Improvement of Computer Networks Security by Using Fault Tolerant Clusters

Serb Aurel
aserb@mta.ro

Patriciu Victor-Valeriu
vip@mta.ro

Radar, Communications and Computer Science Faculty
Military Technical Academy
Bucharest, Romania

Abstract

A fault tolerant system is one that can continue to operate reliably by producing acceptable outputs in spite of occasional occurrences of component failures. A fault tolerant cluster is a cluster with a set of independent nodes, connected over a network, and always with external storage devices connected to the nodes on a common input/output bus. The cluster software is a layer that runs on top of local operating systems running on each computer. Clients are connected over the networks to a server application that is executing on the nodes. One of the most important problems in implementing fault tolerant system is the identification of single points of failure and elimination of these single points of failure by using replaceable units.

The need for interoperability between the M&S world and the C4ISR world has been formulated in several publications. The challenge even increases when NATO and PfP Nations demands to train using their own simulation systems as well as their own command and control systems.

Today, can be identified some key words that are in common in modern modeling and simulation systems, command and control systems and in fault tolerant clusters. Some of them can be:

- Open and distributed systems.
- Networks.
- High level operating systems.
- Segments federates (federations) and packages.
- Hierarchical architecture.
- Commercial standards, specifications, and products.
- Interoperability and reusability.
- High availability systems.

Keywords: fault tolerant cluster, modeling, simulation, command, control.

1. Introduction

Advances in information technology have created major efficiencies in the design of large-scale command and control systems and modeling and simulation systems. Hardware is constantly improving and more sophisticated and powerful software packages are available commercially. And the ability to network both locally and over worldwide systems can be fully utilized for the management of these systems.

The development of improved cyberdefense capabilities is gaining increasing importance and attention as the civilian, government, and military sectors become more reliant upon networking and computer technologies to conduct routine activities and manage crisis situations. Within the area of command and control and modeling and simulation, distributed systems can provide the foundation for the development, testing, and evaluation of defensive layering technologies. A layered defense strategy must pay attention to a balance of avoidance, detection, and response techniques designed to improve security, performance, and functionality.

The natural growth in scope and importance of the computing environment has been accelerated by recent trends toward globalization and mergers. The result: ”mission-critical” has a broader meaning than ever before, and downtime can affect operations in the next room and on the next continent. The increasing reliance on networked applications and information means that all parts of an organization or military system can be seriously affected by physical, design, or human-machine interaction faults, by viruses and malicious acts or by an isolated local disaster such as an earthquake, flood, hurricane, fire, or theft. When these faults or disasters strikes a data center, it can mean much more than a temporary loss of computing power. Work delays, data degradation, and data loss can quickly translate into an unacceptable loss, which can’t be allowed.

2. Fault Tolerance in High Availability Clusters

2.1 Fault Tolerance

The open and distributed systems, which are the most important systems used for command and control, and modeling and simulation must never fail. But only ideal systems would be perfectly reliable and never fail. This, of course, is impossible to be achieved in practice, because the systems fail for many reasons. Fault tolerance is the best guarantee that high-confidence systems will not succumb to physical, design, or human-machine interaction faults, or will allow viruses and malicious acts to disrupt essential services.

A fault tolerant system is one that can continue to operate reliably by producing acceptable outputs in spite of occasional occurrences of component failures.

The basic principle of fault-tolerant design is the use of redundancy, and there are three basic techniques to achieve fault tolerance: spatial (redundant hardware), informational (redundant data structures), and temporal (redundant computation).

The classical hardware and software fault tolerant techniques are modular redundancy, N-version programming, error-control coding, checkpoints, rollbacks, and recovery blocks.

2.2 Replaceable Units

Modern systems are partitioned at several levels based on functions provided by specific subsystems. A fault-tolerant system displays similar functional partitioning, but in addition it contains redundant components and recovery mechanisms, which may be employed in different ways at different levels. It is reasonable to view a fault-tolerant system as a nested set of subsystems each of which may display varying levels of fault tolerance. Recovery from a fault within a redundant partition may be effected within the domain itself, or may require action by higher levels within the system.

Fault tolerant architectures package these redundant partitions into replaceable units. A replaceable unit is a unit of failure, replacement and growth - that is, a unit that fails independently of other units, which can be removed without affecting other units, and can be added to a system to augment its performance, capacity, or availability.

2.3 Clusters

A distributed system is a collection of computers (called nodes) that communicate with each other through a communication medium. Under the control of systems software, the nodes can co-operatively carry out a task. An open system allows system integration, so the customers can choose various hardware and software components from different vendors and integrate them to create a custom configuration suiting their needs and cost requirements.

