Topstat

Practice on High Performance Statistics in Kernel

Zhuo Song

Oct. 2016 Alibaba

Why

- Support top-style statistics in a common way
- Provide great large numbers of ordering in a limited and controllable overhead (for instance, in a network with many connections)
- Easily support for second-level monitoring

Name

top-style statistics

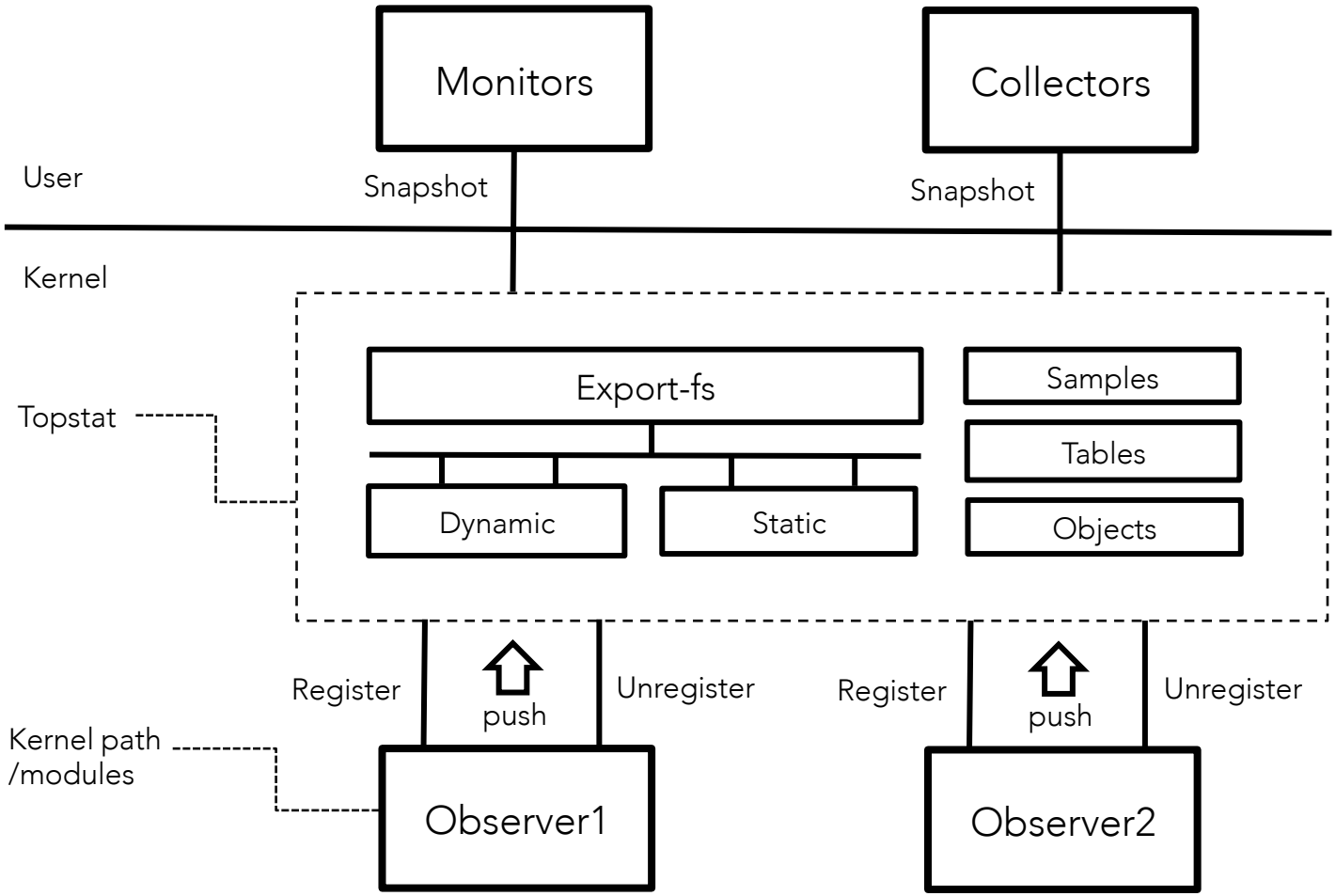
top + stat

Types

- The static, like parameters, values, or something stand-alone, e.g. packets received till now

- The dynamic, e.g. packets received per second (pps), time-related

Architecture



NOTE:

Objects' lifecycle could be maintained by observers, e.g. keep them in memory even when topstat was removed, then plug and push next time when topstat is back. The capability is sometimes useful when upgrading Is needed in production environment. See object and upgrade section for more information.

Implementation

NOTE

The examples below are only related to networking, but topstat is a common work, so it could also be used for other purposes.

Terms

- **Root**

The file system exposed to user-space for showing statistics, e.g. proc/sysfs

The root of topstat can be: “/proc/topstat/”

There is no ioctl, netlink or any other implementation of user-space interfaces in topstat, since one of the targets for topstat is file-friendly.

