VM Live Migration Speedup in Xen

Xiaowei Yang, Tao Hong

www.huawei.com

HUAWEI TECHNOLOGIES CO., LTD.

HUAWEI

Agenda

- **Background**
- **Issues and Proposals**
- **Evaluation**

Background

- VM live migration scenarios
  - Host evacuation on maintenance
  - Load balance between hosts
  - Power consumption optimization inside the cluster

- Challenges
  - Success ratio
    - Fallback on exception
  - Migration time
    - 1-VM or multi-VMs
  - VM downtime
  - VM performance impact
  - Required resources (CPU%, network TX/RX)

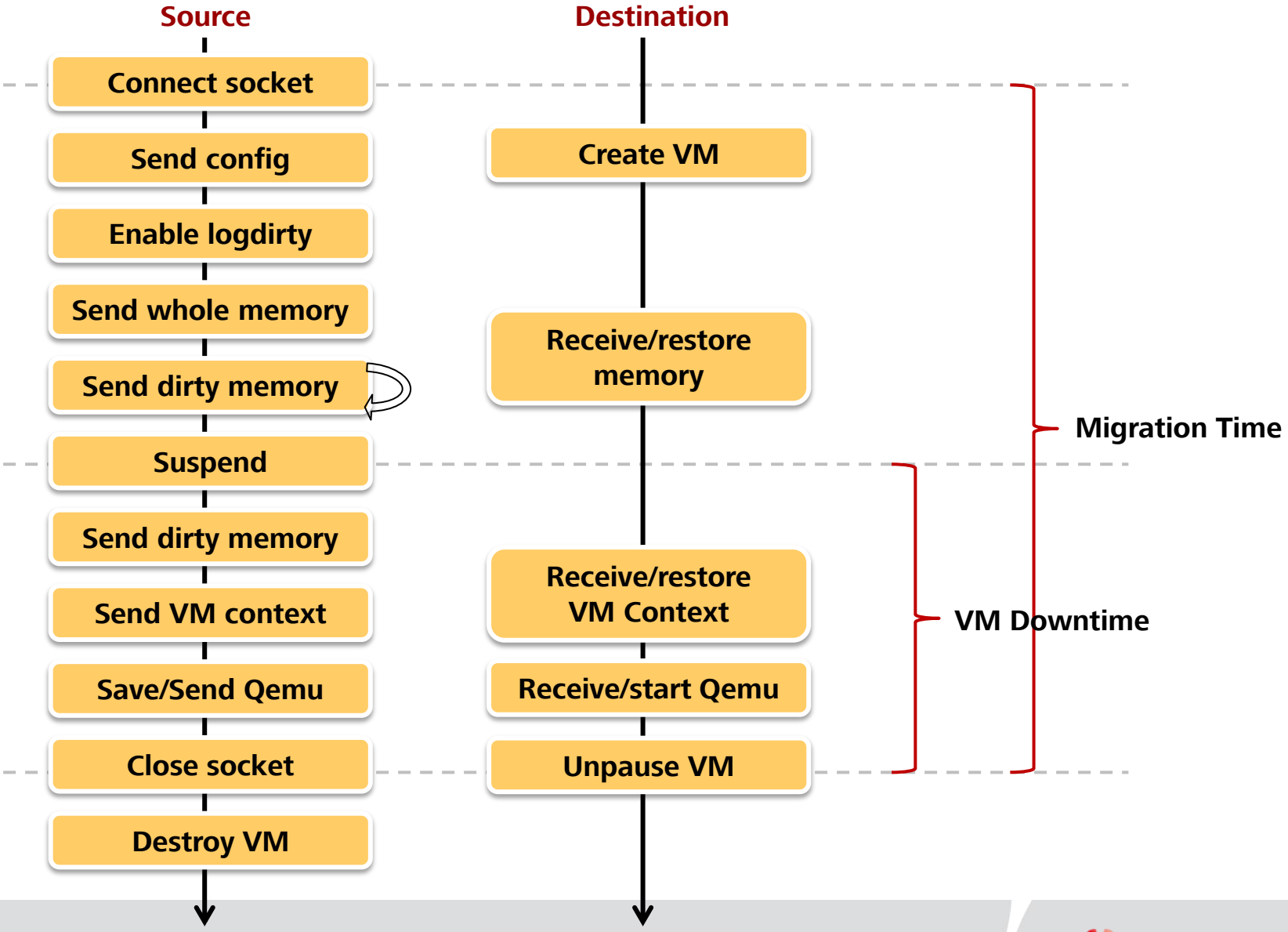
Goals

- Evaluate the function and performance of VM live migration in Xen

- Increase VM live migration success ratio, especially when the host is heavily loaded; fallback on exception