A cluster is a set of loosely coupled, independent computer systems, connected over a network that behave as a single system. The cluster software is a layer that runs on top of local operating systems running on each computer. Client applications interact with a cluster as if it is a single high-performance, highly reliable server. System managers view a cluster much as they see a single server. Most applications will run on a cluster without any modification at all. And only standard-based hardware components such as SCSI disks and Ethernet LANs are used to create a cluster.

Additional systems can be added to the cluster as needed to process more complexes or an increasing number of requests from the clients. If one system in a cluster fails, its workload can be automatically dispersed among the remaining systems. This transfer is frequently transparent to the client.

2.4 Fault Tolerant Clusters

A fault tolerant cluster is a cluster with a set of independent nodes, connected over a network, and always with external storage devices connected to the nodes on a common input/output bus. Clients are connected over the networks to a server application that is executing on the nodes. The nodes of a cluster are connected in a loosely coupled manner, each maintaining its own separate processors, memory, and operating system. Special communications protocols and system processes bind these nodes together and allow them to cooperate to provide outstanding levels of availability and flexibility for supporting mission-critical applications. Fault tolerant clusters maintain strict compliance to the principles of open systems. Most applications will run on a fault tolerant cluster without any modification at all.

The top-level software of a fault tolerant cluster can be designed to maximize the flexibility of configurations within a local cluster. Clusters may be formed with a different number of nodes. This flexibility in system selection and cluster configuration protects customer investments in installed systems and allows the processing power of each node to be matched with the specific requirements of each application service.

If the failure of any component in a cluster results in the unavailability of service to the end user, this component is called a single point of failure for the cluster. One of the most important problems in implementing fault tolerant system is the identification of single points of failure and elimination of these single points of failure by using replaceable units.

The elimination of a single point of failure, by using replaceable units, always has a cost associated with it. Usually, what can be done is only to attempt to make a service highly available if the cost of losing the service is greater than the cost of protecting it.

The possible single points of failure that a cluster could have are:

- Nodes in the cluster,
- Disks used to store application or data, adapters, controllers and cables used to connect the nodes to the disks,
- The network backbones over which the user are accessing the cluster nodes and network adapters attached to each node,
- Power sources,
- Applications.

A fault tolerant cluster is a grouping of servers having sufficient redundancy of software and hardware components that a failure will not disrupt the availability of computer services. The modern fault tolerant clusters are able to eliminate all single points of failure. To develop a complete high availability solution, is necessary to be maintained high availability within a hierarchy of system levels, some of which go beyond the cluster level. Failure at all levels must be detected quickly and a fast response provided. When a component becomes unavailable, fault tolerant cluster software detects the loss and shifts that component's workload to another component in the cluster. The failure recovery is done automatically, without any human intervention.

2.5 Eliminating Nodes as Single Points of Failure

The node in a fault tolerant system consists of a group of components, any of which can fail. The most important components are:

- One or more central processing units,
- Memory boards,
- Input/output controllers.

The use of cluster architecture lets the system eliminate a node as a single point of failure without losing service.

The nodes are connected to each other by a local area network, which allows them to accept client connections and to transmit messages that confirm each other's health. If one node fail, the failured node is removed from the cluster and, after only a brief delay, its resources are taken over by the node configured to do so, so called the takeover node. This process is known as failover. The process of failover is handled by special high availability software running on all nodes in the cluster. Different types of clusters use different cluster management and failover techniques. There are specific differences in cluster types and their high availability software.

In fault tolerant clusters, disks containing data are physically connected to multiple nodes on a common input/output bus. When a node that owns a disk fail, a surviving node assumes control of the disk, so that the critical data remains available.

Clients and other devices are connected over the networks to the nodes. After the failover, all the clients and network devices connected to the failed node can access the second node as easily as the first. When a node failed during the running of a critical application, a takeover node can restart that application so that the service is not lost.

2.6 Eliminating Disks as Single Points of Failure

A fault tolerant solution improves data availability by allowing that a number of nodes to share the same hard disks within a cluster. When a node in the cluster fails, the fault tolerant cluster software will recover and disperse the work from the failed node to another node within the cluster. As a result, the failure of a system in the cluster will not affect the other systems, and in most cases, the client applications will be completely unaware of the failure.

Each node in a cluster has its own root disks, but each node may also be physically connected to several other disks in such a way that multiple nodes can access the same data. On such systems, this cluster-oriented access is provided by a software cluster component called Logical Volume Manager. Access may be exclusive or shared, depending on the kind of cluster created. Redundancy is necessary to prevent the failure of disk media or a disk controller.

