Contribution ID: **291** Type: **not specified**

# XDP bulk packet processing

*Monday, 9 September 2019 15:00 (45 minutes)*

It is well known that batching can often improve software performance. This is mainly because it utilizes the instruction cache in a more efficient way. From the networking perspective, the size of driver's packet processing pipeline is larger than the sizes of instruction caches. Even though NAPI batches packets over the full stack and driver execution, they are processed one by one by many large sub systems in the processing path. Initially this was raised by Jesper Brouer. With Edward Cree's listifying SKBs idea, the first implementation results look promising. How can we take this a step further and apply this technique to the XDP processing pipeline?

To do that, the proposition is to back down from preparing xdp_buff struct one-by-one, passing it to XDP program and then acting on it, but instead we would prepare in driver an array of XDP buffers to be processed. Then, we would have only a single call per NAPI budget to XDP program, which would give us back a list of actions that driver needs to take. Furthermore, the number of indirect function calls, gets reduced, as driver gets to jited BPF program via indirect function call.

In this talk I would like to present the proof-of-concept of described idea, which was yielding around 20% better XDP performance for dropping packets with touching headers memory (modified xdp1 from linux kernel's bpf samples).

However, the main focus of this presentation should be a discussion about a proper, generic implementation, which should take place after showing out the POC, instead of the current POC. I would like to consider implementation details, such as:
- would it be better to provide an additional BPF verifier logic, that when properly instrumented (make use of prologue/epilogue?), would emit BPF instructions responsible for looping over XDP program, or should we have the loop within the XDP programs?
- the mentioned POC has a whole new NAPI clean Rx interrupt routine; what should we do to make it more generic in order to make driver changes smaller?
- How about batching the XDP actions? Do all the drops first, then Tx/redirect, then the passes. Would that pay off?

## I agree to abide by the anti-harassment policy

Yes

## I confirm that I am already registered for LPC 2019

**Primary author:** FIJAŁKOWSKI, Maciej

**Presenter:** FIJAŁKOWSKI, Maciej

**Session Classification:** Networking Summit Track