- Increase VM live migration efficiency in terms of the indicators mentioned in the previous page

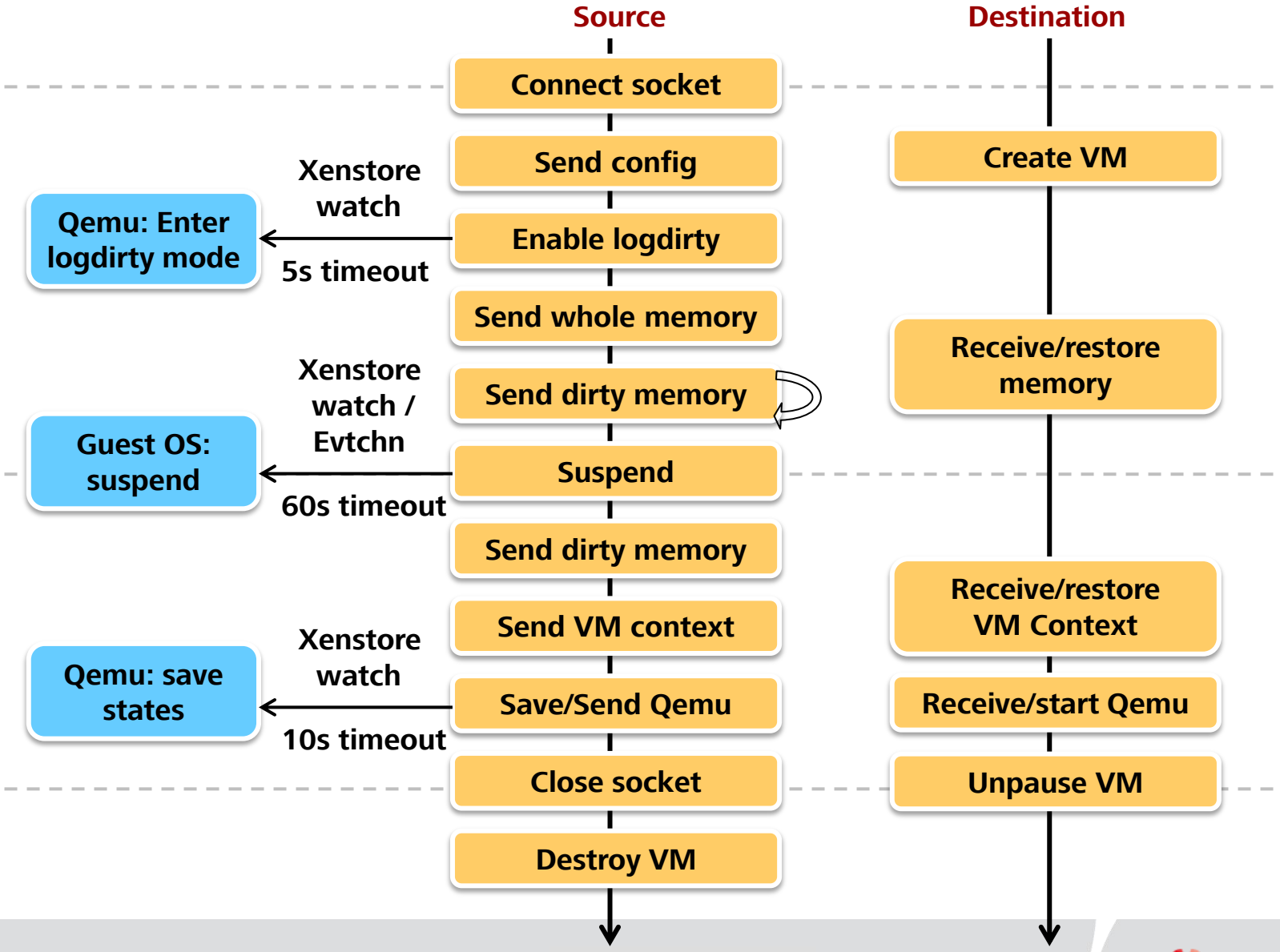
VM Live Migration Algorithm



Agenda

- Background
- Issues and Proposals
- Evaluation

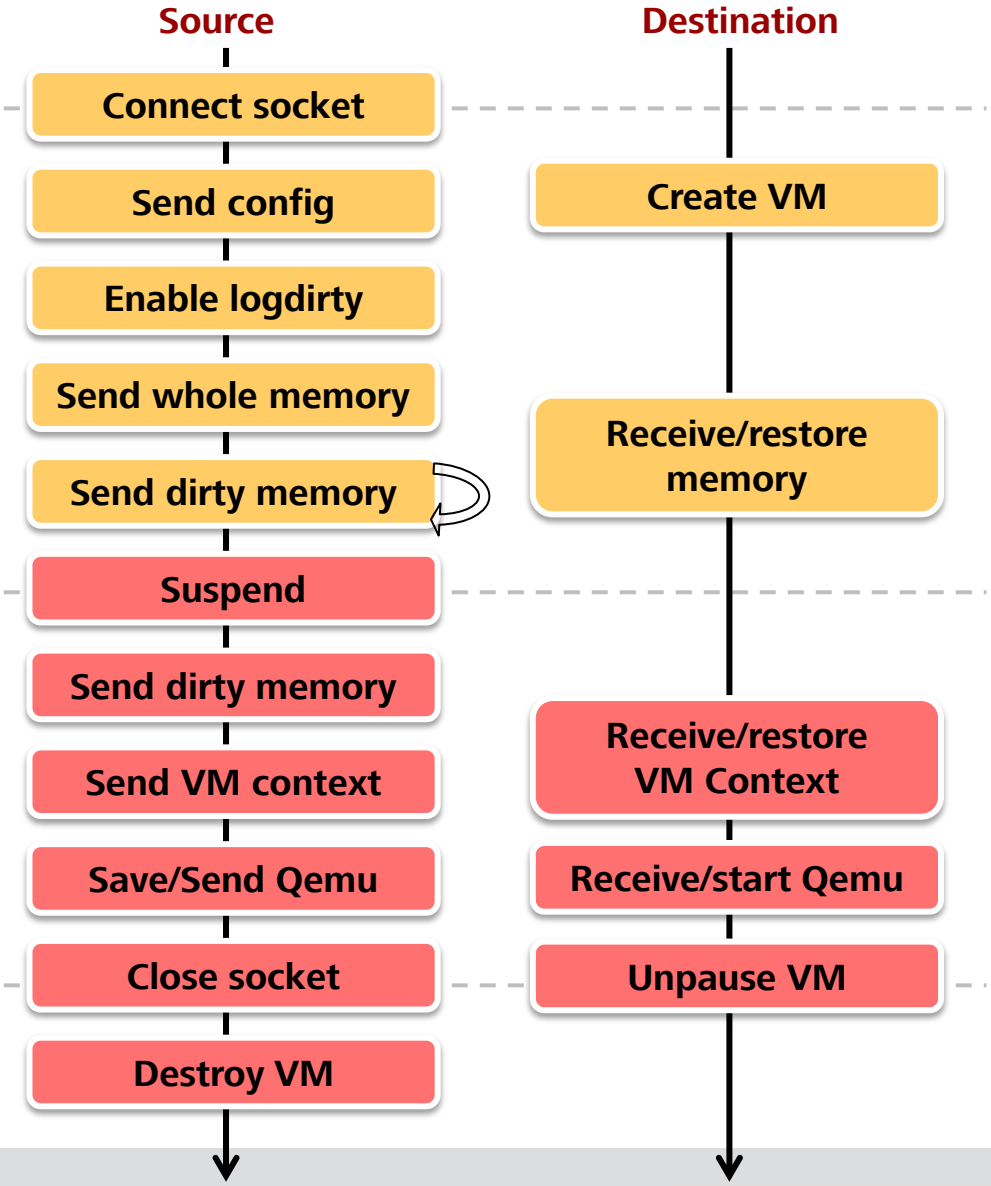
Issue 1: Timeout



Proposal

- Main reason for timeout
  - Long delay before Xen watch triggers, when dom0 serves multi-VM and is stressed. Extreme case: delay = 13s
- Root cause
  - There are claims that xenstore has performance & scalability issue, e.g. *Oxenstored* paper
- Workaround
  - Inter domain: use evtchn
  - Inter process: use Linux IPC, e.g. shared memory, semaphore, named pipe

Issue 2: Lack of Fallback



Without fallback on failure, VM will stop execution on either side!

