A System Architecture for Processing Flows

Raj Srinivasan

Director, Software Engineering

Bivio Networks, Inc,

Email: raj@bivio.net

FLOCON, 9 October 2006

System Requirements

Due to the ever increasing number of network applications, and the proliferation of threats, requirements on flow processing systems are increasing dramatically with time. The main requirements are:

- Performance – this is currently in the 1 Gbps to 10 Gbps range, and increasing
- Scalability – the system architecture must be capable of addressing the requirements at lower and higher end applications economically, by scaling the available compute resources
- Functionality – the numbers and types of applications to be supported are increasing every year. In addition to general processing capabilities, these applications require special functions to be accelerated. Common functions are:
  - cryptography
  - compression
  - regular expression matching
- Manageability – this is a key function for almost all situations, and includes a number of functions such as the ability to configure, monitor, and maintain.
- Application maintainability and portability – this is often overlooked. The architecture should be general enough so that applications can maintain generality, and address special needs through appropriate APIs. The cost of transforming applications to suit special architectures has been proven to be very costly, again and again.

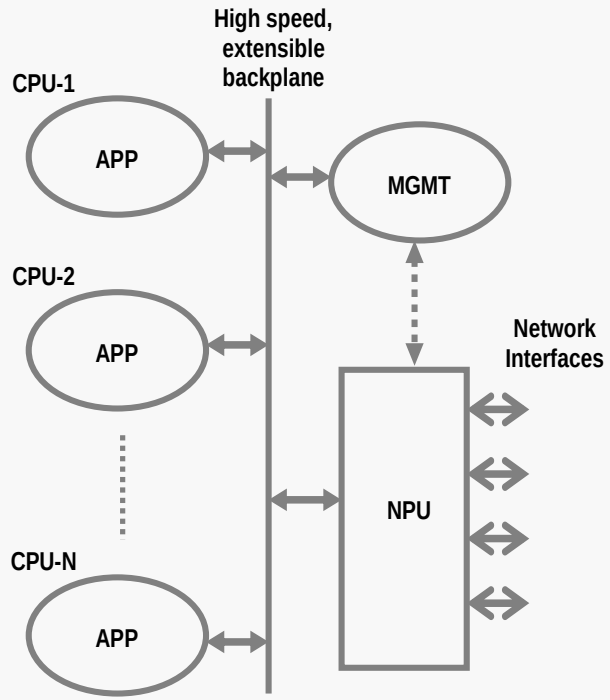
A General Architecture that meets these requirements

We propose the following clustering architecture, and show how it meets all the requirements we have listed. Such a system has been implemented for commercial use.

This architecture has the following highlights:

- General purpose CPUs with facilities (over PCI) for specialized coprocessors
- A Network Processing Unit (NPU) for managing flow distribution
- An extensible, high-speed backplane for interconnecting CPUs and NPU
- A designated CPU for system management
- Facilities to attach multiple types of network interface cards to NPU

Architecture for Flow Processing



The Application Environment - Software

Software Environment

- All application CPUs run the Linux operating system. They are loosely coupled, and run identical versions of the kernel and system applications.
- In the current implementation, each CPU is a PPC 7447A processor. The architecture does not depend on the processor type, though. In the next generation, each CPU is a multi-core unit that can be configured to run as single cores or in SMP mode.
- There is no shared memory between the CPUs. All communication is over the system backplane.
- Each CPU has attached to it a PCI bus. This enables to attach coprocessor/accelerator units to each CPU.
- The management CPU has storage attached to it, that is accessible to all the application processor CPUs.
- All of the system management and configuration functions – the databases, CLI, GUI – run on MGMT.

The Application Environment – Networking, File System

Network Environment is identical in every CPU

- The backplane forms a private network which is accessed through a special Ethernet interface.
- Each external Ethernet interface is virtually extended into each CPU, so the external networks are accessed as though they were directly attached to each CPU. The NPU (Agere APP 540) is transparent.
- Applications (running in CPUs and MGMT) communicate with each other using standard internet protocols over standard socket mechanisms.
- Routing protocols run on MGMT, but forwarding tables are propagated to application CPUs.
- Interfaces are configured on MGMT, and interfaces states are propagated to application CPUs.

File system environment is identical in every CPU

- All CPUs boot over the backplane using standard protocols with MGMT serving as the boot server.
- File system is mounted from MGMT
- Each CPU has local directories so applications can have individual logs or temporary storage.

