**IX: A Protected Dataplane Operating System for High Throughput and Low Latency**

*Adam Belay, Stanford University;*

*Ana Klimovic, Samuel Grossman, and Christos Kozyrakis, Stanford University;*

*George Prekas and Edouard Bugnion, École Polytechnique Fédérale de Lausanne (EPFL);*

**Link:**<https://www.usenix.org/conference/osdi14/technical-sessions/presentation/belay>

**Summary:**

The motivation behind IX is very similar to the motivation behind the Arrakis operating system i.e. to decrease the additional overhead due to kernel mediation. As discussed in Paper 1, direct access to hardware resources gives much better performance than accessing I/O through a kernel. It is believed that networking requirements such as high packet rates, microsecond tail latency etc. are best addressed outside the kernel in the user – level networking stack. In traditional OS, software acts like a bottle neck. During network processing, a traditional OS such as a regular Linux machine traverses through complex code paths which are convoluted by interrupts and scheduling thereby adding overhead and in turn degrading the performance. Conventionally to improve the performance, systems usually try to bypass the kernel by moving the TCP/IP stacks to user space or on the hardware, avoiding the connection scalability bottleneck by using proxies at the expense of latency or using datagrams or replacing the classic Ethernet by using a lossless fabric or offload memory access. However, there are some limitations with these approaches and they may not increase the performance a lot.

IX, a data-plane operating system, is designed to break the 4-way tradeoff between high throughput (high packet rates), low microsecond tail latency, strong protection and resource efficiency. It uses virtualization to provide high I/O performance while maintaining the key advantage of strong protection offered by existing kernels. It leverages the protection boundary offered by the kernel by not bypassing it, utilizes the connection scalability of protocols such as TCP and tries to use the conventional commodity Ethernet. It uses hardware virtualization to separate management, resource configuration, provisioning and scheduling functions of the kernel (control plane) from network processing (data plane). The control plane has a full Linux kernel and a user-level program - IXCP and the data plane runs as protected, library-based operating systems running on dedicated hardware threads. Each data plane runs a single application and is given one CPU core. It runs as a single address-space OS. ICXP implements resource allocation procedures and also monitors resource usage and data-plane performance.

The IX API’s design is guided by the commutativity rule and hence does not follow the POSIX API. However compatibility with existing applications is provided with the help of a user-level library – libIX. In IX, the Linux kernel is run in ring 0 in root mode and the Dune module is run within the Linux kernel to enable application specific OS’s to run in non-root mode in ring 0. IX relies on the Linux kernel for file system support. The applications run in VMX non-root mode in ring 3. IX supports elastic threads and background threads within a shared user-level address-space. Each of these threads is capable of issuing POSIX system calls that are validated for security before forwarding to the Linux kernel. The current implementation of IX depends on Dune and hence requires VT-x features available on an Intelx64. IX follows the notion of run to completion which helps in improving data cache locality and a lot of the scheduling unpredictability is removed since the operations are executed in a well-defined order of steps. This notion helps in getting low latency in IX. The second design principle followed by IX is adaptive batching which improves instruction cache locality, good branch prediction behavior and prefetching the next cache line thereby increasing performance. The IX data-plane is optimized for multi-core scalability since each of the elastic threads are synchronized and operate in a coherent free manner. Since the IX core does not use IOMMU, the data-plane is trusted code and has access to descriptor rings with host physical addresses. IX gives 10X better throughput than Linux when sending short messages using the 10 GB Ethernet. It outperforms Linux by 3.6X for throughput when using memcached, a widely deployed key-value store, thereby reducing the kernel time from approximately 75% in a regular Linux machine to <10% in IX. When connection scalability is considered, IX performs 10X better than Linux.

In my opinion this study can be very useful in situations where we are dealing with large overheads due to delays in the network stack. I feel there are a few limitations with this approach. The number of applications that can be run at a time depends on the limited number of CPU cores available. Communication between applications running on the same machine needs to be done over a network since each one runs on an individual dedicated core. However, if Dune’s capability is fully leveraged, then with the help of regular system calls to the Linux kernel, we can overcome this limitation.