Different fault tolerant configurations provide a range of solutions that address varying levels of fault protection requirements, including multi-site resiliency solutions. There are solutions that combine local fault tolerant cluster configurations with Fibre Channel mass-storage devices in order to provide a disaster-tolerant solution for a clustering environment up to 40 kilometers apart.

The most important two methods available for providing disk redundancy are: using disk arrays in a RAID configuration and using software mirroring. Each approach has its own advantages.

RAID (Redundant Array of Inexpensive Disk) is a disk technology that is designed to provide improved availability, security and performance over conventional disk systems. While appearing logically to the operating system as a single disk drive, a RAID array is actually made up of several disks, which have their data spread across the drives in any of several different methods. The group of disks that function together in a well-defined arrangement is known as RAID level. RAID Level 1 allows hardware mirroring, while others provide protection through the use of parity data. The RAID levels allow the array to reconstruct lost data if a disk fails.

In addition, arrays can be configured in independent mode, which means that each member of the array is seen as an independent disk.

An alternative technique for providing protected data storage is the use of software mirroring, which allows that a single logical filesystem to be implemented on multiple physical copies in a way that is transparent to users and applications. It means that if a disk or sectors of a disk, containing on copy of the data should fail, the data will still be accessible from another copy on another disk. Note that the mirror copy is on a separate input/output bus. This arrangement eliminates the disk, the input/output card and the bus as single points of failure.

2.7 Eliminating Networks as Single Points of Failure

Networks are configured and used in clustered systems for access to an application by clients or other systems, and for communication between cluster nodes. In a fault tolerant cluster, the software establishes a communication link knows as a heartbeat. It is recommended that the fault tolerant cluster to be designed with more than one network, so that high level cluster software has at least one network at all times that it can use to monitor the status of cluster nodes.

This special use of networking must itself be protected against failures. Points of failure in the network include the LAN interfaces and cables connected to each node. In the cluster the entire communication link from the client system to the application server is subject to failures of various kinds. Depending on the type of LAN hardware, failures may occur in cables, interface cards, network routers, hubs, or concentrators.

All these single points of network failure can be eliminated by providing fully redundant LAN connections, and by configuring local switching of LAN interfaces. To protect against network adapter failure, a second network adapter would be configured to the same network backbone. If the fault tolerant cluster is designed with more than one network, two network adapters will be used for all the network backbones. For eliminating the loss of client connectivity, can also be configured redundant routers or redundant hubs through which clients can access the services of the cluster. Another way to eliminate points of failure is to configure local switching, which means shifting from a configured LAN interface card to a standby.

2.8 Eliminating Power Sources as Single Points of Failure

Different methods can be used for eliminating power sources as single points of failure. The use of multiple power circuits with different circuit breakers reduces the likelihood of a complete power outage. An uninterruptible power supply provides standby in the event of an interruption to the power source. Small local uninterruptible power supply can be used to protect individual system processor units and data disks. Large power passthrough units can protect the power supply to an entire computer system.

2.9 Eliminating Applications as Single Points of Failure

The software of a fault tolerant cluster is a layer that runs on top of local operating systems running on each computer. The cluster management software provides services such as failure detection, recovery, load balancing, and the ability to manage the servers as a single system. This high level software monitors local hardware and software subsystems, tracks the states of the nodes, and quickly responds to failures in a way that eliminates or minimizes applications downtime, and provides a number of important other benefits, including improved availability, easier manageability, and cost-effective scalability.

The critical applications and data are housed on disk devices that are physically cabled to cluster nodes. This shared physical connection allows the ownership of shared logical volumes and their contents to be quickly switched from one node to another. Load balancing technique allows the performance of a server-based program to be scaled by distributing its client requests across multiple servers within the fault tolerant cluster. The load balancing management software can specify the load percentage that it will handle, or the load can be equally distributed across all of the hosts. If a host fails, the load balancing mechanism dynamically redistributes the load among the remaining hosts. Load balancing technique is used to enhance scalability, which boosts throughput while keeping response times low.

When a host fails or goes offline, the high level software of a fault tolerant cluster automatically reconfigures the cluster to direct client requests to the remaining computers. In addition, for load-balanced ports, the load is automatically redistributed among the computers still operating, and ports with a single server have their traffic redirected to a specific host. While connections to the failed or offline server are lost, once the necessary maintenance is completed, the offline computer can transparently rejoin the cluster and regain its share of the workload.

If there is a node failure, it shuts down, and the cluster reconfigures itself; services that were on the failed node are made available on another system. There are different methods used for providing services after the shutting down of a node.

