DPDK

DATA PLANE DEVELOPMENT KIT

DPDK-based userspace TCP/IP stack testing

SHU MA

EBS – KUAFU

ALIBABA CLOUD

Agenda

1 Background

2 Current status

3 Our practice

4 Q&A

Background

✓ Luna

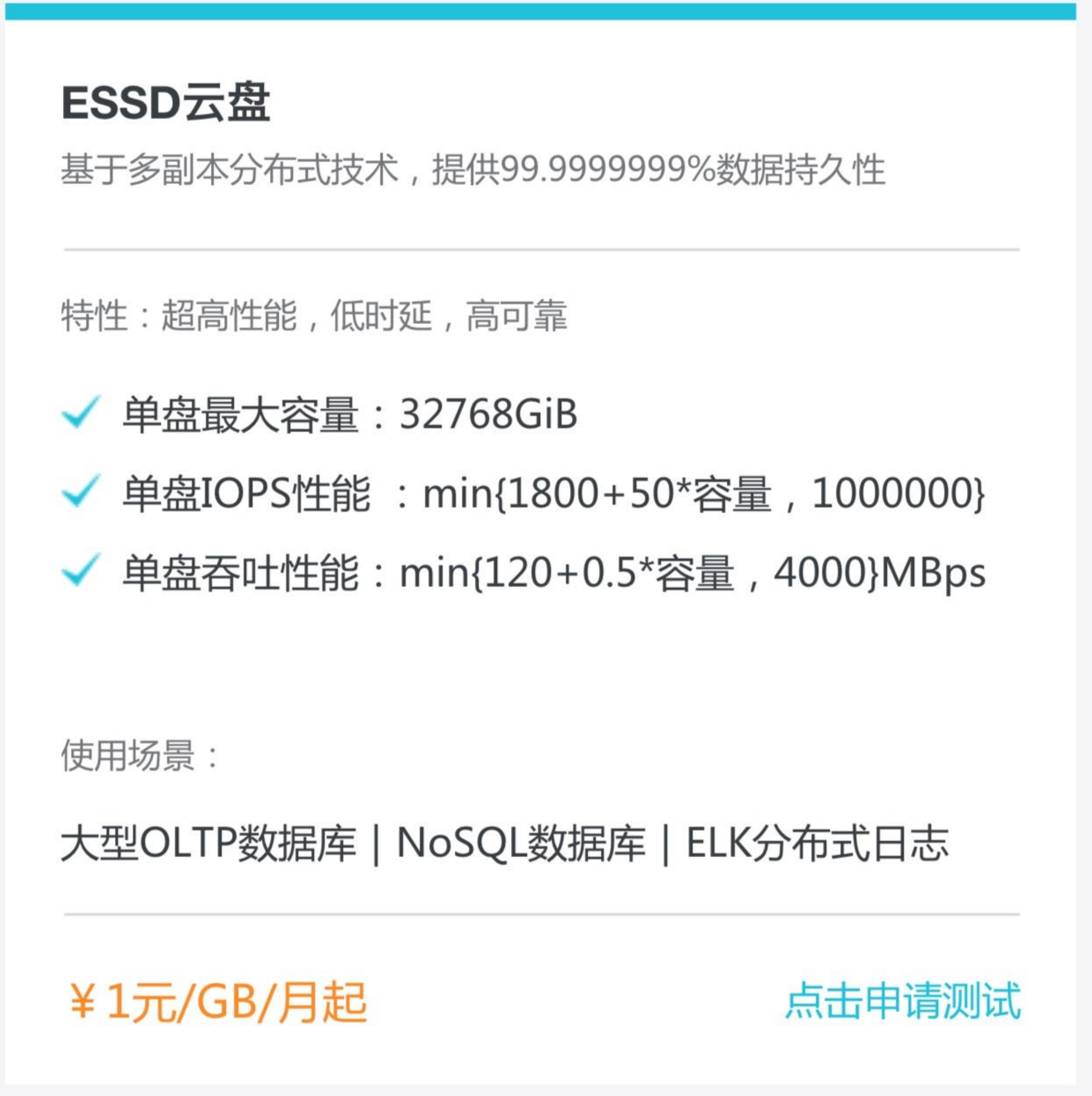
- high performance network framework
- DPDK
- Luna Stack (userspace lightweight TCP/IP stack)

✓ Product

- ESSD (cloud disk)
- hundreds of production clusters
- tens of thousands of machines

