BDTC
2014 中国大数据技术大会
BIG DATA TECHNOLOGY CONFERENCE
暨第二届CCF大数据学术会议

构建低时延大数据系统

Low Latency BigData System

Dr Zhaohong Lai (CEO)

北京卓越讯通科技有限公司

2014.12.08

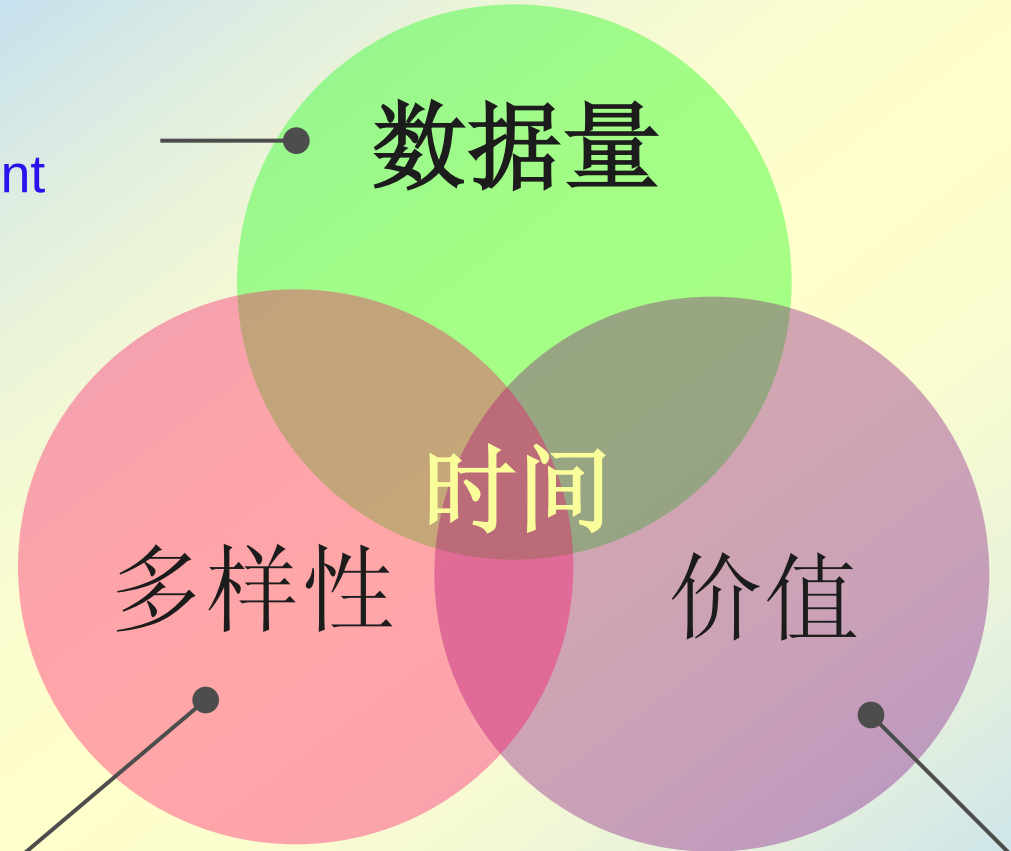
目录

大数据：时间驱动

人类数字化进程:

Explosive increment

Time matters



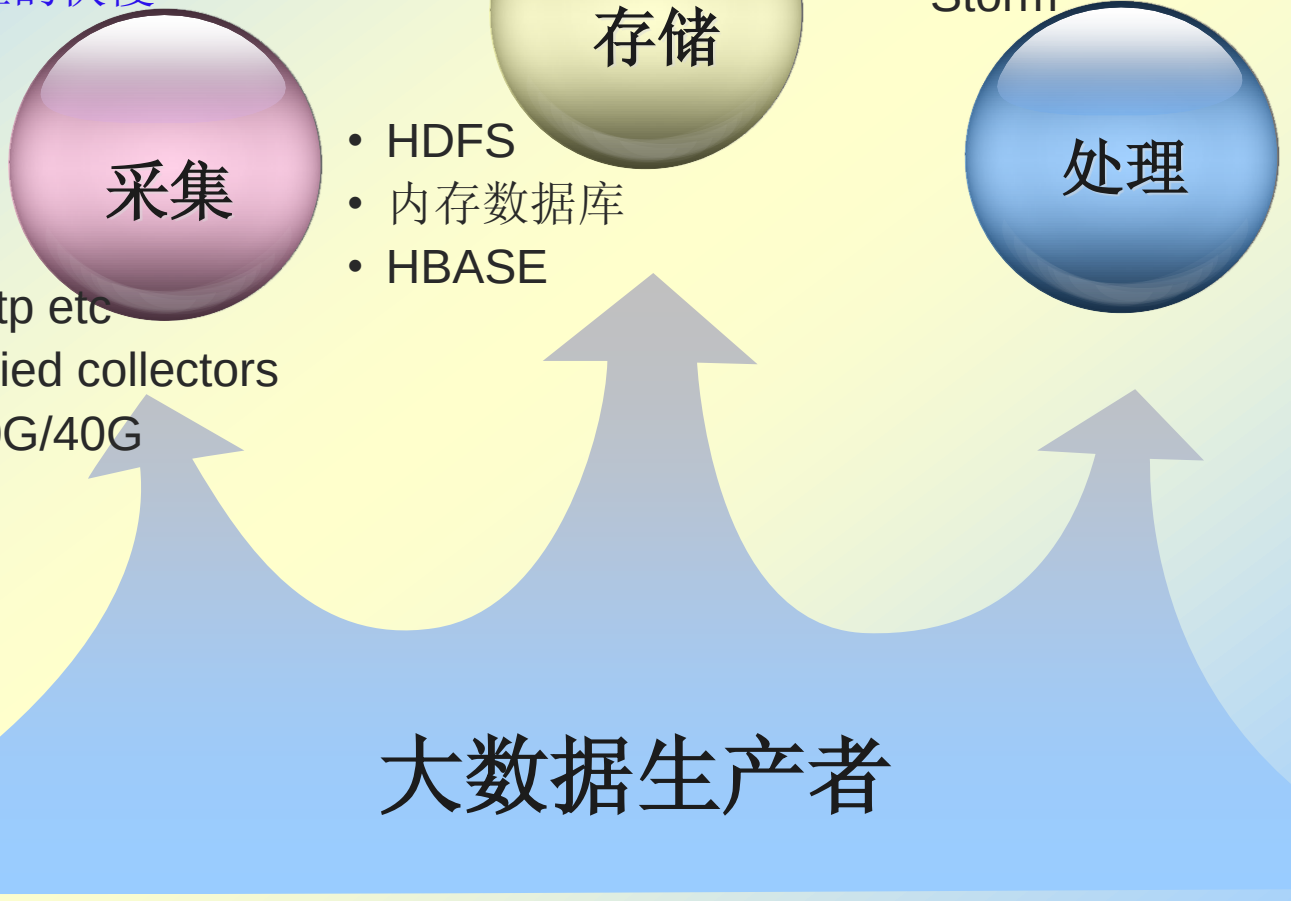
结构化转为

非结构化为主流

价值和时间紧密相关

Velocity & Latency

- 在大数据中，Velocity 可以理解为多个含义
  1. 数据产生的快慢
  2. 数据采集的快慢
  3. 数据传输的快慢
  4. 数据处理的快慢



Latency Importance: 大数据的交易平台

- Latency is the key
- 50% Signal
- 50% Latency
- 3us limit (tick to trade)

