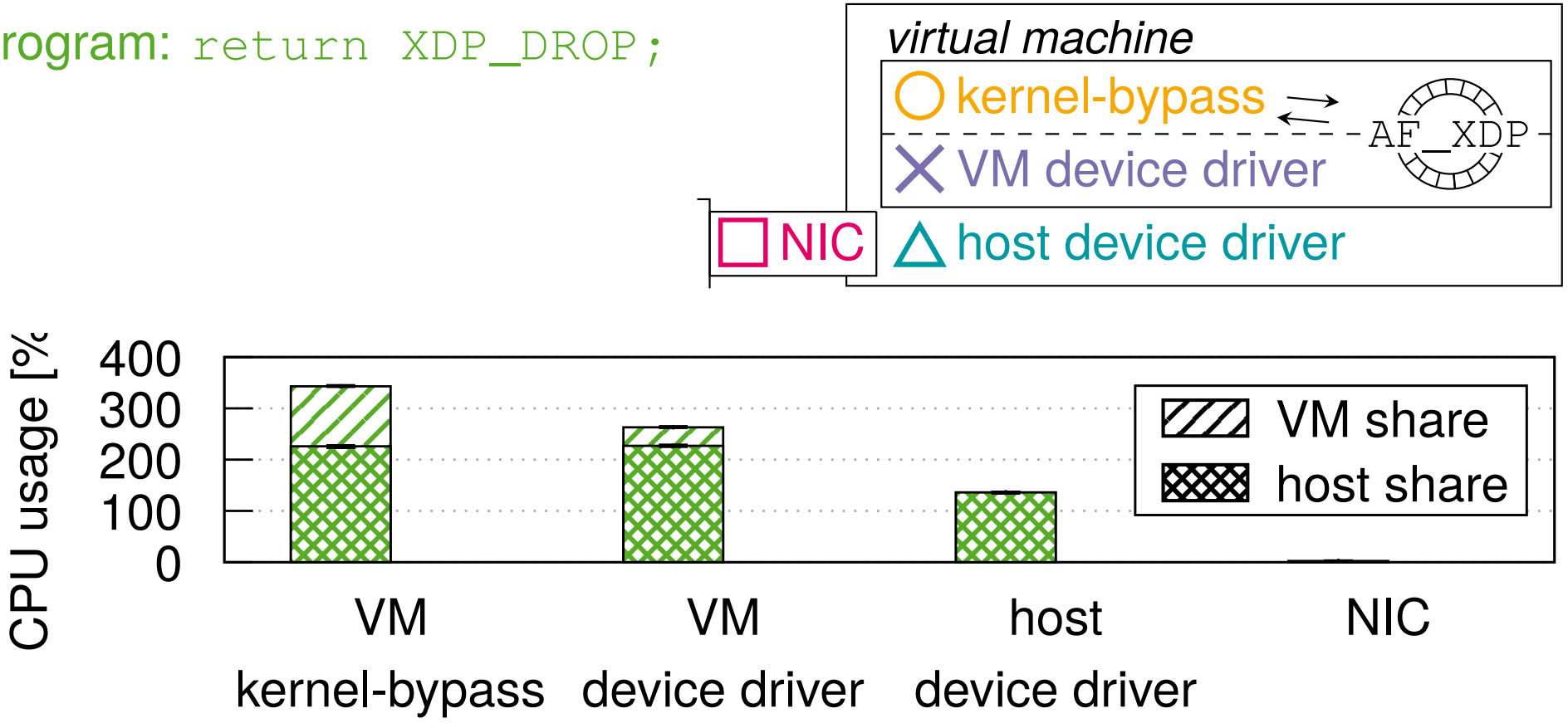
Program: `return XDP_DROP;`



- Only minimal benefit in offloading within a VM
- But offloading to the host requires isolation