✓ Latency

- 1/3 kernel
- nearly as fast as RDMA



ESSD云盘

基于多副本分布式技术，提供99.9999999%数据持久性

特性：超高性能，低时延，高可靠

✓ 单盘最大容量：32768GiB

✓ 单盘IOPS性能 ：min{1800+50*容量，1000000}

✓ 单盘吞吐性能：min{120+0.5*容量，4000}MBps

使用场景：

大型OLTP数据库 | NoSQL数据库 | ELK分布式日志

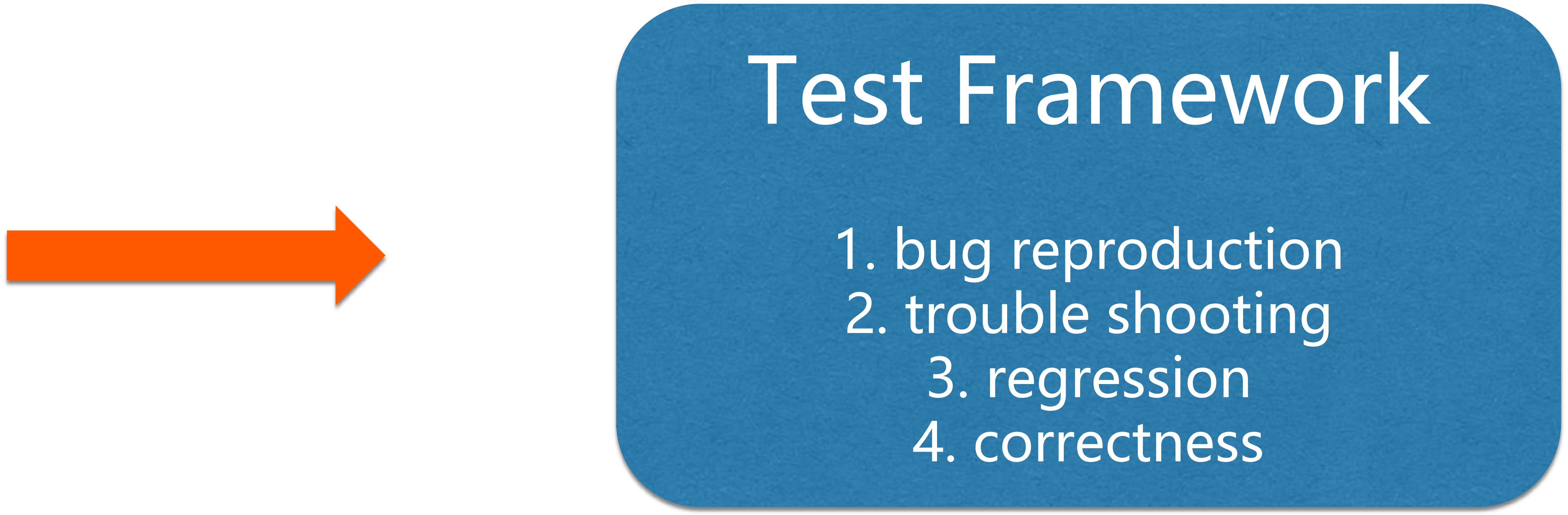
￥1元/GB/月起

https://www.aliyun.com/product/disk

Background

✓ Challenges in developing Luna Stack

- Bug is time-series-related
  - hard to reproduce
  - hard to troubleshoot
- Large number of corner cases
  - hard to fix
  - easy to break other cases
- Convince upper-layer developers
  - correctness
  - robustness