One way is to have another node that take over the applications that were running on the failed system. By using the high-level cluster software, application services and all the resources needed to support the application can be putted together into special entities called application packages. This application packages are the basic units that are managed and moved within the fault tolerant cluster. Packages simplify the creation and management of highly available services and provide outstanding levels of flexibility for workload balancing. When a package is failed over between nodes, all of the contents of the package are moved from the failed node to the new node. The ability to easily move application packages within a fault tolerant cluster provide outstanding availability during system maintenance activities such as hardware or software upgrades. Packages can be moved from node to node with simple operator commands, allowing scheduled maintenance to be performed on one node of a cluster while other nodes continue to provide support for critical applications. When the maintenance is complete, the node rejoins the cluster and assumes its normal workload of application packages.

Another approach for providing services after the shutting down of a node is to provide different instances of the same application running on multiple nodes so that when one node goes down, users need only reconnect to an alternate node.

3. Modeling and Simulation Systems, Command and Control Systems, and Fault Tolerant Clusters

The need for interoperability between the Modeling and Simulation (M&S) world and the Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) world has been formulated in several publications. The challenge even increases when NATO and PfP Nations demands to train using their own simulation systems as well as their own command and control systems. The key issue for the C4ISR community is the interoperability between live or real C4ISR systems and M&S systems.

Within the simulation community, the new and promising approach of using High Level Architecture (HLA) and Synthetic Environment Data Representation and Interchange Specification (SEDRIS) is promoted to gain interoperability and reuse of the systems. For realising the interoperability, the C4ISR community is moving to standardize the Joint Technical Architecture (JTA) and the Defense Information Infrastructure Common Operating Environment (DII COE). Unfortunately, over the last decade, uncoordinated standards for M&S-to-C4ISR interoperability have been and are currently being developed by both communities.

Today, can be identified some key words that are in common in modern modeling and simulation systems, command and control systems and in fault tolerant clusters. Some of them can be:

- Open and distributed systems.
- Networks.
- High level operating systems.
- Segments, federates (federations) and packages.
- Hierarchical architecture.
- Commercial standards, specifications, and products.

- Interoperability and reusability.
- High availability systems.

3.1 Open and Distributed Systems

All modern systems used for command and control must be open and distributed systems. They must be flexible and extensible, able to be kept up-to-date with state-of-the-art technology, and to offer the best capabilities for reuse and interoperability. The architecture of all modern fault tolerant systems is that of a cluster, which is one of the best open and distributed system.

3.2 Networks

A fault tolerant cluster is a set of independent computers (nodes) connected over a network, and always with external storage devices connected to the nodes on a common input/output bus. Clients are connected over the networks to a server application that is executing on the nodes.

The basic HLA protocol establishes that the communications path between any federates is over the network. This rigor requires substantial effort to design the models in the federate and the common functions of HLA for each federate, the interface data structure, and the message transactions or services required for this highly object-oriented architecture. The resulting architecture, however, offers the flexibility to support multiple configurations of the architecture needed for specific modeling and simulation, and command and control objectives, and the ability to sustain changes in design over the program life cycle.

3.3 High Level Operating Systems

In a fault tolerant system the nodes of the cluster are connected in a loosely coupled manner, each maintaining its own operating system. The cluster software is a layer that runs on top of local operating systems running on each computer. The high availability applications in the fault tolerant cluster run at the top level cluster software.

In the HLA the RTI is defined as a distributed operating system for federates and federations. The top-level cluster software can be a good support for RTI and for the command and control systems.

3.4 Segments, Federates (Federations) and Packages

The HLA requires that all *federates* incorporate specified capabilities to allow the objects in the simulation to interact with objects in other simulations through the exchange of data supported by services implemented in the RTI. The RTI is a distributed operating system for the federation which provide a set of general purpose services that support federate-to-federate interactions and federation management and support functions. The basic components of the HLA are the simulations themselves, or more generally, the federates.

In DII-COE-based systems, all software and data are packaged in self-contained units called *segments*. This is true for COE infrastructure software and for mission-application software as well. Segments are defined in terms of the functionality they provide, not in terms of ”modules,” and may in fact consist of one or more ”modules.” They are defined as a collection of related functions as seen from the perspective of the end user, not the developer.

By using the high-level cluster software, application services and all the resources needed to support the application can be putted together into special entities called application *packages*. This application packages are the basic units that are managed and moved within the fault tolerant cluster. Packages simplify the creation and management of highly available services and provide outstanding levels of flexibility for workload balancing. When a package is failed over between nodes, all of the contents of the package are moved from the failed node to the new node.