Possible reasons of failure after suspension:

- Network / socket connection breaks
- VM fails to boot up, e.g. Qemu dump image corrupts, etc.

Proposal

- Network / socket connection breaks
  - Source: domain resume is OK - VM's memory still exists; devices needs recreation
- VM fails to boot up at destination
  - Source: socket connection remains till the very end
  - Destination:
    - On failure, nack Source -> domain resume
    - On success, ack Source -> domain destroy

Issue 3: Poor Performance

	Migration time (s)	VM downtime (s)	Dom0 CPU%	Throughput (Mbps)	Perf impact
Idle	25.8	1.2	81.6	633.0	n/a
VDI	26.3	1.4	115.0	661.0	n/a
Webserver	44.6	~2.0	108.0	799.0	~30%

- **Migration time:** long
- **Required resource:** high
  - Network throughput becomes a bottleneck for multi-VM live migration
- **VM downtime:** long
- **Performance Impact:** high

Proposal – Compression

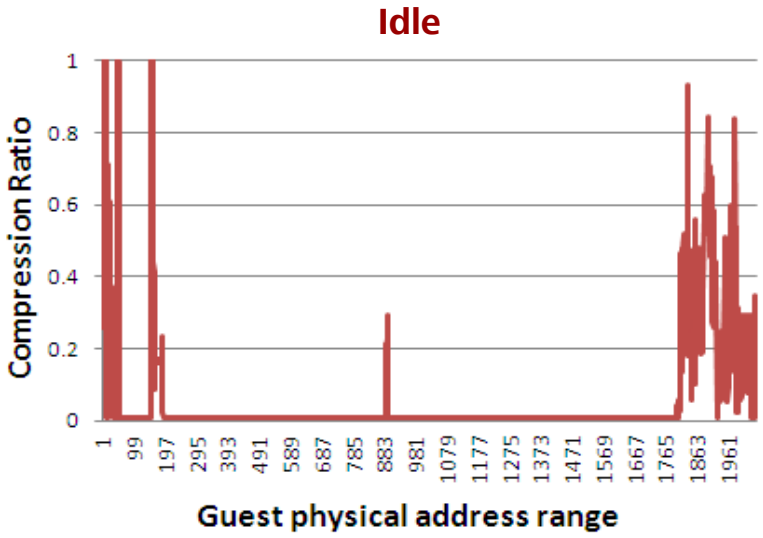
	Migration time (s)	VM downtime (s)	Dom0 CPU%	Throughput (Mbps)	Compression ratio %
Orig	25.8	1.2	81.6	633.0	1:1
Libz	25.7	1.2	105.2	32.0	22:1

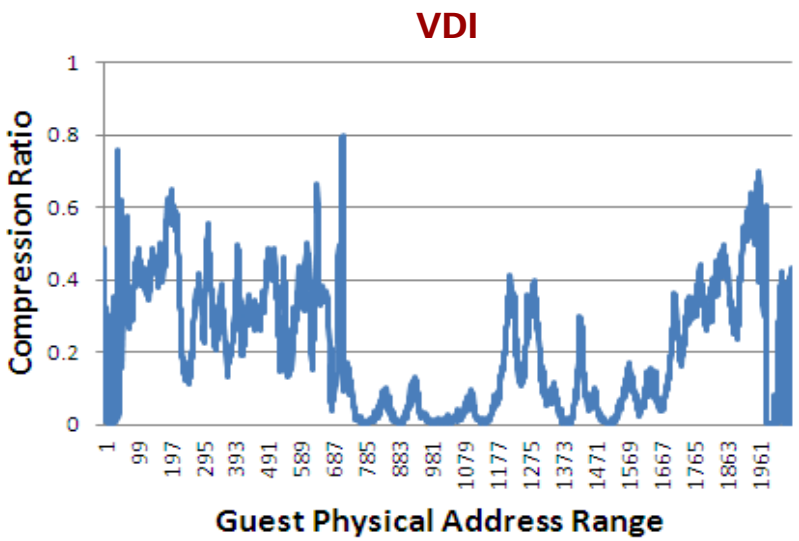
- Throughput decreases dramatically
- Migration time unchanged. Why?