交易用户

1. 用户资料
2. 统计数据
3. 历史数据库等等

期货市场行情

国内外市场行情，价格等等

外汇数据

FX Broker 数据接口

新闻动态

1. 行业新闻
2. 热点论坛，微博数据等等

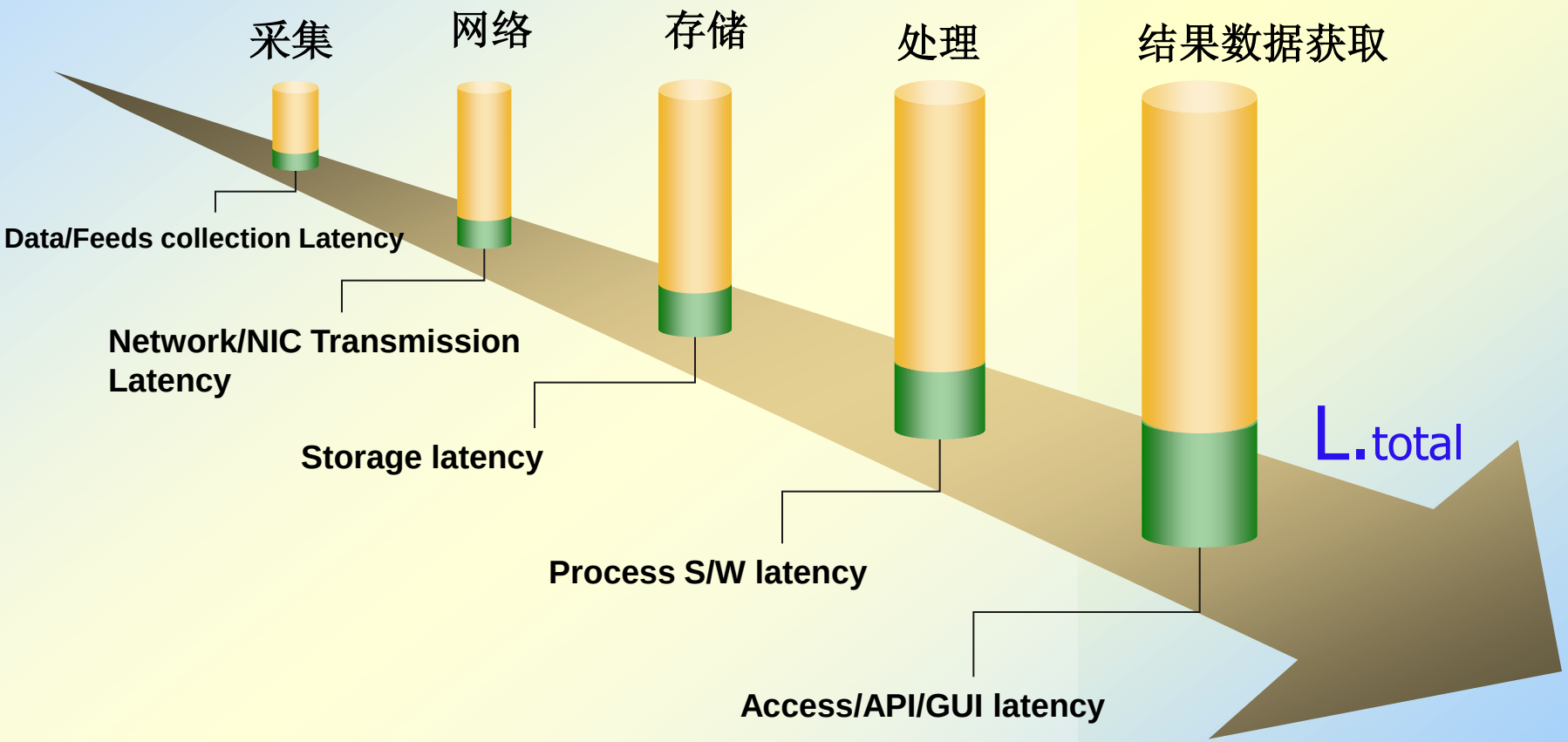
交易所数据

Equity, Option, Future Exchange 数据接口

时延: Where is the limit?

- For human being: Light Speed in theory
- But fiber/electronic transmission travels at 70% of light speed (考虑30% 的衰减) in reality.

