# An Overview of Security Issues in Cluster Interconnects

Manhee Lee  Eun Jung Kim  Ki Hwan Yum[†]  Mazin Yousif[⋆]

Department of Computer Science
Texas A&M University
College Station, TX 77843
{manhee, ejkim}@cs.tamu.edu

[†] Department of Computer Science
University of Texas at San Antonio
San Antonio, TX 78249
yum@cs.utsa.edu

[⋆] Corporate Technology Group
Intel Corporation
Hillsboro, OR 97124
mazin.s.yousif@intel.com

**Abstract**

Widespread use of cluster systems in diverse set of applications has spurred significant interest in providing high performance cluster interconnects. A major inefficiency in utilizing such interconnects has been the send/receive communication overheads at the sender/receiver hosts. Various techniques such as User-Level Communication (ULC) have been proposed to mitigate this communication inefficiency. However, due to recent security breaches, focus on cluster communication security research has spurred. Such research is non-trivial due to the high-speed nature of the cluster interconnect.

This paper explores possible schemes to ensure secure cluster intra-communication encompassing the host processor, secure coprocessor and the Network Interface Card (NIC). We then compare these schemes in terms of host processor offload, end-to-end latency, security transparency and cryptographic processing performance. Then we give an overview of security issues of cluster interconnects. In addition, we propose a cache architecture enabling fast session key access in Security-Enhanced Network Interface Card (SENIC).

# 1 Introduction

As of November 2005, more than 70% of the top 500 supercomputers are clusters [15]. Such trend is expected to continue because of cluster system's high cost-efficiency and high availability. Meanwhile, the recent security breaches in several companies mandated that in addition to providing high performance, these servers should be robust to security attacks such as data stealing and modifications [2]. Firewalls and Intrusion Detection Systems (IDS) may prevent many attacks, but also have vulnerabilities exploited by other attacks harming the cluster interconnect [28, 43].

The majority of research on cluster security has focused on securing data dormant in storage through encrypting files or blocks [20, 27, 32, 3, 10, 6, 34]. However, such schemes fall short in securing dynamically modified or generated data. Clearly confidential data, whether dormant in storage or in-flight movement, will need to be kept secret and unaccessible by attackers. General security measures such as application-level encryption and authentication would be enough when the amount of data is small. But if the amount of data is huge, the security processing overhead of a host CPU will be significant. For example, the Grid Security Infrastructure (GSI) does not recommend to encrypt or authenticate the communicating data for performance reasons. This means that if highly classified data need to be processed through Grid, it may need a physically separate Grid network.

This security performance problem becomes more complicated as cluster systems use very high speed cluster interconnects. Traditionally, all communication tasks such as packetizing and interrupts handling are done by the operating system, consuming a host CPU's computing power. As cluster interconnects become faster, the communication-handling overhead becomes more pronounced, adversely affecting the overall performance of cluster systems. ULC could mitigate this overhead because it off-loads the overhead to the NIC, freeing host processor cycles to execute user applications.

Because security operations require a great deal of computing power, securing in-flight data in a ULC-based environment becomes even harder. For this and to enforce strong security with minimal overall system performance degradation, we propose to incorporate security-specific hardware in the NIC.

The remainder of this paper is organized as follows: Section 2 presents an overview of a few popular cluster interconnects including Myrinet, InfiniBand, Quadrics and Gigabit Ethernet with their respective ULC protocols; Section 3 explores possible ULC security schemes including those relying on the host processor, secure coprocessor and SENIC and compares them in terms of overhead

off-load from host processor, end-to-end latency, security transparency, and cryptographic processing performance; Section 4 presents an overview of security issues of cluster interconnects; Section 5 presents our proposed security-enhanced NIC; and finally Section 6 concludes and presents future research.

# 2 Cluster Interconnect Overview

This section presents a summary of a few popular cluster interconnects covering basic features, usage trend in the supercomuting market, how ULC could be deployed and how user applications could utilize the architecture.

## 2.1 Myrinet

Myrinet was proposed for gigabit local area network in 1995 [22]. Later in 1998, it became an ANSI (American National Standards Institute) Standard [17]. As of November 2005, 20.2% and 14.4% of cluster computers among the top 500 supercomputers are Myrinet-interconnected in terms of the number of nodes and the amount of computing power, respectively. Even though its portion has been decreasing for the past two years, Myrinet is still the most widely used network in cluster systems next to Ethernet [15].