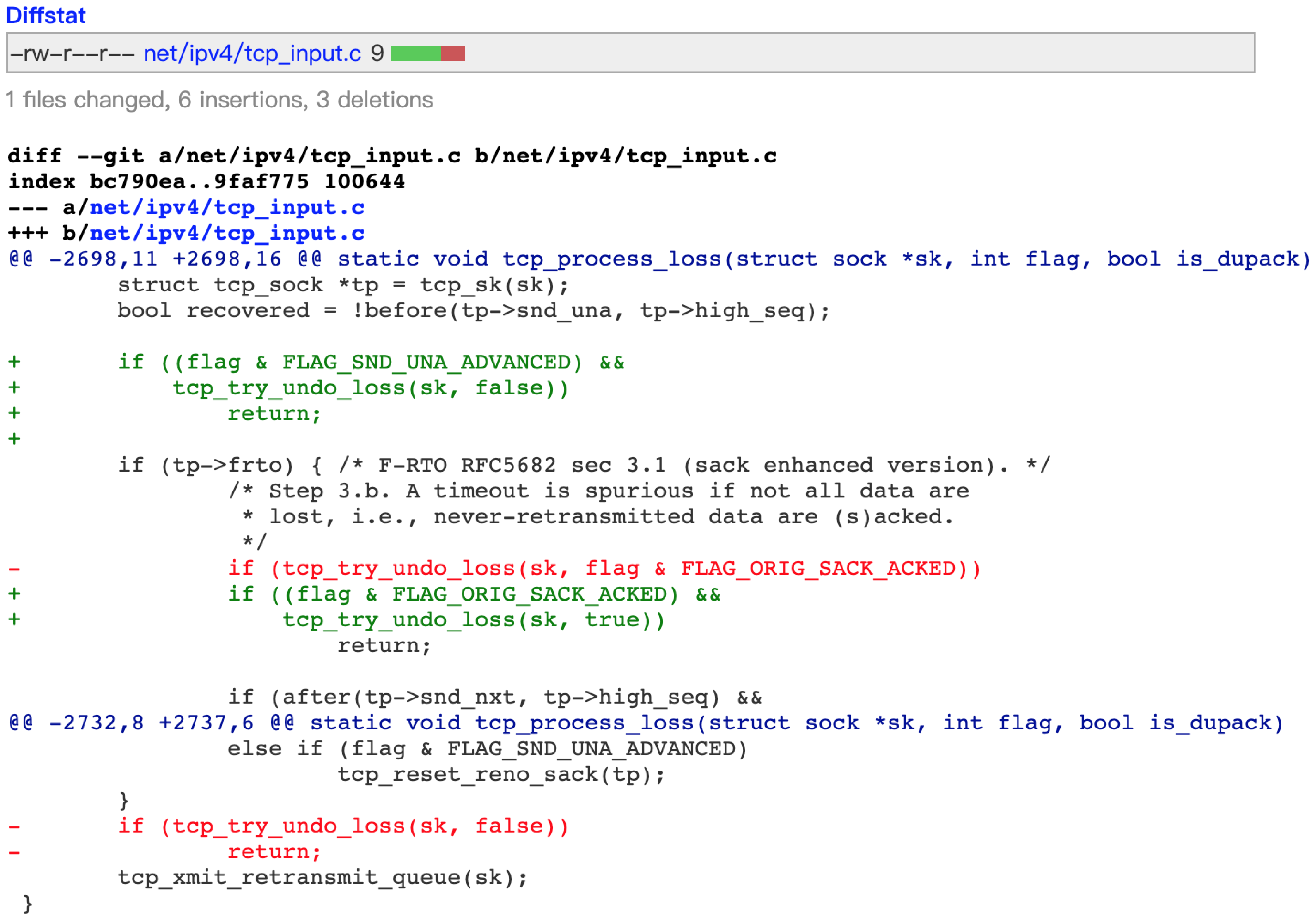
Test Framework

1. bug reproduction
2. trouble shooting
3. regression
4. correctness

Current status

- ✓ Linux kernel, FreeBSD
  - Internal
    - Low unit test coverage
  - External (LTP)
    - 20+ scripts for TCP/IP

- ✓ Testing approaches
  - Unit test （white box）
    - need to know code detail, hard to write
  - Function test （black box）
    - hard to create scenarios with strict time-series
  - packetdrill （grey box）
    - Google, open source
    - USENIX ATC 2013
    - 3 new TCP features, 10 kernel bugs



Diffstat

-rw-r--r-- net/ipv4/tcp_input.c 9

1 files changed, 6 insertions, 3 deletions

```
diff --git a/net/ipv4/tcp_input.c b/net/ipv4/tcp_input.c
index bc790ea..9faf775 100644
--- a/net/ipv4/tcp_input.c
+++ b/net/ipv4/tcp_input.c
@@ -2698,11 +2698,16 @@ static void tcp_process_loss(struct sock *sk, int flag, bool is_dupack)
        struct tcp_sock *tp = tcp_sk(sk);
        bool recovered = !before(tp->snd_una, tp->high_seq);

+       if ((flag & FLAG_SND_UNA_ADVANCED) &&
+           tcp_try_undo_loss(sk, false))
+               return;
+
        if (tp->frto) { /* F-RTO RFC5682 sec 3.1 (sack enhanced version). */
                /* Step 3.b. A timeout is spurious if not all data are
                 * lost, i.e., never-retransmitted data are (s)acked.
                 */
-               if (tcp_try_undo_loss(sk, flag & FLAG_ORIG_SACK_ACKED))
+               if ((flag & FLAG_ORIG_SACK_ACKED) &&
+                   tcp_try_undo_loss(sk, true))
                        return;

                if (after(tp->snd_nxt, tp->high_seq) &&
@@ -2732,8 +2737,6 @@ static void tcp_process_loss(struct sock *sk, int flag, bool is_dupack)
                else if (flag & FLAG_SND_UNA_ADVANCED)
                        tcp_reset_reno_sack(tp);
        }
-       if (tcp_try_undo_loss(sk, false))
-               return;
        tcp_xmit_retransmit_queue(sk);
 }
```

