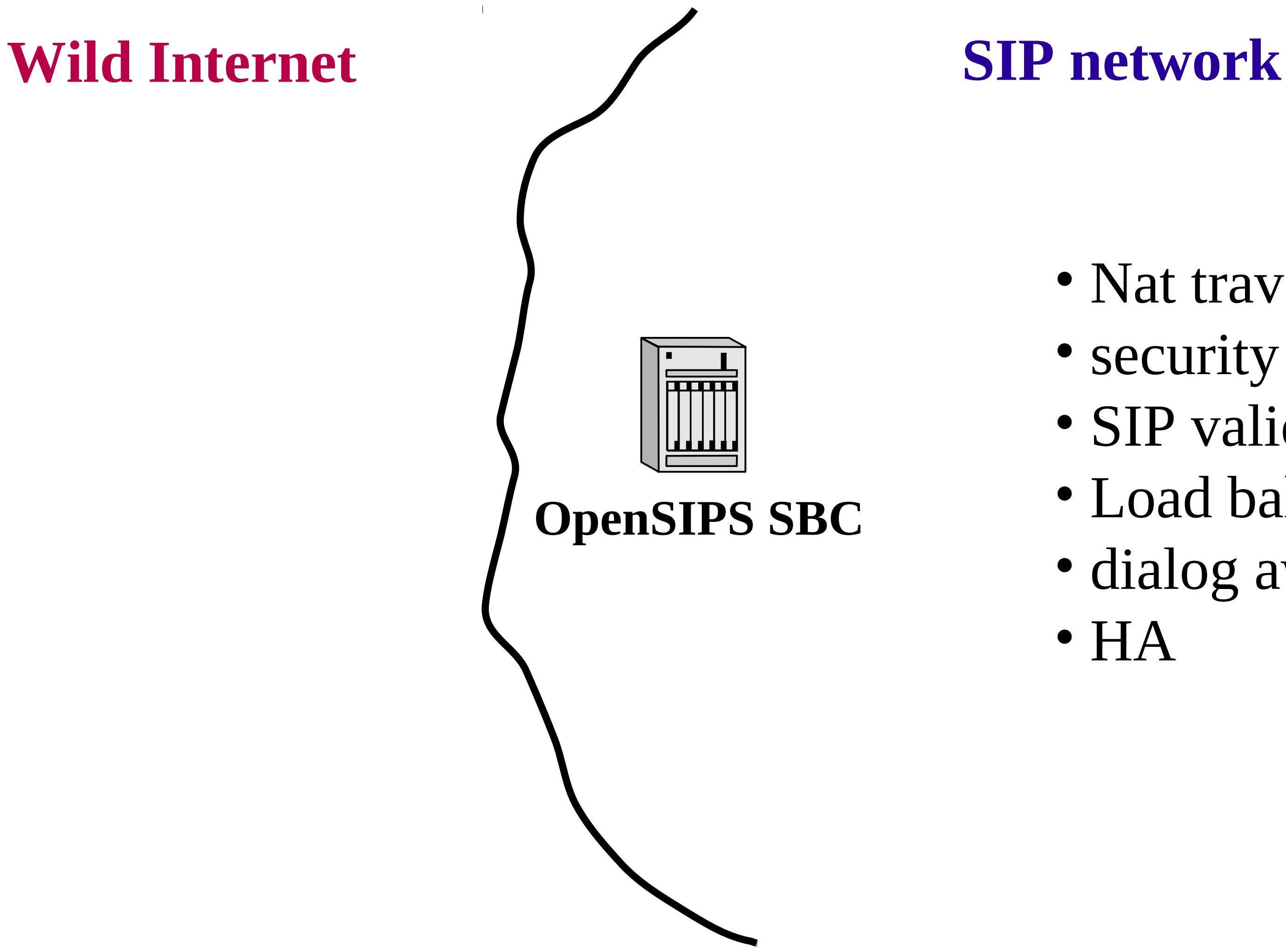
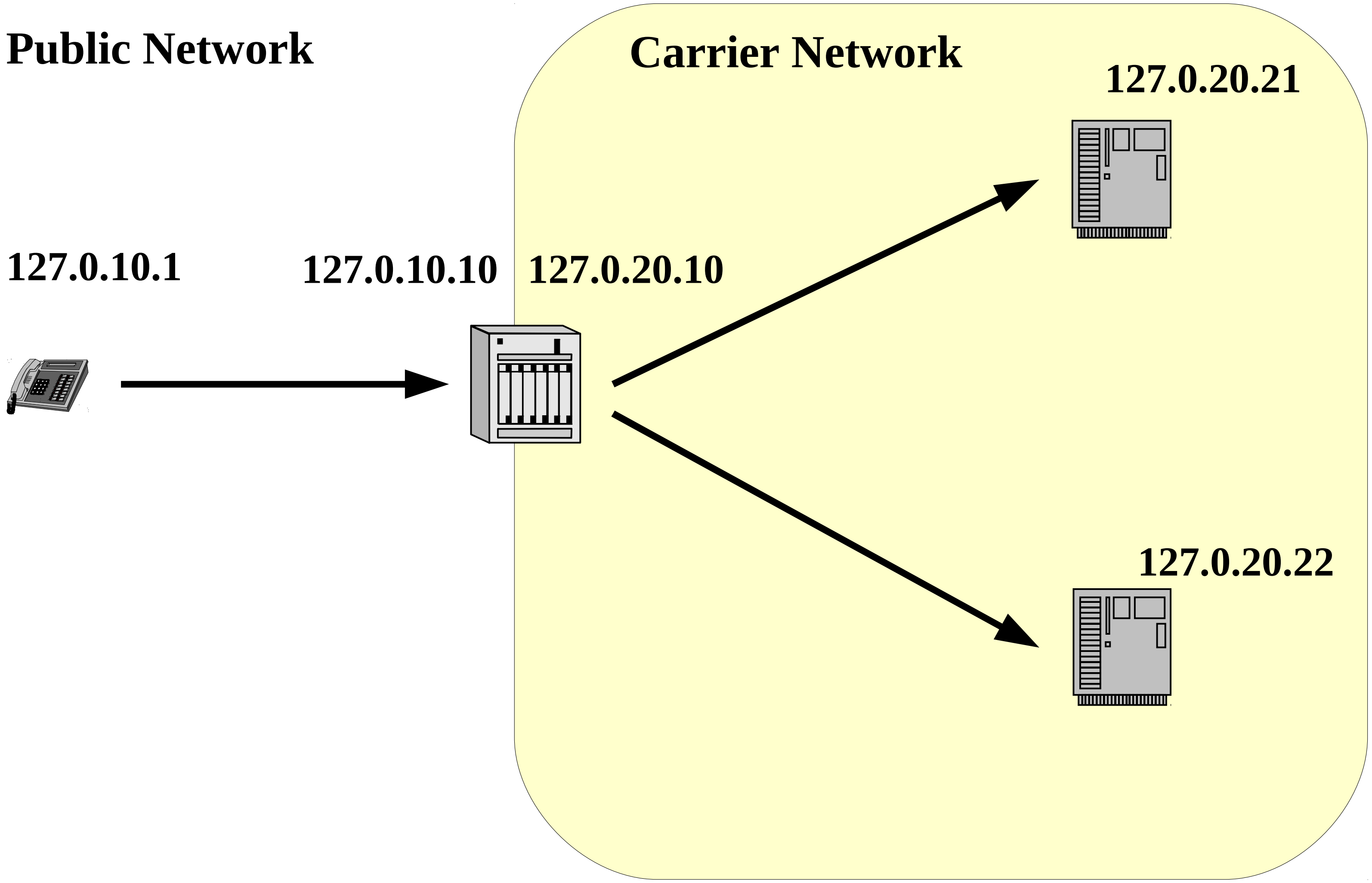
OpenSIPS as Frontend for PBXes

