PRODUCT GUIDE

Linux networking with SAND driver components

By J.C. Utter

ImageStream Internet Solutions has been engineering and marketing Linux networking products since 1995. ImageStream was the first company to ship commercial Linux-based routers, and continues to be a leading OEM supplier of Linux networking solutions. In this article, ImageStream President and CEO J.C. Utter shares some of the benefits of using ImageStream's core technology, the SAND driver component architecture for Linux.

Modern UNIX-style operating systems provide a kernel capable of supporting advanced IP routing applications. Linux represents one of the most capable operating systems available to developers today. But in 1998 when ImageStream began to develop its next-generation networking architecture for Linux, none of the available platforms provided an extensible driver component architecture that addressed the needs of the networking OEM.

After ImageStream developed some of the industry's first Linux routers in 1996, the company committed to support the unique requirements of networking and telephony OEMs. At that time, Linux already supported loadable kernel modules. Loadable kernel modules eliminated the need to compile drivers directly into the Linux kernel, but it did nothing to address the need to support shared protocol components, modular driver customizations, and driver components with diverse licensing.

The SAND architecture

To address these needs, ImageStream developed the Standard Architecture for Network Drivers (SAND). SAND supports the ability to chain binary driver components together dynamically. SAND also defines a standard linkage mechanism and five stages where these components can be installed to provide cooperative data stream processing. The five SAND stream processing and control stages are listed below:

- **Physical Interface Control (PIC):** Configures line interface hardware such as CSU/DSUs and Ethernet transceivers.
- **Hardware Interface Control (HIC):** Hardware-specific driver component for network data input and output.
- **Encoded Data Processor (EDP):** Provides developer access to data link encapsulation.
- **Data Link Protocol (DLP):** Supports plug-in LAN and WAN protocol stacks.
- **Decoded Data Processor (DDP):** Provides developer access to network encapsulation.

The Physical Interface Control (PIC) is the only SAND component that does not process streaming data. PIC components control physical layer setup options, so the PIC is only executed when a network interface is configured.

All of the other pre-defined stages in SAND are used to process streaming network data. The Hardware Interface Control (HIC) is the hardware-specific driver component that handles interface initialization and shutdown, as well as hardware-specific data I/O. Data Link Protocol (DLP) components process Layer 2 link encapsulation. The DLP stage is typically used to implement standards-based LAN and WAN protocol encapsulation and decapsulation.

The Encoded Data Processor (EDP) and the Decoded Data Processor (DDP) represent two pre-defined stages where custom data stream processing components can be installed. The EDP is typically used to manipulate link layer encapsulation before the inbound data stream reaches the standard DLP module and before the outbound data stream from the DLP is transmitted to the physical layer. The DDP is used to manipulate network encapsulation before the inbound stream is passed to the kernel for routing and before the outbound data stream is processed by the DLP module.

Each SAND component is a software object that encapsulates standard SAND data structures and functions, as well as the unique binary executable. The SAND system uses these driver components to dynamically construct a binary chain in host memory. Each of these chains represents the code that will be executed for each network interface.

When inbound data arrives at a network interface, the interrupt service routine or polling thread copies the inbound data into a standard structure in host memory, and then wakes up the SAND receive threads to have them move the inbound data through the inbound processing chain for the specific interface. On the outbound side, SAND uses the Linux kernel's queue processing threads to execute the outbound chain. Driver-originated transmissions such as keep-alives are handled by special kernel threads that service the timer device. Timer-driven transmission handlers must construct their data messages using standard SAND data structures, and then drop the data into the appropriate stage of the outbound chain.

The benefits of using SAND

SAND offers many benefits to developers and end users. Some of these benefits include shorter development cycles, enhanced performance, improved scalability, increased memory efficiency, the ability to mix commercial and Open Source driver elements, and the ability to independently update driver components that originate from different sources.

One significant benefit of using SAND is the availability of off-the-shelf software components. The ability to plug in standards-based protocol components like PPP, HDLC, frame relay, ATM, ASCII, and others may provide significant cost and time-to-market savings when compared to developing such protocols in-house or licensing the protocols from a third party vendor.

SAND can also reduce time-to-market by providing a high-level of isolation between different software components. With monolithic source code, OEM developers typically open the source files provided by the hardware vendor and then begin making changes. When the time comes to debug an application, the developer may end up chasing bugs in the original source, or searching for an interaction between the newly developed software and the vendor-supplied code.

