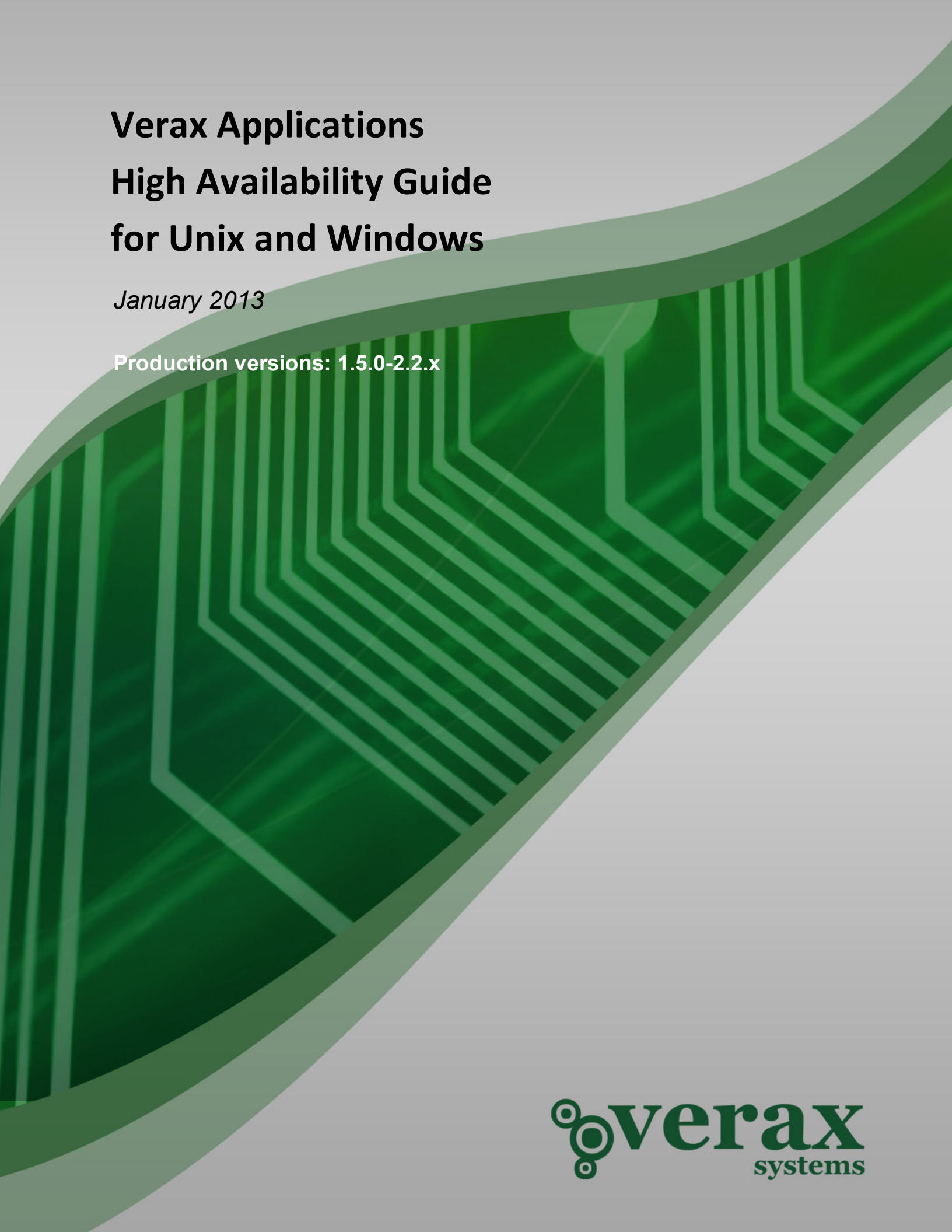
Verax Applications High Availability Guide for Unix and Windows

January 2013

Production versions: 1.5.0-2.2.x



Contact Information:

E-mail: office@veraxsystems.com
Internet:http://www.veraxsystems.com

Technical support:

E-mail: support@veraxsystems.com

CONFIDENTIAL AND PROPRIETARY INFORMATION

TABLE OF CONTENTS

How to use this guide?

This guide contains information on advanced installation and configuration options for Verax products for building **high availability, fault tolerance** and **performance** installations.