- **Sample**

It is a collector of tables

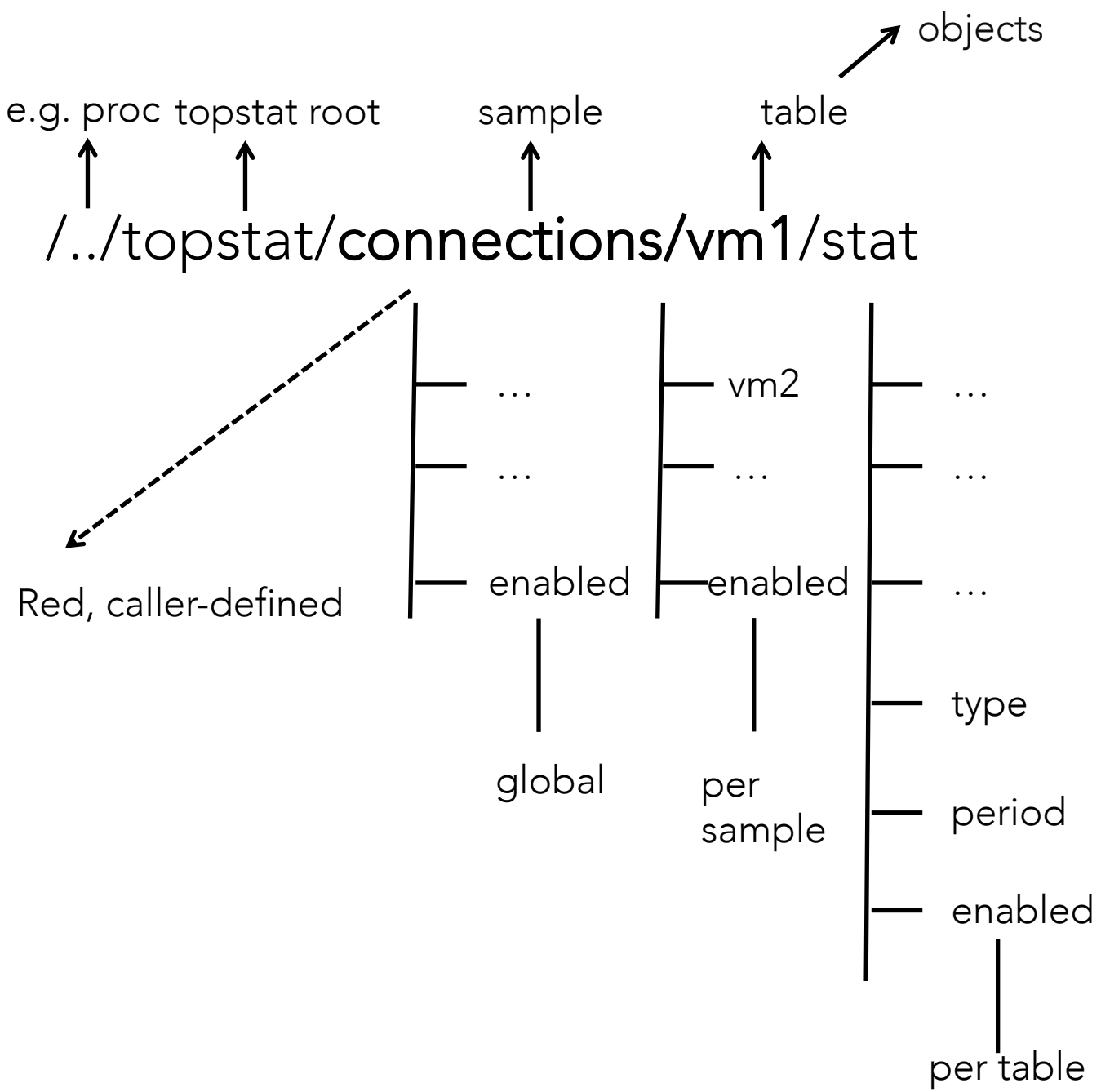
- **Table**

It collects objects to be ordered according to some norm, e.g. packets per second (pps)

- **Object**

Caller can allocate and maintain its lifecycle. And it stands for the real objects (e.g. connections) as a kind of abstraction which will be pushed into topstat for calculating.

Example


objects
e.g. proc
topstat root
sample
table
/../topstat/connections/vm1/stat
Red, caller-defined
…
…
enabled
global
vm2
…
enabled
per sample
…
…
…
type
period
enabled
per table

Root

- It contains samples (list) that the observers (e.g. some kernel path) have registered.

- The root for topstat is: “/proc/topstat/”

- There is only one exported file here, i.e. “/proc/topstat/enabled”, which can be read or written on-the-fly. The default value is 1 that indicates topstat is enabled. If it is set to 0, all statistics from topstat will be disabled, but data structures registered by the callers are not going to be released since it may be enabled in the future so that the result can be watched soon.

Sample

- A collector for the tables (parent directory from point of view of file system)

- Naming freely, in the example, it is “connections”

- Also, there is only one file exported, i.e. enabled
- If 0, all tables belonging to the sample would not be updated, and data structures will not be released since it may be enabled in the future and the result can be watched soon.
- Initially, there is no table added in.
- It can *only* be registered on *process-context* for callers.

Table

- It collects objects ordered in one norm (e.g. pps in the example)
- It can be one of two types, i.e. dynamic or static
- The table contains per-cpu part in its data structure, so in theory, it takes different results for each cpu, and there is an summary for merging all the results on those cpus (see: snapshot section)
- Users of topstat need to register description when creating object (e.g. IP, port), which should be called when snapshot on the table
- All entries in one table are ordered and saved in RB tree (per-cpu), and there is also a time tree to invalidate old entries when pushing and snapshot for dynamic tables (see: algorithm and snapshot section)
- There are bitmaps for the entries so as to reduce overheads just from (de)allocation
- There is limit number (e.g. 128) for the entries (i.e. the size of RB tree), and could be changed only at compilation time, currently. It says, top N (128, maybe)
- It can *only* be added or deleted on process context.
- See the exported files in a table:

Entries	Path (example)	Attribute	Range	Default
period	/proc/topstat/ connections/vm1/ period	R/W	In ms (1~3600000)	1000
type	/proc/topstat/ connections/vm1/type	RO	1 – dynamic, 2 – static	1
delay	/proc/topstat/ connections/vm1/delay	R/W	In ms (0~3600000)	0
enabled	/proc/topstat/ connections/vm1/ enabled	R/W	0 – disabled, 1 - enabled	1
stat	/proc/topstat/ connections/vm1/stat	RO	N/A	N/A
dump	/proc/topstat/ connections/vm1/dump	RO	N/A	N/A
sync	/proc/topstat/ connections/vm1/sync	R/W	0 – no sync 1 – sync	1

Name

- Name, is specified when creating table, in the example, it is “vm1” which is unique id for the table in one sample (table in different sample is allowed to have same name). It is a directory name

Type

- Type, is specified when creating table, in the example, its value is “1” which means the table is dynamic. It says, the table will be used to time related statistics, like pps in the example

- It can not be changed after the table is created. It is read-only.

Period

- Period, can be changed both by topstat api and exported fs after the table is created. If not, the default value is 1000 ms, i.e. 1s. It says, the statistics rate is based on second, i.e. xxx per second

NOTE:

Actually, all time conditionals are based on jiffies, so the minimal period is 1 ms (usually, architecture-dependent)

Delay

- Delay, can be changed both by topstat api and exported fs after the table is created. If not, the default value is 0 ms, i.e. no delay for updating table. It is necessary when pushing is too fast that updating table is so frequent that it downgrades the performance in some level. If set, updating (ordering) will never happen (but still be summed and returned directly from pushing routine) until the delayed time arrives.

NOTE:

Actually, all time conditionals are based on jiffies, so the minimal delay is 1 ms (usually, architecture-dependent)

Stat

- Stat(Read-only), can be read to show statistics of a table which is so-called snapshot.
- It will show ordered objects with values and caller-defined descriptions
- It will do snapshot on each cpu and finally make an average summary for all-cpu, see snapshot later.
- Firstly, see an example:

```
Topstat table snapshot:

name:                           1
type:                           1
delay:                          0
period:                         1000
enabled:                        1
owner:                          obs
owner enabled:                  1
topstat enabled:                1

cpu 0:

obj                 value
1                   1932300
10                  900
9                   900
8                   900
7                   900
6                   900
5                   900
4                   900
3                   900
2                   900

cpu 3:

obj                 value
1                   2101500

all:

obj                 value
1                   2101500
10                  900
9                   900
8                   900
7                   900
6                   900
5                   900
4                   900
3                   900
2                   900
```

Dump

- Dump (read-only), mostly used to debug and check table’s content. It is not synchronized as snapshot, since there may be no need to get a real-time result for only debugging purposes compared with snapshot by kthreads