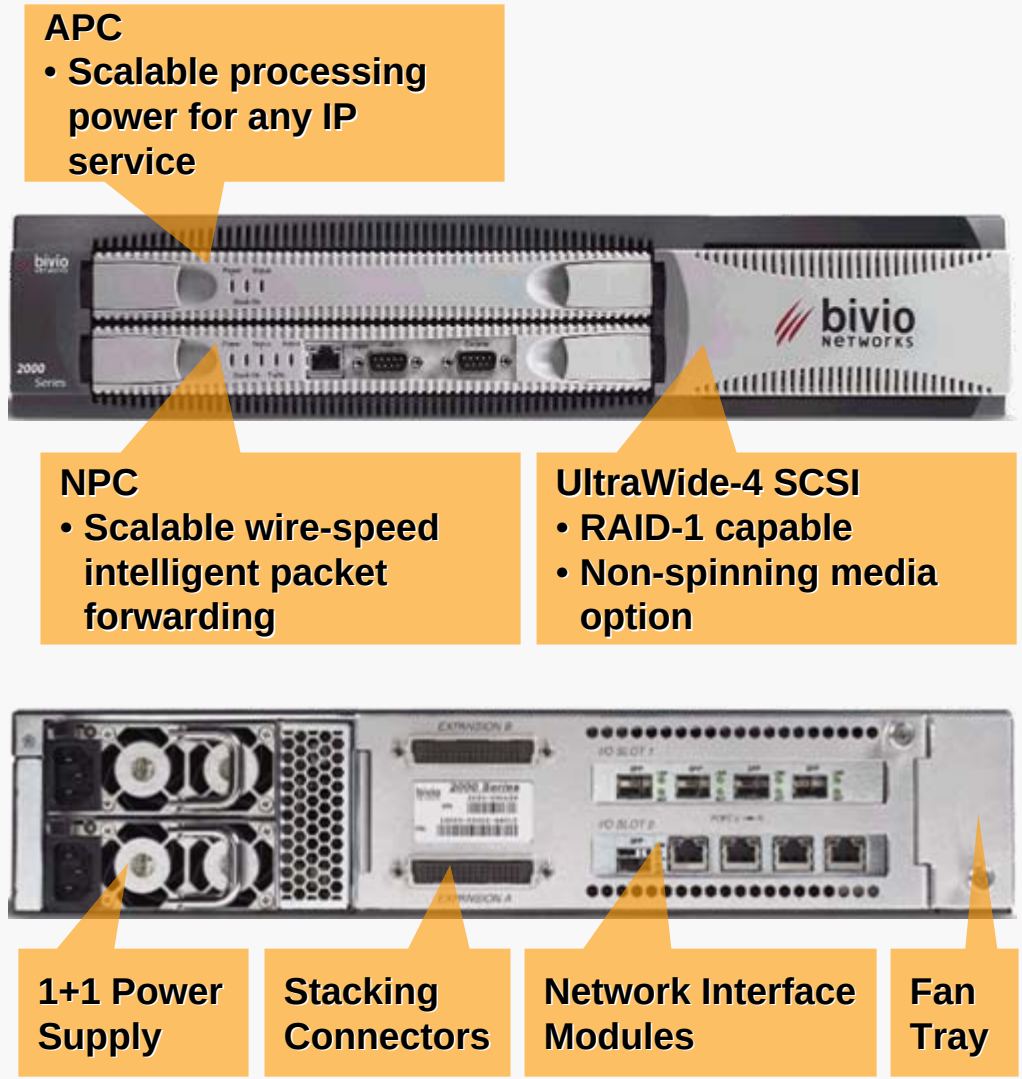
NPU Functions

The NPU (Agere APP 540) performs the following functions:

- Load-share flows to application CPUs based on multiple (configurable) algorithms
- Support special APIs for
  - Cutting through flows that are not interesting to applications
  - Bind specified flows to specific CPUs (application instances)
- Perform QoS functions
- Maintain and report flow statistics as required by applications
- Partition traffic between different applications using the Configurable Inspection Groups (CIG) feature
- (Future) Provide hooks for third parties to insert special inspection routines to maintain application specific state. This is not possible in the current generation, but will be an integral part of the next generation system.