bug fix for Linux kernel

Packetdrill: script

- ✓ 4 statements
  - packets
    - tcpdump-like syntax
    - **inbound**, **outbound**
  - **system calls**
    - strace-like syntax
  - **shell commands**
  - **python scripts**

- ✓ time model
  - relative time
    - *+0, +.1*
  - absolute time
    - *0.100, 0.100…0.200*

```
0    socket(..., SOCK_STREAM, IPPROTO_TCP) = 3
+0   bind(3, ..., ...) = 0
+0   listen(3, 1) = 0

+0   < S 0:0(0) win 32792 <mss 1460, nop, wscale 7, nop, nop, TS val 0 ecr 0>
+0   > S. 0:0(0) ack 1 <mss 1460, nop, nop, TS val 0 ecr 0, nop, wscale 7>
+0   `netstat -anp | grep 8080 | grep SYN_RCVD`  // examine TCP state

+.1  < . 1:1(0) ack 1 win 100
+0   accept(3, ..., ...) = 4
+0   %{ assert tcpi_snd_cwnd = 10 }%   // examine TCP_INFO

+0   write(4, ..., 1000) = 1000  // send 1 packet
+0   > . 1:1001(1000) ack 1

+.2  > . 1:1001(1000) ack 1 // RTO retrans, 200ms
+.4  > . 1:1001(1000) ack 1 // RTO retarns, 400ms
```