Myrinet comes in two speeds: 2Gbps and 10Gbps. By combining dual ports, the 2Gbps speeds can be extended to 4Gbps. Myrinet is a switching network with low-latency cut-through switches with technologies for detecting and isolating faults. It can scale up to tens of thousands of nodes enough for high-end clusters. One major advantage of Myrinet is its interoperability with Ethernet.

For maximum network utilization, Myrinet's software stack includes provisions for user-level access, OS-bypass and CPU offloading through firmware execution of the network protocol in the NIC. The two basic Myrinet software stacks are called GM and MX for 2Gbps and 10Gbps, respectively. Both are open-source supprting most common architectures and OSes. Besides, because of Myrinet's firmware programmability, researchers have used Myrinet to propose and develop other user-level protocols such as AM, FM, U-Net and VMMC [40, 33, 39, 24]. Santosh, *et al*. summarized research conducted on Myrinet in [35].

However, a common difficulty in using user-level protocols is that applications often need to be modified to access NIC directly. To remove this problem, Myrinet provides several software libraries to allow legacy software to utilize tranparent OS-bypassing features. Currently available libraries are MPI (Message Passing Interface), Sockets, DAPL (Direct Access Programming Library), VI (Virtual Interface Architecture), and PVM (Parallel Virtual Machine).

## 2.2 InfiniBand

The InfiniBand Architecture (IBA) emerged as a result of merging two competing I/O technologies: System I/O and Next Generation I/O. Its specification is being defined by the InfiniBand Trade Association – an industry consortium [18]. IBA's main success has been as a cluster interconnect. According to Top500 organization, as of November 2005, 5∼6% of cluster systems use InfiniBand. Compared to the market segments of the other two leading interconnects – Ethernet and Myrinet, IBA remains third distant. However, it is continuously growing with more IBA-based clusters making the Top500 list.

The basic link speed for IBA is a Single Data Rate (SDR) of 2.5Gbps (1x); by aggregating multiple links, speeds can reach 10Gbps (4x) and 30Gbps (12x). And with Double Data Rate (DDR) and Quad Data Rate (QDR), speeds up to 60Gbps and 120Gbps, respectively, can be reached. To achieve such high bandwidth, IBA offloads a great deal of its network processing to the adapters, referred to as Host Channel Adapter (HCA) and Target Channel Adapter (TCA).

A good deal of software support for IB is currently available including open-software stack from the OpenIB Alliance. Most software stacks supply IPoIB (IP over IB), SDP (Socket Direct Protocol), SRP (SCSI RDMA Protocol), uDAPL (user Direct Programming Library) and MPI.

## 2.3 Quadrics

QsNet, a product of Quadrics, has long been the interconnect of choice for high-end supercomuters. According to the Top500 list of November 2005, 2.8% and 3.9% of clusters use Quadrics in terms of number of systems and total performance of systems, respectively. The latter number clearly indicates that Quadrics is preferred for high-performance supercomputers. However, according to [15], this percentage had been decreasing, losing ground to IBA.

In addition to typical features of network adapters, QsNet provides outstanding features, including, globally shared virtual memory space, programmability on a network processor and fault detection and tolerance. QsNet-II, a follow-on to QsNet, can deliver up to 7.2Gbps from a user space to a user space [14], which is still way below what IBA can deliver.

Quadrics also provides open-source software stack to use Quadrics. Elanlib provides the lowest-level, user space programming environment. On top of this, Shmem and MPI programming are possible.

## 2.4 Ethernet

Ethernet has extended its presence from the Local Area Network (LAN) focus to even supercomputing. Again, as of November 2005, 360 systems of the Top500 list clusters use Ethernet. 249 of them use Gigabit Ethernet.

Gigabit Ethernet was first standardized in 1998 as IEEE 802.3z whose interconnecting medium was limited to optical fiber. Later, Gigabit Ethernet standard, IEEE 802.3ab, using copper cables was ratified in 1999. Both standards' theoretically maximum bandwidth is 1000 Mbps but its practical throughput without any tuning techniques is hardly over 500 Mbps. However, since its performance meets the requirements of mid- and low-end cluster systems, the number of Ethernet-based cluster systems has been increasing rapidly.