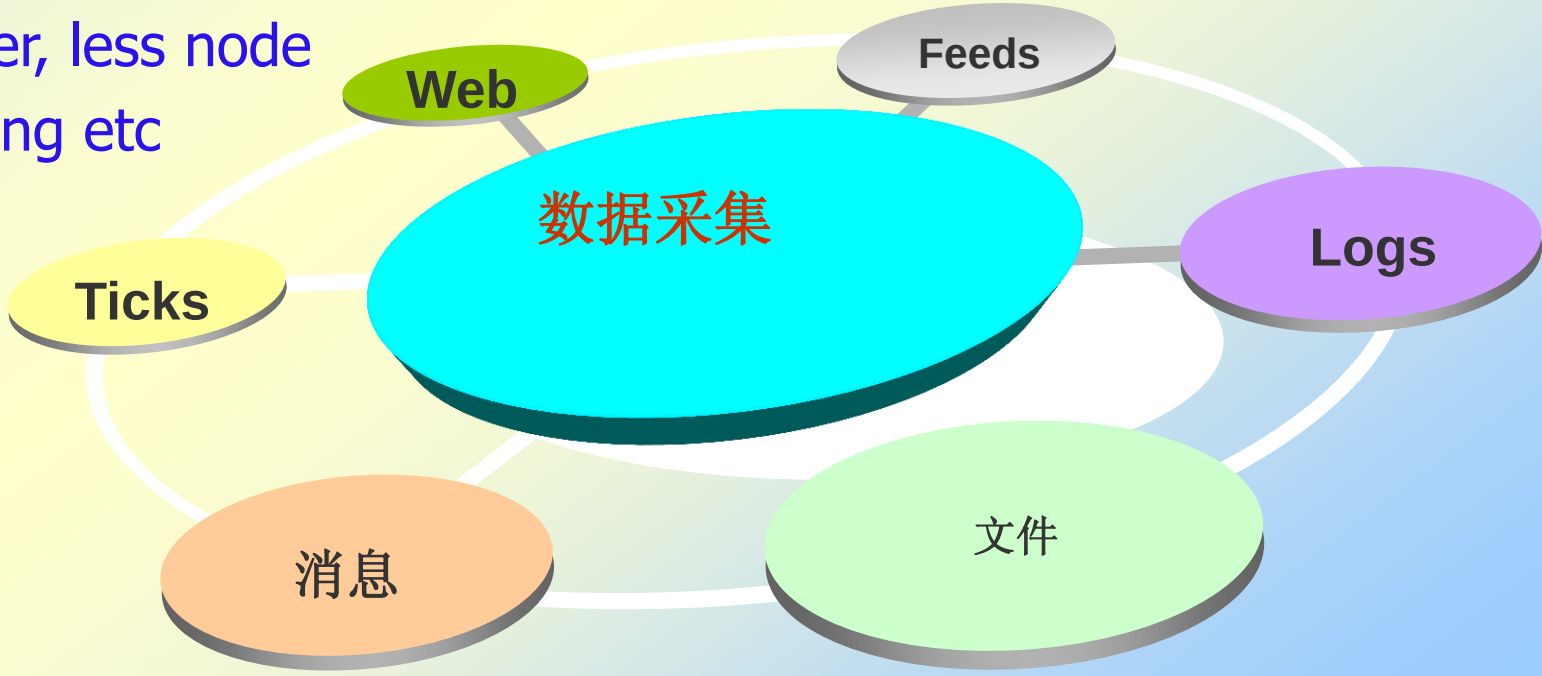
时延：是一个Chain



$L.total = L_1 + L_2 + \ldots + L_n$

采集时延

- 文件批处理, 日志实时采集, 消息实时,交易所Ticks
- Batching 设计： FTP 典型 (TCP Based, secured line), Batching Mode
- Streaming 设计：Storm
- 软件时延：缓存、TCP connection setup, transmission
- 硬件实延：通用芯片处理问题 (ARM eg)
- TCP Buffer, less node
- BDP setting etc

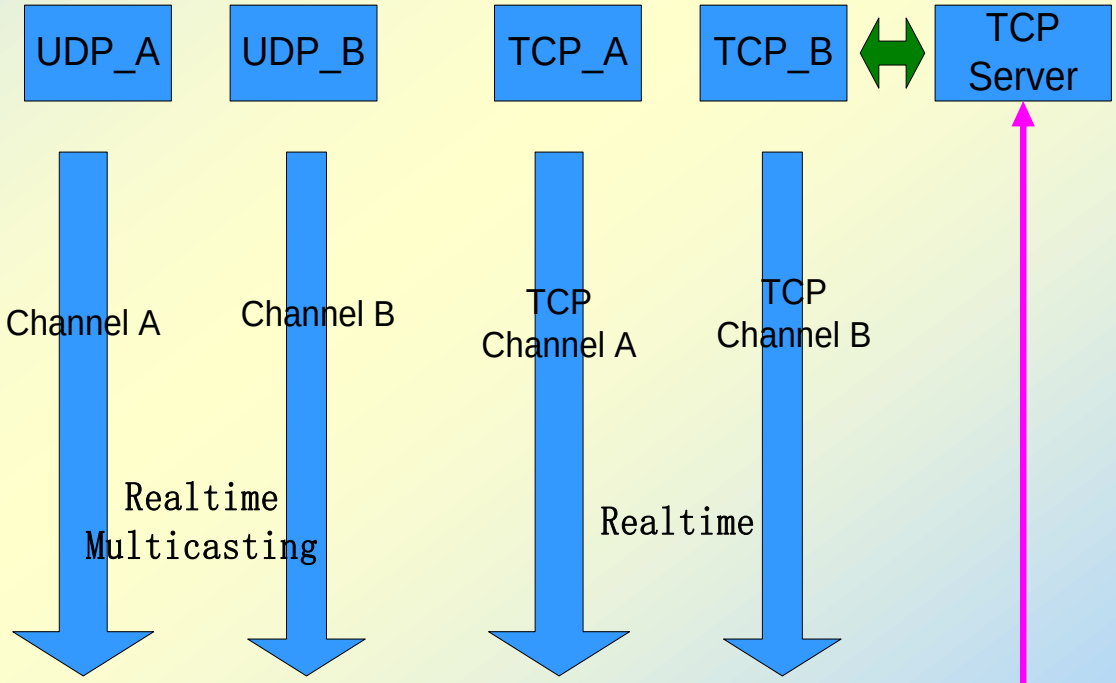


采集时延: FPGA FeedHandler

- Through FPGA dedicated processing
- Example, HFT: 10G NIC card, FPGA to fast process pkts + FeedHandler
- All the work completed on FPGA card
- CPU is only for management and data backup.
- From tick to trade , about 3-10us,
- At least 10times faster!