Purpose and scope

The guide contains information on advanced installation and configuration steps. For details on basic installation, please refer to the Verax application installation guides.

Notation used

Source code, commands, user-entered data, on-screen messages and user interface elements (menus, choice lists, etc.) are shown using Courier font. In order to improve readability, indentation has been used, for instance:

```
ls -al
```

! **This notation (Information) is used to indicate important information.**

This notation (Warning) is used to flag actions that can lead to the loss of data, system malfunction, etc.

This notation (Hint) is used to indicate additional information.

The following symbols are used to flag information pertaining to a particular type of operating system:

Linux

Oracle Solaris

Microsoft Windows

IBM AIX

Intended audience and guide overview

This installation guide is intended for system administrators or other IT personnel responsible for Verax products installation.

The guide consists of the following chapters:

- **Chapter 1. Introduction:** contains general product information required for the installation.
- **Chapter 2. Designing OSS hardware configuration:** provides introductory information on how to determine hardware configuration and sizing for running Verax applications.
- **Chapter 3. High availability:** describes how to configure Verax applications to run in high-availability (HA) mode.
- **Chapter 4. Scalability:** provides information on how to use hardware clustering in order to increase performance and capacity of the system.
- **Chapter 5. Virtualization:** provides information on how to use virtualization in order to maximize the utilization of the IT infrastructure.
- **Chapter 6. Failover:** provides information concerning the procedure that involves automatically offloading tasks to a standby system component so that the procedure is as seamless as possible to the end user.

Please refer to the Verax applications on-line help for additional information (press the F1 key in application to display the on-line help).

1 Introduction

Verax products are built according to a three-tier application design principle, including:

- **Presentation tier** (a.k.a. the front-end) responsible for presenting the user interface. Verax applications run in a web browser and are built using the industry-standard HTML and Adobe Flex technologies in order to provide the best operator productivity and user experience.
- **Business logic** tier (a.k.a. the back-end) consisting of services and batch processes. Services constitute the interface to the system (for instance, there are separate services for user management, security, event logging, NMS alarms, etc.). Verax applications interface with the business logic exclusively via services. Application/service interaction is achieved via a BlazeDS service bus. Applications services are running in the application server context (Verax applications use Apache Tomcat server by default, however, other types of Java application servers (such as IBM WebSphere) can be used. (For further information, please contact Verax Systems' technical support). Batch processes (such as alarm aggregation counters) are non-interactive processes which run without user interface under the control of the Batch Manager service.
- **Data tier** which provides database access. Verax applications use both object-oriented access and direct relational access. The direct relational access is used where performance issues are critical (performance data aggregation, substantial data inserts, etc.).

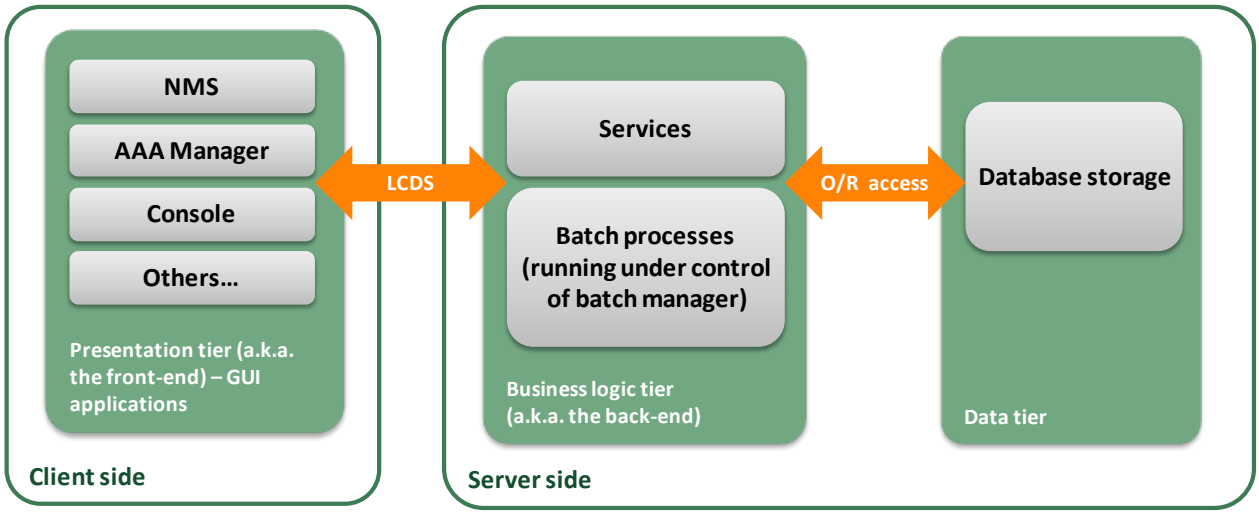


Figure 1: Verax applications architecture.

