

腾讯信鸽实时推送演进与实践

甘恒通

腾讯大数据高级工程师



数据平台部

目录

01

背景与挑战

02

解决方案

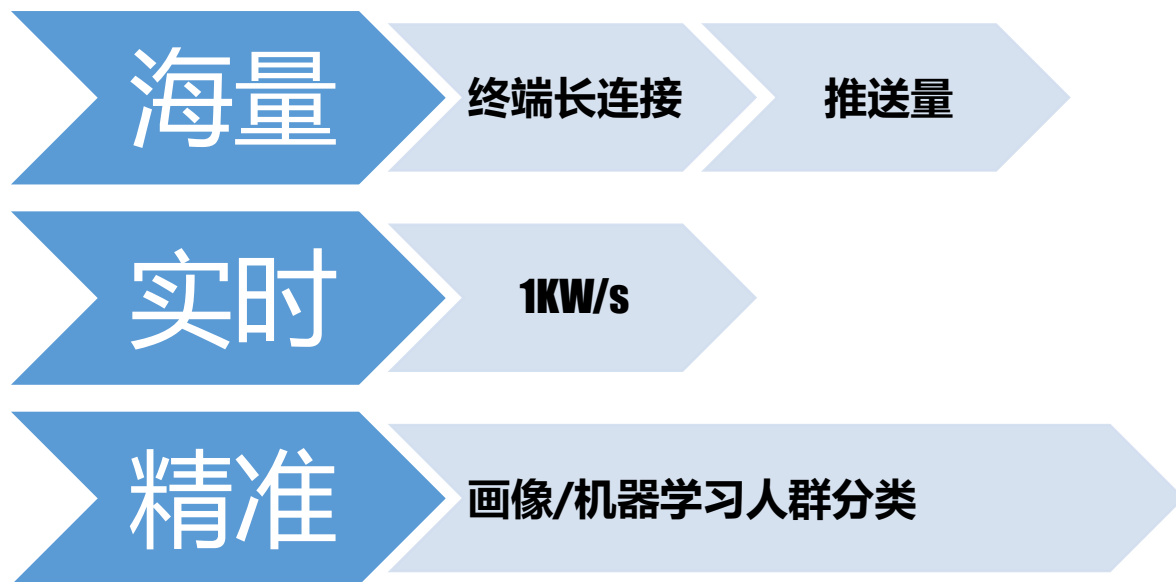
03

案例

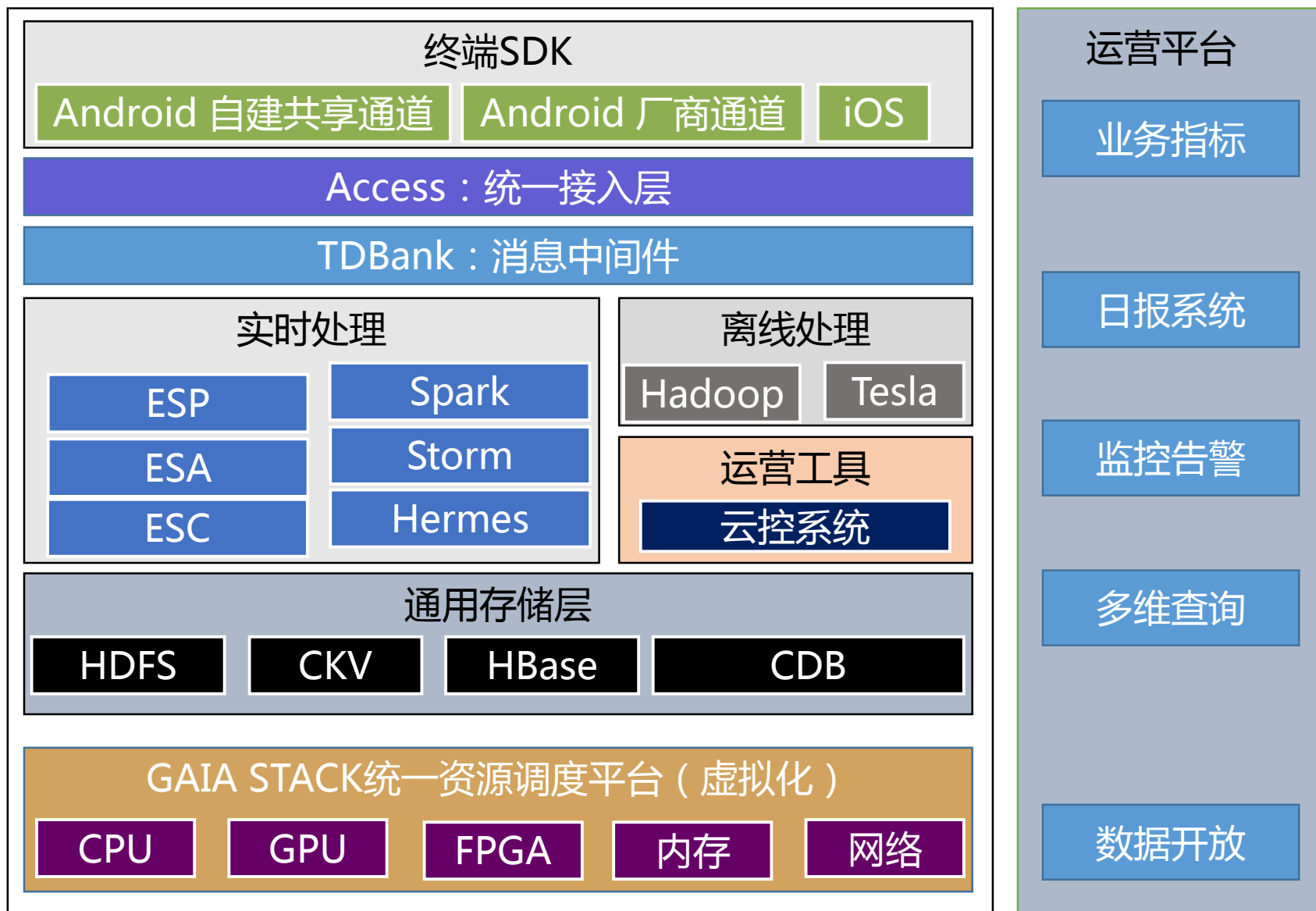
04

运营
系统建设

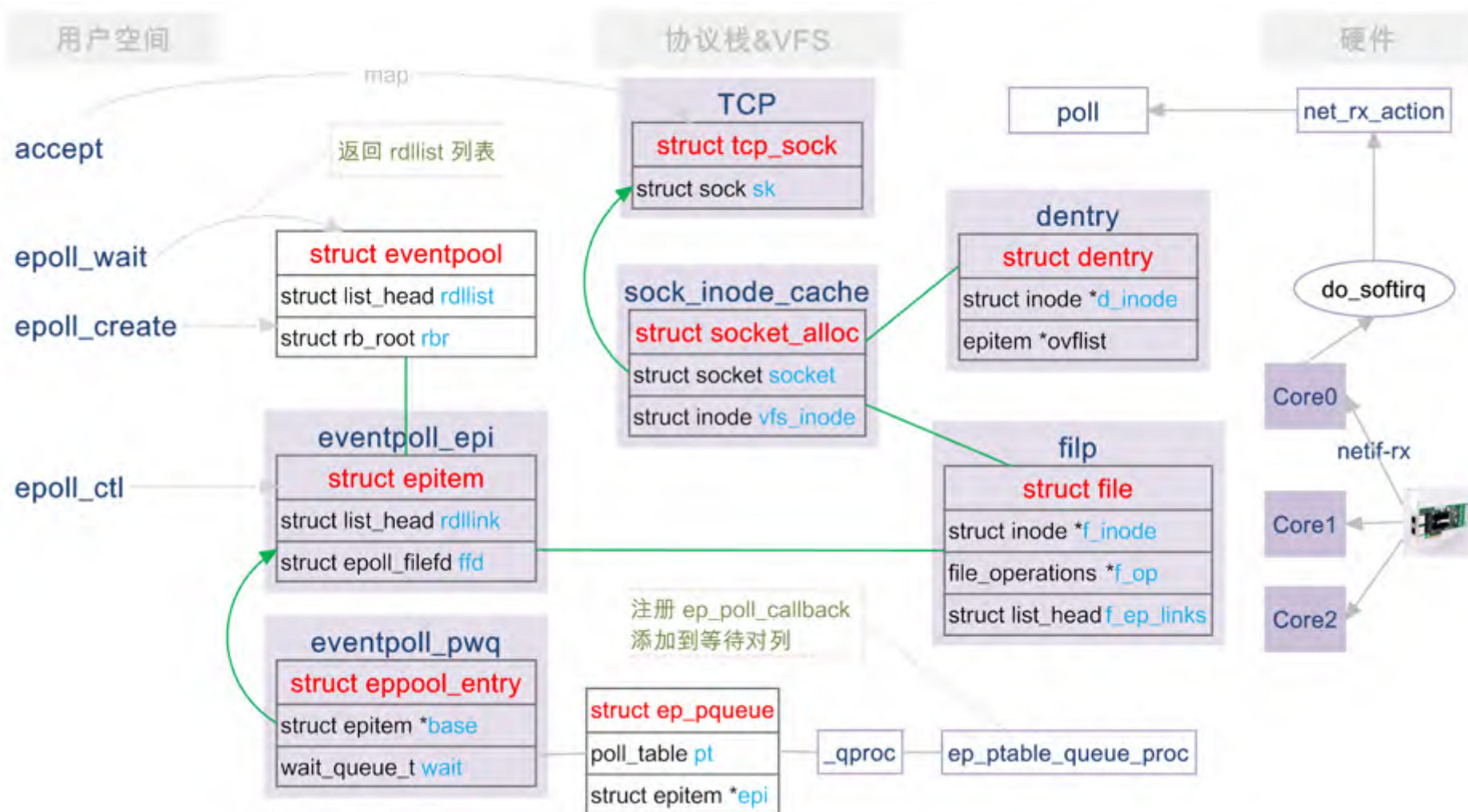
信鸽推送的挑战



信鸽推送系统解决方案



单机性能优化--关键环节



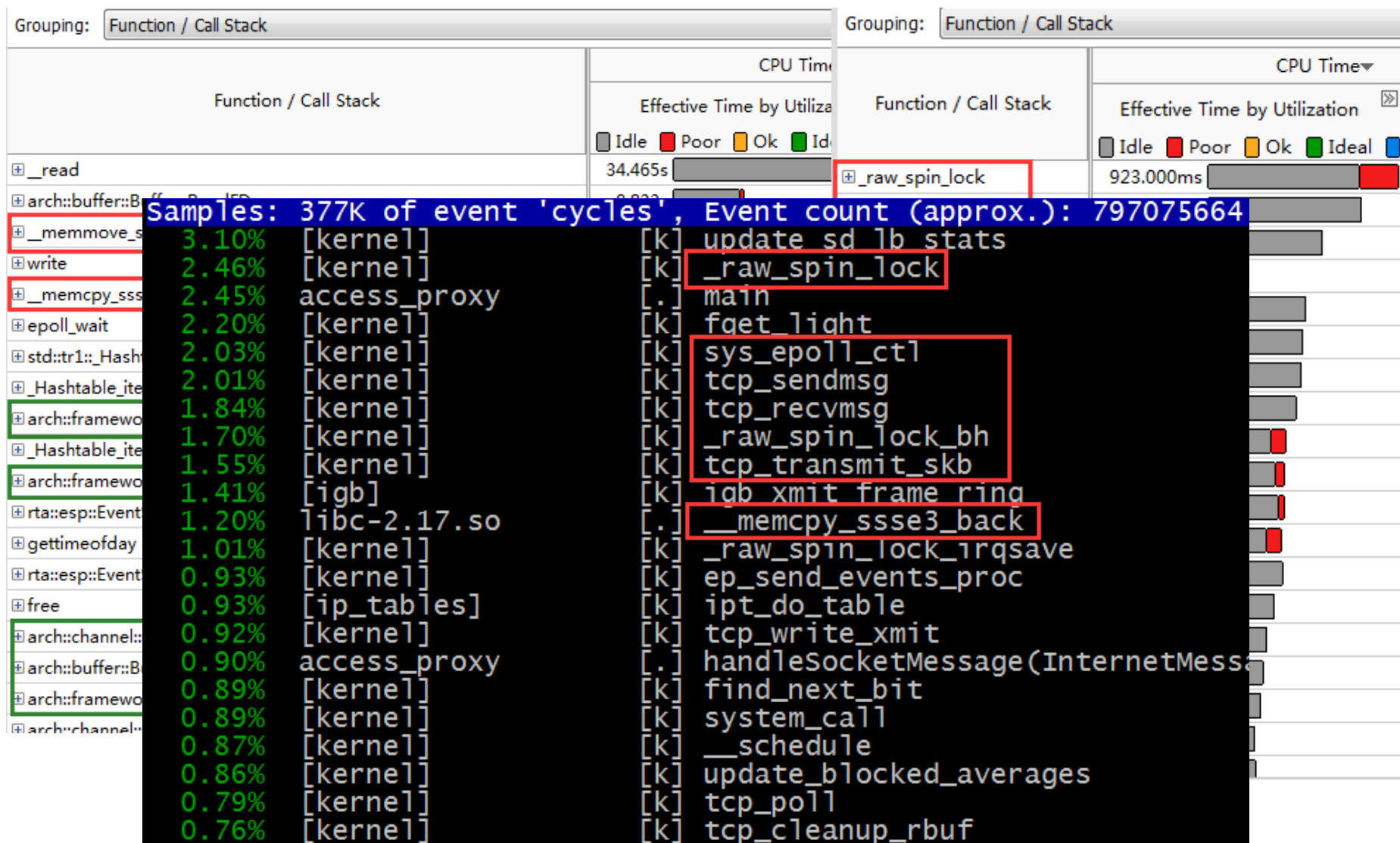
单机性能优化--操作系统--配置部分

配置类型	配置项	值
系统打开文件资源限制	/proc/sys/fs/file-max	2500000
进程打开文件句柄限制	/proc/sys/fs/nr_open	2500000
Epoll监听fd个数限制	/proc/sys/fs/epoll/max_user_watches	2500000
内核分配给TCP内存	/proc/sys/net/ipv4/tcp_mem	754752 1006336 1509504
Listen fd backlog	/proc/sys/net/core/somaxconn	1024
fast recycling of TWsocketsv	/proc/sys/net/ipv4/tcp_tw_recycle	1
Reuse TW socket	/proc/sys/net/ipv4/tcp_tw_reuse	1
拥塞控制算法	/proc/sys/net/ipv4/tcp_westwood	0
Eth scatter-gather	Eth sg	Off
Eth generic-segmentation-offload	Eth gso	Off

单机性能优化--性能评测

Grouping: Function / Call Stack		Grouping: Function / Call Stack	