- See a simple screen shot:

```
Topstat table dump:

name:                       1
type:                       1
delay:                      0
period:                     1000
delay jiffies:              0
period jiffies:             1000
slide jiffies:              100
enabled:                    1
owner:                      obs
owner enabled:              1
topstat enabled:            1
topstat obj tracker:        2561
snapshot show cpu old:      -2
snapshot show cpu:          -1
snapshot unused bitmap:     1024

ffffffffffffffff ffffffffffffffff ffffffffffffffff ffffffffffffffff
ffffffffffffffff ffffffffffffffff ffffffffffffffff ffffffffffffffff
ffffffffffffffff ffffffffffffffff ffffffffffffffff ffffffffffffffff
ffffffffffffffff ffffffffffffffff ffffffffffffffff ffffffffffffffff
0

cpu 0:

bitmap: 128
fffffffffffffc00 ffffffffffffffff 0

index             value             time              count             object
0                 1990200           4295733740        607711            ffff880061992c00
1                 1000              4295733741        4491              ffff880061991000
2                 1000              4295733741        4491              ffff880061991400
3                 1000              4295733741        4491              ffff880061991800
4                 1000              4295733741        4491              ffff880061991c00
5                 1000              4295733741        4491              ffff880061990000
6                 1000              4295733741        4491              ffff880061990400
7                 1000              4295733741        4491              ffff880061990800
8                 1000              4295733741        4491              ffff880061990c00
9                 1000              4295733741        4491              ffff88006198f000
10                0                 0                 0                 (null)
11                0                 0                 0                 (null)
12                0                 0                 0                 (null)
13                0                 0                 0                 (null)
14                0                 0                 0                 (null)
15                0                 0                 0                 (null)
16                0                 0                 0                 (null)
17                0                 0                 0                 (null)
18                0                 0                 0                 (null)
19                0                 0                 0                 (null)
20                0                 0                 0                 (null)
21                0                 0                 0                 (null)
22                0                 0                 0                 (null)
```

Sync

- Sync (R/W), to enable kthread for snapshot on each cpu. Actually kthread will scan the the table (tree), do aging and then record result for statistics, a little bit slower when enabled (put into work queue) but more clearer and accurate

- It could be disabled for stress from some monitoring policies when necessary. When disabled, the snapshot will not touch tree, instead it will only get the result from array

Enabled

- Like topstat root or sample, but the difference is scope. It only controls the statistics for the table.

Object

- Abstraction for each statistic object

- Each time when caller pushes data to topstat table, there will be fields swap* to record last value, timestamp, and so on in the current object, which will be used for next calculation(summed) and make decision for table updating. On the other hand, the table entry stores the caller-defined object (see: algorithm)

- Actually, the lifecycle of objects could be different with topstat's (see upgrade)

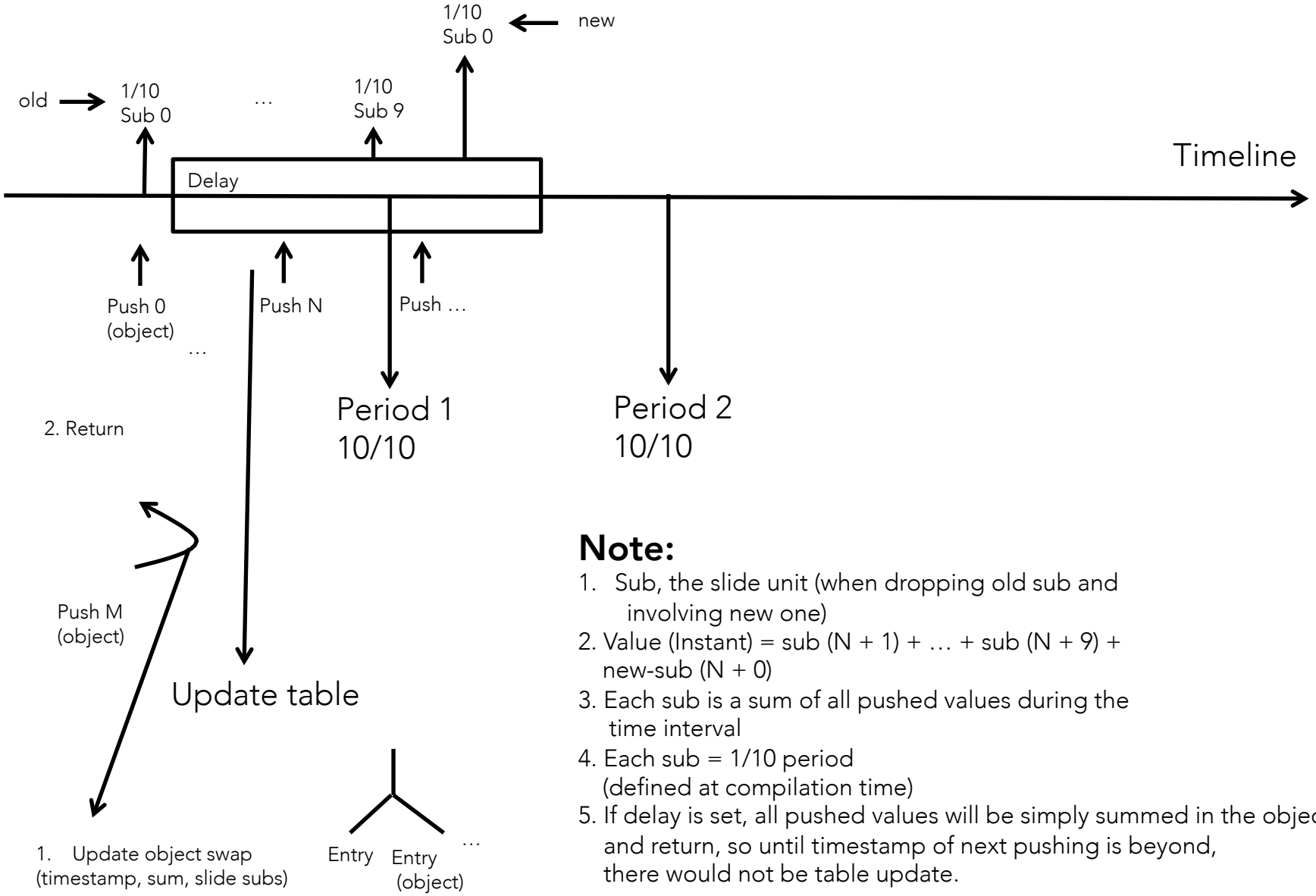
Algorithm

Dynamic

- Dynamic algorithm is to calculate final values in dynamic table.

- Since it is timer-free in topstat, it fits well into large numbers of objects ordering. It is designed to balance between accuracy and performance.

- For each pushing (dynamic table), it will do next:



Note:

1. Sub, the slide unit (when dropping old sub and involving new one)
2. Value (Instant) = sub (N + 1) + … + sub (N + 9) + new-sub (N + 0)
3. Each sub is a sum of all pushed values during the time interval
4. Each sub = 1/10 period (defined at compilation time)
5. If delay is set, all pushed values will be simply summed in the object and return, so until timestamp of next pushing is beyond, there would not be table update.

