Содержание

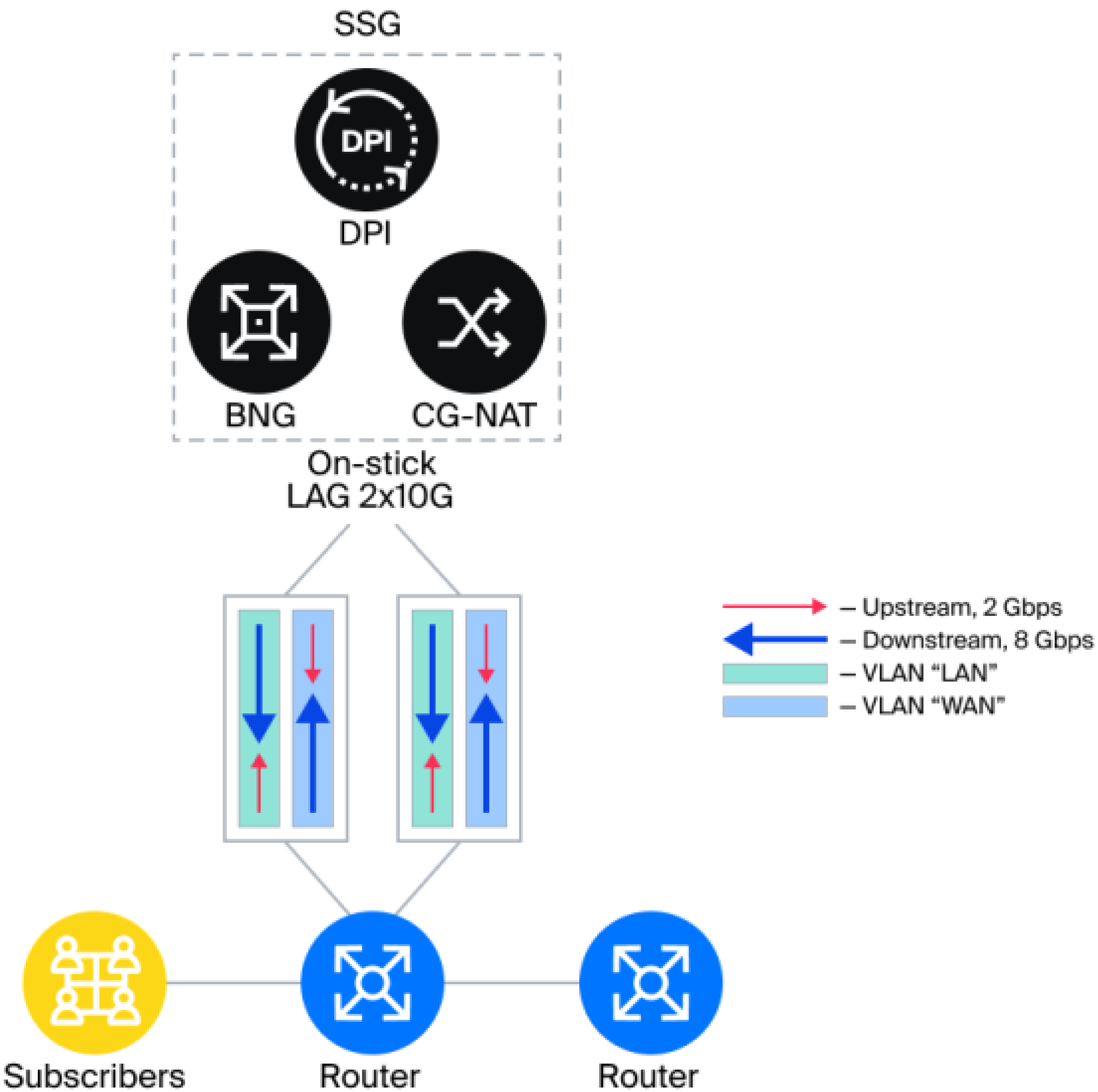
Configuring on-stick and LAG/LACP

Configuring on-stick

Blog: On-stick — a new mode of the Stingray Service Gateway to save router ports

[FastDPI 12+]

On-stick allows you to save on physical hardware. FastDPI usually works with bridges, bridging two physical ports (devices). For an on-stick device the physical port is one, on which fastDPI itself creates virtual ports - on the subscriber side (`subs`) and internet side (`inet`).