Bivio Architecture Overview

- System architecture optimized for Packet Handling
- Multi-dimensional scaling
- Key hardware elements:
  - Powerful computation platform
  - Hardware acceleration
- Network computing platform
  - Integrated management
  - Extensive QoS features
  - Built-in high availability options
  - Architectural survivability
- Open development environment
  - Linux execution environment
  - Full featured SDK

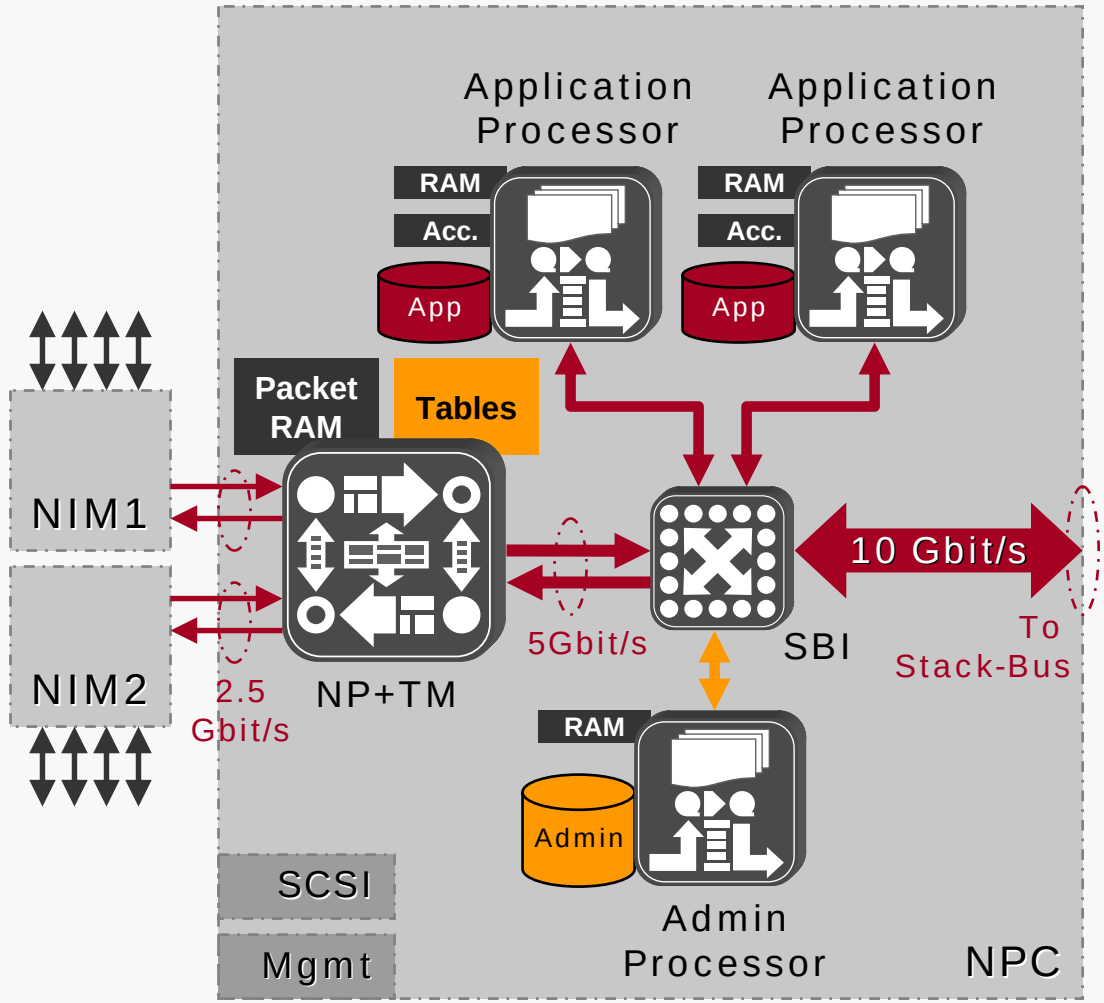


NPC: Network Processor Card

- 3 Gbps aggregate throughput
- Wire speed intelligent forwarding operations
  - Packet inspection & classification
  - Modification & transformation (NAT)
  - Traffic Management
    - Diffserv, Intserv ..
- CPU load sharing control
- Internal Dual SCSI HD and RAID support
- Chassis-level management
- DB-9 and 10/100 RJ-45 management ports

NPC Functional Overview

- PowerPC-based Application and Admin Processors
  - ~6,000 MIPS on local APs for application processing
- Network Processor and Traffic Management subsystem
  - 5Gbit/s full duplex performance
  - QoS processing