Static

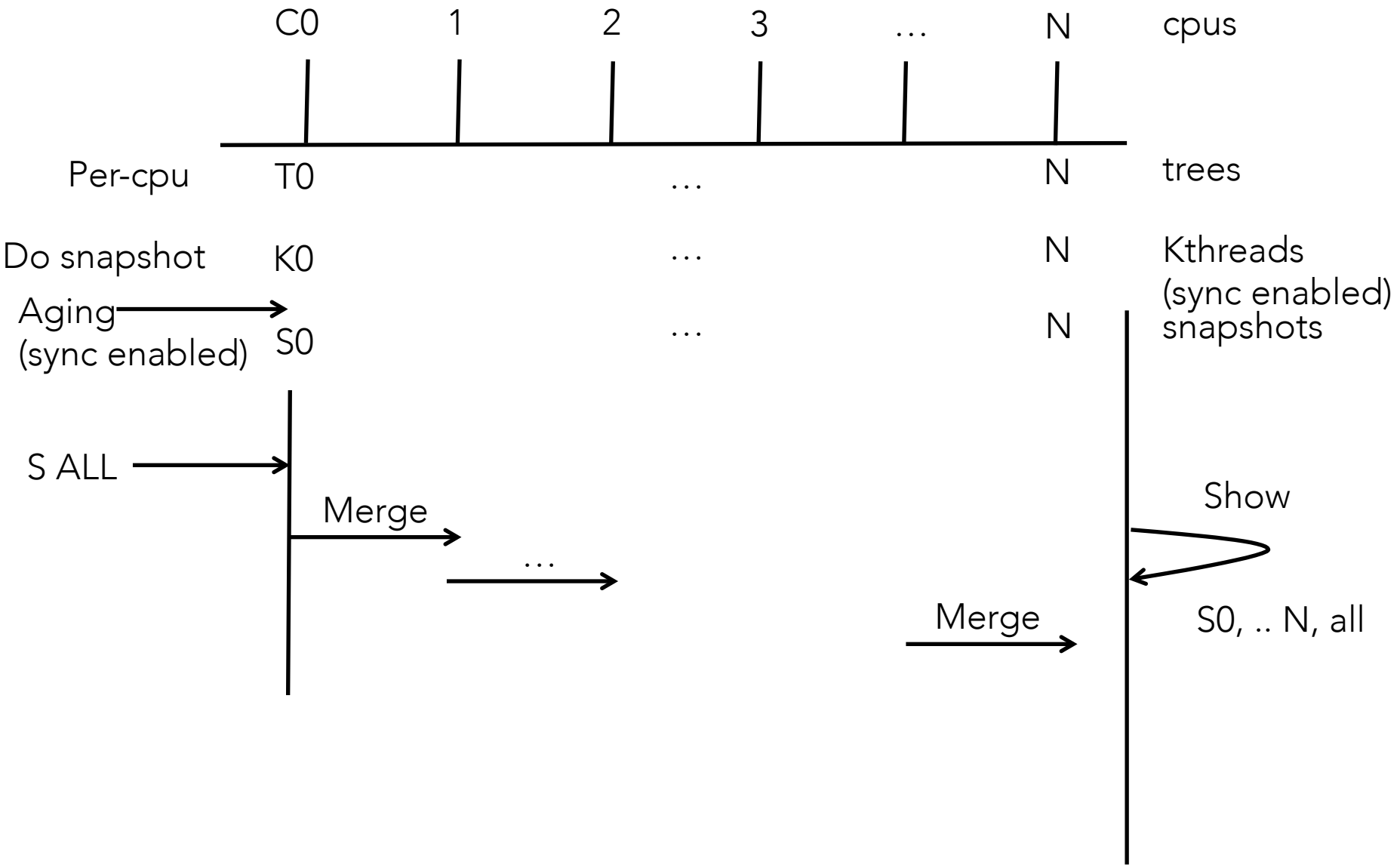
- Static algorithm is to calculate final values in static table.

- It is simple enough to calculate sum and decides when to update table (after delay).

Update

- Aging, to invalidate old entries (objects) in the table according to period and time tree
- If the object has been released by caller (live = 0), and exist in the table, remove the object (i.e. entry) from the per-cpu table
- If the reference counter of the object comes to 0 finally, free it
- Look for the old entries (expired, not being updated in a period), and kick them out of the table and set the ready bitmap to 1 to indicate the slot (entry) can be used directly by other objects
- Get the free slot (entry), and update the entry with newly pushed object if possible
- If no free entries, walk through the value tree with the newly pushed object and update the entry, then insert it at the proper node if the value is bigger than the existing smallest one

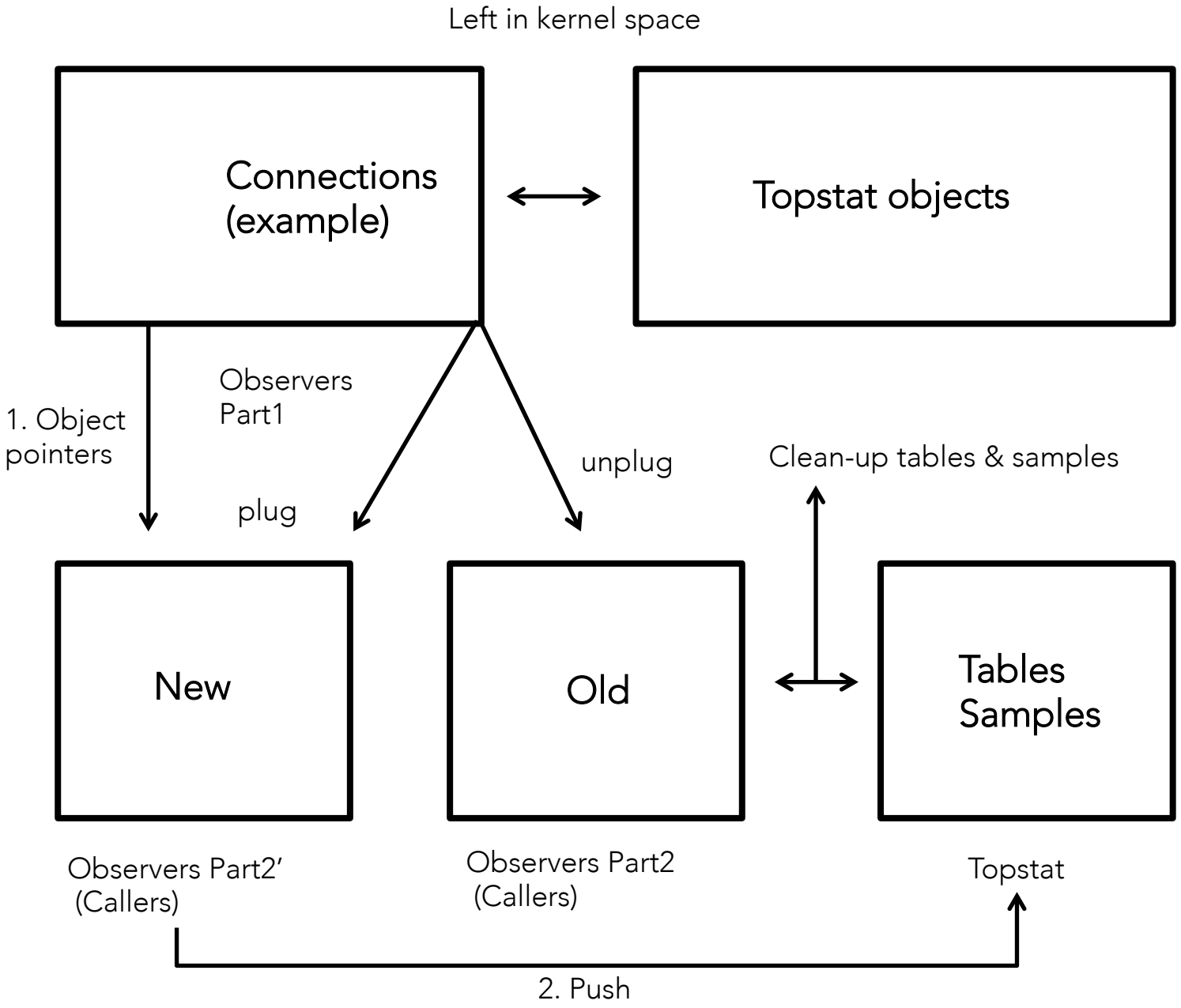
Snapshot


C0
1
2
3
…
N
cpus
Per-cpu
T0
…
N
trees
Do snapshot
K0
…
N
Kthreads
(sync enabled)
Aging
(sync enabled)
S0
…
N
snapshots
S ALL
Merge
…
Show
Merge
S0, .. N, all

Upgrade

- Only observers update is covered here. The objects could be alive in kernel memory space until someone releases them explicitly even if topstat or observer has been removed. Consider the observer keeps the object pointers in another module (e.g. connection tables) , and the observer has been removed. Next time when the observer comes back (like, insmod), it could use the object pointers in connection tables directly and push the data when necessary.