The quest of Ethernet community for high speed networking resulted in 10G Ethernet. IEEE 802.3ae 10G Ethernet was standardized in 2002, specified only on fiber optic medium and interfaces. In 2004, a copper cable-based 10G Ethernet, IEEE 802.3ak 10GBASE-CX, was approved as an IEEE standard. It uses high-grade copper cables, similar to 4x IBA. Despite of the 15m distance limitation, it is expected to lower the total cost of cluster systems by combining IEEE 802.3ae for long distance and IEEE 802.3ak for short distance. To lower the cost more and go beyond the distance limitation, 10GBASE-T group is trying to complete its standard to use the structured, twist-pair cabling (Category 5 or 7) by 2006. Once ratified, it will be more cost-effective to build Ethernet-interconnected cluster systems.

Different from the aforementioned cluster interconnects, Ethernet was not originally designed for high performance communication. Among several reasons, a high CPU overhead has been a major hurdle for Ethernet to be used in such high performance communication environment. Without help

of ULC, a host CPU has to do all communication processing including TCP/IP processing, buffer-to-buffer copy, system calls to initiate communications, and so on. To alleviate this overhead, a non-standard tweak, Jumbo Frames, has been widely used, especially in cluster systems. By increasing the Maximum Transmission Unit (MTU) size from the standard 1500 Bytes up to 9000 Bytes, the packetization overhead of CPU was greatly decreased. As [21] showed, Jumbo Frames enhances Ethernet throughput by 20$\sim$30%. However, since Jumbo Frame is not inter-operable with standard Ethernet networks due to its differt MTU size, it causes another re-packetizing overhead in gateway routers or switches connecting two Ethernet networks.

To provide more fundamental solutions to CPU overhead problem, there have been some research to apply ULC or OS bypassing techniques to Ethernet. These can be categorized into ULC or non-ULC approach. The ULC approach provides ULC protocols in Ethernet environment while the non-ULC approach tries to apply several CPU offloading techniques without providing user-level handles to access a NIC directly.

Well-known user-level protocols for Ethernet are MESH, U-Net, M-VIA, EMP, and iWARP [9, 39, 7, 36, 5]. MESH [9] is a user-level library enabling the fast context switching and zero-copy communication. Through these techniques, it can obtain around 900 Mbps incurring only 5% more or less host CPU overhead in both sending and receiving nodes. U-Net, developed by Eicken, *et al*. exposes a set of in-memory queues to user software instead of providing API calls [39]. This allows user applications to transfer message to and from network without CPU overhead. In [41], a NIC design was proposed for fast Ethernet network implementing U-Net. Its design greatly influenced the design of VIA. M-VIA [7] is an implementation of VIA for Linux on Ethernet. In addition, its modular design makes it easy to port M-VIA to new hardware and M-VIA can run either in hardware accelerated mode or in full software mode, depending on hardware support. EMP [36] is a use-level communication protocol facilitating zero copy, OS bypass, and NIC-driven communication on Gigabit Ethernet. Using programmable NICs, it shows good performance with a latency of 23$\mu$s and throughput of 880 Mbps. iWARP, shorthand of ”Internet Warp,” is one of the most promising Ethernet-based ULC solutions led by the iWARP consortium [5]. iWARP describes three protocols: Remote Direct Memory Access (RDMA), Remote Direct Data Placement (RDDP), and Marker PDU Alignment (MPA). The most important feature of this specification is that it provides zero copy and OS bypass while it uses TCP/IP as its underlying communication protocol which is also offloaded, thus providing low latency, high performance, and high interoperability. Products are now available

in [11].

As described earlier, for better portability of legacy software using socket libraries to Ethernet ULC, Balaji, *et al*. proposed a mapping layer between the socket library and EMP [19]. This allows legacy software to keep using its existing socket programming. We categorize it as non-ULC approach although it is using EMP. In addition to this, there are three more non-ULC techniques to briefly introduce. GAMMA [4] is a low latency, high throughput communication system for Linux clusters systems interconnected by Gigabit Ethernet. The core of GAMMA is a custom Linux network device driver. This driver substitutes heavy communication system calls which requires an expensive context switching with light-weight ones which allow user processes to directly call kernel routines, thus reducing host CPU overhead. PC Cluster Consortium [12] is providing a cluster system software package, SCore. One of its communication library, PM/Ethernet, is designed to improve communication performance by reducing interrupt handling overhead through *interrupt reaping* [38]. Different from GigaE-PM [37], it can work on any Ethernet NIC without help of hardware functionalities. TCP Offloading Engine (TOE) is designed to offload TCP/IP processing overhead from CPU to a processor or ASIC on NIC. For example, as analyzed in [26], a TOE implementation has a latency of 8.9$\mu$s and a throughput of 7.6 Gbps. Since TOE is also hidden from software using socket libraries, data centers or web-servers whose communications are mostly TCP/IP can get much benefit.

