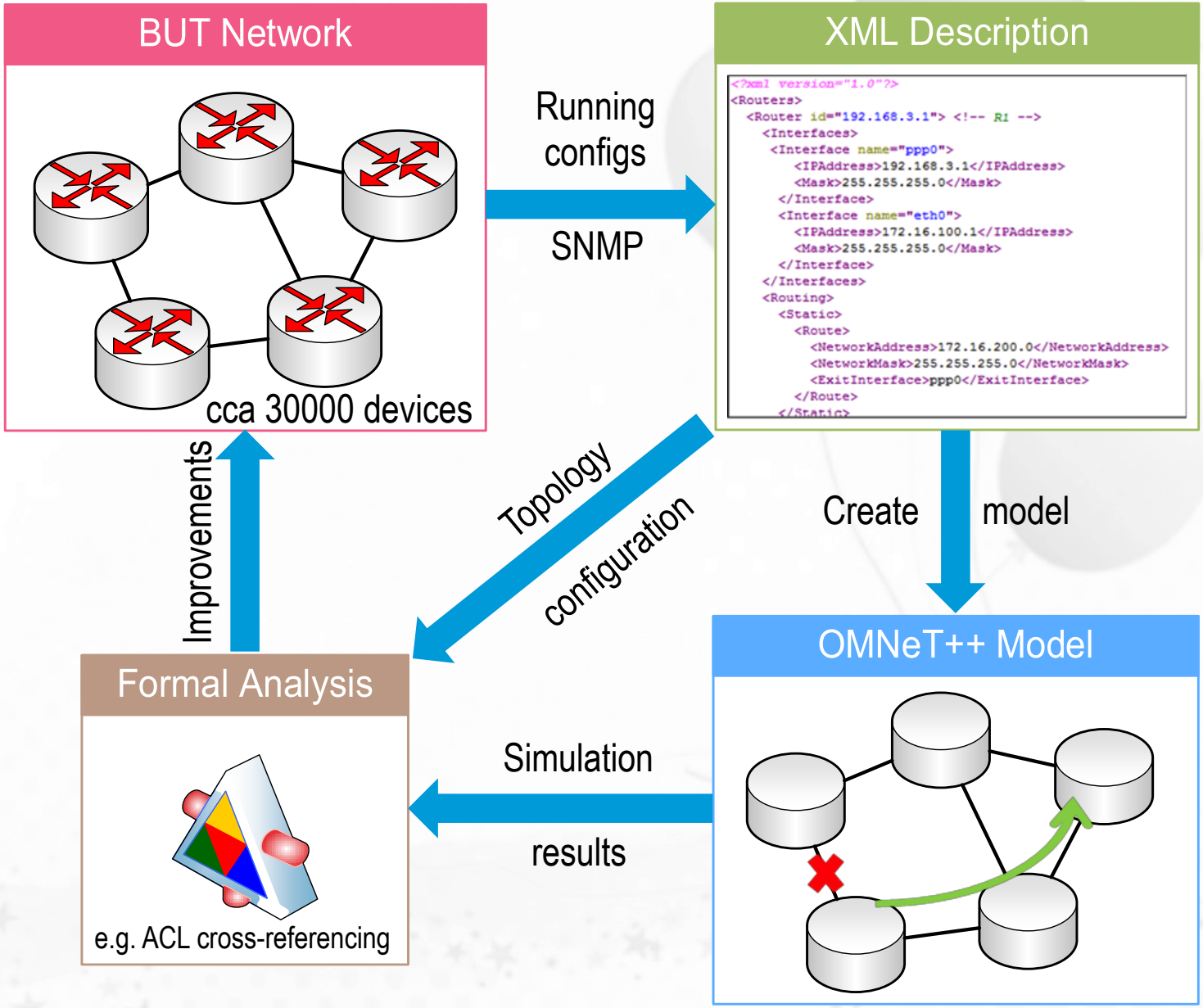
IPv6 Unicast and IPv4 Multicast Routing in OMNeT++