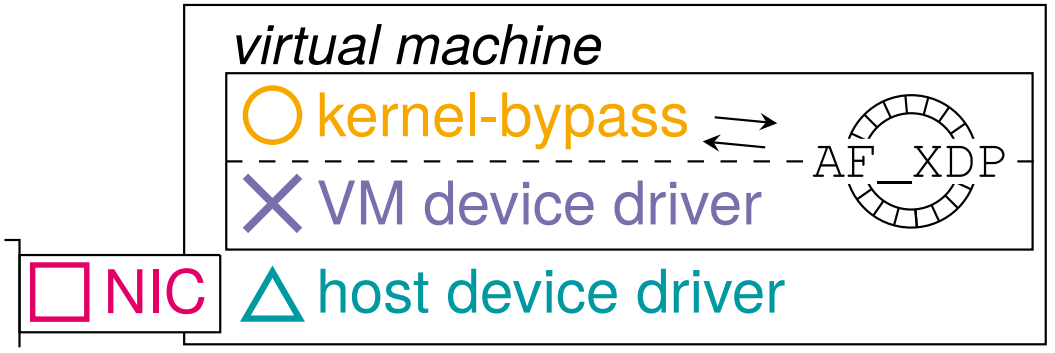
Influence on VM Host

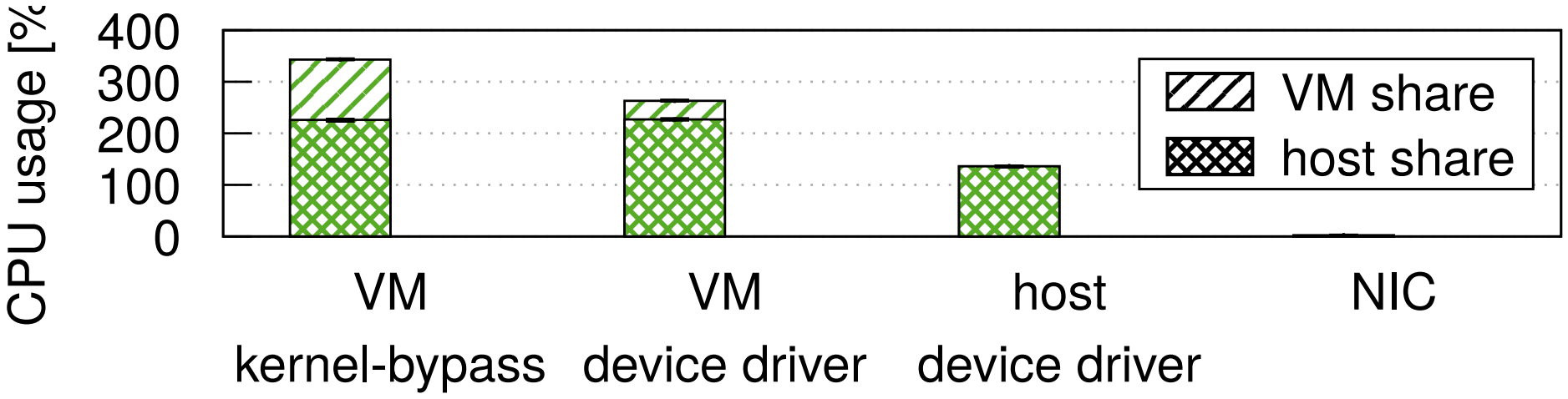
Program: `return XDP_DROP;`



Influence on VM Host

Program: `return XDP_DROP;`





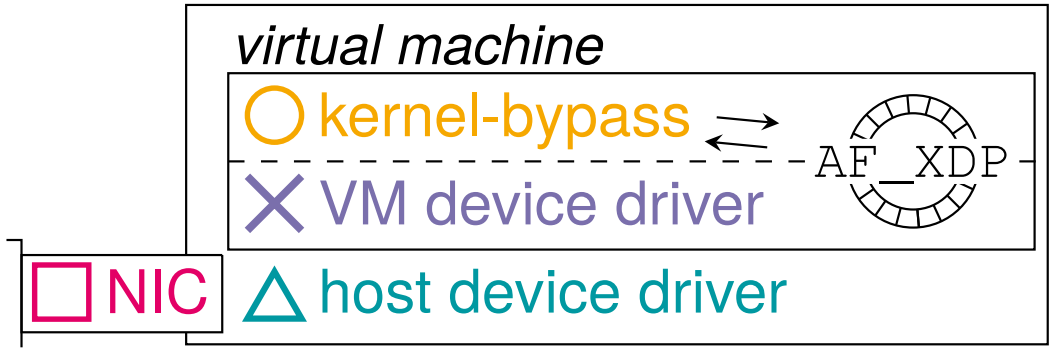
- Offloading can reduce CPU usage in VM and host