	Compression time (s)	Transfer time (s)
Orig	0	22.6
Libz	21.2	0.7

- Compression (libz) consumes most of the time

Proposal – Selective Compression





- Compression ratio complies to principle of locality somehow
- Introduce selective compression: skip xMB if compression ratio < y:1

	Migration time (s)	Compression time (s)	Transfer time (s)	Throughput (Mbps)	Compression ratio %
Orig	25.8	0	22.6	633.0	1:1
Libz	25.7	21.2	0.7	32.0	22:1
Libz+opt	21.9	15.4	2.6	99.6	7.6:1

Proposal – Multithread

- Executes compression and transfer simultaneously

	Migration time (s)	Compression time (s)	Transfer time (s)
Orig	25.8	0	22.6
Libz	25.7	21.2	0.7
Libz+opt	21.9	15.4	2.6
Libz+opt+m.t.	20.2	15.3	3.2

	Throughput (Mbps)	Dom0 CPU%	Compression ratio %
Orig	633.0	81.6%	1:1
Libz	32.0	105.2	22:1
Libz+opt	99.6	104.0	7.6:1
Libz+opt+m.t.	114.0	120.0	7.6:1

Agenda

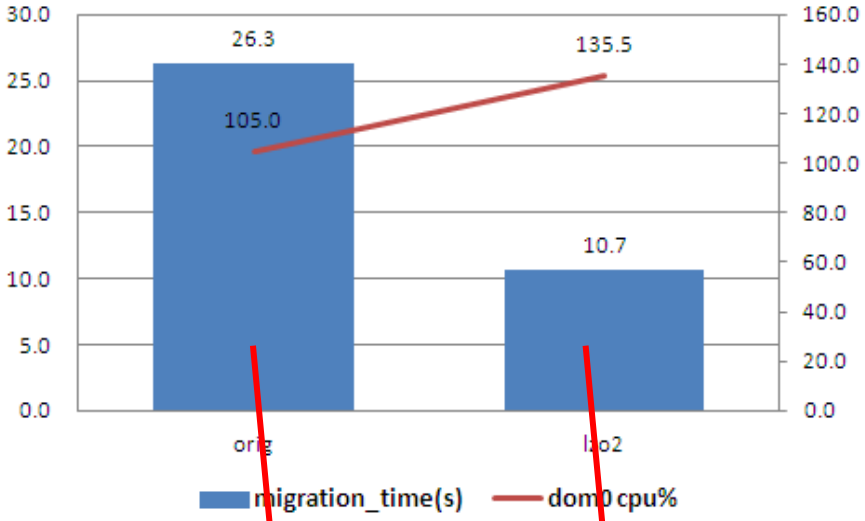
- Background
- Issues and Proposals
- Evaluation