2 Designing OSS hardware configuration

While designing the hardware architecture to run Verax applications, the following items have to be considered:

- **Availability** – what is the required availability level? For some Verax applications (such as billing) the 24/7 availability is not essential; others, like the Self-Care Portal or NMS, must be up at all times. In general, availability and cost are in contradiction, and some tradeoffs must be made, for instance:
    - Can 100% redundancy be afforded, as additional hardware and software licenses have to be purchased?
    - Should HA be only applied to mission-critical applications (or their parts)?
    - Should redundant hardware be located in a single data center or two remote ones?

  See also section 3 on how to implement High Availability (HA) Verax applications installations.
- **Performance** – what are the required performance levels? Should a single, multi CPU server be used, or should a hardware cluster configuration be built? See also section 4 for details on how to build high performance Verax applications installations.
- **Utilization of hardware** – software systems typically have processing "peaks-and-valleys" which cause the hardware to be idle for extended periods of time. Maximizing utilization of hardware (and thus ROI) is a costly issue. It can be achieved by using virtualization, process priorities and design in which components can be collocated within a single host or virtual machine.

- **Growth strategy** – how to make the system saleable with growing numbers of subscribers, managed elements and services without having to replace the hardware overtime? Virtualization and clustering (described in sections 4 and 5 respectively) provide the means of addressing growth problems.
- **Hardware and software costs** –most database systems are licensed per CPU (or a core in the CPU). The investment into a high-end database server might be more cost-effective than using two low-end database servers in a clustered configuration.

To determine the best configuration for your business, please contact Verax Systems' technical support.

3 High availability

This chapter describes how to build Verax applications in high availability mode, focusing on maximizing uptime and availability, rather than system performance (see also section 4). Additionally, it focuses on solutions for each hardware platform as recommended by the hardware and operating system vendor.

! **Please note that these are only exemplary configurations as built and tested by Verax Systems. Other configurations may also be applicable to specific solutions.**

All scenarios described in the subsequent sections assume HA configurations located in a single data center. For high-end applications however, for maximum availability, the HA should be distributed across a number of data centers. A sample schema of HA with backup data center connected by two fiber connections is presented in Figure 2 below.