- But samples and tables must be cleaned up before removing observer and topstat.


Left in kernel space
Connections (example)
Topstat objects
1. Object pointers
Observers Part1
plug
unplug
Clean-up tables & samples
New
Old
Tables Samples
Observers Part2' (Callers)
Observers Part2 (Callers)
Topstat
2. Push

Performance

- The test is based on tcp v4
- It will make top-128 statistics for connections according to pps (packets per second)
- Inbound & outbound
- “/proc/topstat/tcp_v4/conns_pps/stat”

Spec

- Environment (redhat el5 x86_64)
- Kernel base (taobao-kernel-2.6.32.358.1193 x86_64)
- netperf-2.4.5-1
- aliperf-0.3.9-9 (from original perf)
- top-3.2.7
- Server ip (10.65.254.50)
- Client ip (10.65.254.51)

- Server model (Huawei, Tecal RH2288H V2-12L）
- CPU (Intel(R) Xeon(R) CPU E5-2630 0 @ 2.30GHz, 24 cores)
- Memory (126 GB)
- NIC (Intel Corporation I350 Gigabit)

Changes

- Build topstat into the kernel and initialize it at early stage
- Register tcp_v4_topstat_sample and tcp_v4_topstat_table
- Add topstat_object to sock and change sk_alloc (active), sk_clone (passive), sk_free, so that the lifecycle will follow sock
- Change tcp_sendmsg and tcp_rcv_established to push when sending and receiving packets
- Only calculate for packets at established stage

Case

Test connection number: 1, 100, 500, 1000, 5000
Watch cpu utilization and hotspot (perf) when topstat enabled and disabled

Server: nothing more

netserver

Example:

```
$sudo netstat -pnlt | grep netserver
tcp        0      0 0.0.0.0:12865           0.0.0.0:*               LISTEN      9921/netserver
```

Client:

netperf -H $SERVER -t TCP_RR -l $TIME -D -- -r 64,64

Example:

```
$ps -ef | grep netperf
root     123678      1 20 11:09 pts/0    00:00:12 netperf -H 10.65.251.50 -t TCP_RR -l 120 -D -- -r 64 64
```

1

```
all:

sip               sport           dip               dport           pps
10.65.251.51      57303           10.65.251.50      53266           18990
10.65.254.248     46535           10.65.251.50      22              2
```

Topstat	Enabled	Disabled
Top		
CPU Utilization	0.5%sy 0.0%si 0.2% us	0.5%sy 0.0%si 0.2% us
Perf		
topstat_push	0.42%	N/A
topstat_full_enabled	0.12%	0.06%
topstat_entry_value_insert	0.12%	N/A
topstat_aging	0.08%	N/A
topstat_entry_time_insert	0.00%	N/A

100

all:

sip	sport	dip	dport	pps
10.65.251.51	44058	10.65.251.50	39403	5002
10.65.251.51	52064	10.65.251.50	34532	4995
10.65.251.51	34861	10.65.251.50	36425	4984
10.65.251.51	56407	10.65.251.50	55119	4978
10.65.251.51	50756	10.65.251.50	59083	4930
10.65.251.51	34861	10.65.251.50	36425	4924
10.65.251.51	59724	10.65.251.50	58812	4924
10.65.251.51	34861	10.65.251.50	36425	4916
10.65.251.51	50756	10.65.251.50	59083	4841
10.65.251.51	52064	10.65.251.50	34532	4805
10.65.251.51	56407	10.65.251.50	55119	4796
10.65.251.51	35827	10.65.251.50	53439	4792
10.65.251.51	44058	10.65.251.50	39403	4789
10.65.251.51	34861	10.65.251.50	36425	4773
10.65.251.51	35827	10.65.251.50	53439	4760
10.65.251.51	52331	10.65.251.50	54197	4729
10.65.251.51	50921	10.65.251.50	39843	4711
10.65.251.51	50756	10.65.251.50	59083	4709
10.65.251.51	43325	10.65.251.50	47458	4704
10.65.251.51	59724	10.65.251.50	58812	4687
10.65.251.51	50382	10.65.251.50	36520	4676
10.65.251.51	47299	10.65.251.50	47681	4651
10.65.251.51	59724	10.65.251.50	58812	4648
10.65.251.51	47299	10.65.251.50	47681	4640
10.65.251.51	44058	10.65.251.50	39403	4626
10.65.251.51	50756	10.65.251.50	59083	4614
10.65.251.51	60235	10.65.251.50	35055	4605
10.65.251.51	52331	10.65.251.50	54197	4596
10.65.251.51	50382	10.65.251.50	36520	4595
10.65.251.51	35827	10.65.251.50	53439	4591
10.65.251.51	48438	10.65.251.50	35444	4588
10.65.251.51	60851	10.65.251.50	44973	4582
10.65.251.51	50678	10.65.251.50	53976	4580
10.65.251.51	47299	10.65.251.50	47681	4575
10.65.251.51	51648	10.65.251.50	45282	4570
10.65.251.51	50961	10.65.251.50	60179	4563
10.65.251.51	39108	10.65.251.50	42457	4561
10.65.251.51	60735	10.65.251.50	35055	4547

Topstat	Enabled	Disabled
Top		
CPU Utilization	21.5%sy 4.8%si 1.0% us	19.2%sy 4.5%si 1.0% us
Perf		
topstat_push	1.01%	N/A
topstat_full_enabled	0.13%	0.14%
topstat_entry_value_insert	0.41%	N/A
topstat_aging	0.12%	N/A
topstat_entry_time_insert	0.12%	N/A

```
   4.55%     netserver  [kernel.kallsyms]              [k] _spin_lock

   1.97%     netserver  [nf_conntrack]                 [k] tcp_packet

   1.80%     netserver  [kernel.kallsyms]              [k] tcp_ack

   1.57%     netserver  [kernel.kallsyms]              [k] tcp_sendmsg

   1.51%     netserver  [kernel.kallsyms]              [k] dev_queue_xmit

   1.50%     netserver  [igb]                          [k] igb_poll

   1.34%     netserver  [igb]                          [k] igb_xmit_frame_ring

   1.25%     netserver  [kernel.kallsyms]              [k] dev_hard_start_xmit

   1.15%     netserver  libc-2.5.so                    [.] __libc_recv

   1.14%     netserver  [kernel.kallsyms]              [k] schedule

   1.01%     netserver  [kernel.kallsyms]              [k] topstat_push
```