3.5 Hierarchical Architecture

HLA simulations are made up of a number of HLA federates and are called federations. Simulations that use the HLA are modular in nature allowing federates to join and resign from the federation as the simulation executes.

Based on functions provided by specific subsystems, all fault-tolerant clusters are partitioned at several levels, but in addition it contains redundant components and recovery mechanisms which may be employed in different ways at different levels. Recovery from a fault within a redundant partition may be effected within the domain itself, or may require action by higher levels within the system.

At top of any fault tolerant cluster, command and control, and HLA compliant system there is a distributed operating system that runs on top of local operating systems running on each computer or on top of federates and federations.

3.6 Utilize Existing Commercial Standards, Specifications, and Products

The commercial marketplace generally moves at a faster pace than the military marketplace and advancements are generally available at a more rapid rate. Use of commercial products has several advantages. Using already built items lowers production costs. The probability of product enhancements is increased because the marketplace is larger. The probability of standardization is increased because a larger customer base drives it.

3.7 Interoperability and Reusability

There is a need for software components to be able to communicate with each other using ”standard” mechanisms and ”open” interfaces for an effective integration to occur. As software and hardware systems get more complex, the need for interoperability among different components becomes critical.

The HLA can be conceptual ”software bus” that allow applications to communicate with one another, regardless of who designed them, the platform they are running on, the language they are written in, and where they are running. HLA also enables the building of a plug-and-play component software environment.

The fault tolerant cluster can offer a good architecture for command and control systems and HLA compliant systems to work with these applications. The fault tolerant cluster is an open and distributed system, flexible and extensible, and its architecture offer compliance to the principle of reusability and interoperability.

3.8 High Availability Systems

A fault tolerant solution improves data and applications availability by allowing that a number of nodes to share the same hard disks within a cluster. When a node in the cluster fails, the fault tolerant cluster software will recover and disperse the work from the failed node to another node within the cluster. As a result, the failure of a system in the cluster will not affect the other systems, and in most cases, the client applications and data will be completely unaware of the failure.

In C4ISR systems information related to the battlespace is complex and dynamic and must flow quickly among all tactical, strategic, and supporting elements. There is an unprecedented increase in the amount of information necessary to conduct operational planning and combat decision-making. For the command and common systems and for the modeling and simulation systems high availability of data and applications is very important.

High availability and security must be designed into the architecture. The increasing importance placed upon system high availability and security has underscored the need to view high availability and security as an engineering discipline. High availability and security considerations must be addressed throughout the entire system life cycle from requirements analysis through maintenance.

4. Conclusion

Fault tolerance is the best guarantee that the systems will be available, and the essential services will be offered in real-time to the users. The modern fault tolerant clusters are able to eliminate all single points of failure in the nodes of the cluster, the disk used to store applications or data, the networks, the power sources, the data, and the applications and to offer the best high availability architecture for command and control systems, and modeling and simulation systems.

5. References

[1] Serb, A.:: Sisteme tolerante la defectari, Military Technical Academy Publishers, Inc., Bucharest, 1996

[2] Patterson, D. A., Hennessy, J. L.: Computer Organization & Design. The Hardware/Software Interface, Morgan Kaufmann Publishers, Inc., San Francisco, California, U.S.A., 1998

[3] Weygant, P.: Clusters for High Availability. A Primer of HP-UX Solutions, Prentice Hall Pt., Upper Saddle River, New Jersey, U.S.A., 1996

[4] High Level Architecture Interface Specification, v1.3, Defense Modeling and Simulation Office, 5 February 1998, http://hla.dmso.mil/hla/tech/ifspec/if1-3d9b.doc

[5] High Level Architecture Object Model Template, v1.3, Defense Modeling and Simulation Office, 5 February 1998, http://hla.dmso.mil/hla/tech/omtspec/omt1-3d4.doc

[6] High Level Architecture Rules, v1.3, Defense Modeling and Simulation Office, 5 February 1998, http://hla.dmso.mil/hla/tech/rules/rules1-3d2b.doc

[7] Joint Technical Architecture. Version 4.0 Draft 1, Department of Defense, 14 April 2000

[8] Serb, A.: Fault Tolerance in Systems Used for Computer Assisted Exercises, NATO's Research & Technology Organization PfP Symposium on Computer Assisted Exercises for Peace Support Operations, The Hague, the Netherlands, 28-30 September 1999.

[9] Serb, A.: Using of Fault Tolerant Distributed Clusters in the Field of Command and Control Systems, NATO's Research & Technology Organization PfP Symposium ”New Information Processing Techniques for Military Systems”, Istanbul, the Turkey, 09-11 October 2000.