Function / Call Stack	CPU Time	Function / Call Stack	CPU Time
	Effective Time by Utilization Idle Poor Ok Id		Effective Time by Utilization Idle Poor Ok Ideal
+ _read	34.465s	+ _raw_spin_lock	923.000ms
+ arch::buffer::Buffer::ReadFD	9.833s	+ update_sd_lb_stats	744.000ms
+ _memmove_sse3_back	9.535s	+ main	557.000ms
+ write	5.702s	+ recv	0ms
+ _memcpy_sse3_back	5.692s	+ Sys_epoll_ctl	482.000ms
+ epoll_wait	4.313s	+ _acct_update_integrals	467.000ms
+ std::tr1::Hashtable<arch::channel::ChannelHandler*, arch::channel::ChannelHandler*	4.056s	+ tcp_recvmmsg	461.000ms
+ _Hashtable_iterator	3.543s	+ tcp_sendmsg	431.000ms
+ arch::framework::ServiceProcessFactory::GetServiceProcessByIndex	2.868s	+ tcp_transmit_skb	384.000ms
+ _Hashtable_iterator	2.828s	+ _raw_spin_lock_irqsave	382.000ms
+ arch::framework::SocketMessageEvent::OnEncode	2.260s	+ _raw_spin_lock_bh	378.000ms
