# STYX: Exploiting SmartNIC Capability to Reduce Datacenter Memory Tax

Houxiang Ji, Mark Mansi, Yan Sun, Yifan Yuan, Jinghan Huang, Reese Kuper,
Michael M. Swift, and Nam Sung Kim
University of Illinois Urbana-Champaign, University of Wisconsin-Madison, Intel Labs

ATC 23

梁恒中　2024.4.10

# SmartNIC

- consists of a network interface controller, CPU, ASIC- and/or FPGA-based accelerator, memory and IO subsystem
- used to offload network functions
  - TCP/IP network stack
- can carry out customised functionalites
  - gpu communication through network bypassing cpu
- saves host cpu resource

# Memory Optimization Kernel Feature

## ksm/kernel same-page merge

- compare two pages and determine whether they contain the same content
- calculate 32-bit checksum of a page
- perform byte-to-byte comparison of two pages

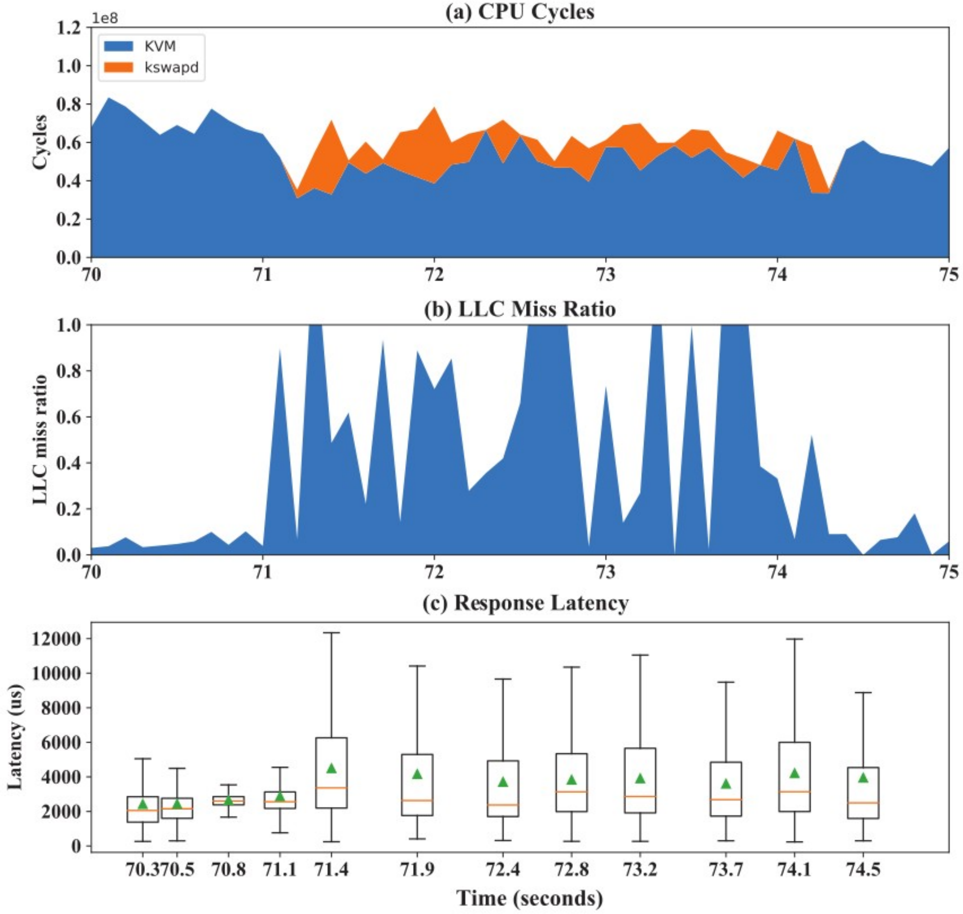
## zswap

- compression backend for kswapd
- compress pages to avoid swap
- decompress when page fault happens
- synchronous direct/asynchronous backgroud

# Impact of Kernel Feature

Both ksm and zswap are memory-intensive and CPU-intensive

- bring large amount of cold data into cache, causing increased cache misses ratio
- consume cpu cycles
- interfere with co-running applications



A snapshot of (a) consumed CPU cycles, (b) LLC miss ratio, and (c) response latency before and after invoking kswapd while running Redis

# STYX overview

Ksm and zswap follow such a pattern:

1) Determine memory regions to operate on
2) Load memory regions to cpu cache from memory
3) Operate on the memory regions
4) Make a decision for the next step according to the result

which can be decomposed into data plane(step 2 and 3) and control plane(step 1 and 4), like a network application.