# 3 Challenges in Cluster Interconnect Security

As we explained in the previous section, ULC is now necessary for a high performance cluster computing. Due to this, the current security paradigm that security operations are done in a host CPU or a secure co-processor can be shifted: NIC with security functionalities can take over much security operations. in cluster systems. In this section, we will compare three possible approaches to secure intra-cluster communication in cluster systems; the host processor, secure coprocessor, ane NIC. Then we will show advantages of NIC-supported security scheme.

| | Host Processor | Secure Coprocessor | Security Enforced NIC |
|---|---|---|---|
| Host Processor Offloading | No | Yes | No |
| End-to-end Latency | High | High | Low |
| Security Transparency | No | Medium | Yes |
| Cryptographic Processing Performance | Good | Good | Medium |

Table 1: User-level communication security model comparison

## 3.1 User-Level Communication Security Schemes

• **Host processor:** The host processor can do all security operations when applications send or receive data. When an application makes a send request, security functions will be called and executed by the host processor. This can be done in an explicit or an implicit way. In the first case, before the application sends data, it encrypts the data by explicitly calling security functions. Similarily, when the data arrive a destination, a receiving application should call security functions to decrypt or authenticate the data. In the implicit way, security functions can be inserted into communication libraries such as Sockets.

• **Secure Coprocessor:** There may be a specialized processor or dedicated logic attached to the memory bus or to the PCI-bus to offload security operations from the host processor. In either explicit or implicit way, a secure coprocessor processes security operations requested by user applications or the kernel.

• **Security-Enforced NIC (SENIC):** This is an intelligent NIC augmented with a secure processor or a security ASIC. When a send request is directly transferred to SENIC, it can do security operations on the data while it is getting the data from user memory and sending into cluster interconnects in a pipeline way. In the same way, a receiving SENIC decrypts or authenticates the data while it transferring the data to a recipient's memory space.

## 3.2 Security Schemes Comparison

In this subsection, we compare the aforementioned three approaches in terms of host processor offloading, end-to-end latency, security transparency, and cryptographic processing performance as

summarized in Table **??**.

• **Host Processor Offloading:** Since depending on the host processor to execute security operations incurs more CPU processing and/or intermediate memory copy operations, this approach is counter to the goals of ULC.

• **End-to-End (E2E) Latency:** E2E latency is the total time from when an application initiates a send request to when the very first part of the data arrives a destination's user memory space. The host processor approach will surely have the longest E2E latency because it cannot send the data before finishing all security operations on the data. The secure coprocessor approach will have shorter lantency than the host processor approach, but not shorter than the SENIC approach. That is because it will not be easy to pipeline security operations in a secure coprocessor and data communication in NIC due to memory bus contention between the two components. E2E latency of SENIC will be the shortest since it does not have such bus contention and more importantly it can pipeline security operations and data transmission. That is, it can immediately send data even before security operations on the whole data are completed.

• **Security Transparency:** We define the security transparency as keeping security operations hidden from applications and/or operating systems as possible. We believe this is the most important feature to minimize (or eliminate) applications' or OSes' code modification as well as reducing performance overhead. In the host processor scheme, applications and OS must handle all security details such as calling security functions. A secure coprocessor on the other hand seems more security transparent, but still requires moderate OS interventions. For instance, if process (or finer)-level security is enforced, that level of key management will be required. To accomplish this in a secure coprocessor, the OS will need to keep the key information up-to-date to compare against keys in incoming and outgoing packets. Security breaches could happen if the key information is not timely updated. By contrast, SENIC is the most security transparent since a NIC includes provisions for ULC contexts making it easier to integrate all key management.