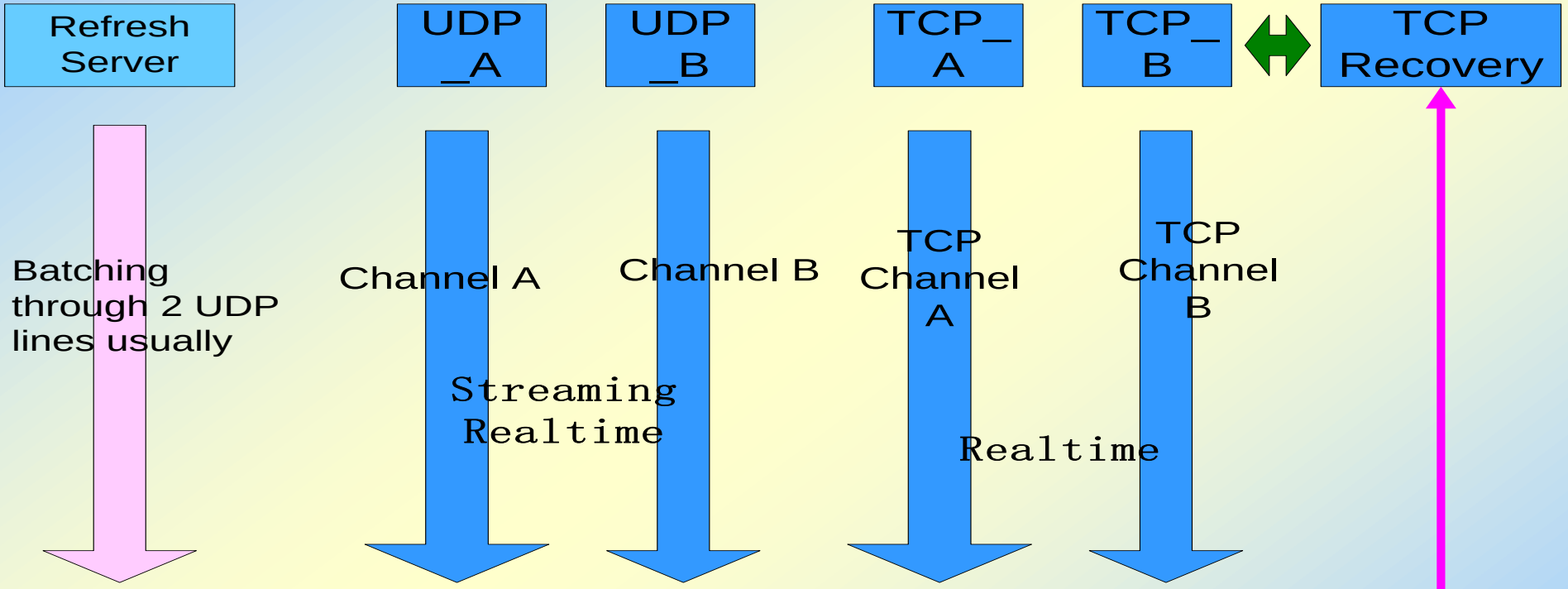
网络时延: 设计是关键

- UDP 比 TCP 快!
- 多通道UDP 采集
- UDP 可靠性通过 TCP Recovery 来弥补
- Always Two lines
- Line Arbitration



Step 1: UDP A and B Channel, get one, ignore the other
Step 2: Pkt loss, cache realtime Ticks, send pkt loss seq to TCP Server for re-trans
Step 3: Rcv all the loss pkts, to sync the cache buffer with the TCP buffer
Step 4: if multiple loss event happens, repeat step 1-3,