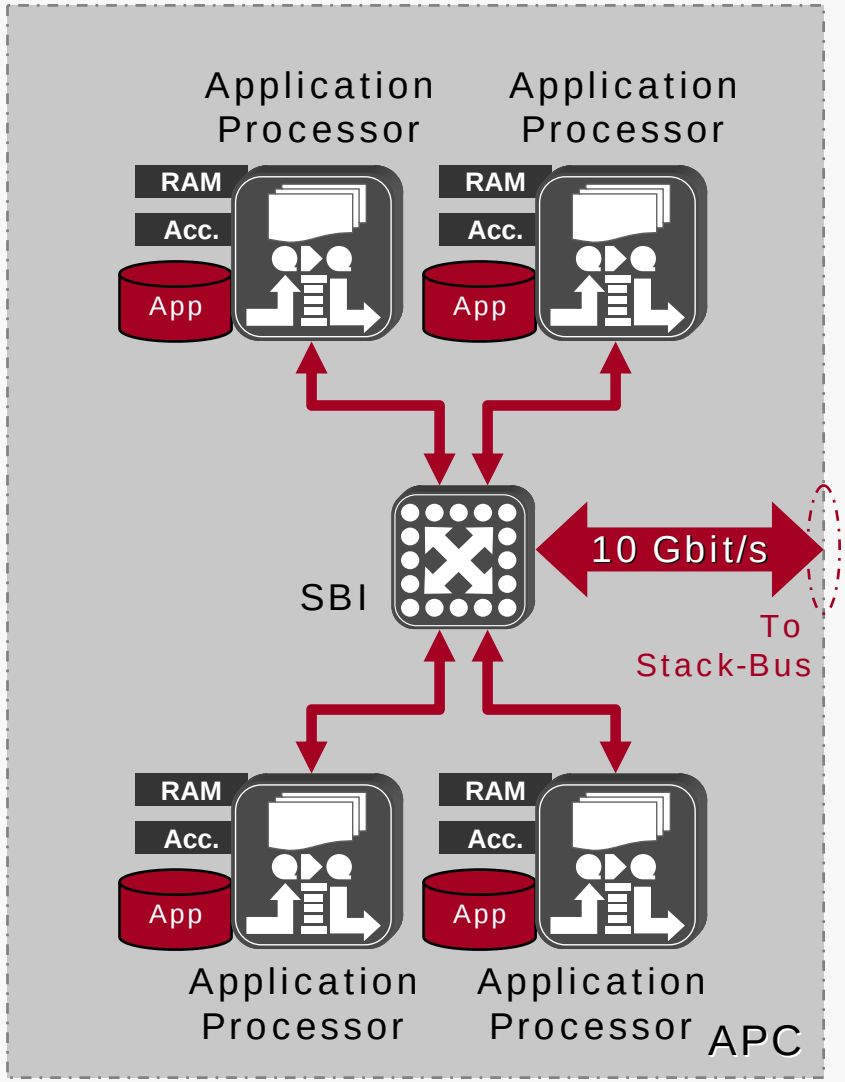


APC: Application Processor Card

- Fully redundant computing architecture
- Parallel Motorola PowerPC subsystems
- Modular hardware acceleration per CPU
- Standard Linux execution environment
- Flows distributed across CPU cluster by NPC