500

all:

sip	sport	dip	dport	pps
10.65.251.51	33066	10.65.251.50	51885	1039
10.65.251.51	53423	10.65.251.50	55064	1038
10.65.251.51	46257	10.65.251.50	56899	1031
10.65.251.51	43798	10.65.251.50	56044	1030
10.65.251.51	37608	10.65.251.50	32924	1028
10.65.251.51	48587	10.65.251.50	32914	1027
10.65.251.51	53508	10.65.251.50	52650	1027
10.65.251.51	53423	10.65.251.50	55064	1025
10.65.251.51	33252	10.65.251.50	59055	1024
10.65.251.51	41131	10.65.251.50	51103	1024
10.65.251.51	43798	10.65.251.50	56044	1024
10.65.251.51	36979	10.65.251.50	58659	1024
10.65.251.51	54192	10.65.251.50	49200	1023
10.65.251.51	59698	10.65.251.50	49287	1021
10.65.251.51	36979	10.65.251.50	58659	1021
10.65.251.51	48314	10.65.251.50	32799	1020
10.65.251.51	50269	10.65.251.50	52985	1020
10.65.251.51	56323	10.65.251.50	44074	1020
10.65.251.51	33066	10.65.251.50	51885	1020
10.65.251.51	37608	10.65.251.50	32924	1019
10.65.251.51	43798	10.65.251.50	56044	1019
10.65.251.51	46048	10.65.251.50	43512	1019
10.65.251.51	41700	10.65.251.50	41028	1018
10.65.251.51	56009	10.65.251.50	44065	1018
10.65.251.51	49728	10.65.251.50	40035	1018
10.65.251.51	56920	10.65.251.50	54453	1018
10.65.251.51	49221	10.65.251.50	33205	1018
10.65.251.51	46257	10.65.251.50	56899	1017
10.65.251.51	48314	10.65.251.50	32799	1017
10.65.251.51	41700	10.65.251.50	41028	1016
10.65.251.51	36979	10.65.251.50	58659	1015
10.65.251.51	56009	10.65.251.50	44065	1015
10.65.251.51	34792	10.65.251.50	34785	1015
10.65.251.51	49728	10.65.251.50	40035	1015
10.65.251.51	60090	10.65.251.50	50320	1014
10.65.251.51	49221	10.65.251.50	33205	1014
10.65.251.51	56009	10.65.251.50	44065	1014
10.65.251.51	36979	10.65.251.50	58659	1013
10.65.251.51	56920	10.65.251.50	54453	1013
10.65.251.51	34483	10.65.251.50	57338	1012
10.65.251.51	38457	10.65.251.50	46667	1012
10.65.251.51	48893	10.65.251.50	32942	1012
10.65.251.51	50269	10.65.251.50	52985	1012

Topstat	Enabled	Disabled
Top		
CPU Utilization	26.8%sy 9.0%si 1.2% us	23.5%sy 7.2%si 1.2% us
Perf		
topstat_push	3.60%	N/A
topstat_full_enabled	0.11%	0.14%
topstat_entry_value_insert	0.16%	N/A
topstat_aging	0.10%	N/A
topstat_entry_time_insert	0.12%	N/A

```
    6.42%     netserver  [kernel.kallsyms]     [k] _spin_lock

    3.60%     netserver  [kernel.kallsyms]     [k] topstat_push

    2.24%     netserver  [kernel.kallsyms]     [k] find_next_zero_bit

    1.68%     netserver  [kernel.kallsyms]     [k] tcp_ack

    1.65%     netserver  [nf_conntrack]        [k] tcp_packet

    1.26%     netserver  [kernel.kallsyms]     [k] dev_queue_xmit

    1.17%     netserver  [kernel.kallsyms]     [k] tcp_sendmsg

    1.17%          init  [kernel.kallsyms]     [k] _spin_lock

    1.13%     netserver  [kernel.kallsyms]     [k] schedule

    1.10%     netserver  libc-2.5.so           [.] __libc_recv

    1.08%          init  [igb]                 [k] igb_poll

    0.99%     netserver  [igb]                 [k] igb_poll
```

1000

all:

sip	sport	dip	dport	pps
10.65.251.51	32800	10.65.251.50	36605	432
10.65.251.51	41648	10.65.251.50	41499	429
10.65.251.51	46693	10.65.251.50	48037	426
10.65.251.51	49105	10.65.251.50	49245	426
10.65.251.51	39710	10.65.251.50	45857	425
10.65.251.51	46693	10.65.251.50	48037	425
10.65.251.51	44379	10.65.251.50	54624	424
10.65.251.51	39349	10.65.251.50	42953	424
10.65.251.51	54254	10.65.251.50	36261	424
10.65.251.51	46838	10.65.251.50	40919	424
10.65.251.51	43241	10.65.251.50	46659	424
10.65.251.51	59523	10.65.251.50	52188	423
10.65.251.51	41420	10.65.251.50	36113	423
10.65.251.51	35485	10.65.251.50	60800	423
10.65.251.51	55794	10.65.251.50	41422	423
10.65.251.51	43241	10.65.251.50	46659	423
10.65.251.51	41925	10.65.251.50	56382	423
10.65.251.51	59523	10.65.251.50	52188	423
10.65.251.51	35098	10.65.251.50	54923	423
10.65.251.51	51555	10.65.251.50	41032	423
10.65.251.51	35485	10.65.251.50	60800	423
10.65.251.51	44197	10.65.251.50	47414	423
10.65.251.51	36556	10.65.251.50	38413	423
10.65.251.51	41925	10.65.251.50	56382	423
10.65.251.51	52380	10.65.251.50	45886	423
10.65.251.51	37705	10.65.251.50	46547	423
10.65.251.51	46838	10.65.251.50	40919	423
10.65.251.51	33284	10.65.251.50	41858	422
10.65.251.51	52822	10.65.251.50	36402	422
10.65.251.51	60422	10.65.251.50	57325	422
10.65.251.51	57970	10.65.251.50	49726	422
10.65.251.51	41560	10.65.251.50	37357	422
10.65.251.51	36827	10.65.251.50	35702	422
10.65.251.51	52314	10.65.251.50	43267	422
10.65.251.51	49594	10.65.251.50	50636	422
10.65.251.51	38840	10.65.251.50	35295	422
10.65.251.51	46693	10.65.251.50	48037	422
10.65.251.51	41560	10.65.251.50	37357	422
10.65.251.51	54929	10.65.251.50	55489	422
10.65.251.51	41420	10.65.251.50	36113	422
10.65.251.51	47148	10.65.251.50	35817	422