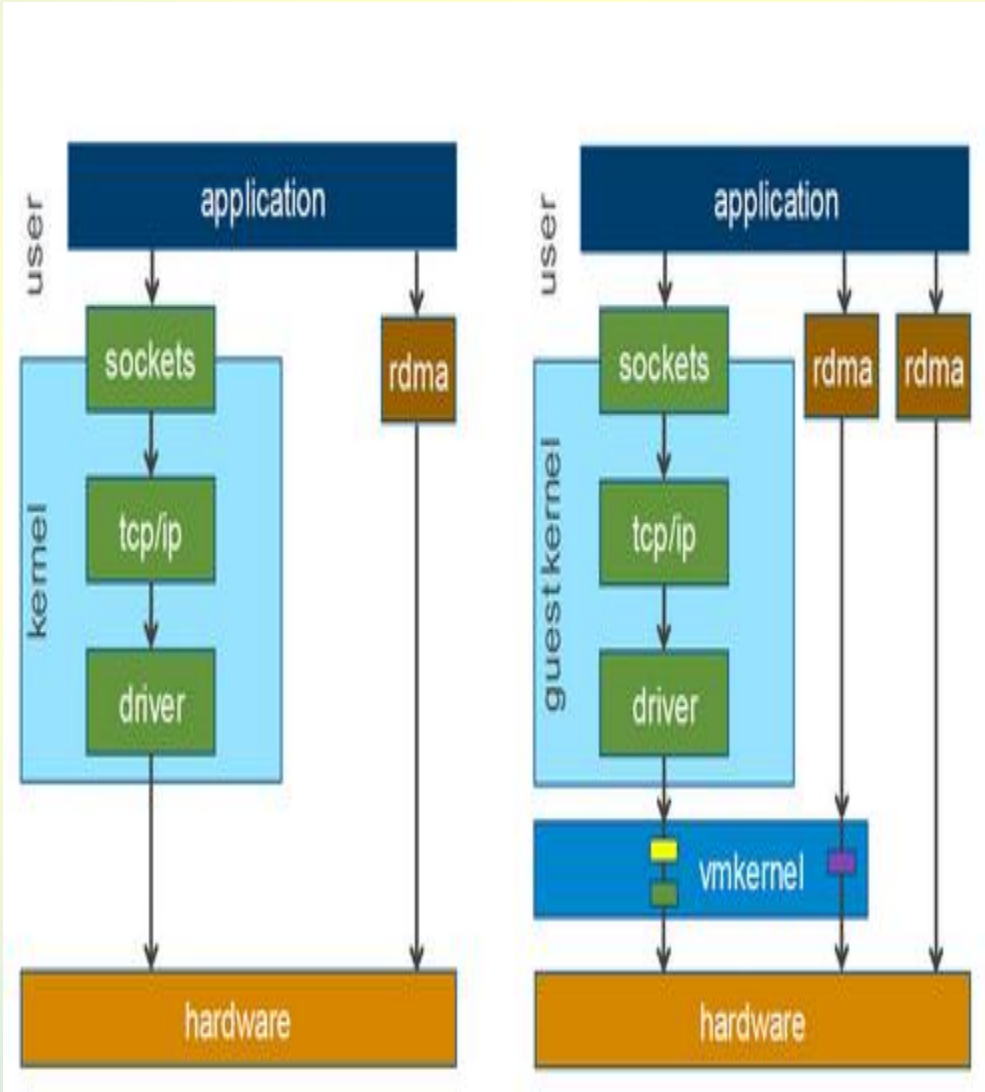
Stock Exchange batching and streaming



Step 1: Request all the Refresh/Reference data from Stock Exchange, while caching realtime UDP channel data
Step 2: UDP A and B Channel, get one, ignore the other
Step 3: Pkt loss, cache realtime Ticks, send pkt loss seq to TCP Server for re-trans
Step 4: Rcv all the loss pkts, to sync the cache buffer with the TCP buffer
Step 5: if multiple loss event happens, call the pkt loss handler

RDMA & kernel bypass

- RDMA: Remote Direct Memory Access
  基本原理，DMA Controller
  直接I/O 访问 （内存），
  不通过CPU
- 特点：高速访问，
  FPGA 前端实现预处理动作
- Kernel Bypassing:
  Direct access to network traffic
- UDP latency: back to back 2-3 us
  with 1 meter distance



存储, Persistency,实时处理

- 经常有场景：交易数据不能丢！
- 比如FIX 消息，要求快速处理和保证每个节点的Persistency
- Memory VS Disk Solution
- KDB, Spark : to process data in memory
- What about memory crashes or hardware Down?
  Data Loss!

  Data written to disk? File write causes milliseconds latency!
  Faster memory write? Strong power support needed!

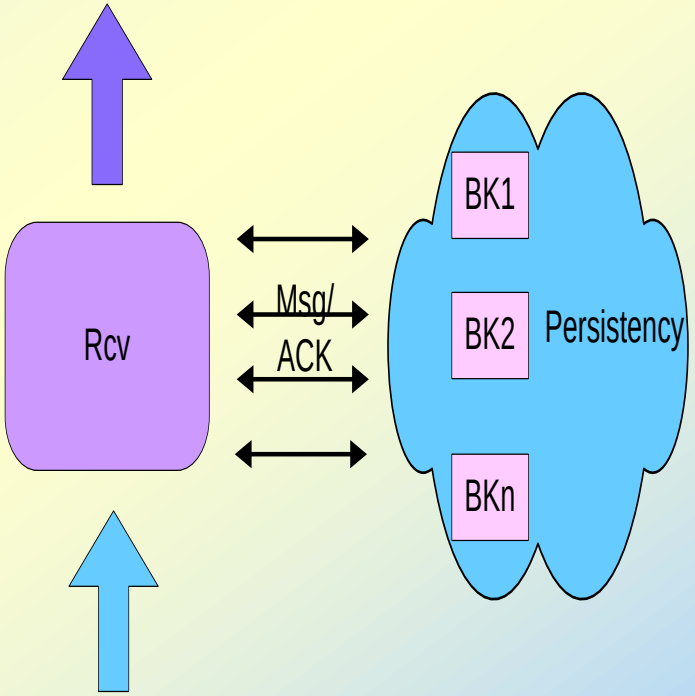
Low Latency Persistency

- Disk 存储再处理，毫秒级别
- 内存有可能丢数据

- Multicasting to a cloud,
  Use network for backup;

- Send the message to multiple machine,
  whichever faster, quicker ack
  Assume persistency done in network!