SAND enables the OEM developer to design components that work seamlessly with vendor-supplied components. SAND can eliminate the need to modify vendor-supplied code, so it may also save time by eliminating the need to become familiar with vendor-supplied sources. By using isolated SAND components, the OEM can independently upgrade custom software and vendor-supplied software without any need to modify the vendor-supplied code.

In most networking applications, SAND provides significant improvements in memory efficiency and performance. SAND components are software objects, and like all objects they must introduce memory and processor overhead. Fortunately, the overhead introduced by SAND is minimal, and most developers inherit enough improvements in performance and efficiency that the improvements exceed this overhead. In complex networking systems that support multiple network cards and protocol stacks, SAND typically improves memory efficiency by providing an optimized architecture for sharing protocol components, and by supporting the ability to unload components when they are not in use.

SAND can also enhance system performance in several ways. For example, the architecture is designed to take full advantage of multiple processors. Most developers would find it difficult (if not impossible) to write multi-threaded drivers in kernel space. Next, SAND provides architectural support for the ability to implement hardware acceleration at any point along a stream processing chain. Significant performance enhancements in a system may be realized by using SAND to redirect packet flows to and from dedicated co-processors or DSP units. SAND also offsets some of its own performance overhead by eliminating the need to test for protocol selection. When the appropriate protocol processing code is “wired” directly to each interface, there is no need to test variables that hold information on protocol selection.

Another key benefit of using the SAND architecture is its scalability. SAND can be scaled by adding plug-in software to support new protocols and distributed processing schemes. SAND was also designed to support a theoretically unlimited number of network interfaces, which may be a significant limitation for some platforms. Interface scalability was one of SAND’s primary design objectives because modern WAN protocols commonly support virtual circuits that can represent thousands of routed interfaces.

Finally, SAND gives OEMs the ability to deploy interdependent commercial and Open Source software under Linux. Some developers have chosen not to use Linux because they believe that Linux precludes their ability to distribute and protect commercially licensed software. From this perspective, the challenge to develop a monolithic driver that uses both Open Source and commercial source can be insurmountable. The SAND architecture addresses this issue by isolating Open Source and commercially licensed software into separate binary components. In this way, it is possible to use commercially licensed software that limits source code distribution, and still comply with Open Source requirements to redistribute source code.

Shortened development cycles

ImageStream developed SAND to enhance the company’s ability to compete with Cisco Systems in the midrange router market. As ImageStream has grown, the company has continued to focus on delivering the most powerful Linux development platform available to router and telephony OEMs. Today, the SAND driver component architecture is helping developers to shorten development cycles, enhance system performance, improve application scalability, increase memory efficiency, and streamline the use of commercial and Open Source code in the same system.

The OEM markets for router and telephony components continue to represent strong growth opportunities for ImageStream. In this context, SAND has been proven in OEM and military applications with customers like Ericsson, Lucent Technologies, Northrop-Grumman, Raytheon, NASA, and Network Instruments. While some OEMs will choose to develop their own networking platform from the ground up, many others have chosen to start with a proven low-cost platform like SAND to accelerate their development.

***J.C. Utter** is a founder and President of ImageStream Internet Solutions. Utter has eleven years experience in the electronics industry directing marketing and publishing technical documents. He is an experienced C programmer, and provides direction to the SAND development team as a system architect.*

For more information, contact J. C. at:
J.C. Utter
ImageStream Internet Solutions
7900 E. 8th Road, Plymouth, IN 46563
Tel: 574-935-8484 • Fax: 574-935.8488
E-mail: jc@imagestream.com
Web site: www.imagestream.com

