

Addressing network surveillance challenges through virtualization technology

The ever-increasing importance of the Internet in our daily lives continues to dramatically influence the way we all live, play, learn, and work. Network protection has advanced to the point where *perimeter* devices such as firewalls and Virtual Private Networks (VPNs) can help prevent unwanted intrusion into business local area networks or computers. But, how do we go about addressing security threats that reside inside the network? Examples of these threats include network abuse/misuse, cyber crime, identity theft, and terrorist activity. Deep packet inspection is a key ingredient for solutions to identifying these kinds of threats within the network. In this column, we will look at how IP Fabrics, Inc. uses their *virtualization technology* to fight these security threats *inside* the network. Thanks go to Kevin Graves, chief technology officer at IP Fabrics, Inc., for providing information and insight.

How does network surveillance work?

At the most basic level, computers identify themselves over the Internet using an IP address. Therefore, at first glance, it seems simple enough to create a system that sifts through packets on a network tap looking for the IP addresses of interest. Unfortunately, IP addresses are almost always allocated dynamically. The use of Dynamic Host Configuration Protocol (DHCP) is the widely accepted approach to assigning IP addresses to hosts. But DHCP is not the only protocol. VoIP uses H.323 standards to make and break connections. Session Initiation Protocol (SIP) is another *session setup/transfer/follow me* protocol. So, how can the surveillance occur if the addressing and connection information is uncertain? Answering this question leads to a surveillance system comprising two components. The first is called the User Identification (UI) component. The UI component is responsible for determining the addressing and connection activity for a given host name of interest. While the specific address identifier in this example is an IP address, it could be a number of things: A user name,

host name, or call identifier for VoIP. The UI component of the surveillance system must be flexible enough to identify users within multiple types of Internet-related protocols.

Once the UI component has identified the address and connection information of the party under surveillance, the second part of the surveillance system kicks in. We will refer to this component of the system as the Packet Traffic (PT) analyzer. The PT analyzer leverages UI-provided addressing and connection information to filter out everything except traffic addressed to or from the person under surveillance.

The UI component tends to be very protocol aware and must support a variety of session establishment protocols, such as dynamic host configuration protocol as well as SIP, FTP, and H.323. Many session establishment protocols are stateful. The user ID component must follow the state machine of these protocols to determine the address and connectivity of the user in question.

The PT analyzer monitors the actual data flows of specific connections, detects malicious or illegal activity, keeps statistics, and takes action when items being looked for in the content are found. It is very important that the PT analyzer be able to maintain deep packet inspection anywhere from hundreds of MBps up to a Gbps on links within the network.

Surveillance equipment approaches

In order to provide line-rate, deep content inspection capability needed for packet traffic analysis, surveillance equipment may incorporate:

- Parallel processing of the packets through multicore CPUs or network processors
- Fixed-function acceleration, for example encrypt/decrypt units, hash units, and Ternary Content Addressable Memories (TCAMs)

- Various memory types to accelerate processing either for lookup and/or content inspection
- Various buses and interconnects to other acceleration silicon
- Interaction with other system elements at the control or management plane

Often the design combines multiple items from this set of approaches to balance cost, time to market, and flexibility. One thing these methods have in common is increased software development complexity on the new-age boards needed to meet wire-speed deep packet processing requirements.

Network processors have an inherent affinity for this kind of application, because they:

- Are optimized for packet processing
- Incorporate a high degree of parallelism
- Provide bus interfaces and interconnects to various memory and silicon types

What's more, the operation of the parallel compute engines is programmed into the part using an instruction set optimized for the network processor, enabling network processors to lend themselves to maximum flexibility.

While network processor components offer many benefits to network equipment developers, they come at a potential cost. Programming the parallel compute engines and software integration and test of the parallel environment can introduce a learning curve and lengthen the development cycle, not only during design and implementation but during the test and validation phase as well.

Kevin points out another drawback: The lengthened software development cycle can mean a functional prototype is unavailable until late in the project cycle, which delays integration and overall

system functional and performance modeling. In conjunction with the complexity, developers may hesitate to modify or enhance working designs due to the risk of performance or functional timing changes causing problems within the highly parallel environment.

IP Fabrics uses virtualization technology to manage this complexity. Specifically, programmers implement runtime software and tools that abstract the underlying Network Processing Unit (NPU) hardware and parallel processing environment using a Virtual Machine (VM). It is a very interesting and compelling notion. Java has been an enabler in the proliferation of Internet-connected software. Why not apply the same principles to network surveillance systems? Implementing a VM on top of a network processor provides the programmer an architecture-independent environment that provides portability and scalability. Further, since the VM is already implemented and tested on the network processor, functional prototypes can be ready for initial integration within hours, which greatly lowers development risk and shortens integration and test time.

In Figure 1, we can see how the virtualization concept works:

1. The developer writes a program using packet-processing policies in a high-level language.
2. The compiler builds the VM *byte code* from the program source that the VM will use to run packet-processing instructions on all the compute engines of the network processor.

This approach eliminates the timing and logic complexities related to parallel programming environments by incorporating these notions directly into the VM.

Using virtualization technology for network surveillance

Now that we have looked at surveillance system requirements and virtualization technology, let's put it all together.

