A Scalable, High Performance Active Network Node

Dan Decasper
dan@arl.wustl.edu

Applied Research Laboratory (ARL), Washington University, St. Louis

Computer Engineering and Network Laboratory (TIK), ETH Zurich, Switzerland
The people

- **PI:** Guru M. Parulkar (ARL)
- **Co-PI:** Bernhard Plattner (TIK)
- **Co-PI:** Jonathan S. Turner (ARL)
- **Staff:** John DeHart (ARL)
- **Grad:** Dan Decasper (ARL/TK)
  - two Grads vacant
- **Start:** July 1st 1998
Project Goal

Design and Implement a Prototype of a Scalable, Active Networking Platform supporting Traffic at Gigabit Rates
The challenge

- Active networking should allow applications to control networking nodes and how their packets are processed and forwarded.

- Requirement should not considerably degrade the performance of each network node.
The challenge (Cont.)

• Fundamental challenge:
  
  – Allow **relocating** part of the processing from the end-systems into the network
  
  – **minimize** the amount of processing on a single node
  
  – **make** the processing as efficient as possible
  
  – **keep the necessary flexibility and customizability** typical to AN
Facing the Challenge

• Building a High Performance Active Network Platform consisting of
  – **Scalable** Hardware Platform
  – Distributed Code Caching
  – **Streamlined** Software Platform

• Applications

• Conclusions and Status
ANN Hardware

Active Network Node (ANN)

WUGS ATM "Backplane"

ANPE - Active Network Processing Element
BI - Bus Interface

Load Balancing
Active Network Processing Element (ANPE)

Default: Processed by first CPU

Load balancing: Processed by second CPU

Non active: cut-through

APIC performance: 1.2 Gbit/s
Software platform

• Important observations guiding our design:

  – Potential active networking functionality is more application specific than user specific
  – Number of active networking functions grows with the number of new applications and communication standards
  – Automatic installation and upgrading of such functions is very desirable
**DAN: Function identifiers**

- Ethernet/IPv4/TCP packet
  - Functions identified by Protocol numbers/Port numbers or hardware

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ethernet</strong></td>
<td></td>
</tr>
<tr>
<td>Protocol</td>
<td></td>
</tr>
<tr>
<td>Vers</td>
<td>Version</td>
</tr>
<tr>
<td>HLen</td>
<td>Header Length</td>
</tr>
<tr>
<td>TOS</td>
<td>Type of Service</td>
</tr>
<tr>
<td>Total length</td>
<td>Total length of the packet</td>
</tr>
<tr>
<td>ID</td>
<td>Identification</td>
</tr>
<tr>
<td>Flags</td>
<td>Flags</td>
</tr>
<tr>
<td>Fragment Offset</td>
<td>Fragment Offset</td>
</tr>
<tr>
<td>TTL</td>
<td>Time to Live</td>
</tr>
<tr>
<td>Protocol</td>
<td>Protocol</td>
</tr>
<tr>
<td>Source Address</td>
<td>Source Address</td>
</tr>
<tr>
<td>Destination Address</td>
<td>Destination Address</td>
</tr>
<tr>
<td>Source Port</td>
<td>Source Port</td>
</tr>
<tr>
<td>Destination Port</td>
<td>Destination Port</td>
</tr>
<tr>
<td>Options (if any)</td>
<td>Options (if any)</td>
</tr>
<tr>
<td>Source Port</td>
<td>Source Port</td>
</tr>
<tr>
<td>Destination Port</td>
<td>Destination Port</td>
</tr>
<tr>
<td>Options (if any)</td>
<td>Options (if any)</td>
</tr>
<tr>
<td>32 bits</td>
<td></td>
</tr>
</tbody>
</table>

Active Nets Workshop 10 980716
Distributed Code Caching

• Abstract view:

    \[
    \begin{array}{cccccc}
    \text{fi}_2 & P_1 & \text{fi}_3 & P_2 & \ldots & \ldots & P_N \\
    \end{array}
    \]

• Today:
  – Function identifiers commonly identify known functions or packet is dropped by the router.

• New:
  – Let router look for the implementation of the identified function on a Code Server!
Distributed Code Caching (Cont.)
ANPE Software Architecture

- Implemented on top of NetBSD/Router Plugins
- Two types of Active Plugins:
  - Class Plugins (contain code)
  - Instance Plugins (run time configuration)
- All types of Plugins can be directly addressed by the upstream node using a Plugin Identifier (PID)
ANPE Software Architecture (Cont.)

- Plugin Management
- Plugin Database Controller
- Security Gateway
- Policy Controller
- Active Plugin Loader

- Plugin Playground
  - ANTS VM
  - ANTS Program A
  - IPv4/IPv6
  - IPv6 flow x
  - DAN
  - DAN plugin D

- Function Dispatcher
- Packet Scheduler

- Device Drivers

- Resource Controller (CPU, Memory, Bandwidth)
Applications

- **Automatic Network Protocol Deployment / Revision**
  - especially well suited for IPv6 options
- **Large-Scale reliable multicast**
  - Faster recovery through topology knowledge
  - Application-specific multicast
- **Congestion control for real-time video and audio**
- **High-performance media gateways for real-time multicast audio/video sessions**
Conclusion and Status

• Most of the ideas presented exist only on paper so far
• Solid background in building high performance, modular router platforms
• Able to leverage results from previous project to jump start this project
• Web site: http://www.arl.wustl.edu/arl/projects/ann/ann.html