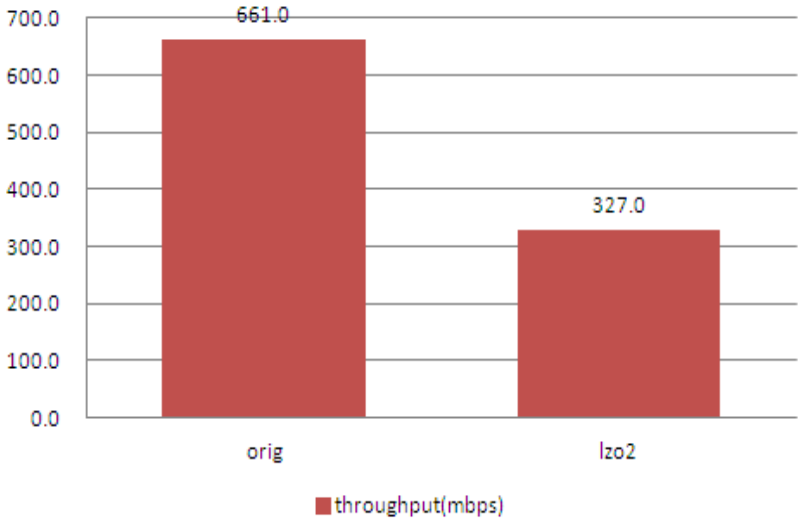
Experimental Environment

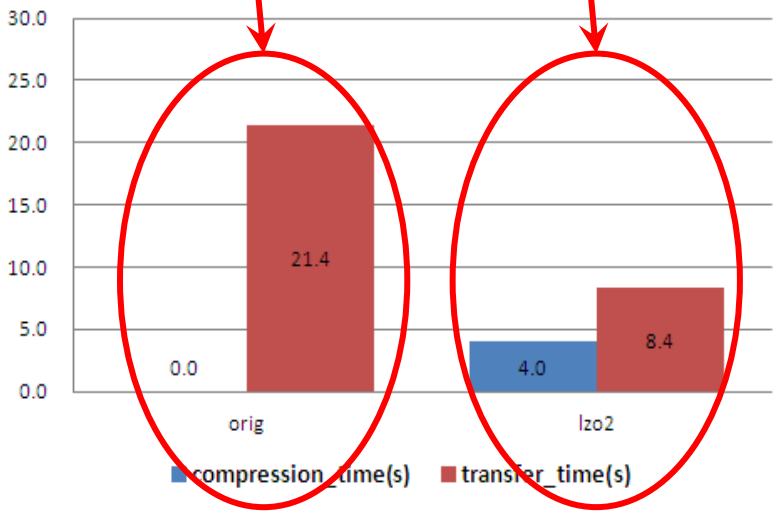
Host Configure	
Processor	2x Intel X5620 @ 2.40GHz, SMT enabled
Memory	96GB (8GB 800GMHz DDR3 x12)
Storage	Huawei Oceanspace S5300, thru IPSAN
Network	Intel Corporation 82576 Gigabit

VM Configure	
VDI Scenario	2 vCPU, 2G vMem; Windows 7 Workload: Office, IE, PDF, Java
Webserver Scenario	2 vCPU, 2G vMem; Suse11 sp1 Workload: Siego, 1024 current users Data: 100K * 35000URLs = ~ 3.5GB Client accesses all URLs randomly

VDI – 1VM

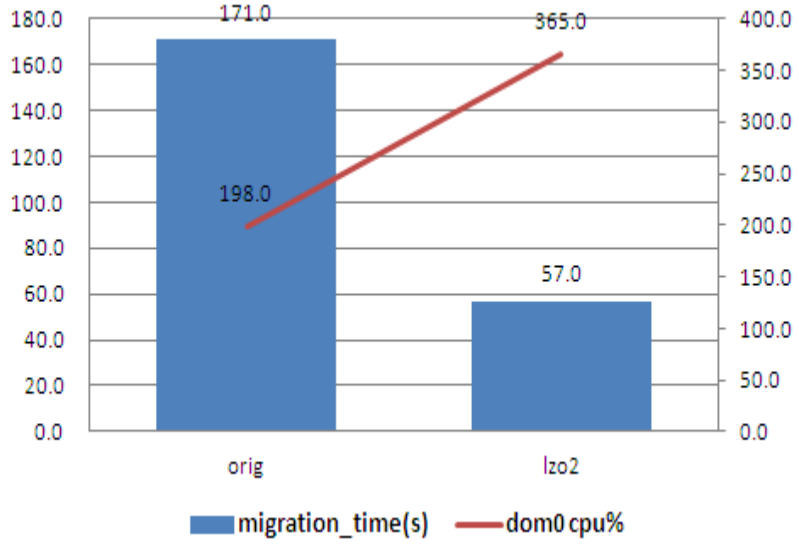


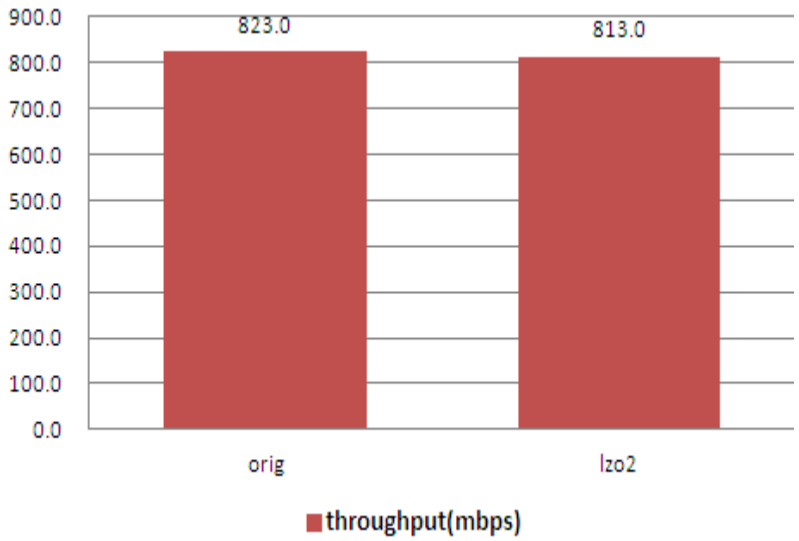




- Migration time is reduced from 26.3s to 10.7s (2.4X faster)
- Multithread (compression/TX) helps further
- With compression (lzo2),
  - TX time down
  - Throughput down – more scalable