STYX leaves control plane operations on host cpu and offloads data plane operations to SNIC.

# STYX overview

## How STYX works

1) The host cpu determines memory regions to operate on;
2) Memory regions are copied from host memory to SNIC memory using RDMA;
3) The SNIC operates on the memory regions;
4) The SNIC transfers back the result to the host memory;
5) The host cpu decides the next step.

# STYX overview

STYX relies on SNICs:

- SNIC is capable of data transfering using its RDMA engine to copy data from host memory
- SNIC is capable of computing using its cpu cores
- SNIC is widely deployed in data centers
- SNIC's cpu cores are not yet fully utilized(?), therefore STYX could offload host operations without dramatically interfering with network applications runing on the SNIC.

# STYX workflow

## (1) Setup

- decide which kernel features to be offloaded
- setup RDMA connection between host cpu and SNIC and allocate resources(one connection for each function)
- use descriptors on host and SNIC to record information

## (2) Submission

- update descriptors to with memory regions to operate on
- the host posts a RDMA send request to SNIC, then the host waits on recv
- RDMA operations can be one-sided or two-sided

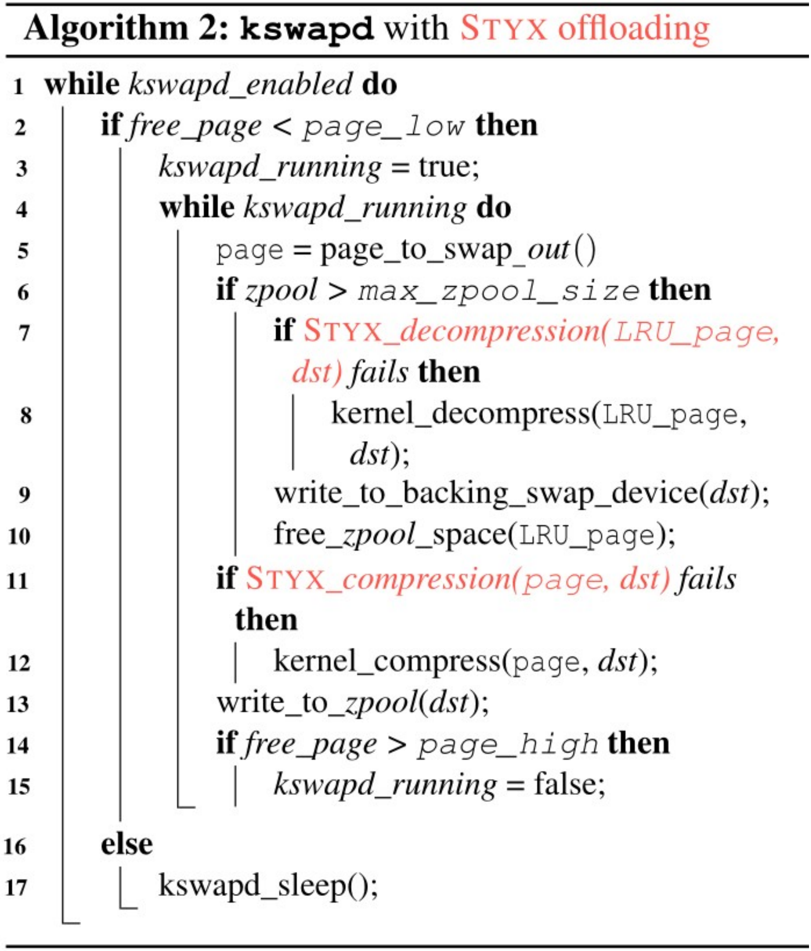
# STYX workflow

## (3) Remote Executation

- STYX on SNIC copies data from host memory
- STYX on SNIC operates on the data
- may interferes with applications on SNIC

## (4) Completion

- STYX on SNIC posts a RDMA send request to host
- The host receives result from SNIC, and resumes execution
- STYX on SNIC waits on recv
- RDMA operations can be one-sided or two-sided