+ rta::esp::EventStreamingProcessorBaseNetworkMessageHandler::Me	2.239s	+ igb_xmit_frame_ring	367.000ms
+ gettimeofday	2.131s	+ fget_light	366.000ms
+ rta::esp::EventStreamingProcessorServiceProcess::GetAssignedProce	2.128s	+ user_exit	325.000ms
+ free	1.820s	+ local_clock	290.000ms
+ arch::channel::Channel::WriteNow	1.736s	+ epoll_wait	271.000ms
+ arch::buffer::BufferHelper::WriteFixUInt32	1.692s	+ native_sched_clock	266.000ms
+ arch::framework::IPCEvent::Encode	1.644s	+ jiffies_to_timeval	236.000ms
+ arch::channel::coder::IntegerHeaderFrameDecoder::Decode	1.550s	+ tracesvs	231.000ms

单机性能优化--性能评测



单机性能优化--硬件性能挖掘

```
int check_support_sse4_2() {
    int res=0;
    __asm__ __volatile__(
        "movl $1,%eax\n\t"
        "cpuid\n\t"
        "test $0x0100000,%ecx\n\t"
        "jz 1f\n\t"
        "movl $1,%0\n\t"
        "1:\n\t"
        : "=m" (res)
        : "eax", "ebx", "ecx", "edx");
    return res;
}
```