- Integrated with RDMA + Cloud
  Latency : 6-7us!



软件: Cache

- Understand Level 1 Cache, Level2 Cache and Memory
- CPU fetches coding/data from register first, then Level1,2,3 and Memory
- Register access at single digit nanosecond level.
- Level1 is about 10 times faster than Level2, 30 times faster than regular memory access
- Pre-fetch could do some help (most is done by CPU automatically)

软件: Context Switch

- Context is Expensive!
- Consider 5000 threads running on a machine!
- CS usually takes 3 microseconds!

For HFT, game is over!

CPU	HZ	CS latency
Intel 5150	2.67GHZ	4300 ns
Intel E5440	2.83GHZ	3600 ns
Intel E5520	2.27GHZ	4500 ns
Intel X5000	2.67GHZ	3000 ns
Intel L5630	2.13GHZ	3000 ns

软件: CPU Affinity

- Affinity: bind a thread to a specific core as to reduce CS!
- With affinity set, latency cut 50%

CPU	HZ	CS latency
Intel 5150	2.67GHZ	1900 ns
Intel E5440	2.83GHZ	1300 ns
Intel E5520	2.27GHZ	1400 ns
Intel X5000	2.67GHZ	1300 ns
Intel L5630	2.13GHZ	1600 ns

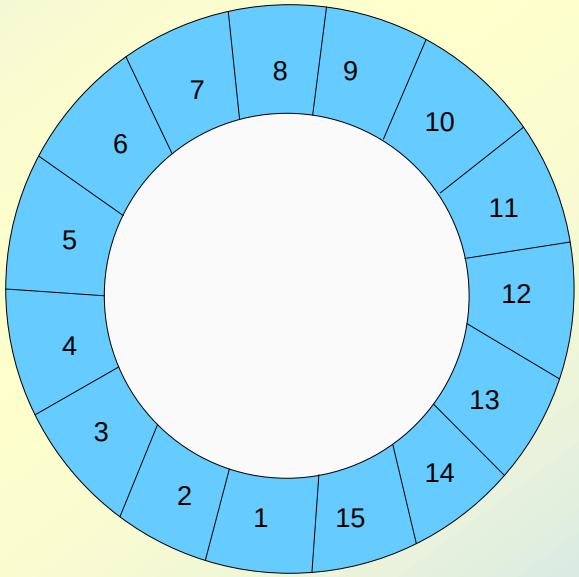
Lock-Free Data Queue

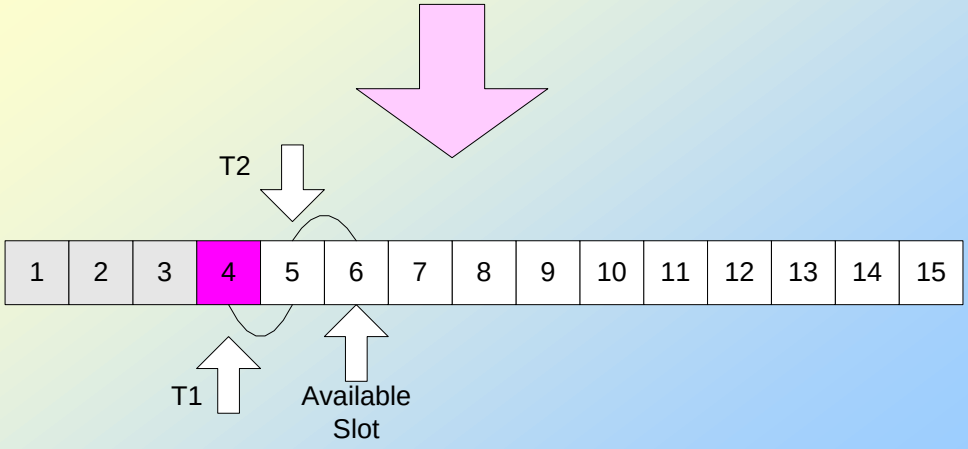
- A lock put a threading into CS wait, causes a 3u latency
- Lock-Free coding with atomic operations
  increment, decrement and CAS (write)
- With increment/decrement for index moving
- Individual thread update its own slot with CAS
- A good example – LMAX Disruptor
  This is not the best impl though, there are still better ones.

Quick Look: Disruptor

- Ring Data Buffer
- Thread reads Avail-Slot
- Current avail-shot=4
- T1 takes slot4, cursor moves to 5
- T2 takes slot5, cursor moves to 6
- Cursor changes with atomic increment
- T1/T2's write with CAS

- Latency cuts to 100-300ns





Memory Address as Hash

- Another extreme fast processing is to use memory address for
- For example, clientID =['20050','20159'] and
  securityID=['30112', '30229'] range is what we want!

rather than the string comparison, we have base = 20050 and 30112
and set: ary[0-158][0-118]=true;

```
if(ary[cID][tID]) { next tradeChecking(xxx); }
```

this usually will be 50-100x faster!