**Algorithm 2: kswapd with STYX offloading**

```
while kswapd_enabled do
    if free_page < page_low then
        kswapd_running = true;
        while kswapd_running do
            page = page_to_swap_out()
            if zpool > max_zpool_size then
                if STYX_decompression(LRU_page, dst) fails then
                    kernel_decompress(LRU_page, dst);
                write_to_backing_swap_device(dst);
                free_zpool_space(LRU_page);
            if STYX_compression(page, dst) fails then
                kernel_compress(page, dst);
            write_to_zpool(dst);
            if free_page > page_high then
                kswapd_running = false;
    else
        kswapd_sleep();
```

# Evaluation Setup

Workload:

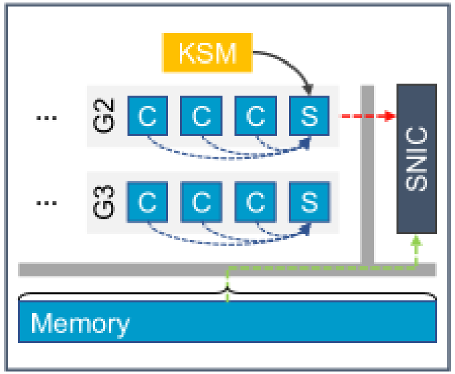
yahoo! Cloud Serving-Benchmark on Redis

(a) update heavy

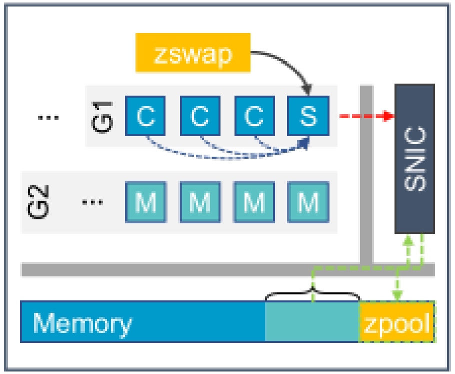
(b) read heavy

(c) read only

(d) read latest



(a) Setup for **ksm**



(b) Setup for **zswap**

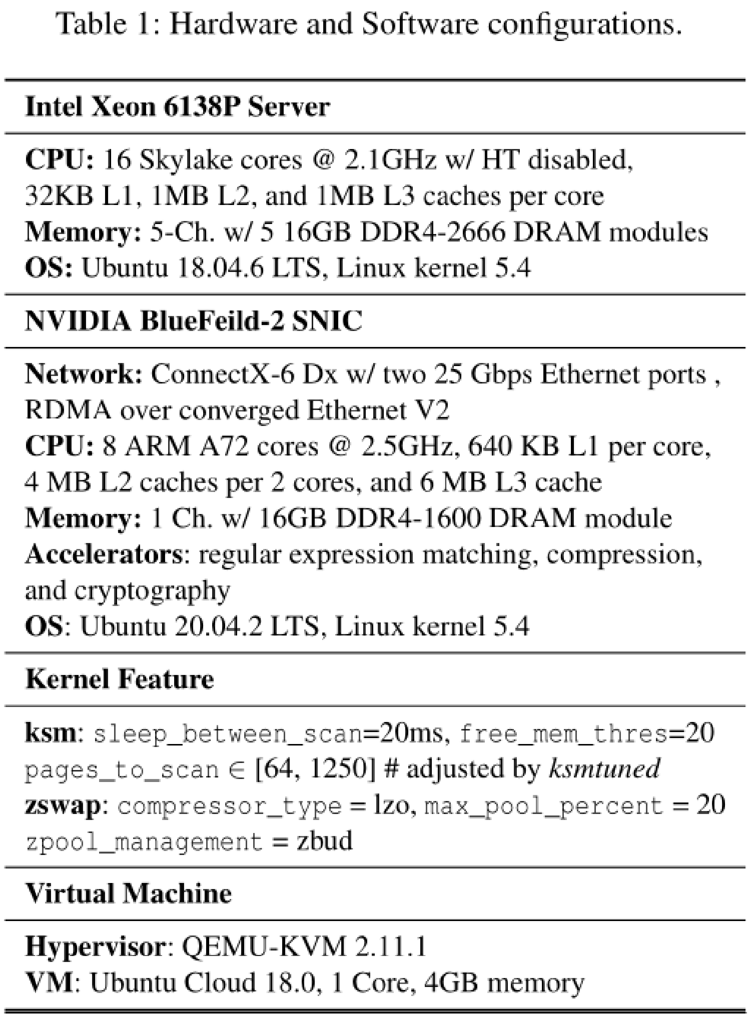
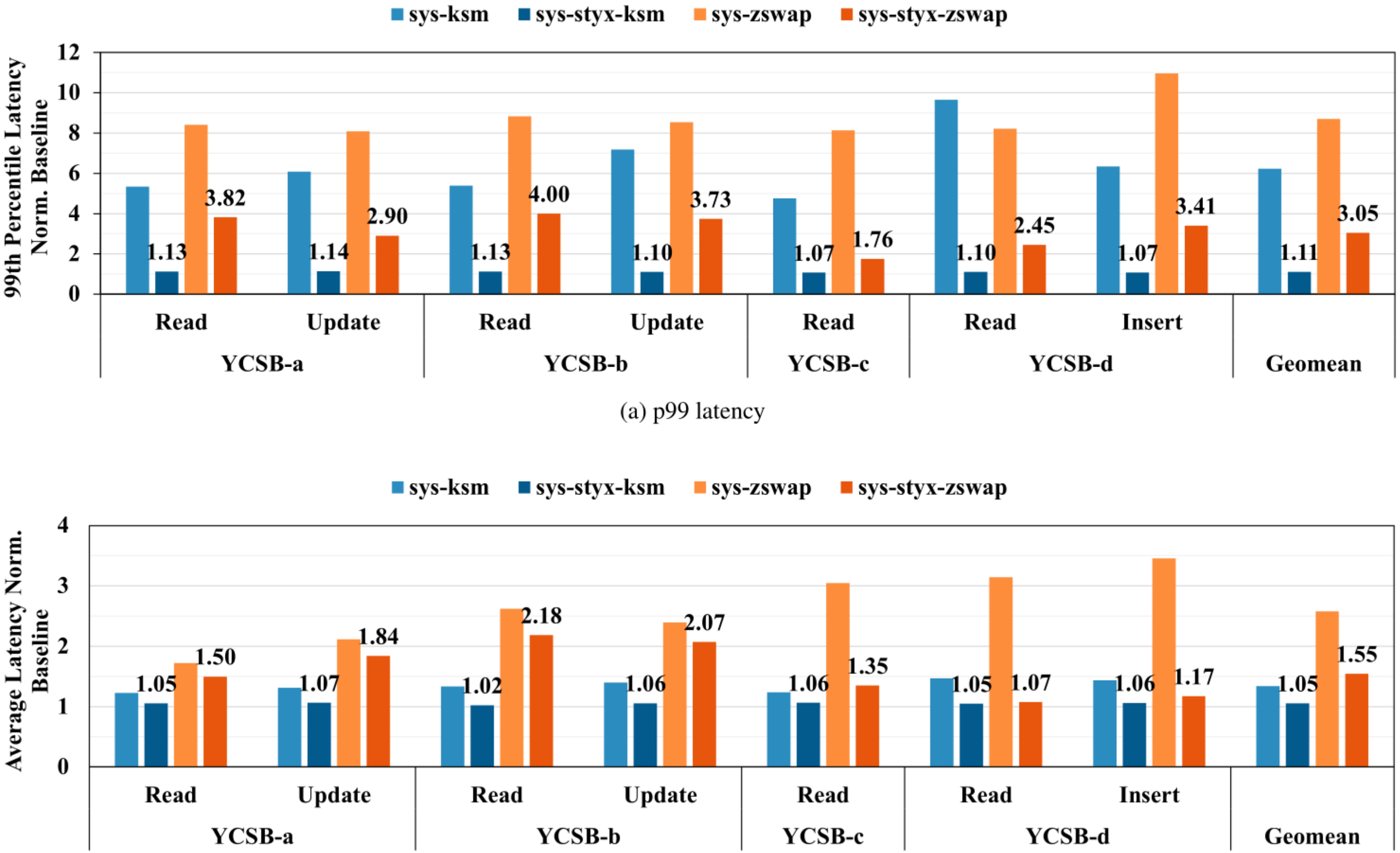


Table 1: Hardware and Software configurations.