Each op-stick port is described in a specific way: first the base physical port is described using `dpdk_device`, then the virtual ports based on the base port are described.
Description of the base devices:

```
dpdk_device=port1:pci:04:00.0
```

```
dpdk_device=port2:pci:04:00.1
```

Description of on-stick based on devices `port`:

```
onstick_device {
    base=port1
    filter=<subs side filter expression>
    subs=subs1
    inet=inet1
}
onstick_device {
    base=port2
    filter=<subs side filter expression>
    subs=subs2
    inet=inet2
}
```

where:
`base` — base device
`filter` — boolean expression to determine the direction of the packet (filter). If this expression returns `true`, the packet is from the `subs` side, otherwise it is from the `inet` side
`subs` — the name of the device on the `subs` side.
`inet` — the name of the device on the `inet` side.

Setting Bridges.

> Basic `port` type devices CANNOT be included in any bridges

```
in_dev=subs1:subs2
out_dev=inet1:inet2
```

Wherever you need to specify a device, you should use virtual devices (in this example, `subs1`, `subs2` and `inet1`, `inet2`). The basic on-stick devices `port1` and `port2` is specified only when describing the on-stick device and nowhere else.

In the on-stick port description, the most important part is the `filter` clause to determine the direction of the packet (subs → inet or inet → subs). Packet direction is an important attribute of a packet in fastDPI on which processing depends. `filter` specifies a boolean expression over the L2 properties of the packet. If this expression returns `true`, the packet is on the `subs` side, otherwise it is on the `inet` side (uplink, internet).

The basis of a `filter` expression are terms that are combined into a logical expression using the logical operators & (AND) and `|` (OR), the parentheses `(` and `')`, and the negation `'!`. The & operator is higher priority than `|`; by analogy with arithmetic expressions, we can think of & as multiplication and `|` as addition, which is the basis for placing parentheses.

Terms specify elementary expressions over L2 properties of a packet. The following terms exist (case is important):

- `vlan(list)`. - a single VLAN packet with the specified VLAN numbers, e.g.:

vlan(56,78,890)
- vlan is a packet with any single VLAN.
- qinq is a Q-in-Q packet.
- pppoe is a PPPoE packet.
- smac(MAC address) - source MAC address of the packet, example: smac(01:02:03:04:05:06)
- dmac(MAC address) - destination MAC address of the packet, example: dmac(01:02:03:04:05:07)

Examples (recall that filter specifies an expression for the subs side):

- subscriber-side Q-in-Q network terminated in a single VLAN: filter=qinq
- heterogeneous network: subscriber-side Q-in-Q or PPPoE in a VLAN: filter=qinq | pppoe. Here, the fact that PPPoE is encapsulated in a VLAN is irrelevant: PPPoE is terminated by BRAS, so PPPoE on the inet side is not possible.
- single VLAN, on the inet side VLAN=609, all other VLANs are subs: filter=vlan & !vlan(609). Here we need to explain in more detail. For the inet side, the filter expression would look like this: filter=vlan(609), but the filter defines the expression for the subs side, so it would seem that the negation is enough: filter=!vlan(609). But this expression will be true for any packet other than the packet with VLAN=609, even without VLAN. So you should specify that the packet **should** contain a single VLAN tag, but except for VLAN=609: filter=vlan & !vlan(609)
- on the inet side, the MAC address of the borderer is 3c:fd:fe:ed:b8:ad: filter=!smac(3c:fd:fe:ed:b8:ad). - all packets with source MAC not equal to the Border MAC address are subs-side packets.

A formal description of the grammar of the filter expression:

```
filter ::= and | and '|' filter
and    ::= mult | mult '&' and
mult   ::= '!' mult | term | '(' filter ')'
term   ::= vlan | qinq | pppoe | smac | dmac

vlan   ::= 'vlan' | 'vlan' '(' list_int ')'
qinq   ::= 'qinq'
pppoe  ::= 'pppoe'
smac   ::= 'smac' '(' mac_address ')'
dmac   ::= 'dmac' '(' mac_address ')'
mac_address ::= xx:xx:xx:xx:xx:xx
```

LAG/LACP port aggregation

Port aggregation by SSG is supported for the following modes in-line и on-stick.