• **Cryptographic Processing Performance:** By using its high-throughput deep pipelines and code optimization, the host processor can do security processing as the secure coprocessor does. The only concern is that it is more desirable for host processors to execute user applications rather than security operations for overall performance. Since a SENIC could be envisioned as a secure coprocessor, it could be used for general cryptographic processing such as encrypting user data or signing a document. However the main purpose of the SENIC is to process communication packets.

Therefore if SENIC needs to process both general cryptographic tasks and secure communication tasks, the SENIC would have to give the secure communication higher priorities, thus limiting the cryptographic processing performance of SENIC.

# 4 Security Issues in User-level Communication

In this section, we will address several security issues which should be considered when SENIC comes into play.

□ **Key Distribution**

To provide security in intra-cluster communication, a secure key distribution mechanism is necessary. There will be several ways in desiging the secure key distribution in a cluster.

Unlike the Internet, a cluster system is a relatively closed system. Therefore, one can assume that there will be a period void of any attacks such as the cluster installation time or the system booting time (even though it is quite rare). Based on this assumption, cluster nodes can predistribute $N$-to-$N$ secret keys or a secret key shared by all nodes without worrying about security attacks. Subsequent key distribution can be protected by those predistributed keys.

When a SENIC is being used, this key distribution can be done partly in the SENIC. For example, for better performance the initial key distribution can be done in the OS level while the subsequent session key distribution done in the SENIC.

□ **Security Algorithm**

In the host processor scheme, virtually any security algorithms are possible by simply running any kinds of security applications. Although limited, the secure coprocessor often provides several security algorithms. Due to its speed limitation, SENIC may not be able to provide such complex and various security hardware. In this case, it is more efficient for the host processor to execute such complex security operations. Therfore, when a SENIC is used, security operation load balancing will be necessary in SENIC-enabled communication for both performance and security reasons.

□ **Cluster Interconnect Protocol Vulnerability**

As shown in [23, 31], there are some security vulnerabilities in cluster interconnect protocols. For example, in InfiniBand the authenticity of a packet is verified the existence of a valid plaintext key. If the packet is captured by any chance, the whole security mechanism can be broken.

To improve its security, encryption and authentication schemes need to be considered in design-

ing the protocols. SENIC can play an important role in security enhancement in cluster interconnect protocols by supplying basic security functions.

□ **Security Attack Response**

Real time detection of security attacks is usually very difficult. The problem becomes more complicated in distributed high-performance communication environment like cluster systems. Accordingly, the task of detection is computing intensive and takes long, so SENIC may not be useful for detection. However, once an attack is detected, the reaction can be done in the NIC-level efficiently. For example, if an access control is applied in the kernel level, a tremendous amount of attacking packets will impose considerable performance overhead on the system since the OS needs to check every packet to determine whether to drop or accept it. Instead, such an access control can be applied in SENIC. Then, the SENIC will filters out all those attacking traffic, thus saving the host processor's computing power.

□ **On-Demand Security or QoS on Security**

On-demand security will be useful when the secure computing/communication is required occasionally. In addition, different levels of security strength can be enforced depending on user/application's security requirements. Without SENIC, only the host processor approach is feasible, costing significant computing power. Instead, if the SENIC is able to accommodate and execute some security requirements of these services, on-demand or security QoS is more feasible in cluster systems.

□ **Key Management in SENIC**

Since most security operations are done in SENIC, a number of secret keys need to be stored in SENIC. Otherwise, SENIC needs to access the system memory for retrieving the keys at every packet. Note that a latency to retrieve those keys should be faster than or equal to the latency to process packets in SENIC. Otherwise, key management will cause substantial delay in packet processing. If the number of keys is large, this will be challenging.

□ **High Speed Security Operation**

One of the most critical requirements of SENIC is security operation speed. Current speed of cluster interconnect is around 10Gbps (Myrinet, Quadrics, Ethernet) or sometimes far faster (InfiniBand). To minimize performance overhead of secure computing, security operation throughput in the SENIC should be at least comparable to the network bandwidth of those cluster interconnects.

□ **Security Vulnerability of SENIC**

One big concern is that SENIC itself may be vulnerable to security attacks. The attacks can be phys-

ical or application-level. To protect against the physical attacks, SENIC has to be tamper-evident or tamper-resistant to be used in highly secure cluster systems. To defend against application-level attacks, SENIC software should be more carefully designed. More protection mechanisms should be integrated to prevent the SENIC from leaking its secret keys or overwriting malicious codes into the system memory.