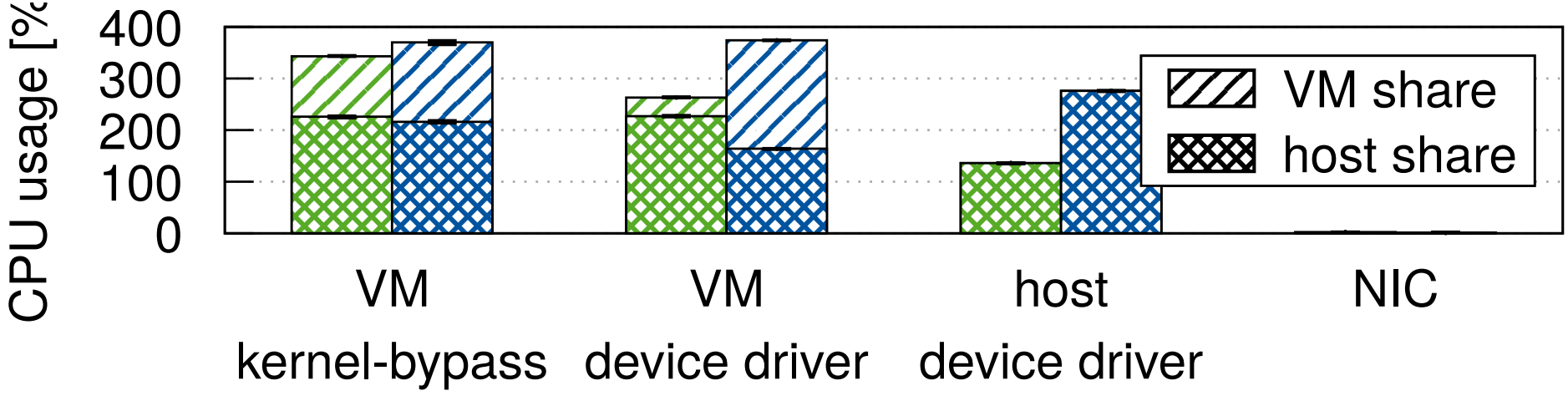
Influence on VM Host

Program: `return XDP_DROP;`

Program:

```
for (i = 1; i < n; i++)
  calculate_checksum();
return XDP_DROP;
```



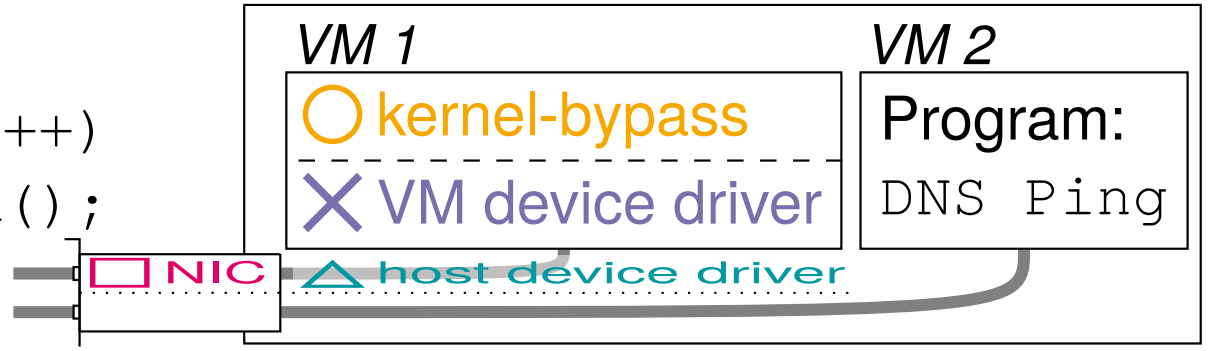


- Offloading can reduce CPU usage in VM and host
- But may shift CPU usage to the host
  - Introduces fairness & accounting problems

Influence on Neighboring VMs

Program:

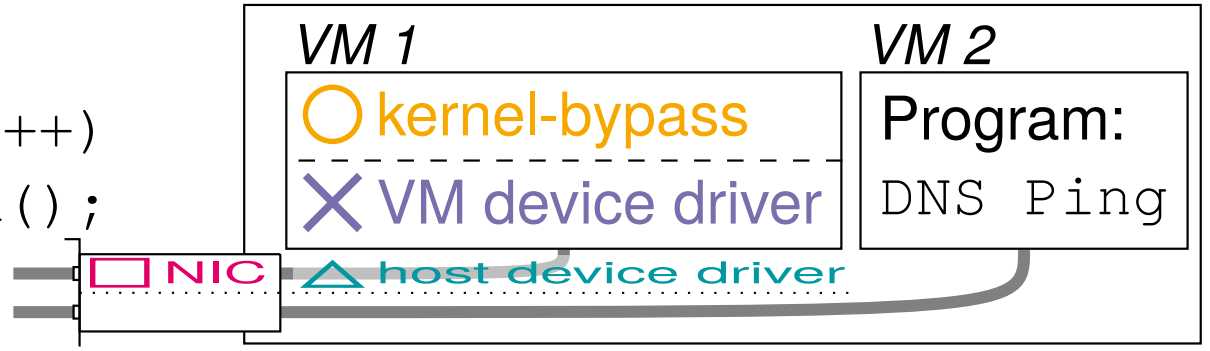
```
for (i = 1; i < n; i++)
  calculate_checksum();
return XDP_DROP;
```

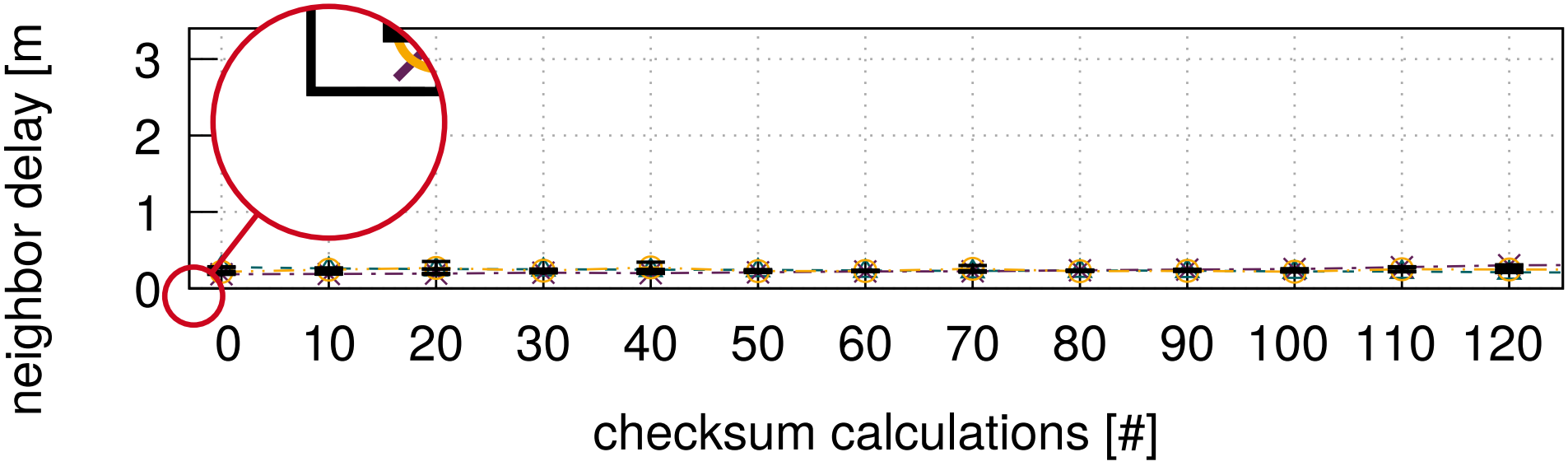


Influence on Neighboring VMs

Program:

```
for (i = 1; i < n; i++)
  calculate_checksum();
return XDP_DROP;
```