100 lines of UT -> 13 lines of script

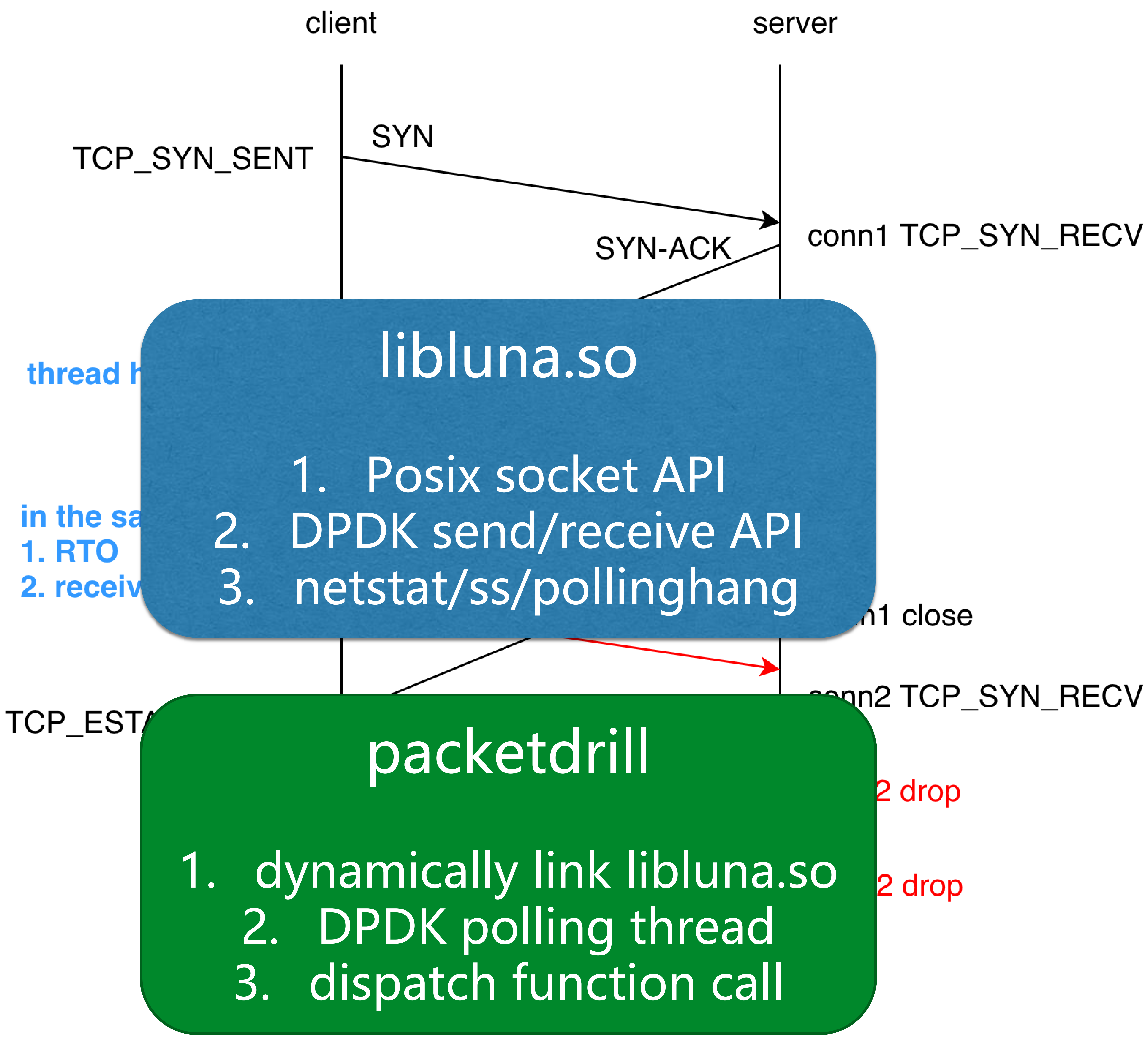
Packetdrill: pros & cons

✓ Pros

- time-series
- developer-friendly script syntax
- high maintainability
- reusable among different stacks

✓ Cons

- kernel TCP/IP
- TCP_INFO/netstat/ss
- **polling related time-series**



libluna.so

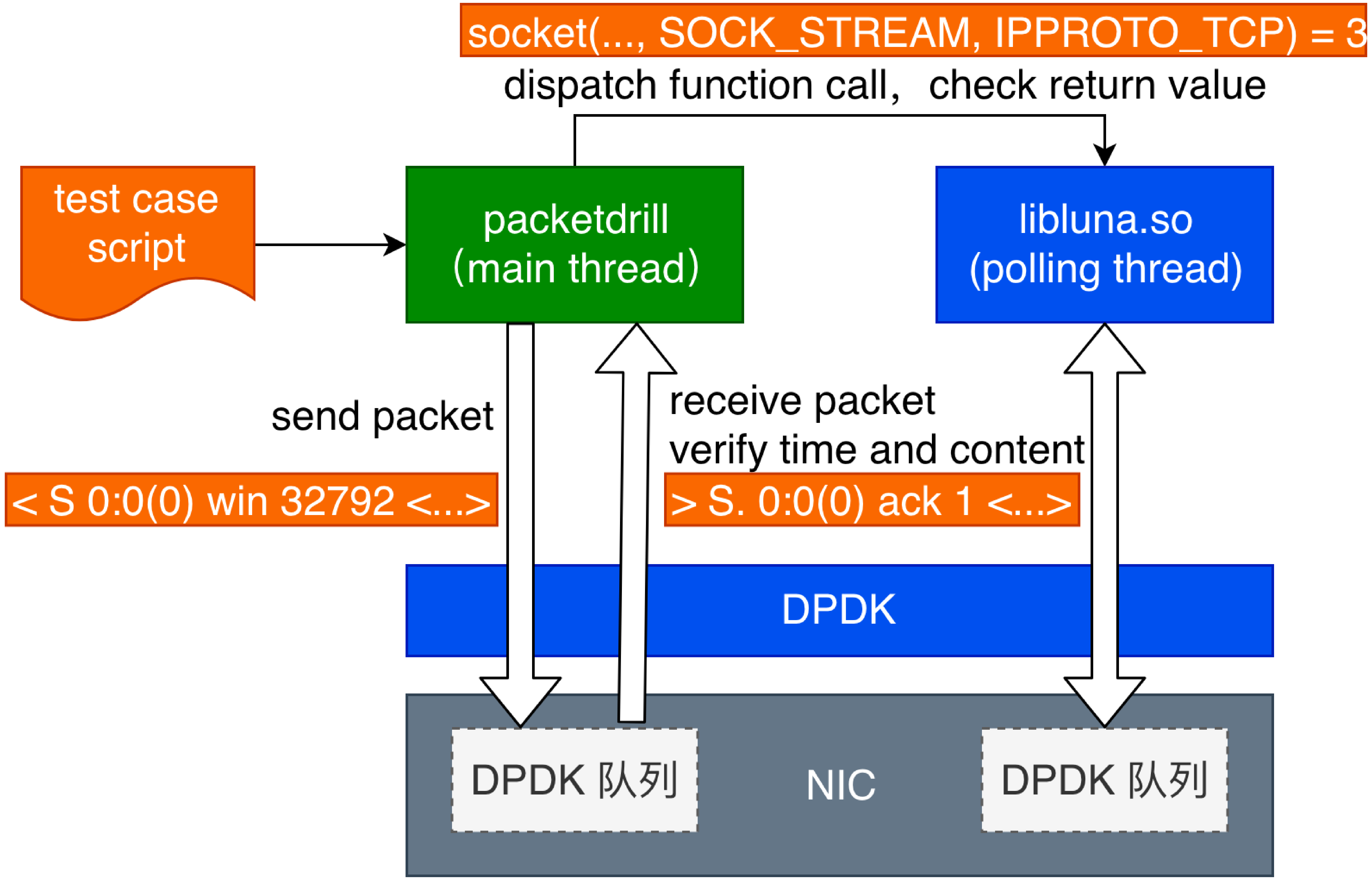
1. Posix socket API
2. DPDK send/receive API
3. netstat/ss/pollinghang