Topstat	Enabled	Disabled
Top		
CPU Utilization	26.6%sy 9.2%si 1.3% us	23.2%sy 8.0%si 1.3% us
Perf		
topstat_push	3.56%	N/A
topstat_full_enabled	0.11%	0.11%
topstat_entry_value_insert	0.11%	N/A
topstat_aging	0.09%	N/A
topstat_entry_time_insert	0.09%	N/A

```
  7.42%     netserver  [kernel.kallsyms]              [k] _spin_lock

  3.56%     netserver  [kernel.kallsyms]              [k] topstat_push

  2.29%     netserver  [kernel.kallsyms]              [k] find_next_zero_bit

  1.73%          init  [kernel.kallsyms]              [k] _spin_lock

  1.63%     netserver  [kernel.kallsyms]              [k] tcp_ack

  1.50%     netserver  [nf_conntrack]                 [k] tcp_packet

  1.30%     netserver  libc-2.5.so                    [.] __libc_recv

  1.21%     netserver  [kernel.kallsyms]              [k] dev_queue_xmit

  1.17%          init  [igb]                          [k] igb_poll

  1.11%     netserver  [kernel.kallsyms]              [k] schedule

  1.07%     netserver  [kernel.kallsyms]              [k] tcp_sendmsg
```

5000

all:

sip	sport	dip	dport	pps
10.65.251.51	50882	10.65.251.50	51741	72
10.65.251.51	59748	10.65.251.50	59568	72
10.65.251.51	60156	10.65.251.50	58596	72
10.65.251.51	36209	10.65.251.50	38573	72
10.65.251.51	51343	10.65.251.50	60473	72
10.65.251.51	57138	10.65.251.50	43919	72
10.65.251.51	49205	10.65.251.50	32792	72
10.65.251.51	52120	10.65.251.50	56069	72
10.65.251.51	54368	10.65.251.50	42477	72
10.65.251.51	35622	10.65.251.50	49420	72
10.65.251.51	49406	10.65.251.50	33671	72
10.65.251.51	38286	10.65.251.50	59618	72
10.65.251.51	38163	10.65.251.50	50306	72
10.65.251.51	60166	10.65.251.50	48523	72
10.65.251.51	37871	10.65.251.50	46896	72
10.65.251.51	34221	10.65.251.50	44008	72
10.65.251.51	38948	10.65.251.50	47377	72
10.65.251.51	47347	10.65.251.50	51071	72
10.65.251.51	38163	10.65.251.50	50306	72
10.65.251.51	49205	10.65.251.50	32792	72
10.65.251.51	33160	10.65.251.50	44241	72
10.65.251.51	34191	10.65.251.50	56242	72
10.65.251.51	52817	10.65.251.50	47195	72
10.65.251.51	51828	10.65.251.50	59311	72
10.65.251.51	58324	10.65.251.50	37547	72
10.65.251.51	60904	10.65.251.50	48532	72
10.65.251.51	37871	10.65.251.50	46896	72
10.65.251.51	35856	10.65.251.50	60511	72
10.65.251.51	35622	10.65.251.50	49420	72
10.65.251.51	60857	10.65.251.50	39197	72
10.65.251.51	54742	10.65.251.50	46535	71
10.65.251.51	46163	10.65.251.50	40825	71
10.65.251.51	54276	10.65.251.50	48223	71
10.65.251.51	44336	10.65.251.50	54048	71
10.65.251.51	42905	10.65.251.50	56502	71
10.65.251.51	40769	10.65.251.50	35305	71
10.65.251.51	48207	10.65.251.50	43118	71
10.65.251.51	53595	10.65.251.50	46496	71
10.65.251.51	57521	10.65.251.50	56848	71
10.65.251.51	43995	10.65.251.50	59067	71
10.65.251.51	53557	10.65.251.50	58963	71

Topstat	Enabled	Disabled
Top		
CPU Utilization	25.2%sy 14.2%si 3.2% us	22.6%sy 13.5%si 3.2% us
Perf		
topstat_push	3.55%	N/A
topstat_full_enabled	0.08%	0.08%
topstat_entry_value_insert	0.04%	N/A
topstat_aging	0.06%	N/A
topstat_entry_time_insert	0.02%	N/A

```
    6.69%     netserver  [kernel.kallsyms]            [k] _spin_lock

    3.55%     netserver  libc-2.5.so                  [.] __libc_recv

    3.55%     netserver  [kernel.kallsyms]            [k] topstat_push

    1.96%     netserver  [kernel.kallsyms]            [k] find_next_zero_bit

    1.91%     netserver  [kernel.kallsyms]            [k] tcp_ack

    1.89%     netserver  [kernel.kallsyms]            [k] copy_user_generic_string

    1.69%          init  [kernel.kallsyms]            [k] _spin_lock

    1.58%     netserver  netserver                    [.] recv_tcp_rr

    1.36%     netserver  [nf_conntrack]               [k] tcp_packet

    1.28%     netserver  [kernel.kallsyms]            [k] schedule

    1.25%     netserver  libc-2.5.so                  [.] __libc_send

    1.19%     netserver  [kernel.kallsyms]            [k] _spin_lock_bh
```

Delay

It will test delay effectiveness compared with 500 connections (1000 pps per connection) with delay 0

10 ms

Topstat	Enabled	Disabled
Top		
CPU Utilization	22.8%sy 7.5%si 1.2% us	22.3%sy 7.2%si 1.2% us
Perf		
topstat_push	0.81%	N/A
topstat_full_enabled	0.11%	0.12%
topstat_entry_value_insert	0.03%	N/A
topstat_aging	0.02%	N/A
topstat_entry_time_insert	0.03%	N/A

- For 500 connections with about 1000 pps per connection, CPU utilization downgrades from (26.8%sy 9.0%si) to (22.8%sy 7.5%si) when topstat enabled

- In this case, only 0.8% overhead if topstat is enabled

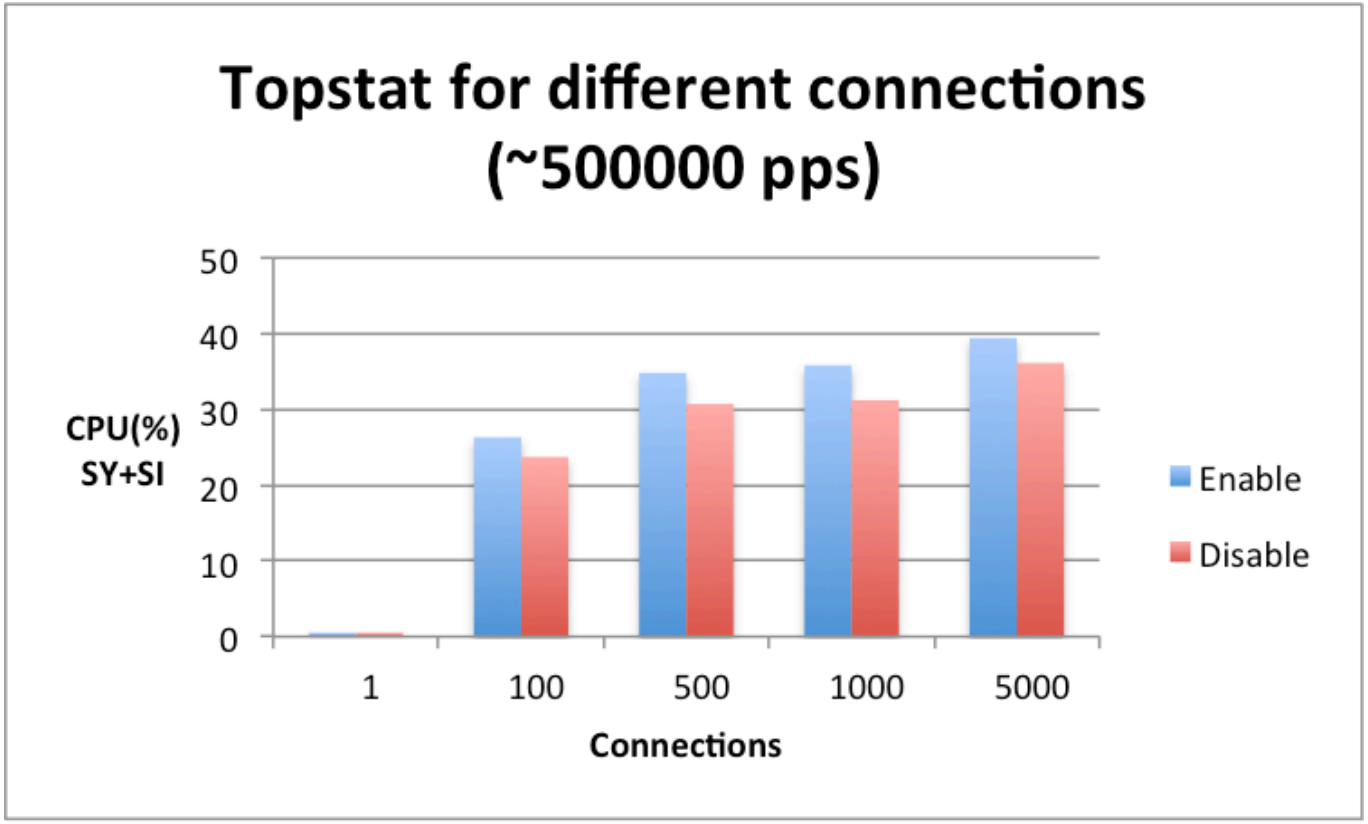
- We can also see the hotspot utilization from perf, topstat_push from 3.60% to 0.81%