SSE4.2检测

local_clock	122.000ms	308,660,000
ip_queue_xmit	121.000ms	111,320,000
tcp_transmit_skb	117.000ms	331,430,000
MD5::BinUtil::crc32	116.000ms	212,520,000
igb_xmit_frame_ring	111.000ms	427,570,000
sched_clock_cpu	106.000ms	199,870,000
tcp_recvmsg	103.000ms	88,550,000

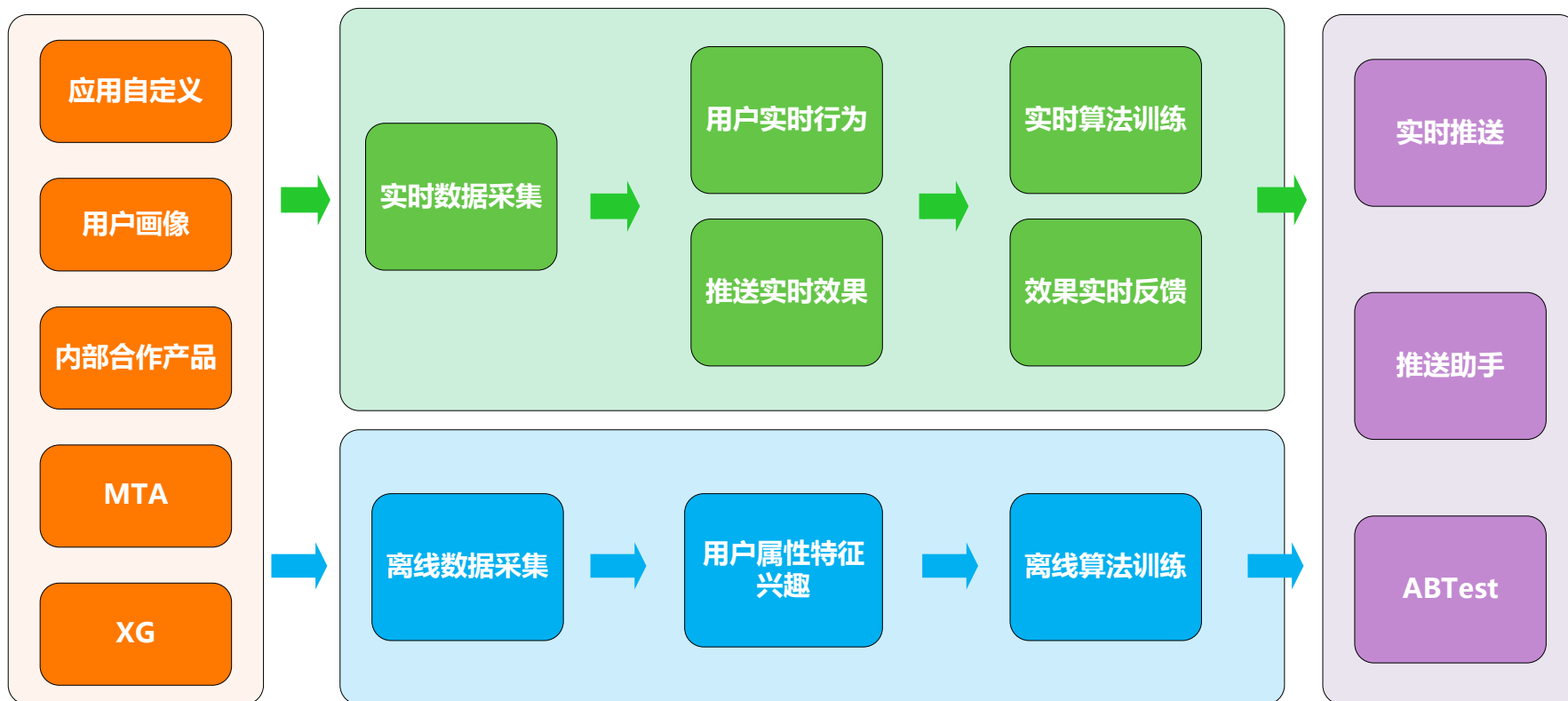
```
unsigned int _mm_crc32_u8(...)
unsigned int _mm_crc32_u16(...)
unsigned int _mm_crc32_u32(...)
unsigned __int64 _mm_crc32_u64(...)
```