PRODUCT GUIDE: LAN, WAN, MAN, AND SAN

Company	Web site	Model	LAN	WAN	MAN	SAN	FIBRE CHANNEL	SOFTWARE
Anatel Communications	www.antel.com	NAC-120 and NAC-240	•					
Anritsu	www.us.anritsu.com	Multiflow 5000			•			
Apcon, Inc.	www.apcon.com	ACI-2046					•	
Architek srl	www.archi-tek.com	Felis Server						•
Aurora Technologies	www.auroratech.com	Linux Drivers						•
		Vanguard 4520CP cPCI Serial Controller		•				
Belobox Networks, Inc.	www.belobox.com	AmpNet				•		
Brooktrout Technology	www.brooktrout.com	NetAccess		•				
Communications Computer Intelligence Integration	www.ccii.co.za	CDDI	•					
		FDDI	•					
		PMC CDDI	•					
		PMC FDDI	•					
Continuous Computing Corporation	www.ccpu.com	Hi-File				•		
Dot Hill	www.dothill.com	SANnet Axis Basic				•		
		SANnet Axis HA				•		
		SANnet Axis Remote				•		
Eurotech	www.eurotech.com	CPU-7630	•	•	•	•		
Gadzoox Networks	www.gadzoox.com	Slingshot 4218					•	
Galileo Technology	www.galileot.com	GT-96122		•				
		GT-96132		•				
Green Hills Software, Inc.	www.ghs.com	INTEGRITY 4.0 for Intel XScale						•
ImageStream Internet Solutions, Inc.	www.imagestream-is.com	WAN cards		•				
Interface Concept	www.interfaceconcept.com	IC-2E1-PMCa		•				
Interphase Corporation	www.iphase.com	4535		•				
		ENTIA 5536	•	•				
		iNAV 4000		•				
		iSAN 4539		•				
		iSAPN 5575	•	•				
		iSPAN 1635		•				
		iSPAN 4531S		•				
		iSPAN 4532		•				
		iSPAN 4534	•	•				
		iSPAN 4575	•	•				
		iSPAN 4576	•	•				
		iSPAN 5535	•	•				
		iSPAN 5539	•	•				
		iSPAN 6275	•	•				
		iSPAN 6535		•				
		iSPAN 6575	•	•				
		PowerSAN 4526				•		
		PowerSAN 5541				•		
		PowerSAN 5550				•		
		PowerSAN 5560				•		
		PowerSAN 6020				•		
		SlotOptimizer 552C	•			•		
		SlotOptimizer 553C	•			•		

Company	Web site	Model	LAN	WAN	MAN	SAN	FIBRE CHANNEL	SOFTWARE
Interphase Corporation cont.	www.iphase.com	SlotOptimizer 554E	•			•		
		SlotOptimizer 5570	•			•		
Lineo	www.lineo.com	SecureEdge				•		
Men Mikro	www.men.de	P9	•					
		P16	•					
Montivista	www.mvista.com	Linux Carrier Grade Edition						•
Nexsan Technologies	www.nexsan.com	InfiniSAN D2D				•		
Odin TeleSystems Inc.	www.odints.com	Arni-32	•					
		Balder-8S-PCI	•					
		Balder-8U-PCI	•					
		Vidar-5x16	•					
		Vidar-6x3	•					
Osicom Technologies, Inc.	www.osicom.com	2201C3 Series	•					
Performance Technologies, Inc.	www.pt.com	ChanneLink		•				
		CPC334		•				
		CPC340H		•				
		CPC344/348		•				
		CPC370PQ		•				
		CPC376		•				
		CPC396	•					
		Ipnexus		•				
		MPS800	•	•				
		Nexus Ware						•
		PCI344P		•				
		PCI370PQ		•				
		PCI37xAP		•				
		PMC334		•				
		PT-CPC388		•				
		Signaling Blade	•					
		WAN Protocol Software						•
QLogic Corp.	www.qlogic.com	SANbox2					•	
RAMiX Inc.	www.ramix.com	PMC941	•					
SBE, Inc.	www.sbei.com	LMC1000M		•				
		LMC5200C		•				
		LMC5200M		•				
SBS Technologies, Inc.	www.sbs-cp.com	ARIES 720/750		•				
		ARIES 800		•				
		MAXIM 520		•				
Siemens	www.smi.siemens.com	CPCI-COM294	•	•				
Software Group Limited	www.group.com	SG4C		•				
Systran Corporation	www.systran.com	RAMplex RX78	•					
Trebia Networks Inc.	www.trebia.com	SNP Architecture				•		
Tricom Technology	www.tricomtech.com	PX010	•					
		PX100 TX/FX	•					
Unison Information Systems	www.unisoninfo.com	RAID I/O FIBRE II					•	
VISTA Controls Corporation	www.vistacontrols.com	PFD/TP-PMC	•					