□ **Power Consumption**

The power consumption has been a hot issue in designing cluster systems so that the consumption does not increase as much as the size of cluster systems increases. Power consumption in communication is not negligible any longer, especially in communication intensive systems such as multimedia web-servers. SENIC will consume more power than normal NICs because security hardware needs additional powers. Note that without SENIC a host processor or secure coprocessor will consume power.

# 5 Session Key Cache in Security-Enforced Network Interface Card

In this section, we propose a SENIC design focusing on the fast key access we mentioned in the previous section. The main design goal is to minimize the performance degradation which might occur when the number of key is huge.

## 5.1 Overall Architecture

To provide a fine-grain security service, so many session keys, called session keys, will be generated per connection or per packets and these keys need to be stored in SENIC memory. If a session key should be retrieved from SENIC memory at every packet processing, it will cause significant E2E delay because SENIC memory access time is in the critical path. To shorten this delay, we propose a Session Key Cache (SKC) in SENIC as shown in Figure 1. Security Logics, Session Key Cache, and Basic Key RAM (BKRAM) are integrated into the original NIC architecture. While the original NIC logics process packets in the context of its user-level communication, Security Logics performs

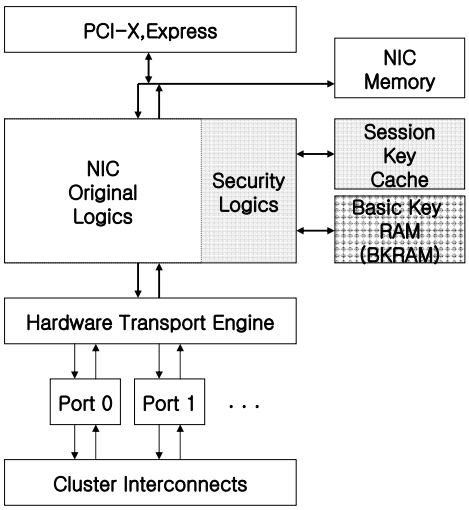


Figure 1: Security enforced NIC architecture with session key cache.

security operations on the packet in parallel.

Basic Key RAM (BKRAM), powered by a battery which can support the power during product lifetime, is used to hold basic keys such as one shared key. Such RAM is also called Battery Backed RAM (BBRAM) [25]. Its contents will be instantly zeroized by cutting off the power supply once a tamper is detected. FIPS 140-1 recommends BBRAM to store sensitive data [16]. The contents of BBRAM should be toggled periodically because if the same data are stored for long time the contents can be imprinted into RAM, so the data might remain recognizable even after zeroization. Thanks to these characteristics of BBRAM, we can store some basic keys such as an initially distributed key, a private key of SENIC, a common secret key shared by multiple SENICs, and so on.

When a new session key is generated, it is stored both in SKC and SENIC memory. The main purpose of SKC is to reduce totoal memory access time by exploiting the temporal locality of packet arrivals. Even though a great number of process-to-process communications and their session keys exist, only a small number of session keys will be used in a short time period. Least Recently Used (LRU) is used as a replacement policy to utilize the temporal locality.

To estimate the cost and power consumption of this design, we analyze a design with two AES hardware blocks (send and receive) with 64KB session key cache. Since BKRAM's size will be very small and its access is also infrequent, we ignore its space and access time in this analysis. Hodjat *et al*. [29, 30] recently designed an AES using 175K Gates gaining 35 Gbps speed. According to [1], AMIS can make an ASIC with high density (100,000 Gates / $mm^2$) and low power (30nW/MHz/Gate @ 1.8V) using 0.18 $\mu$s technology. By using their technology, one AES block will

| | Space | | | Power | | |
|---|---|---|---|---|---|---|
| | AES($mm^2$) | Cache($mm^2$) | Overhead(%) | AES(W) | Cache(W) | Overhead(%) |
| 1x | 3.5 | 187.15 | 15.56 | 2.1 | 1.21 | 27.59 |
| 4x | 14 | 187.15 | 16.42 | 8.4 | 1.21 | 80.09 |
| 12x | 42 | 187.15 | 18.71 | 25.2 | 1.21 | 220.09 |

Table 2: Security Enforced CA space and power overhead

consume 17.5$mm^2$ space and 1.05 W with 200MHz clock speed. Suppose we implement SENIC on InfiniHost MT23018, a CA chip developed by Mellanox [8], Table 2 summarizes space and power consumption of SENIC up to 12X security blocks. Metrics of the cache are quantized using CACTI model [42]. Compared to the space overhead, the power overhead is rather big. But, it will be 10$\sim$20% increase in overall system power consumption [13].

