OS and Hardware

Adil Ahmad
Categorizing the papers at OSDI and ATC

Centered around three important OS duties

A. Process scheduling
B. Memory management
C. Network management
D. Novel hardware designs

And new ways to improve the current hardware
A. Process scheduling

**Two relevant concepts:**

- **Fairness:** each process should be allowed to use the CPU equally
- **Latency:** each process should complete as soon as possible

OS decides which process runs on CPU.
A: Process scheduling

Paper #1

“Fair Scheduling for AVX2 and AVX-512 Workloads”
USENIX ATC 2021

Identify that AVX causes scheduling unfairness and compensate the affected processes
Background: AVX instructions

Non-AVX process data in 64-bit registers

AVX can process data in 256 or 512-bit wide registers at a time

Better performance for certain applications!
Problem: unfairness from AVX instructions

AVX instructions force a CPU core into a lower clock speed

Other processes executing on the same processor core are affected!
Design a scalable hybrid event queue to reduce application latency
Background: event queues

Service packet when a thread is free

Event (e.g., network packet) is received

Buffer packet in a queue
A: Process scheduling

Problem: latency in current event queues

1:1 model
Queue for CPU 1
Queue for CPU 2
Under-utilized CPUs because events cannot be migrated easily!

1:N model
Shared queue for CPU 1 and 2
Expensive synchronization between CPUs on each event!

“SKQ: Event Scheduling for Optimizing Tail Latency in a Traditional OS Kernel” USENIX ATC 2021
B. Memory management

Rely on the OS for memory allocation

**Two relevant concepts:**

- **Memory conservation:** support many programs with limited memory
- **Maximum performance:** reduce the performance impact of memory accesses
Paper #1

“Beyond Malloc Efficiency to Fleet Efficiency: A Hugepage-aware Memory Allocator” USENIX OSDI 2021

Create a memory allocator which helps conserve memory for hugepages
Background: translation and hugepages

Fast translations using TLB

Slow translations using page tables

Better performance!
Problem: fragmentation from hugepages

Allocate two buffers of 2 MB each

Free 1 MB from each buffer

2 MB is free but can only allocate 1 MB buffers!
Design an OS kernel which is optimized for server architectures (i.e., NUMA)
Background: non-uniform memory access

Can access both local and remote DRAM

Memory access speed depends on locality

Non-uniform memory access (NUMA)
Problem: NUMA slows monolithic kernels

A shared kernel state is accessed by all CPUs.

Frequent accesses to remote DRAM harm performance!

“NrOS: Effective Replication and Sharing in an Operating System” USENIX OSDI 2021
“Exploring the Design Space of Page Management for Multi-Tiered Memory Systems”
USENIX ATC 2021

Design a memory management scheme optimized for server memory architectures
Background: multi-tiered memory (MTM)

Memory access speeds depend on memory tier and locality.

DRAM is a “fast” memory tier.

DCPMM is a “slow” memory tier.

Multi-tiered memory with NUMA

“Exploring the Design Space of Page Management for Multi-Tiered Memory Systems” USENIX ATC 2021
Problem: inefficient MTM usage

Local DRAM is full of data
OS keeps data in local DCPMM to preserve locality

There is space in remote DRAM, which is faster!
C. Network management

Rely on the OS for network access

Two concepts are relevant:

1. Network flows are **short** (e.g., web requests) or **long** (e.g., database snapshots)

2. **Short** network flows are more **latency-sensitive** than long ones
Paper #1

“A Linux Kernel Implementation of the Homa Transport Protocol”
USENIX ATC 2021

Implements Homa, a transport protocol that prioritizes short flows to reduce latencies
C: Novel hardware designs

**Background: Homa transport protocol**

**TCP:**
Sender does not control how packets are handled by network

<table>
<thead>
<tr>
<th>Normal router queue</th>
<th>Latency-sensitive short flows could be delayed!</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short</td>
<td></td>
</tr>
<tr>
<td>Long</td>
<td></td>
</tr>
</tbody>
</table>

**Homa:**
Sender configures packets to use priority queues in routers

```
High priority router queue
| Short |                                               |

Low priority router queue
| Long  |                                               |
```

Short flows are prioritized, and latency is reduced!
Problem: Homa is not tested practically

Simulations through software network simulators
User-level implementation with kernel-bypass

"A Linux Kernel Implementation of the Homa Transport Protocol" USENIX ATC 2021
D: Novel hardware designs

- Improve **security** of remote computation
- Improve **performance** in emerging real-world scenarios
A new trusted hardware that is more scalable and robust than previous solutions.

“Scalable Memory Protection in the Penglai Enclave”
USENIX OSDI 2021
Background: enclaves

CPU stops the OS from accessing enclave memory

Even physical attackers cannot leak enclave memory

D: Novel hardware designs

“Scalable Memory Protection in the Penglai Enclave” USENIX OSDI 2021
Problem: existing enclaves do not scale

D: Novel hardware designs

Limited enclave memory (e.g., 128-256 MB)

Slow startup speeds (e.g., many seconds)
D: Novel hardware designs

Paper #2

“A Fast and Flexible Hardware-based Virtualization Mechanism for Computational Storage Devices” USENIX ATC 2021

A new hardware to virtualize computational storage devices without relying on slow software
Background: computational storage

Offload computation to the FPGA

Reduces data moved through slow PCIe

“A Fast and Flexible Hardware-based Virtualization Mechanism for Computational Storage Devices” USENIX ATC 2021
Problem: CSD virtualization is inefficient

- CSDs do not support hardware virtualization
- Must rely on slow software interfaces!

D: Novel hardware designs

“A Fast and Flexible Hardware-based Virtualization Mechanism for Computational Storage Devices” USENIX ATC 2021
“The nanoPU: A Nanosecond Network Stack for Datacenters”
USENIX OSDI 2021

A new hardware to realize nanosecond response times for very short-lived datacenter requests
Background: remote procedure calls

Can call a function (or procedure) on a remote machine as if it was a local function.

RPCs in datacenters are becoming more and more short-lived.

Few nano-second (ns) – 1 micro-second (us)

“The nanoPU: A Nanosecond Network Stack for Datacenters” USENIX OSDI 2021
Problem: existing network stacks are slow

Software stacks (e.g., Linux) take ~100s of microseconds

Hardware stacks (e.g., RDMA) take 1-2 microseconds

D: Novel hardware designs

“The nanoPU: A Nanosecond Network Stack for Datacenters” USENIX OSDI 2021
Conclusion

OS and hardware
Thursday, July 15 @ 7 AM PDT

My tail never has any latency
Friday, July 16 @ 8.30 AM PDT