packetdrill

1. dynamically link libluna.so
2. DPDK polling thread
3. dispatch function call

Modified packetdrill

- ✓ Main thread
  - read script line by line
  - send/receive packets via DPDK
  - dispatch function
  - run shell tools
    - inspect: **netstat**, **ss**
    - interfere: **pollinghang**?time=10
- ✓ Stack thread
  - polling mode
  - userspace stack initialization
  - call dispatched function
- ✓ Usage
  - ./packetdrill ./test.pkt
  - ./packetdrill --userspace_stack --so_filename=libluna.so ./test.pkt
  - Compare between Luna TCP and kernel TCP


socket(..., SOCK_STREAM, IPPROTO_TCP) = 3
dispatch function call， check return value
test case script
packetdrill (main thread)
libluna.so (polling thread)
send packet
< S 0:0(0) win 32792 <...>
receive packet
verify time and content
> S. 0:0(0) ack 1 <...>
DPDK
DPDK 队列
NIC
DPDK 队列

Modified packetdrill

```
0    socket(..., SOCK_STREAM, IPPROTO_TCP) = 3
+0   bind(3, ..., ...) = 0
+0   listen(3, 1) = 0

+0   < S 0:0(0) win 32792 <…>
+0   > S. 0:0(0) ack 1 <…>
+0   `netstat -anp | grep 8080 | grep SYN_RCVD`

+.1  < . 1:1(0) ack 1 win 100
+0   accept(3, ..., ...) = 4
+0   %{ assert tcpi_snd_cwnd = 10 }%

+0   write(4, ..., 1000) = 1000
+0   > . 1:1001(1000) ack 1

+.2  > . 1:1001(1000) ack 1
+.4  > . 1:1001(1000) ack 1
```

script for kernel TCP

```
0    socket(..., SOCK_STREAM, IPPROTO_TCP) = 3
+0   bind(3, ..., ...) = 0
+0   listen(3, 1) = 0

+0   < S 0:0(0) win 32792 <…>
+0   > S. 0:0(0) ack 1 <…>
+0   `curl http://127.0.0.1:8899/netstat | grep 8080 | grep SYN_RCVD`

+.1  < . 1:1(0) ack 1 win 100
+0   accept(3, ..., ...) = 4
+0  `curl http://127.0.0.1:8899/ss | grep 8080 |
     sed 's/^.*\(cwnd:[0-9]*\).*$/\1/' | grep 10`

+0   write(4, ..., 1000) = 1000
+0   > . 1:1001(1000) ack 1

+.2  > . 1:1001(1000) ack 1
+.4  > . 1:1001(1000) ack 1
```

script for userspace TCP

Experience in Alibaba

- ✓ 75 test cases for Luna TCP
  - TCP state transmission
  - exceptional packet handling
  - congestion control、keep alive、custom features …
  - RFC 793, 1122, 3042, 5681, 6582

- ✓ reproduction
  - fix 3 bugs in production

- ✓ regression
  - added to Jenkins

DPDK
DATA PLANE DEVELOPMENT KIT

“

Thank You !

”

Q&A