A 64KB fully associative cache can contain 4K entries of 128bit session key with 4.3ns access time. Considering the latency of current cluster interconnect is microsecond scale, additional several nanosecond scale will not be significant. Therefore, the cache hit rate in SKC is the most important factor in performance overhead because frequent cache misses will require much longer memory accesses.

# 6 Conclusions

This paper addresses security issues in cluster interconnects, especially with user-level communication (ULC). The important conclusions of this work are the following. First, we compared possible schemes to enable ULC security; host processor, secure coprocessor, and security enforced network interface card (SENIC). We strongly advocated SENIC for its low end-to-end latency and high security transparency while retaining the performance benefit of ULC through bypassing the host processor.

Second, we raised several security issues that should be reconsidered when the NIC-supported security enhancement is in effect. Some traditional security problems such as the key distribution and security attack responses can be dealt by the SENIC more efficiently. Besides, we also addressed

new security & design issues such as the key management and security vulnerabilities.

Third, as a solution to an efficient key management in SENIC, we designed a Session Key Cache (SKC) inside NIC to enable fast key accesses and fine-grain security granularity. We analyzed a SENIC with a 64 Kbytes cache in terms of power and space overheads. It incurs 15.56% space and 27.59% power overhead by comparing Mellanox InfiniHost MT23018.

We are currently examining a number of possible extensions to this work. First, we are analyzing the efficiency of SKC of SENIC by modeling scientific workloads. Furthermore, we will simulate using the realistic Web-based or Database servers. In-depth experiment with those server applications should fortify our SENIC design. Second, we are exploring the possibilities to make use of SENIC in enhancing Grid security by integrating SENIC concepts into Grid Security Infrastructure. Finally, we would like to apply our research to seek for feasible solutions in practical cluster interconnects such as InfiniBand or 10 Gigabit Ethernet.

# References


[1] AMIS, http://www.amis.com/asics/standard_cell.html.

[2] Data under siege, http://www.washingtonpost.com/wp-dyn/articles/a19982-2005mar9.html.

[3] Decru, http://www.decru.com/.

[4] GAMMA project, http://www.disi.unige.it/project/gamma/.

[5] iwarp consortium, http://www.iol.unh.edu/consortiums/iwarp/.

[6] Kasten Chase, http://www.kastenchase.com/.

[7] M-via, http://crd.lbl.gov/ftg/mvia/mvia.shtml.

[8] Mellanox, http://www.mellanox.com/.

[9] Mesh projects, http://atlas.web.cern.ch/atlas/project/cern/ep-atr/mesh/.

[10] Neoscale, http://www.neoscale.com/.

[11] NetEffect, http://www.neteffect.com/.

[12] Pc cluster consortium, http://www.pcluster.org/.

[13] Power consumption, http://techreport.com/reviews/2005q2/athlon64-x2/index.x?pg=15.

[14] Quadrics, http://www.quadrics.com/.

[15] Top500 supercomputers, http://www.top500.org/.

[16] FIPS 140-1: Security requirements for cryptographic modules. *National Institute of Standards and Technology*, 1994.

[17] ANSI/VITA 26-1998: Myrinet-on-VME Protocol Specification. *American National Standard*, 1998.

[18] Infiniband architecture specification, volume 1, release 1.1. *InfiniBand Trade Association*, 2002.

[19] P. Balaji, P. Shivam, P. Wyckoff, and D. K. Panda. High performance user level sockets over gigabit ethernet. In *CLUSTER*, pages 179–186, 2002.

[20] M. Blaze. A cryptographic file system for UNIX. In *ACM Conference on Computer and Communications Security*, pages 9–16, 1993.

[21] B. M. Bode, J. J. Hill, and T. R. Benjegerdes. Cluster Interconnect Overview.

[22] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W.-K. Su. Myrinet: A Gigabit-per-Second Local Area Network. *IEEE Micro*, 15(1):29–36, 1995.