SSE4.2 CRC32支持

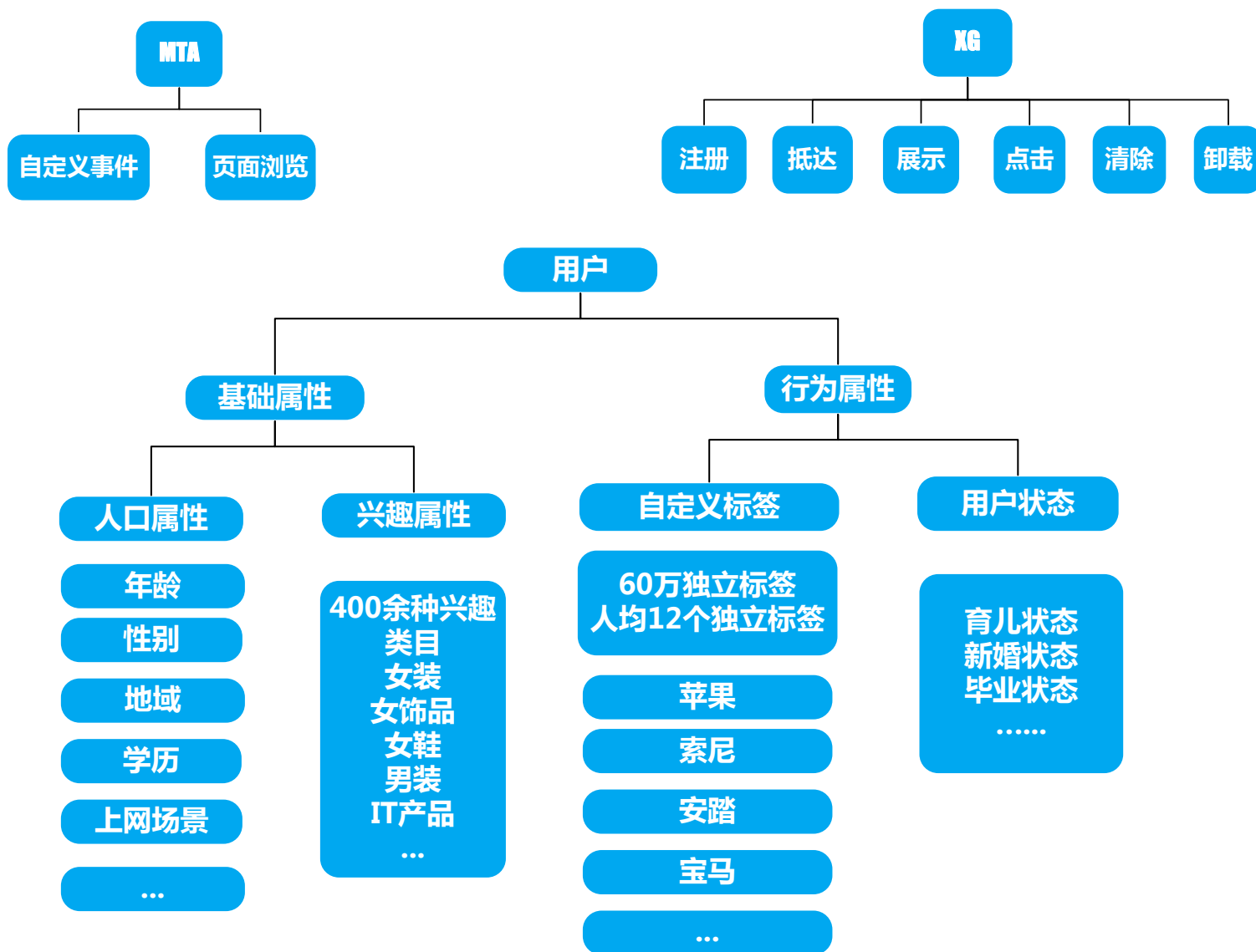
_errno_location	0ms	5,060,000
update_cfs_shares	20.000ms	15,180,000
retransmits_timed_out	20.000ms	32,890,000
MD5::Util::crc32sse	20.000ms	48,010,000
ipv4_dst_check	20.000ms	43,010,000
ip_output	20.000ms	50,600,000
inode_init_once	20.000ms	22,770,000

Intel SSE4.2 CRC32与常规CRC32性能对比
执行 35W 次计算结果 (SSE 快接近 **6** 倍)