| **Intel Xeon 6138P Server** |
|---|
| **CPU:** 16 Skylake cores @ 2.1GHz w/ HT disabled, 32KB L1, 1MB L2, and 1MB L3 caches per core<br>**Memory:** 5-Ch. w/ 5 16GB DDR4-2666 DRAM modules<br>**OS:** Ubuntu 18.04.6 LTS, Linux kernel 5.4 |
| **NVIDIA BlueFeild-2 SNIC** |
| **Network:** ConnectX-6 Dx w/ two 25 Gbps Ethernet ports , RDMA over converged Ethernet V2<br>**CPU:** 8 ARM A72 cores @ 2.5GHz, 640 KB L1 per core, 4 MB L2 caches per 2 cores, and 6 MB L3 cache<br>**Memory:** 1 Ch. w/ 16GB DDR4-1600 DRAM module<br>**Accelerators**: regular expression matching, compression, and cryptography<br>**OS**: Ubuntu 20.04.2 LTS, Linux kernel 5.4 |
| **Kernel Feature** |
| **ksm**: `sleep_between_scan`=20ms, `free_mem_thres`=20 `pages_to_scan` ∈ [64, 1250] # adjusted by *ksmtuned*<br>**zswap**: `compressor_type` = lzo, `max_pool_percent` = 20 `zpool_management` = zbud |
| **Virtual Machine** |
| **Hypervisor**: QEMU-KVM 2.11.1<br>**VM**: Ubuntu Cloud 18.0, 1 Core, 4GB memory |