APC Functional Overview

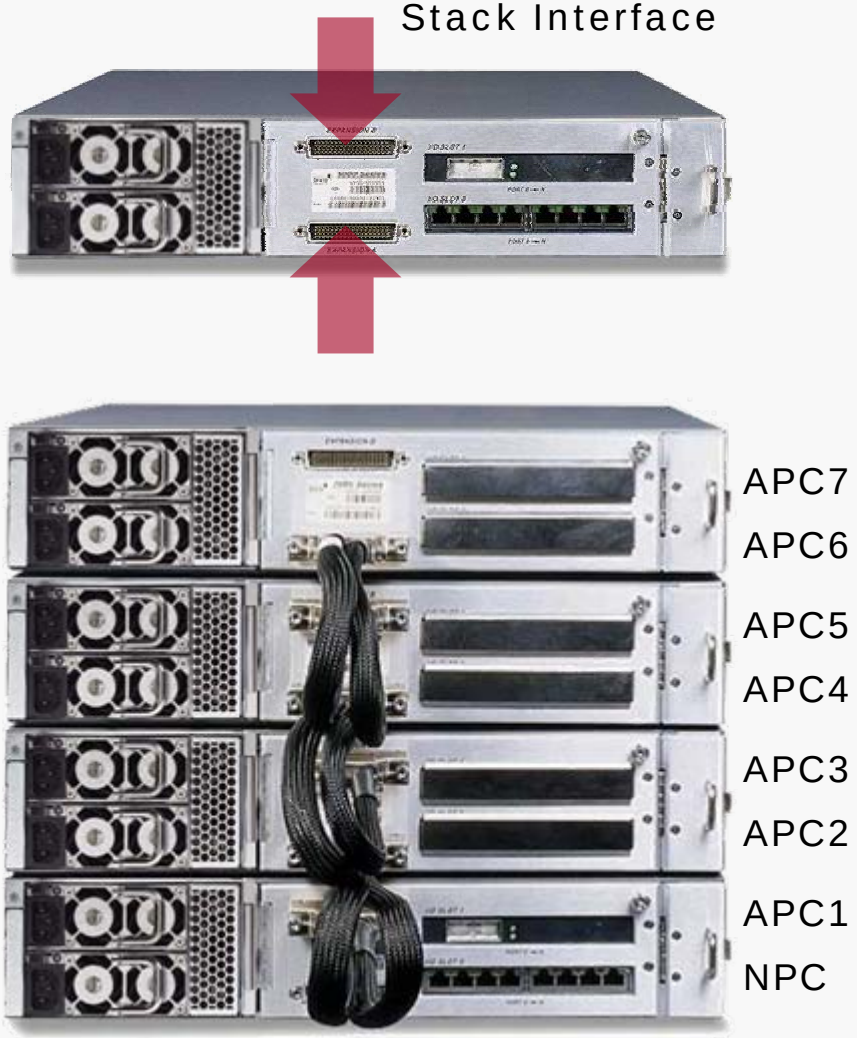
- PowerPC-based Application
- Any number of APCs can be stacked



Scaling

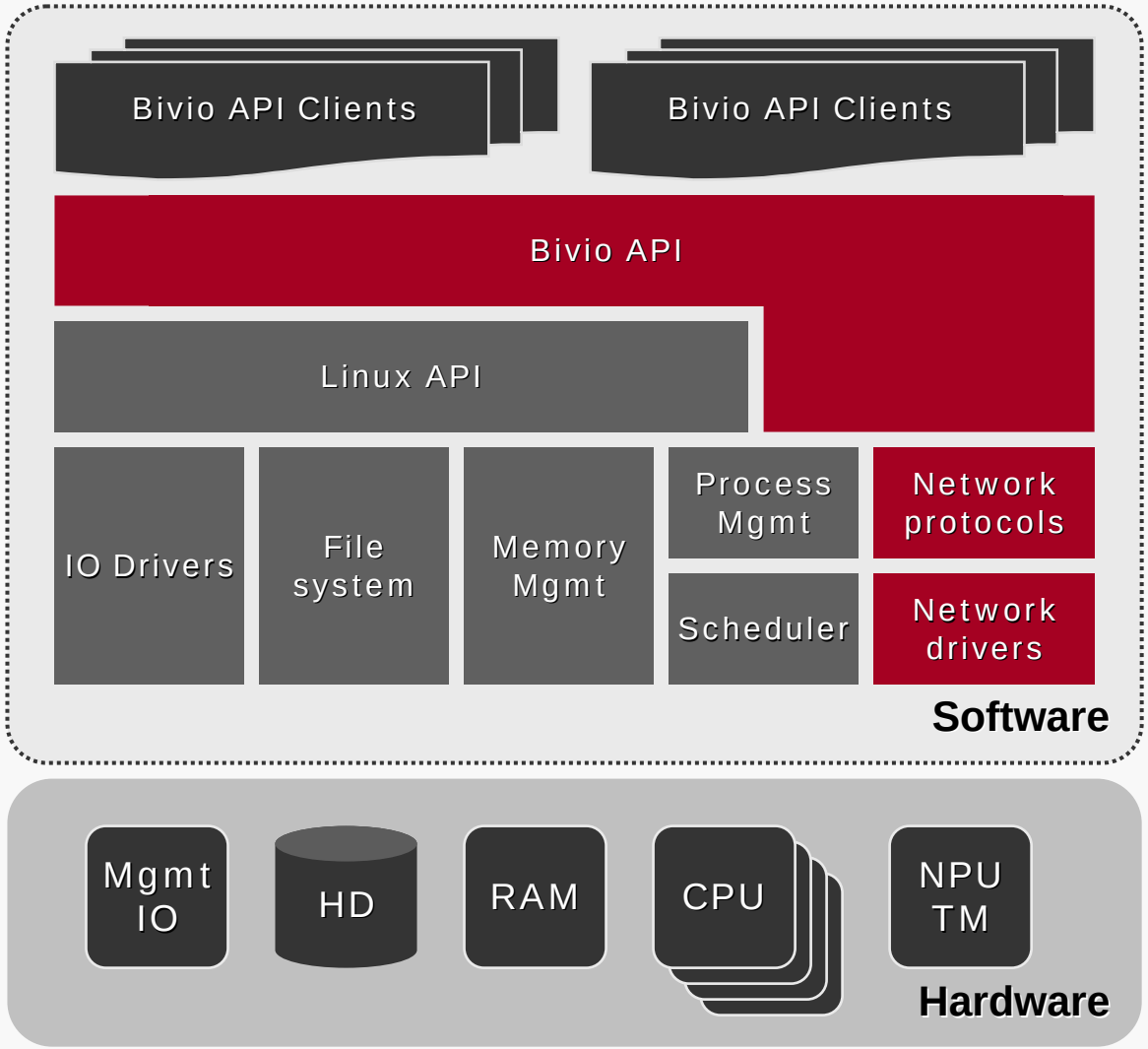
- 10 Gbps full duplex stack interface
- Multi-dimensional scaling
  - Additional APCs scale processing
  - Additional NPCs scale throughput
- Uniquely suited to demands of current and future Packet Handling applications