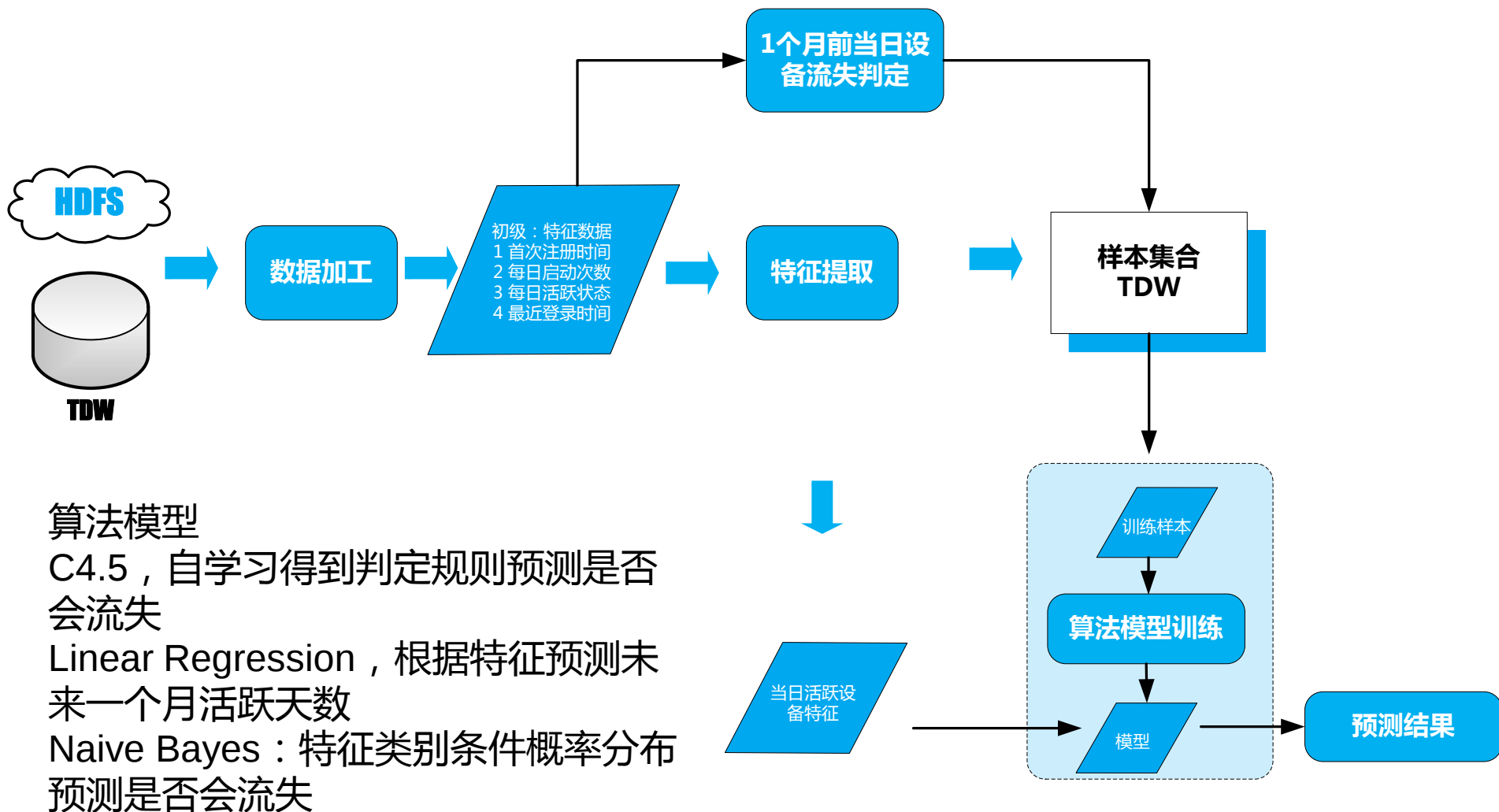
实时精准推送系统



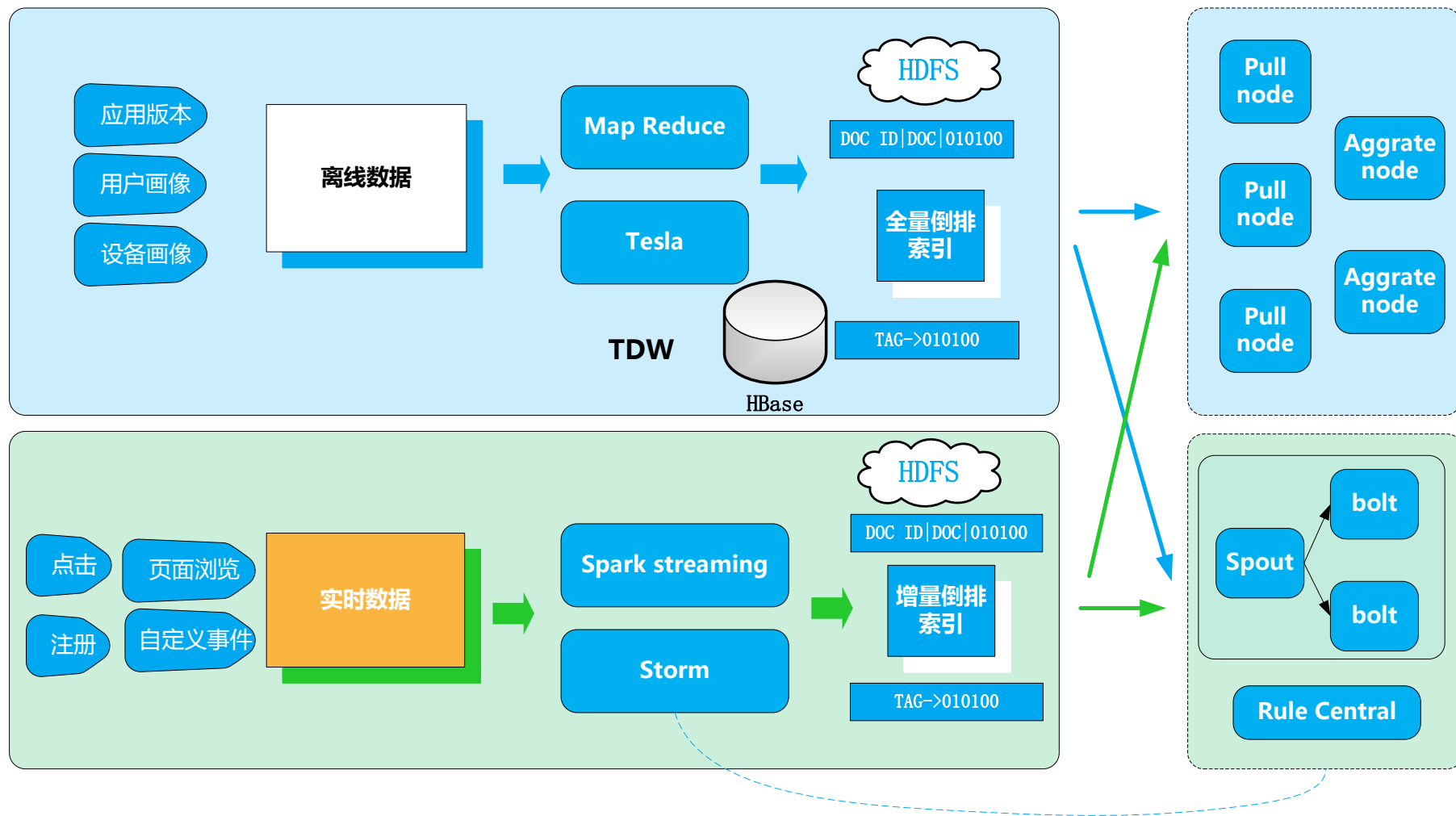
数据



人群挖掘

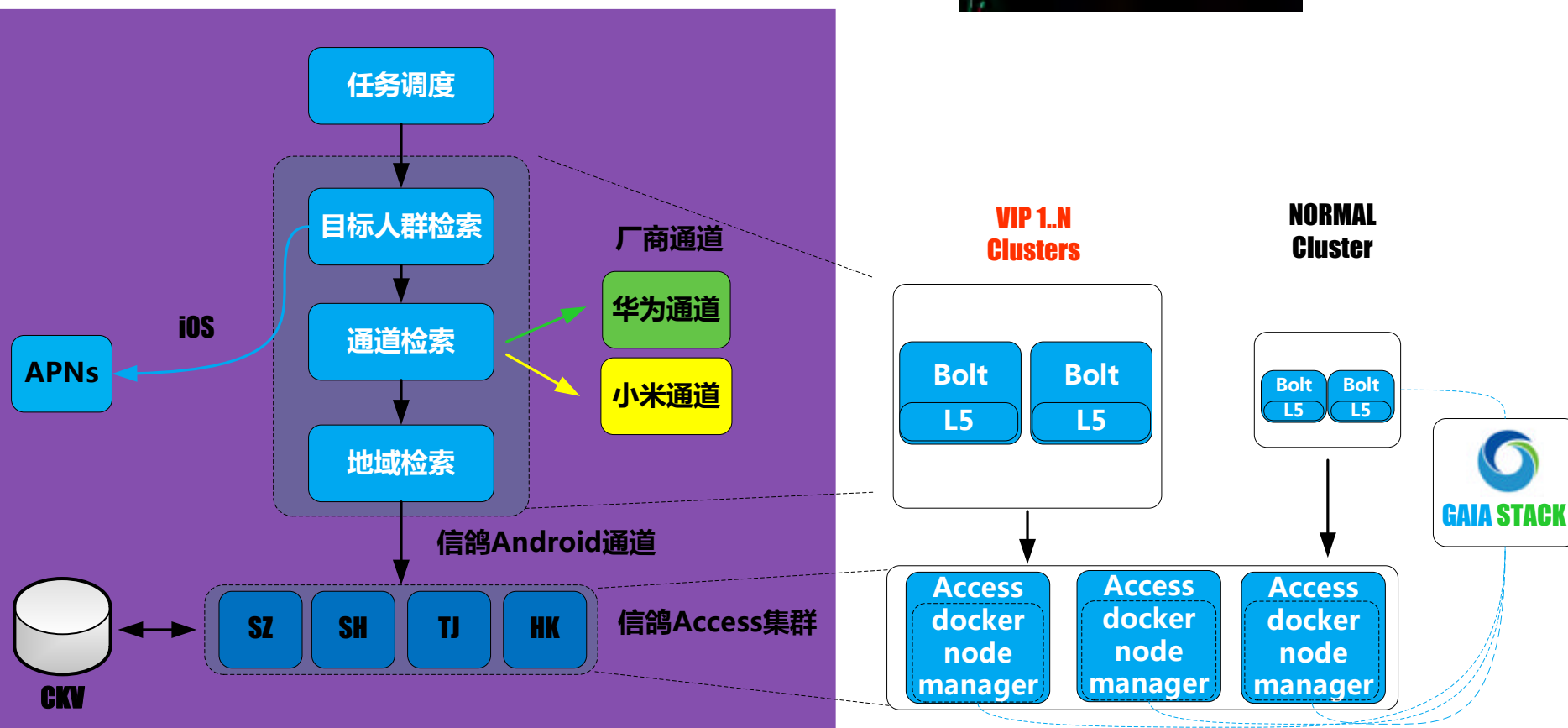


实时推送&多维分析



实时推送

```
union {  
    uint64_t    allFlow;  
    struct router {  
        uint8_t ucRegion;  
        uint8_t ucSetID;  
        uint8_t ucBoxID;  
        uint8_t ucNodeID;  
        uint32_t uiSockIdx;  
    };  
};
```