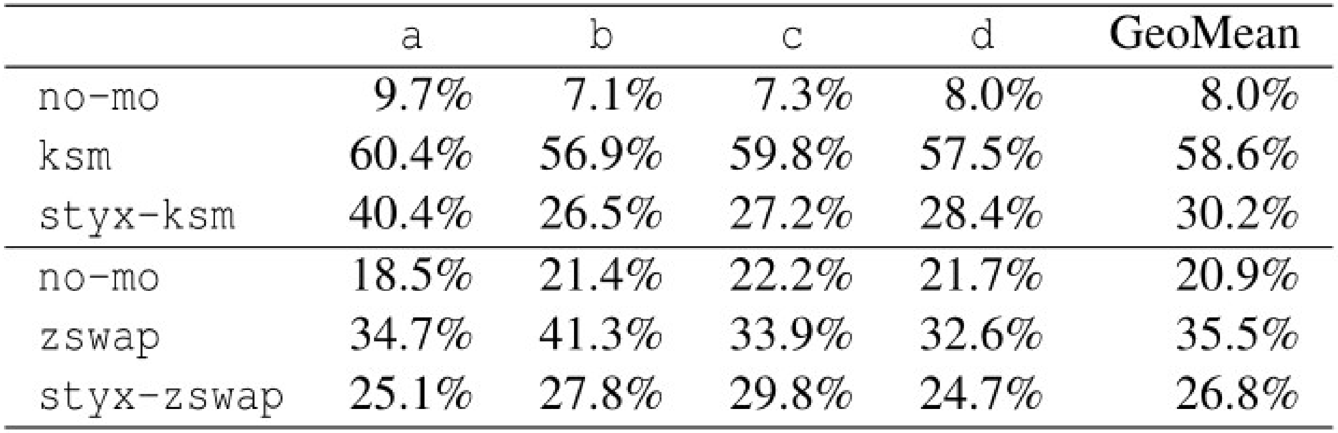
# Evaluation



(a) p99 latency

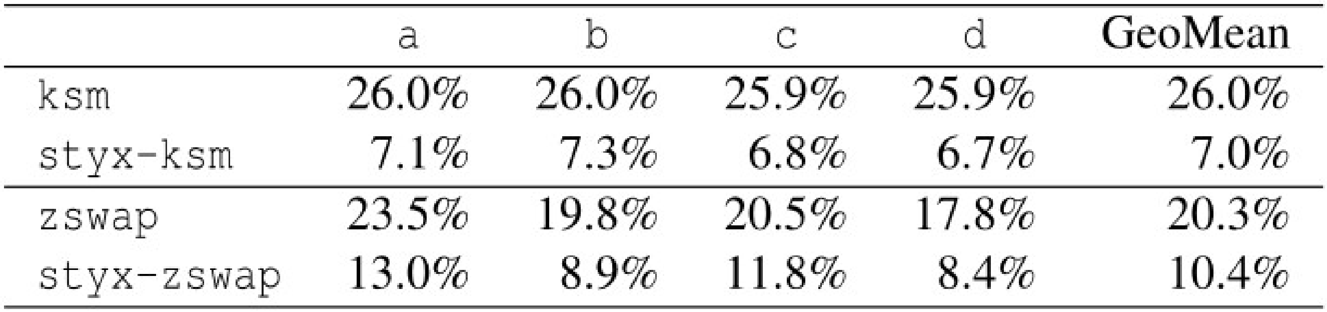
(b) Average latency

# Evaluation



| | a | b | c | d | GeoMean |
|---|---|---|---|---|---|
| no-mo | 9.7% | 7.1% | 7.3% | 8.0% | 8.0% |
| ksm | 60.4% | 56.9% | 59.8% | 57.5% | 58.6% |
| styx-ksm | 40.4% | 26.5% | 27.2% | 28.4% | 30.2% |
| no-mo | 18.5% | 21.4% | 22.2% | 21.7% | 20.9% |
| zswap | 34.7% | 41.3% | 33.9% | 32.6% | 35.5% |
| styx-zswap | 25.1% | 27.8% | 29.8% | 24.7% | 26.8% |

LLC miss ratio under different configuration



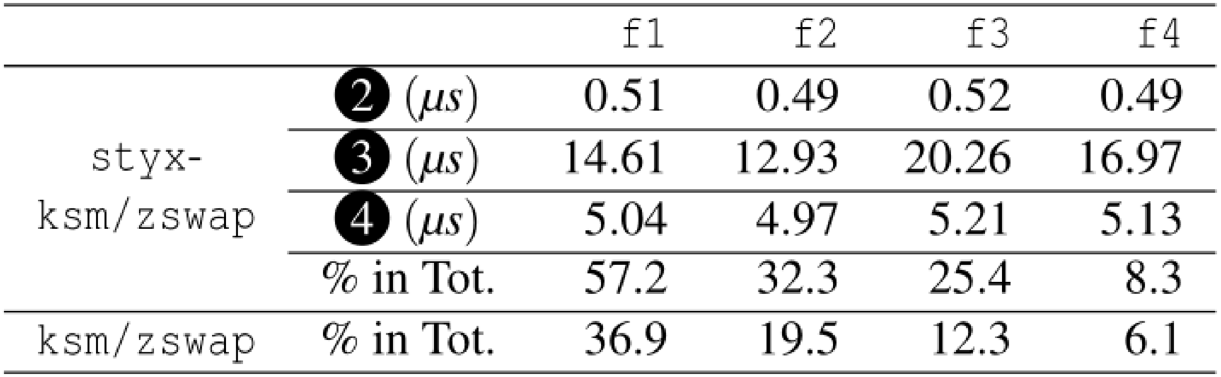
| | a | b | c | d | GeoMean |
|---|---|---|---|---|---|
| ksm | 26.0% | 26.0% | 25.9% | 25.9% | 26.0% |
| styx-ksm | 7.1% | 7.3% | 6.8% | 6.7% | 7.0% |
| zswap | 23.5% | 19.8% | 20.5% | 17.8% | 20.3% |
| styx-zswap | 13.0% | 8.9% | 11.8% | 8.4% | 10.4% |