Vladimír VESELÝ,
Ondřej RYŠAVÝ, Miroslav ŠVÉDA

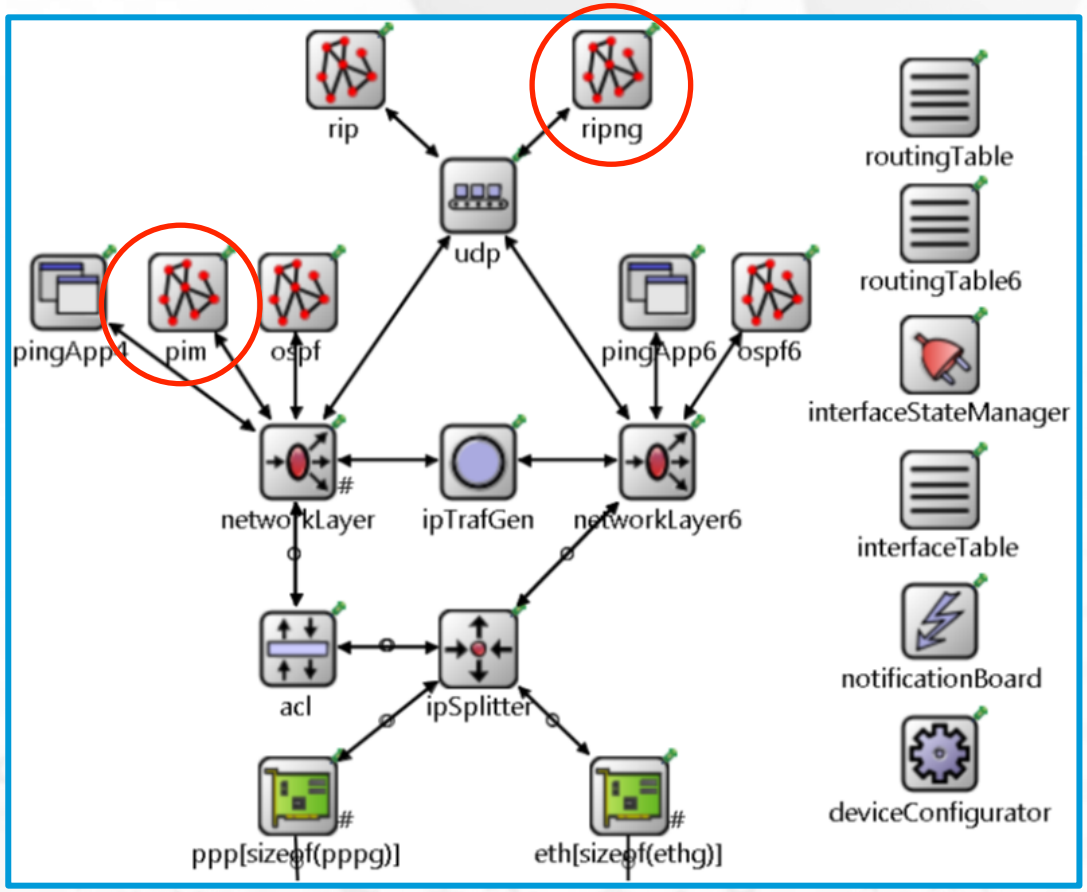
MOTIVATION



CONTRIBUTION

- Extending functionality of the INET framework
  - IP based wired networks and protocols

- Unicast routing
  - RIPv2
  - RIPng
  - OSPF v2
  - ISIS
- QoS
  - DSCP
  - Queues
- Multicast
  - IGMP/MLD
  - PIM-DM



ANSARouter compound module inspired by Cisco

Basic Info

- IPv6 twin of distance vector RIPv2
- Hop count as the metric (15 max, 16 unreachable)
- Destination address FF02::9, destination UDP port 521
- Split-horizon rule to prevent counting to infinity
- Bellman-Ford algorithm
- RIPng Request
  RIPng Response
  messages:

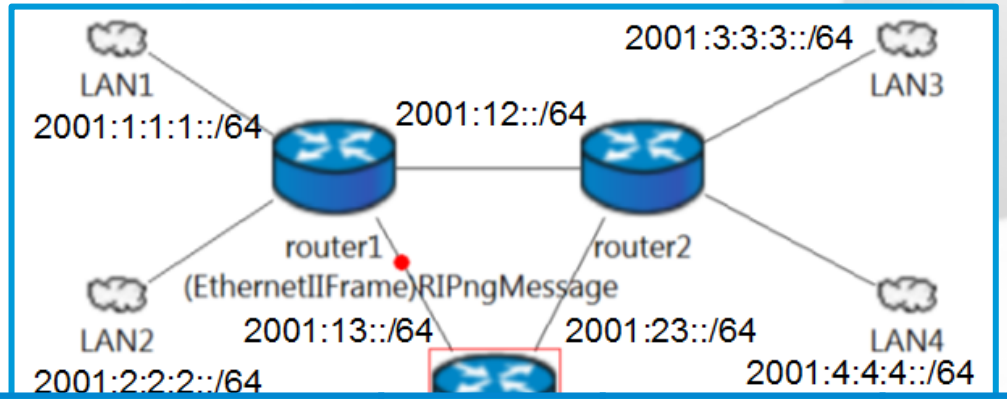
```
//  RIPng command
enum RIPngCommand
{
    RIPngRequest = 1;
    RIPngResponse = 2;
}

// Routing table entry structure
class RIPngRTE
{
    IPv6Address IPv6Prefix;   //128 bytes
    uint16_t    routeTag;     //2 bytes
    char        prefixLen;    //1 byte
    char        metric;       //1 byte
}

// Represents a RIPng message
packet RIPngMessage
{
    char     command enum(RIPngCommand); //1 byte
    char     version = 1;                //1 byte
    uint16_t mustBeZero  = 0;            //2 bytes
    RIPngRTE rtes[];                     //RTE
}
```