US Patent Application #10/078,324
"Systems and methods for fair arbitration between multiple request signals"



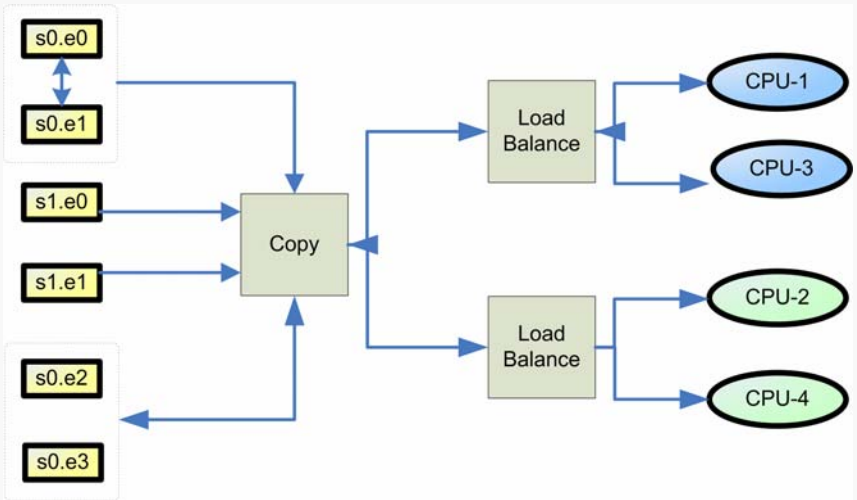
Open Software Architecture

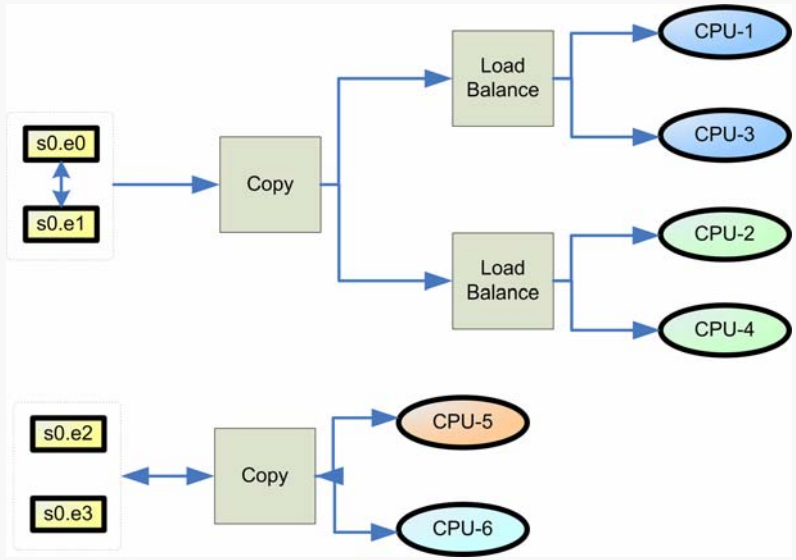
- Standard Linux execution environment, enhanced with:
  - Performance booster features such as **ZeroCopy** drivers
  - Congestion control mechanisms
  - Configurable Inspection Group feature
- SDK enables applications to leverage platform features
  - FW/VPN
  - IPS/IDS
  - Load balancing
  - Managed services
  - UTM
  - Any proxy applications
- Application porting support