cpu utilization under different configuration

# Evaluation

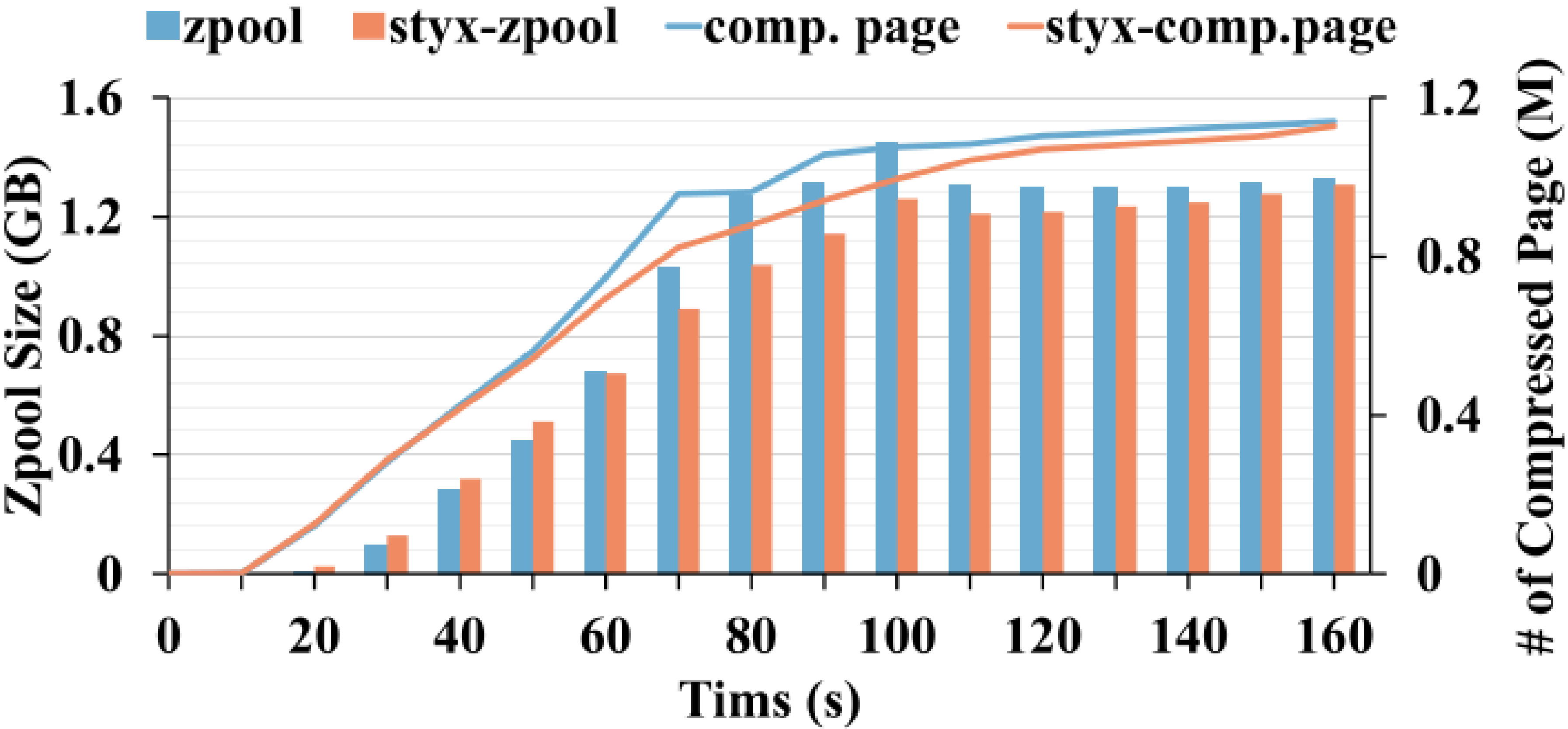
Time breakdown of styx offloaded kernel feature.

- f1: comparision of ksm
- f2: checksum of ksm
- f3: compress of zswap
- f4: decompress of zswap



| | | f1 | f2 | f3 | f4 |
|---|---|---|---|---|---|
| styx-ksm/zswap | ❷ ($\mu s$) | 0.51 | 0.49 | 0.52 | 0.49 |
| | ❸ ($\mu s$) | 14.61 | 12.93 | 20.26 | 16.97 |
| | ❹ ($\mu s$) | 5.04 | 4.97 | 5.21 | 5.13 |
| | % in Tot. | 57.2 | 32.3 | 25.4 | 8.3 |
| ksm/zswap | % in Tot. | 36.9 | 19.5 | 12.3 | 6.1 |

# Evaluation


zpool
styx-zpool
comp. page
styx-comp.page
Zpool Size (GB)
1.6
1.2
0.8
0.4
0
# of Compressed Page (M)
1.2
0.8
0.4
0
0
20
40
60
80
100
120
140
160
Tims (s)

# Impact on SNIC application

Under maximum 25Gbps network bandwidth:

- running regular expression matching (rem) on SNIC
- SNIC application needs at most 5 cores at a package size of 128B, and need only 1 core at a package size of 1024B
- STYX utilizes only ~30% of a core when running compression of zswap
- STYX has little impact on SNIC application(13.83us → 13.85us)