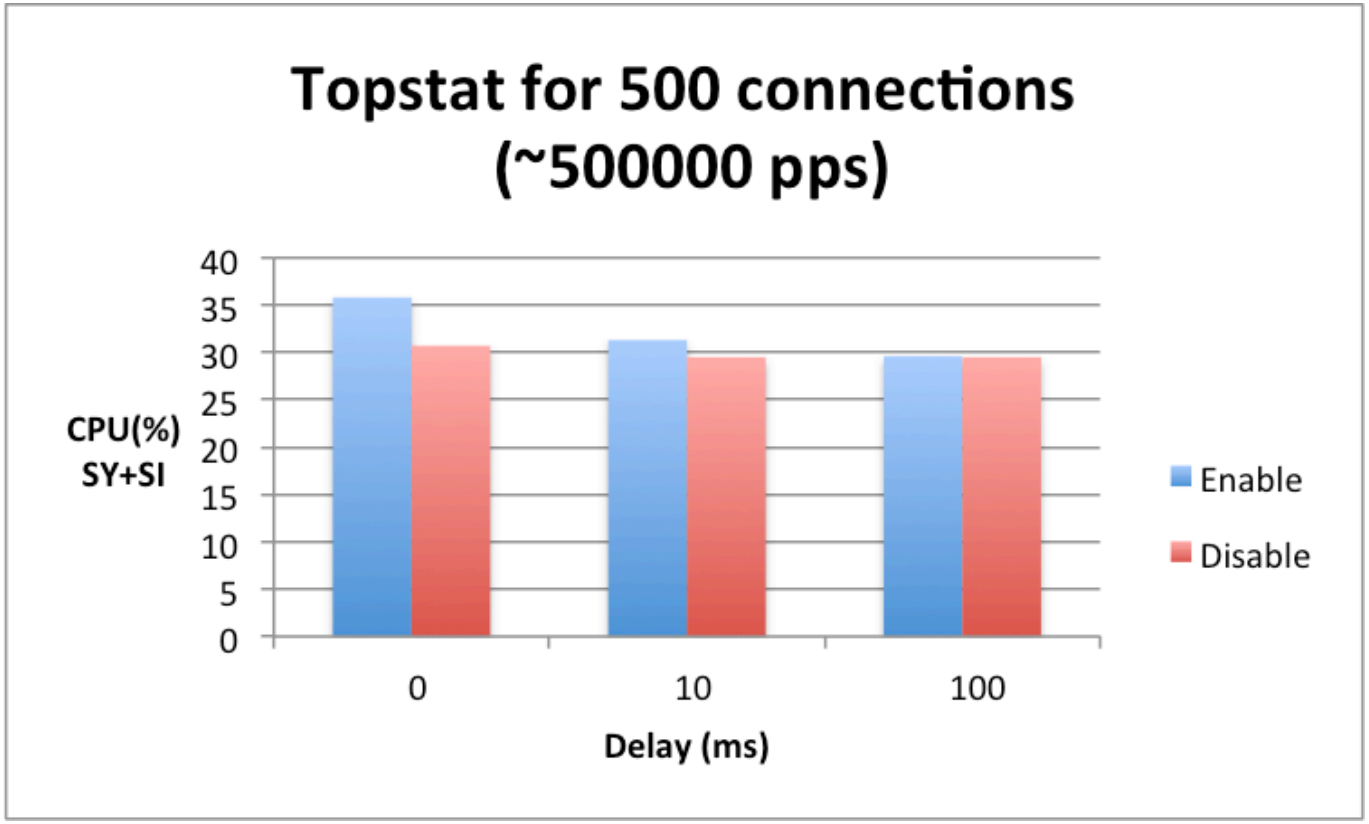
100 ms

Topstat	Enabled	Disabled
Top		
CPU Utilization	22.4%sy 7.2%si 1.2% us	22.3%sy 7.2%si 1.2% us
Perf		
topstat_push	0.39%	N/A
topstat_full_enabled	0.12%	0.12%
topstat_entry_value_insert	0.00%	N/A
topstat_aging	0.00%	N/A
topstat_entry_time_insert	0.01%	N/A

- In this case, only 0.1% overhead for CPU if topstat is enabled
- We can also see the hotspot utilization from perf, topstat_push from 3.60% to 0.39%
- It could be very useful and effective when there are many many connections and throughput is big

Summary


Topstat for different connections
(~500000 pps)
CPU(%)
SY+SI
50
40
30
20
10
0
1
100
500
1000
5000
Connections
Enable
Disable


Topstat for 500 connections
(~500000 pps)
CPU(%)
SY+SI
40
35
30
25
20
15
10
5
0
0
10
100
Delay (ms)
Enable
Disable

- Consequently, topstat will effect the performance (only consider CPU) when enabled, depending on the connections and pps, but can be limited and controlled effectively by delay (only less than 0.2% with 100ms delay). In a network, e.g. over hundreds of pps per connection, the delay will even not impact on statistics accuracy, but downgrade the overhead largely.

- One can design to balance between the performance and accuracy according to policy in the real production environment. In the test, for accuracy , tcp stack will push each packet to topstat, but one can also change the policy, e.g. 100 for each push.

- It has been running in production environment for over two years.
- We plan to open source and feedback to community in the near future.

Thanks

Q&A

Some recent work by Alibaba cloud kernel team:

- Container-like operating system (e.g. vlinux)
- Storage and file system (e.g. Instance Storage, Acache)
- Networking (e.g. RDMA*, e2eqos, RTshift, mirror network)

To join us: boyu.mt@taobao.com