A proposed architecture for integrating Active Networks and MPLS

Sanda Dragos and Martin Collier

email: dragoss@eeng.dcu.ie
collierm@eeng.dcu.ie
Multiprotocol Label Switching is a label-based packet switching technique.
Active Networks

“active” in two ways:
- nodes can perform computations;
- users can program the network.

There are three architectures approaches:
Active Networks

“active” in two ways:
- nodes can perform computations;
- users can program the network.

There are three architectures approaches:
Active Networks

“active” in two ways:
- nodes can perform computations;
- users can program the network.

There are three architectures approaches:
1. Active Nodes architecture
**Active Networks**

“active” in two ways:
- nodes can perform computations;
- users can *program* the network.

There are three architectures approaches:
1. Active Nodes architecture
2. Active Packets architecture
Active Networks

“active” in two ways:
- nodes can perform computations;
- users can *program* the network.

There are three architectures approaches:
1. Active Nodes architecture
2. Active Packets architecture
3. Active Packets and Nodes architecture
Integrating Active Networks and MPLS
Integrating Active Networks and MPLS

Where?
Integrating Active Networks and MPLS

Where? In access networks.
Integrating Active Networks and MPLS

Where? In access networks.

Why?
Integrating Active Networks and MPLS

Where? In *access networks*.

Why? **MPLS** - suitable for traffic engineering and QoS;
Integrating Active Networks and MPLS

Where? In access networks.

Why? MPLS - suitable for traffic engineering and QoS;
Active Networks - supports dynamic control and modification of network behavior.
Integrating Active Networks and MPLS

Where? In *access networks*.

Why? *MPLS* - suitable for traffic engineering and QoS; *Active Networks* - supports dynamic control and modification of network behavior.

How?
Integrating Active Networks and MPLS

Where? In access networks.

Why? MPLS - suitable for traffic engineering and QoS; Active Networks - supports dynamic control and modification of network behavior.

How?
Implementation
Implementation

A. Set up a minimal MPLS network

LSR A → LSR B → LSR C → LSR D
Implementation

A. Set up a minimal MPLS network

B. Modify the source address of packets labeled with a certain label
Implementation

A. Set up a minimal MPLS network

B. Modify the source address of packets labeled with a certain label
Linux implementation using Netfilter

*Netfilter* is a framework inside Linux kernel which enables “packet mangling”.

---

### Network Layers

- **Physical Layer**
- **Data Link Layer (Ethernet)**
- **Network Layer (IP)**
- **Transport Layer (TCP)**
- **MPLS**
**Linux implementation using Netfilter**

*Netfilter* is a framework inside Linux kernel which enables “packet mangling”.

Netfilter is a framework inside Linux kernel which enables “packet mangling”.

- **Transport Layer (TCP)**
- **Network Layer (IP)**
- **MPLS**
- **Data Link Layer (Ethernet)**
- **Physical Layer**

![Diagram of networking layers and Netfilter functions](image)
Linux implementation using Netfilter

Netfilter is a framework inside Linux kernel which enables “packet mangling”.

User application (modify the packet)

Transport Layer (TCP)

Network Layer (IP)

2
2
NF_IP_LOCAL_IN

3
NF_IP_FORWARD

5
NF_IP_LOCAL_OUT

ROUTE

MPLS

Data Link Layer (Ethernet)

Physical Layer

1

1
NF_IP_PRE_ROUTING

3

3
NF_IP_POST_ROUTING

4

4

User application
(modify the packet)

Transport Layer (TCP)

Network Layer (IP)

2
2
NF_IP_LOCAL_IN

3
NF_IP_FORWARD

5
NF_IP_LOCAL_OUT

ROUTE

MPLS

Data Link Layer (Ethernet)

Physical Layer

1

1
NF_IP_PRE_ROUTING

3

3
NF_IP_POST_ROUTING

4

4

User application
(modify the packet)

Transport Layer (TCP)

Network Layer (IP)

2
2
NF_IP_LOCAL_IN

3
NF_IP_FORWARD

5
NF_IP_LOCAL_OUT

ROUTE

MPLS

Data Link Layer (Ethernet)

Physical Layer

1

1
NF_IP_PRE_ROUTING

3

3
NF_IP_POST_ROUTING

4

4

User application
(modify the packet)
Linux implementation using Netfilter

Netfilter is a framework inside Linux kernel which enables “packet mangling”.

User application
(modify the packet)

Transport Layer (TCP)
Network Layer (IP)
MPLS
Data Link Layer (Ethernet)
Physical Layer

1. NF_IP_PRE_ROUTING
2. NF_IP_LOCAL_IN
3. NF_IP_FORWARD
4. NF_IP_POST_ROUTING
5. NF_IP_LOCAL_OUT
ROUTE

User application
(modify the packet)
Linux implementation using Netfilter

Netfilter is a framework inside Linux kernel which enables “packet mangling”.

User application (modify the packet)

Transport Layer (TCP)

Network Layer (IP)

NF_IP_LOCAL_IN

NF_IP_POST_ROUTIG

NF_IP_FORWARD

NF_IP_LOCAL_OUT

ROUTE

MPLS

Data Link Layer (Ethernet)

Physical Layer

An example

Conclusions
An example

**Bid rejection:** web servers hosting online auctions.
An example

Bid rejection: web servers hosting online auctions.
Conclusions

- Integration of two edge technologies: **MPLS** and **Active Networks**
- Example which proves that such integration is possible;
- Overcomes the MPLS limitation to perform switching above layer 2;
- Offers a flexible network which can control packets;
- We can use active code to control the MPLS traffic either within a domain as well as between different administrative domains.
Conclusions

► integration of two edge technologies: **MPLS** and **Active Networks**

► example which proves that such integration is possible;

► overcomes the MPLS limitation to perform switching above layer 2;

► offers a flexible network which can control packets;

► we can use active code to control the MPLS traffic either within a domain as well as between different administrative domains.
Conclusions

► integration of two edge technologies: MPLS and Active Networks

► example which proves that such integration is possible;

► overcomes the MPLS limitation to perform switching above layer 2;

► offers a flexible network which can control packets;

► we can use active code to control the MPLS traffic either within a domain as well as between different administrative domains.
Conclusions

- integration of two edge technologies: **MPLS** and **Active Networks**

- example which proves that such integration is possible;

- overcomes the MPLS limitation to perform switching above layer 2;

- offers a flexible network which can control packets;

- we can use active code to control the MPLS traffic either within a domain as well as between different administrative domains.
Conclusions

- integration of two edge technologies: **MPLS** and **Active Networks**

- example which proves that such integration is possible;

- overcomes the MPLS limitation to perform switching above layer 2;

- offers a flexible network which can control packets;

- we can use active code to control the MPLS traffic either within a domain as well as between different administrative domains.
Conclusions

- integration of two edge technologies: **MPLS** and **Active Networks**
- example which proves that such integration is possible;
- overcomes the MPLS limitation to perform switching above layer 2;
- offers a flexible network which can control packets;
- we can use active code to control the MPLS traffic either within a domain as well as between different administrative domains.