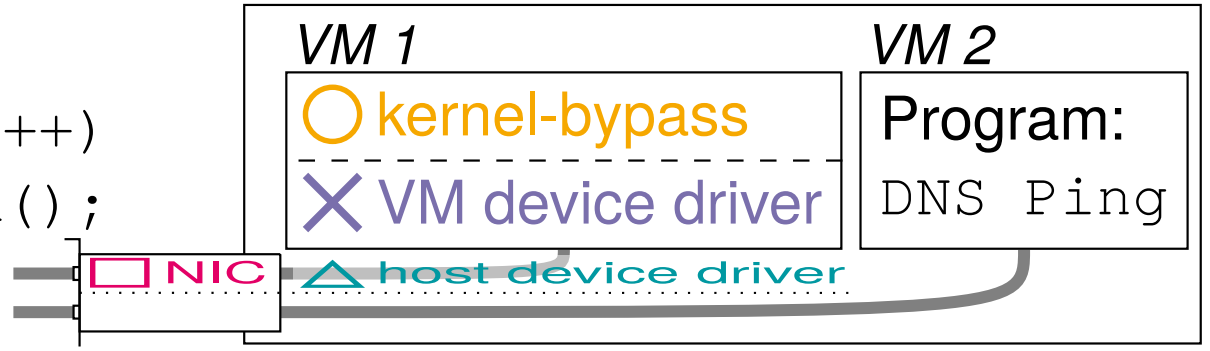


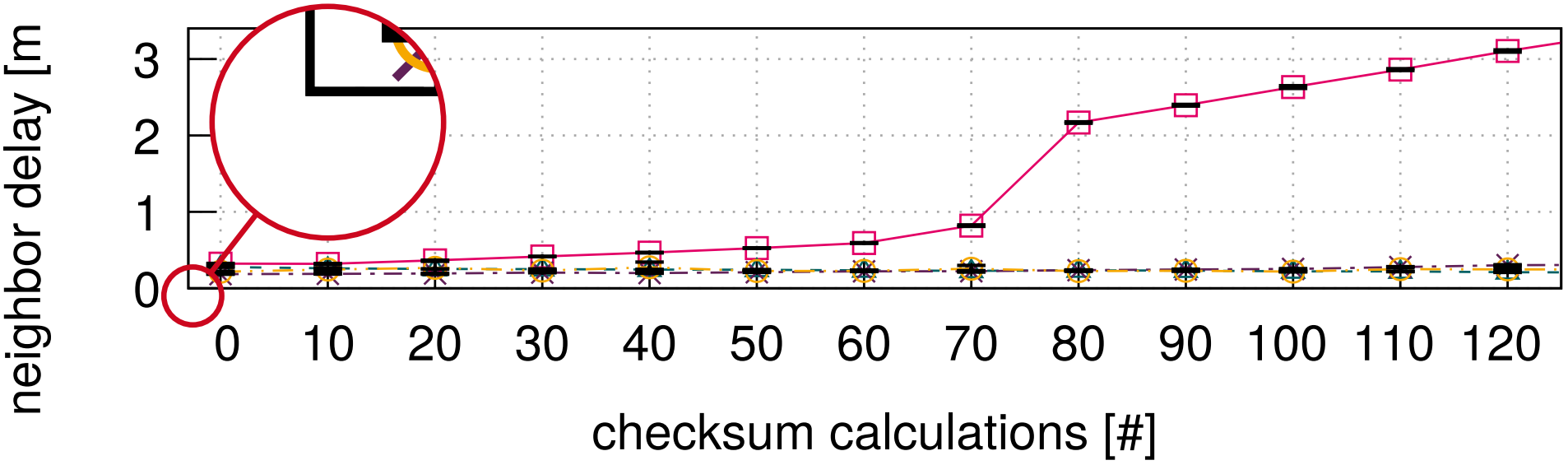


Influence on Neighboring VMs

Program:

```
for (i = 1; i < n; i++)
  calculate_checksum();
return XDP_DROP;
```