- Require good data structure design with memory space as tradeoff

高频交易 latency

- After all these latency cut + system optimization
- High Frequency Trading from Tick to Trade could be done within 5 us, including the HFT algo execution time.

低时延高速10G/40G大数据采集场景

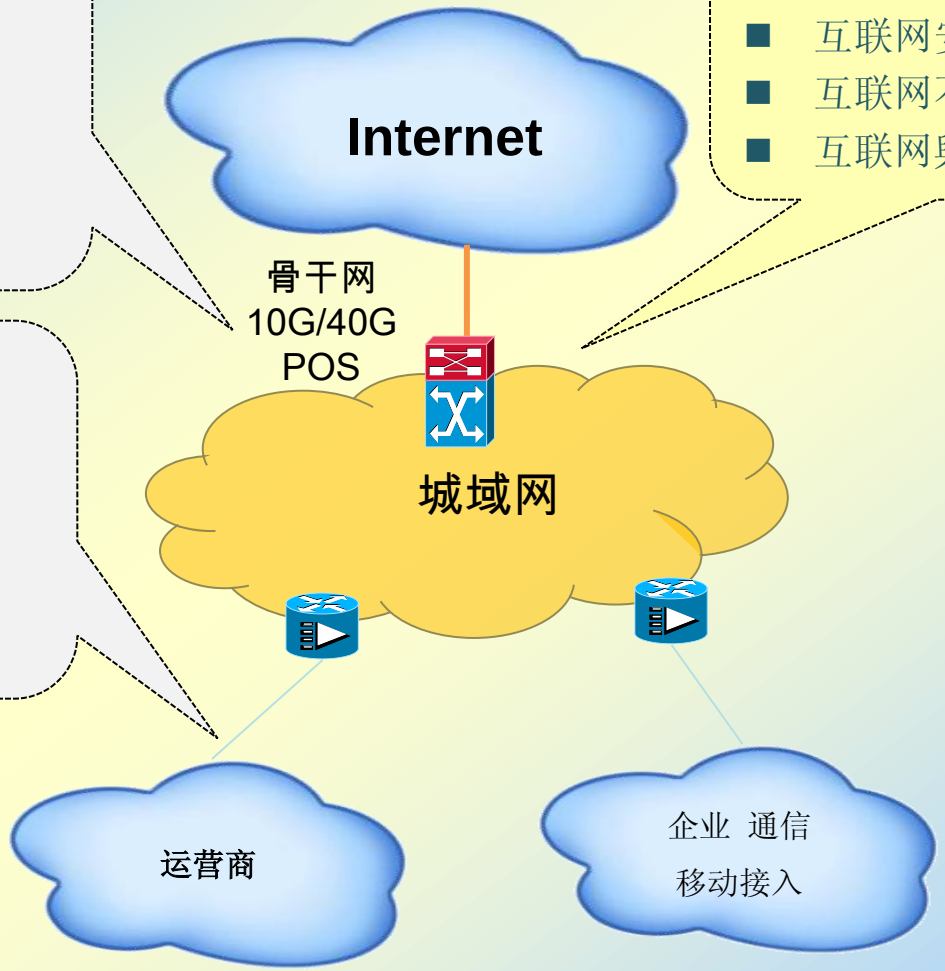
安防、反恐

- 侦听、安防
- 语音监听

互联网内容监控解决方案

- 互联网安全监控
- 互联网不良信息检测
- 互联网舆情监测

- 海量数据需要拦截
- 组建低迟延系统



10G/40G/100G 高速大数据采集设备

Latency: line rate – cut to single microsecond level

- ✓2 x Cavium CN68xx 多核处理器
- ✓32 cores，1.2GHz
- ✓32 GB DDR3 DRAM / PDB，共64GB
- ✓操作系统：Linux、Cavium SE
- ✓全双工 40G 包处理能力
- ✓单板支持 4 x 10G（SFP+）
- ✓可通过RTM扩展至12 x 10G/1G（SFP+/SFP）
- ✓板级处理器： Freescale QorIQP2020 （800MHz）
- ✓支持热插拔
- ✓支持 IPMI v1.5 / PICMG 3.0 规范
- ✓支持 SNMP 管理协议

多U 10G/40G/100G 高速采集设备

包交换板

包处理板

总结

- 过去几年，基于Hadoop 等技术推动了大数据产业
- 但是低时延的核心技术并没有什么改变，低时延是一个latency chain, 总结如下：
  - 基于FPGA的高速采集 and InfiniBand
  - 用UDP multiple lines 的传输方式
  - RDMA/Kernel bypass 降低网络延时
  - Persistency 通过网络cloud ack方式
  - 减少context switch
  - 设计lock-free data queue
  - 内存地址as hash