Bogdan-Andrei Iancu
Founder OpenSIPS Project
OpenSIPS Solutions

OpenSIPS frontend



- Nat traversal
- security filter
- SIP validation
- Load balancer
- dialog aware
- HA


Public Network
Carrier Network
127.0.20.21
127.0.10.1
127.0.10.10
127.0.20.10
127.0.20.22

LB versus dispatching

- Dispatching ?
  - no information on the load of the peers
  - does an probabilistic distribution of the traffic among the peers.
  - assumes all the peers are identical
  - it is fast as nothing more than transaction state is needed
- Load Balancing ?
  - based on dialog module, counts the load of the peers
  - peers may be different (as resources and load)
  - can receive feedback from peers
  - does not require any intelligence from the peers.

Dispatching and failover

- as dispatching assumes multiple peers, it is natural to do failover between them if one is not responding
- how to detect if a peers is down
  - no reply from it (internal timeout)
  - 5 class reply – internal server error
  - 6 class reply – global error
  - some particular 4 class reply – about load or availability

Testing the failure case in failure route:

```
if ( t_check_status( "[56][0-9][0-9]" ) ||                # peer error
(t_local_replied("all") && t_check_status("408") ) ||    # local 408
t_check_status("409") )                                  # special
```

```
route {
        if ( !ds_select_dst("2", "0")) {  # over CallID
             sl_send_reply(“503”,”Service Unavailable”);
             exit;
        }
        t_on_failure(“1”);
        t_relay();
}
failure_route[1] {
        if (failure_condition) {
                ds_mark_dst();   #avoid this peer for next dispatching
                if (!ds_next_dst()) {
                        t_reply(“503”,”Service Unavailable”);
                        exit;
                }
                t_on_failure(“1”);
                t_relay();
        }
}
```

Failover logic

(1)collect an ordered initial set of peers

(2)use the first one from the set

(3)send out the request

(4)if failure is detected, mark the peer as disabled and check if there is any other peer left in the set

- use the next peer from the set and add it a branch for serial forking

(1)go to step (3)

Thank you for your attention
You can find out more at www.opensips.org
bogdan@opensips.org
www.opensips-solutions.com

Questions are welcome