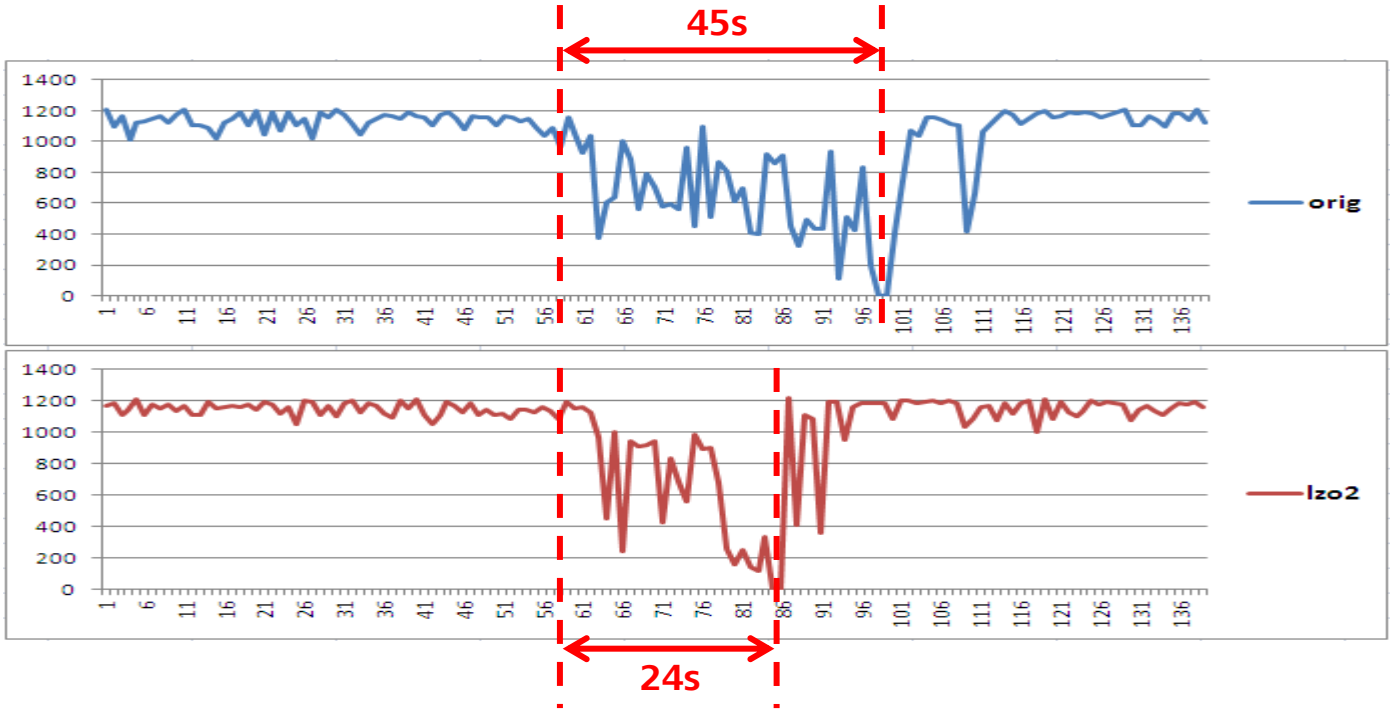
VDI – Multi-VM





- Total migration time is reduced from 171.0s to 57.0s (3X faster)
- The advantage of scalability is more obvious

Webserver



- Total migration time is reduced from 44.6s to 23.8s (1.9X faster)
- Performance impact is big – need further improve

Takeaways

- To increase VM live migration success ratio
  - Avoid timeout：Improve the notification efficiency between migration and other processes
  - Make sure VM lives at least on one side: fallback whenever there is a failure at Destination
- To increase VM live migration efficiency
  - Compress memory data before TX. Different algorithms matter a lot
  - Do compression and data transfer parallelly
  - Choose a wise compression ratio to skip data block
  - Live migration can be 1.9X – 3X faster per our test
- There is still room to improve

Thank you

www.huawei.com