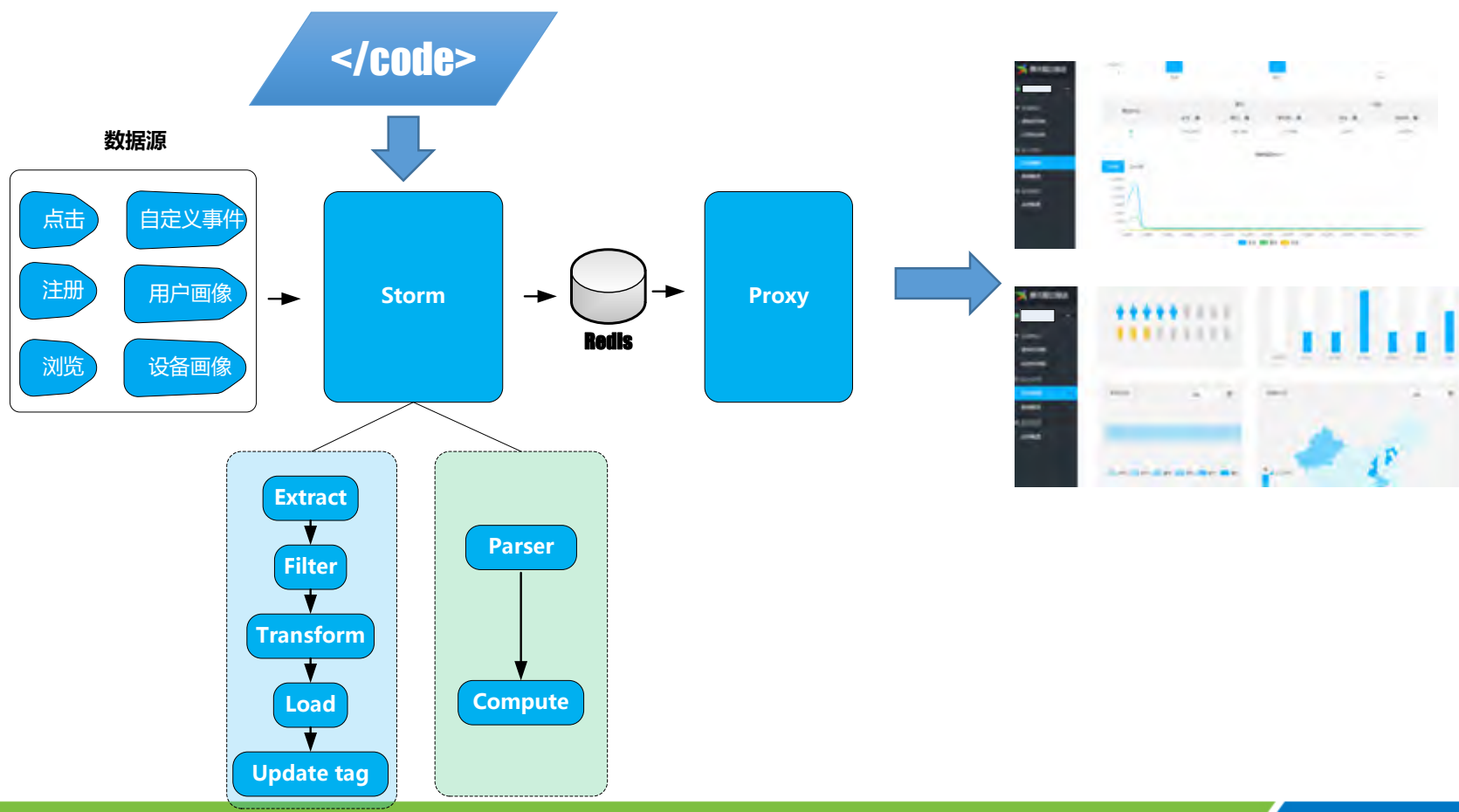
效果评估



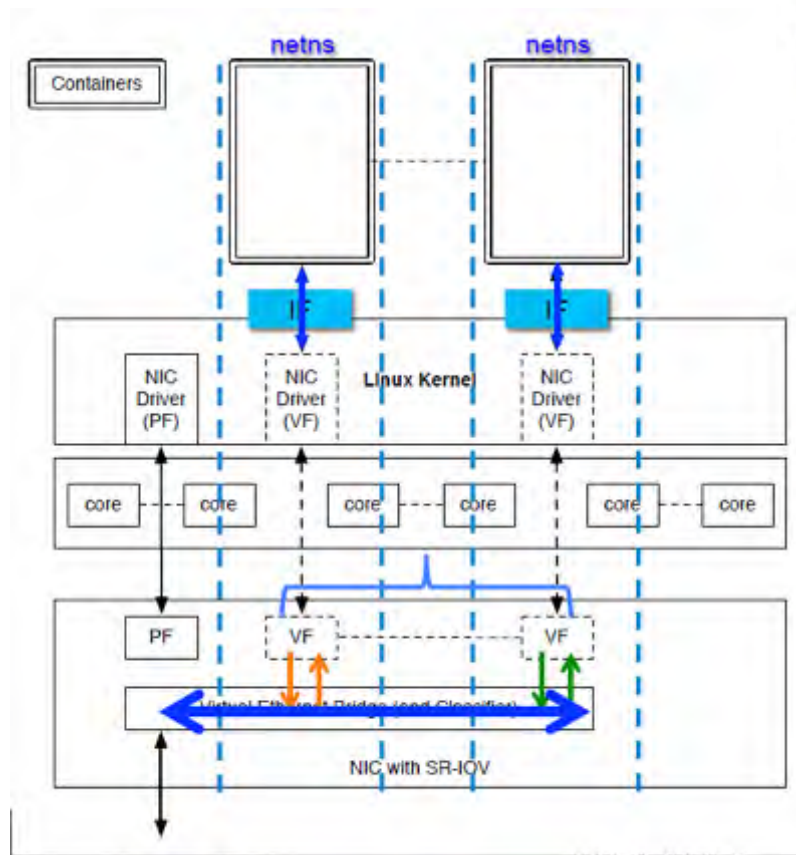
目标页面集合



自定义参数事件



软硬虚拟化



isolcpus:

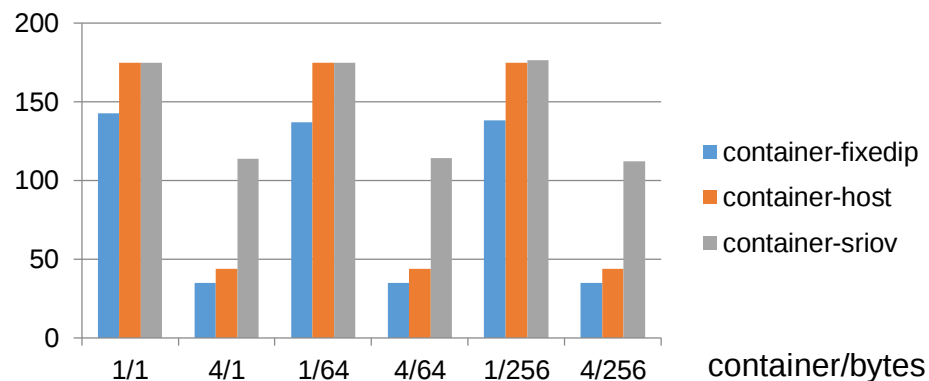
- Boot-time (Kernel boot parameter)
- ... default_hugepagesz=1G ... **isolcpus=12-15**
- Isolation from timers from other CPUs.

Cgroups/cpuset.cpus:

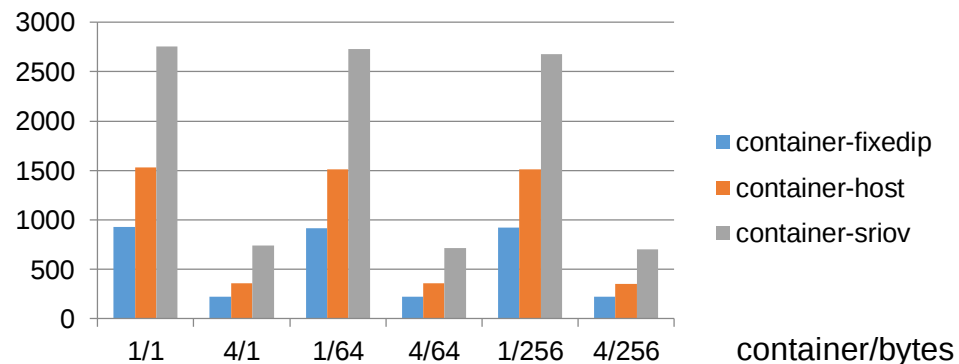
- Run-time
- **Isolate target CPUs (Next Page)**
- Run Container on those CPUs
- Same as isolcpus except the hrtimer issue

```
$ docker run -t1 --cpuset-cpus="12-15"
```

TCP_CRR(短连接)



TCP_RR(长连接)





腾讯云分析



腾讯大数据



信鸽推送

技术支持

dtsupport@tencent.com

商务咨询

data@tencent.com