[23] R. Dimitrov and M. Gleeson. Challenges and new technologies for addressing security in high performance distributed environments. In *Proceedings of the 21st National Information Systems Security Conference*, pages 457–468, 1998.

[24] C. Dubnicki, A. Bilas, and K. Li. Design and implementation of virtual memory-mapped communication on myrinet. In *11th International Parallel Processing Symposium (IPPS '97)*, page 388, 1997.

[25] J. Dyer, R. Perez, S. Smith, and M. Lindemann. Application support architecture for a high-performance, programmable secure coprocessor. In *Proceedings of the 22nd National Information Systems Security Conference*, October 1999.

[26] W.-C. Feng, P. Balaji, C. Baron, L. Bhuyan, and D. Panda. Performancf Characterization of a 10-Gigabit Ethernet TOE. In *Hot Interconnects 13, (HotI 2005)*, Stanford, CA, August 2005.

[27] K. Fu. Group sharing and random access in cryptographic storage file systems. *Master's thesis, MIT*, 1999.

[28] D. Geer. Just how secure are security products? *Computer*, 37(6):14–16, 2004.

[29] A. Hodjat and I. Verbauwhede. A 21.54 gbits/s fully pipelined aes processor on fpga. In *FCCM '04: Proceedings of the 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 308–309, Washington, DC, USA, 2004. IEEE Computer Society.

[30] A. Hodjat and I. Verbauwhede. Minimum area cost for a 30 to 70 gbits/s aes processor. In *IEEE Computer Society Annual Symposium on VLSI*, pages 83–88, 2004.

[31] M. Lee, E. J. Kim, and M. Yousif. Security enhancement in infiniband architecture. In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium*, 2005.

[32] D. Mazieres. Security and decentralized control in the SFS global file system. *Master's thesis, MIT*, Aug. 1997.

[33] S. Pakin, M. Lauria, and A. Chien. High performance messaging on workstations: Illinois fast messages (fm) for Myrinet. In *Supercomputing '95: Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM)*, page 55, New York, NY, USA, 1995. ACM Press.

[34] E. Riedel, M. Kallahalla, and R. Swaminathan. A framework for evaluating storage system security. In *Proceedings of the Conference on File and Storage Technology*, pages 15–30, Monterey, CA, Jan. 2002.

[35] R. D. Santos, R. Bianchini, and C. L. Amorim. A Survey Of Messaging Software Issues And Systems For Myrinet-Based Clusters. *Parallel Distributed Computer Practices, Special issue High-Performance Comput. Clusters*, 2(2), 1999.

[36] P. Shivam, P. Wyckoff, and D. Panda. Emp: zero-copy os-bypass nic-driven gigabit ethernet message passing. In *Supercomputing '01: Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM)*, pages 57–57, New York, NY, USA, 2001. ACM Press.

[37] S. Sumimoto, H. Tezuka, A. Hori, H. Harada, T. Takahashi, and Y. Ishikawa. The design and evaluation of high performance communication using a gigabit ethernet. In *International Conference on Supercomputing*, pages 260–267, 1999.

[38] S. Sumimoto, H. Tezuka, A. Hori, H. Harada, T. Takahashi, and Y. Ishikawa. High performance communication using a commodity network for cluster systems. In *HPDC*, pages 139–146, 2000.

[39] T. von Eicken, A. Basu, V. Buch, and W. Vogels. U-net: a user-level network interface for parallel and distributed computing (includes url). In *SOSP '95: Proceedings of the fifteenth ACM symposium on Operating systems principles*, pages 40–53, New York, NY, USA, 1995. ACM Press.

[40] T. von Eicken, D. E. Culler, S. C. Goldstein, and K. E. Schauser. Active messages: a mechanism for integrated communication and computation. In *ISCA '92: Proceedings of the 19th annual international symposium on Computer architecture*, pages 256–266, New York, NY, USA, 1992. ACM Press.

[41] M. Welsh, A. Basu, and T. von Eicken. ATM and fast ethernet network interfaces for user-level communication. In *Proceedings of Third International Symposium on High Performance Computer Architecture*, February 1996.

[42] S. Wilton and N. Jouppi. CACTI: An enhanced cache access and cycle time model. *IEEE Journal of Solid-State Circuits*, 31(5):677–688, 1996.

[43] A. Wool. A quantitative study of firewall configuration errors. *Computer*, 37(6):62–67, 2004.