RIPng OMNeT++ MSG structure

VERIFICATION

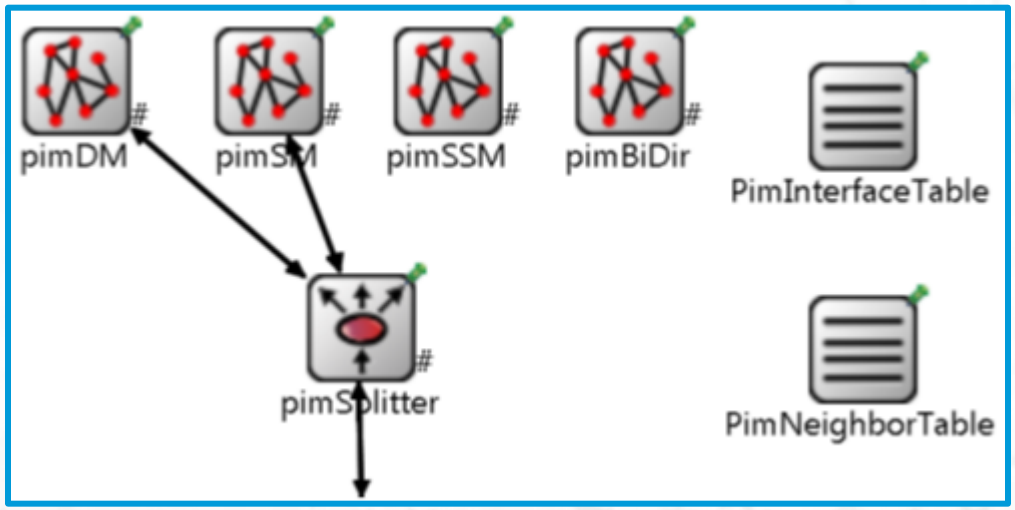


Message	Sender	Simul. [s]	Real [s]
RIPng Request	R1	0.000	0.000
RIPng Response	R1	0.000	0.321
RIPng Request	R2	0.001	0.620
RIPng Response	R1	0.001	0.640
RIPng Response	R2	211.000	211.888

[1] = ROUTING_PROT 2001:1:1:1::/64 [120/2] via fe80::8aa:ff:fe00:2, eth0
[2] = ROUTING_PROT 2001:2:2:2::/64 [120/2] via fe80::8aa:ff:fe00:2, eth0
[3] = ROUTING_PROT 2001:3:3:3::/64 [120/2] via fe80::8aa:ff:fe00:6, eth1
[4] = ROUTING_PROT 2001:4:4:4::/64 [120/2] via fe80::8aa:ff:fe00:6, eth1
[5] = STATIC 2001:13::/64 [1/10] via <unspec>, eth0
[6] = STATIC 2001:5:5:5::/64 [1/10] via <unspec>, eth2
[7] = STATIC 2001:6:6:6::/64 [1/10] via <unspec>, eth3
[8] = STATIC 2001:23::/64 [1/10] via <unspec>, eth1

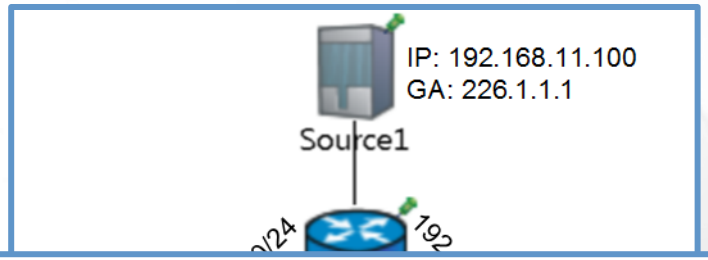
Basic Info

- Independent on unicast routing protocol
- Uses source trees to describe path from sender to receiver(s)
  - Incoming RPF interface
  - Outgoing interface list
- Simple “flood and prune” mechanism…
- …but with complex signalization (plenty of messages)



Structure of PIM module

VERIFICATION



showMRoute (std::vector<std::string>)
showMRoute[2] (std::string)
[0] = (192.168.11.100, 226.1.1.1), flags: C
Incoming interface: eth0, RPF neighbor 192.168.12.1
Outgoing interface list:
eth2, Forward/Dense

[1] = (192.168.33.100, 226.2.2.2), flags: P

Message	Sender	Simul. [s]	Real [s]
PIM Hello	R1	30.435	25.461
PIM Prune/Join	R3	87.000	87.664
PIM Graft	R2	144.000	144.406
PIM Graft-Ack	R1	144.000	144.440
PIM Prune/Join	R2	366.000	364.496

Conclusion

- "Not just merely" code contribution…
  - …so far our simulation models work fine (proved by enterprise)
  - …more interesting simulations benefiting our extensions will follow
- Near future
  - wrap PIM-SM and ISIS
  - FHRP
  - OSPFv3
- More distant near future ☺
  - RINA
  - LISP ITR Map Cache synchronization problem
  - EIGRP (yes, Cisco published specs via IETF)

The End

- *Feel free to ask any question…*

- Why are there no mentions about other OMNeT++ PIM implementations that “do exist”?
  - Because their authors do not want to share them ☹
  - http://nes.fit.vutbr.cz/ansa