Configurable Inspection Groups

- Advanced Modes
  - Multiple Inspection Groups
  - Copy mode
  - Tap mode
  - Load balancing





Performance & Scalability

Until recently, Moore's law ensured that performance goals could always be met using more and more powerful processing units. But this has changed recently, as evidenced by the move by major processor vendors to multi-core chips. However, this too has some problems:

- SMP units are bound by memory bandwidth. Even with two memory controllers, the classic architecture of a powerful dual-core SMP CPU operating on very fast memory (600+ Mhz DDR2 memory) will be hard pressed to do a 1-2 Gbps of flow processing. There is no way to get to 10 Gpbs with such an architecture, even with multiple cores, operating on a single memory subsystem.
- ASICs are limited by the same issues. Ultimately, an ASIC incorporates some sort of a special purpose CPU operating on special memory. For extremely special purposes, high performance can be obtained, at the cost of flexibility and maintainability of software. Even these have to be clustered for 10 Gbps and beyond.
- When multiple applications need to be working in a cooperative environment, such as in a Unified Threat Management System (UTM), each with differing requirements on processing and memory resources, a cleaner separation than afforded by an SMP is sometimes preferable. Virtual architectures provide the separation, but are often performance limited due to the overheads.

One of the great advantages of the Clustering architecture we have presented is the inherent linear scalability. There is also a lot of built-in redundancy, since CPUs can be used interchangeably.

Scalability - continued

The NPU makes a big contribution to system scalability, if used appropriately.

NPUs are typically good at traffic processing, and a certain level of inspection. In our architecture, we do not depend on NPUs to perform special functions such as RegEx or Cryptographic processing. These are performed on CPUs, and so can be scaled.

Within NPUs, we do the following:

- Intelligent load sharing, to ensure that both directions of a flow are processed in the same CPU. This cuts out a lot of synchronization traffic between CPUs.
- (Future) Deep packet inspection to ensure that related flows will stay on the same CPUs. A simple example is the SIP signaling and media sessions, or the FTP control and data sessions. This also reduces synchronization traffic drastically.
- Configure special computing groups for special traffic classes. This feature organizes the available computing resources into groups that process special (specified) traffic classes.
- Recognize failed CPUs and redistribute traffic to other compute resources.

Doing these, we have found that performance and functionality can be scaled LINEARLY by adding more CPUs and NPUs. We believe that the biggest cause of less than linear scaling is synchronization traffic, and reducing this is one the NPUs very important contributions to this architecture.

System Manageability

This is such an important feature that we have dedicated a CPU for management. This has the following benefits:

- The system is always accessible, configurable, and manageable under the heaviest load conditions. The importance of this property cannot be overemphasized since there are many situations that need to be addressed immediately, and delays cannot be tolerated.
- The application CPUs are spared some of the more expensive management functions involving log processing, configuration updates, etc.
- MGMT in our architecture does some special functions such as system health monitoring, and upon detecting critical failures, can trigger failovers with very small latency.
- Many system maintenance functions can be done on the fly.

Summary & Ongoing Work

In summary, we have presented a simple architecture that has the following features:

- Scalable for CPU performance
- Scalable for specialized coprocessor functions
- Scalable NPU capabilities
- Linearly scalable by choosing the appropriate mode of operation for NPU
- Standard application environment that promotes ease of maintenance and application portability
- Manageability

In addition, this architecture has been implemented for commercial applications, in a highly integrated 2U chassis.

The next version of this product is under development, and will incorporate more powerful processors, utilizing the best of SMP and Clustering features in CPUs, faster backplane, and incorporate an NPU that can support 10 Gbps operation.