LACP port aggregation

```
lacp=0
```

Allowable values of the `lacp` parameter:

- 0 (default) - LACP is disabled, SSG does not hold LAG, but freely skips it
- 1 - LAG in passive mode: we don't send periodic LACPDUs, but we respond to LACPDUs that arrive
- 2 - LAG in active mode: we send periodic LACPDUs.

The aggregation traces the traffic balancing.

Configuring LAG

LAGs can include either regular ports or on-stick ports, no mixing is allowed. LAG on on-stick is organized on the base (physical) port. LAG is implemented in fastDPI at logical level: there is no single bond device, inside fastDPI work is done with ports as before.

The maximum number of ports in a LAG is 18.

3 different configurations are possible:

- LAG on the `subs` side, no LAG on the `inet` side
- LAG on the `inet` side, no LAG on the `subs` side.
- LAG on the `subs` side AND on the `inet` side

Requirements for devices included in a LAG:

- a device can only be part of one LAG device;
- all devices in a LAG must have the same speed;

Currently, LAGs are configured in `fastdpi.conf` without the ability to be applied on the fly, i.e. a fastDPI restart is required when the LAG configuration is changed.

LAG Description:
Each LAG requires a separate `lag` section. Only physical interfaces can be included in a LAG; mixing physical and logical interfaces is not allowed.

```
lag {
    name=

    device=
    device=

    system_id=
    priority=
    short_timeout=
}
```

where:

name - optional name of the LAG, used for log output.
`device` - lists all physical interfaces included in the LAG. The LAG must contain at least 2 devices.
`system_id` - MAC address is the `system_id` of this LAG. If not specified, "arp_mac" is used.
`priority` - system priority for this LAG, number in the range 1 - 65535, default is 32768.
`short_timeout` - short (on) or long (off) LACP timeout.

Applying balancing to outgoing LAG traffic

The type of the applied balancing algorithm is specified by the `lag.balance_algo` parameter:

- `-1` — without load balancing — the packet is sent to the paired bridge port
- `0` (default) — load balancing by internal `session_id` (default balancing). `session_id` is used as the hash
- `1` — hash based on `flow key <srcIP, dstIP, srcPort, dstPort, proto>`. if there is no `flow`, balancing is performed by `session_id`

Additional hash configuration parameters in the `lag` section: `hash_seed`, `hash_offset`, `hash_bits`
How many significant bits we take from the 64-bit hash during balancing. The balancing algorithm in the general case looks like this:

- calculate a 64-bit hash of certain packet fields and `hash_seed`;
- from the 64-bit hash we take the `hash_bits` bits starting from the `hash_offset` bit;
- use the resulting number N to determine the port number in LAG: `port := N mod LAG_active_port_count`, i.e.

```
port := ((hash(packet, hash_seed) >> hash_offset) & (2^hash_bits - 1))
mod LAG_active_port_count
```

Example:

```
//       +-------------------------------------------------+
// hash: |                               XXXXXXXXXX------|
//       +-------------------------------------------------+
//                                       ^         ^
//                                       |         hash_offset = 6
//                                       hash_bits = 10
```

Only basic on-stick devices must be specified in the LAG description. Mixing on-stick and conventional devices in one LAG is not allowed.

During aggregation, a traffic balancing trace is traced.

Example of on-stick + LACP configuration for two physical interfaces

In this scenario, subscriber traffic in VLANs 101, 102, and QinQ

```
dpdk_device=port1:pci:86:00.0
dpdk_device=port2:pci:86:00.1

lag {
    name=LACP
    device=86:00.1
    device=86:00.0
    lacp=1
    system_id=6c:b3:11:60:fa:66
    priority=32768
    balance_algo=0
}

onstick_device {
    base=port1
    filter=vlan(101,102) | qinq
    subs=subs1
    inet=inet1
}

onstick_device {
    base=port2
    filter=vlan(101,102) | qinq
    subs=subs2
    inet=inet2
}

in_dev=subs1:subs2
out_dev=inet1:inet2
```

Diagnosis

LACP diagnostics is performed using the pcap entry. To do this, the parameter `pcap` must be added to the LAG description.
Parameter values:

- on — enable pcap recording
- off — disable pcap recording

```
# tracing (writing to pcap) of LACP packets of the given LAG
    #pcap=on
```

Logging is enabled via the `bras_trace` parameter. See the BRAS tracing section for possible values.