Turning our attention to the packet traffic analyzer, it must:

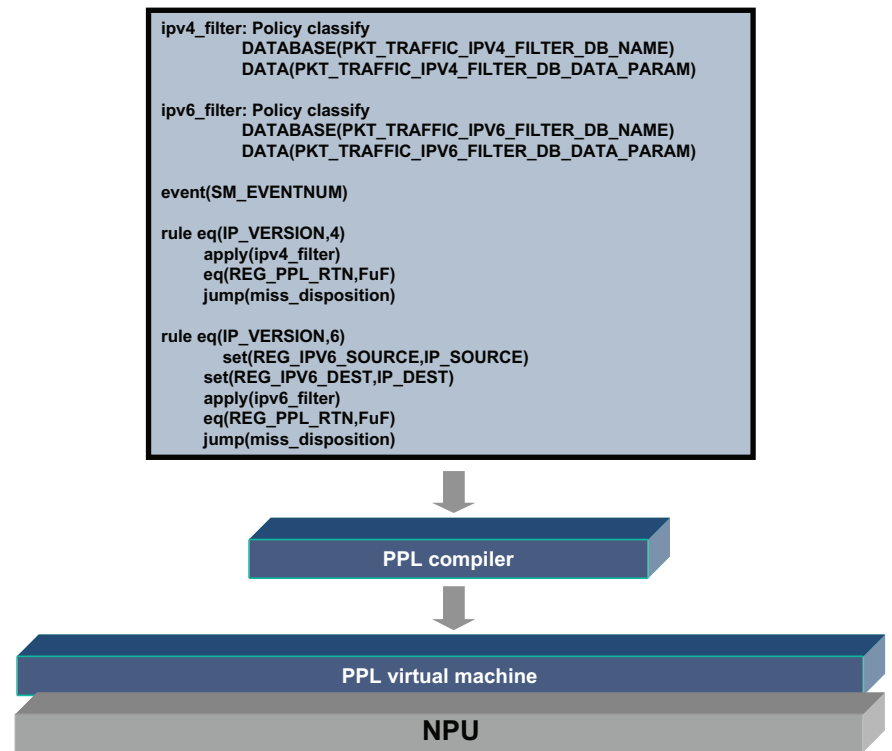


Figure 1

- Monitor specified packet flows
- Perform flow lookups to get state information
- Perform packet classification
- Perform payload scanning

Generally, the programmer of the PT analyzer must program the processor to first classify the packet to determine if it is a packet of interest. If it is, perform a flow/session lookup to determine if the packet is part of a session already being monitored. If not part of an existing session, the packet traffic analyzer software must allocate and initialize a session entry for this packet. Lookups are often performed on the IP packet five-tuple (IP source and destination address, protocol, and Layer 4 source and destination ports). Sometimes lookups are performed on a Virtual LAN (VLAN) tag. Since session tracking may include up to one million independent flows, the session table typically requires some kind of hash table, further complicating the packet analyzer design and implementation.

Expressing this logic using assembly language code takes thousands of lines of code involving the packet logic noted earlier as well as code to deal with the parallel processing environment (mutual

exclusion and semaphores to prevent table entry allocation duplication, and so on).

Implementing the logic in C is also a challenge. C does not include constructs for parallel computing, so the developer must incorporate these things in the design and implementation. Further, the lack of a *parallel engine operating system* requires the C programmer to interact directly with the hardware. Efficient payload scanning and pattern matching can also be extremely complex. Programming effective algorithms involves researching and coding state-of-the-art algorithms in order to take advantage of every cycle in the hardware.

The IP Fabrics virtualization approach uses a high-level programming model that provides a set of built-in algorithms (which they call policies in the language) optimized for the hardware it runs on and incorporating the latest optimized algorithms that can be simply and directly invoked by the program running on the virtual machine. Algorithms built into the virtual machine include connection lookup, header encapsulation or de-encapsulation, rate monitoring, and content inspection using strings, regular expressions, or a patterns database, to name a few. IP Fabrics

DeepSweep™ product uses virtualization technology for network surveillance at gigabit data rates (Figure 2).

When asked about overhead of the virtual machine, Kevin mentions that the virtual machine uses highly optimized, best-of-breed algorithms in addition to the VM architecture using both pipelined and parallel processing across the compute engines. In turn, these hand-tuned aspects of the virtual machine will often outperform those written by general purpose network processor software developers. The more complex the processing becomes, the higher performance the virtual machine achieves when compared with complex network processor software implementations.

When asked about future products from IP Fabrics, Kevin alluded to plans for a series of network surveillance products using IP Fabrics' virtualization technology. Like network processors, the new Intel and AMD multicore processors could also be targets for IP Fabrics' virtualization technology.



Figure 2

Conclusion

As network surveillance requirements continue to drive more surveillance systems deeper into the network, it will be interesting to see how virtualization technologies can revolutionize line-rate, complex packet processing for these applications. Virtualization technology

aims to remove complexities in the development cycle, shorten time to market, and enable system engineers to bring better, faster, more flexible products to market sooner.

For more information, contact Curt at cschwaderer@opensystems-publishing.com.