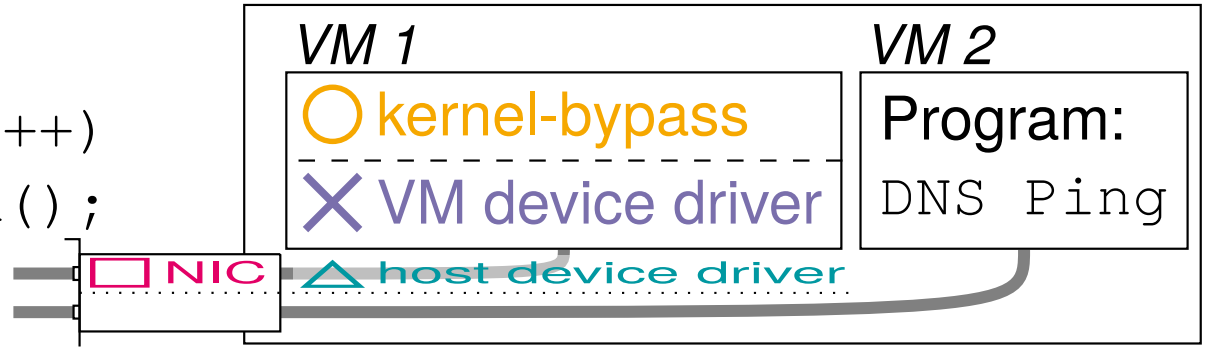


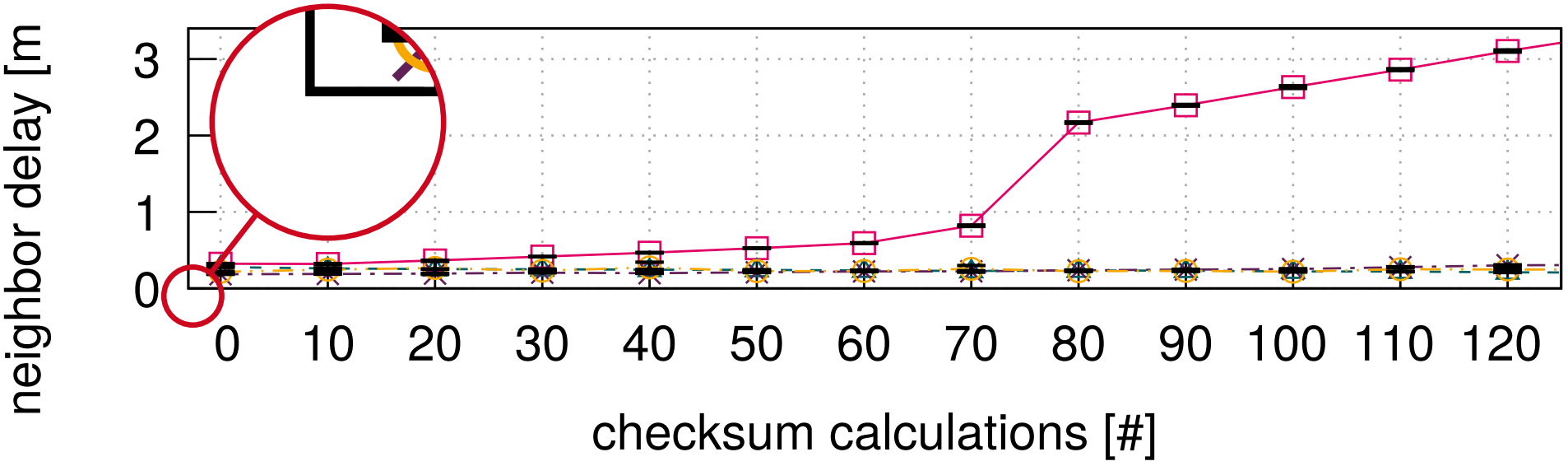


Influence on Neighboring VMs

Program:

```
for (i = 1; i < n; i++)
  calculate_checksum();
return XDP_DROP;
```





- The NIC can introduce delays for non-offloaded applications

Conclusion

- **Offloading performance depends on specific scenario**
  - Offloading may be beneficial
  - But, device driver offloading may not be sufficient
  - But, slow NIC CPU may get overloaded
  - But, offloading may affect traffic of non-offloaded applications

Conclusion

- **Offloading performance depends on specific scenario**
  - Offloading may be beneficial
  - But, device driver offloading may not be sufficient
  - But, slow NIC CPU may get overloaded
  - But, offloading may affect traffic of non-offloaded applications

- **More measurements and results in our paper**