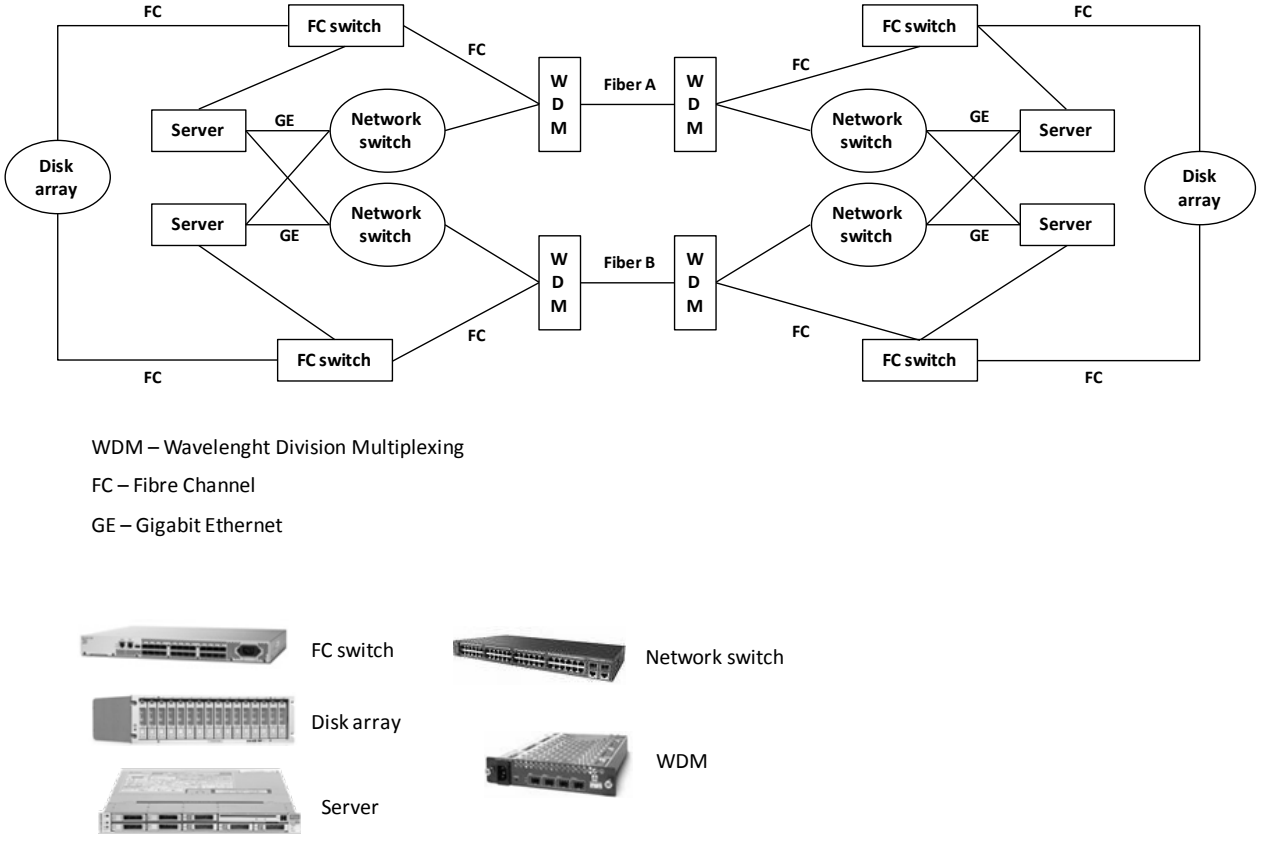


Figure 2: HA configuration with backup data center.

The benefits of such an approach include:

- Pay-as-you-grow scalable system,
- Maximum availability,
- Redundant connections to all devices and Internet,
- Scalability.

3.1 IBM AIX

AIX

Verax applications use IBM PowerHA to ensure high-availability on IBM Power Systems. IBM PowerHA is a High Availability solution for AIX, Linux and i5/OS systems provided by IBM running on the IBM Power platform. Please note that only AIX configuration is described in this document – for more information on other IBM operating systems, please contact Verax Systems' technical support.

PowerHA provides reliable monitoring, failure detection and automated recovery of business application environments to backup resources, allowing administrators to:

- Determine the primary node an application should start on.
- Select backup node(s) and their priorities to move application onto if the primary node fails.
- Manage users and groups across the cluster nodes, keeping user and group accounts and identifiers consistent across the cluster. This ensures that applications are run properly on different nodes.
- Perform configuration management, such as reverting to the last known good configuration.
- Configure restore policy, for instance if a higher priority node comes back online after a failure.

3.1.1 Supported platforms

Software. The following versions of AIX are required to run PowerHA:

- AIX V5.2
- AIX V5.3
- AIX V6.1

ⓘ In order to check the AIX version on your system use the following command:

```
oslevel -g
```

Detailed software requirements for running PowerHA are provided by IBM:

http://www.ibm.com/common/ssi/cgi-bin/ssialias?infotype=dd&subtype=sm&appname=pseries&htmlfid=897/ENUS5765-F62#SHeader_10

Hardware. PowerHA V5 works with virtually any IBM system capable of running the AIX operating system. **Special attention** has to be paid to the storage system, since PowerHA has limited storage compatibility.

The platforms and storage supported by PowerHA are provided by IBM:

http://www.ibm.com/common/ssi/cgi-bin/ssialias?infotype=dd&subtype=sm&appname=pseries&htmlfid=897/ENUS5765-F62#SHeader_9

3.1.2 Exemplary configuration

This section provides a description of an exemplary AIX-based HA configuration tested in the Verax Systems' labs.

The following Verax applications environment has been used:

- Operating system: AIX 5.3.
- Database system: Oracle Database 10 release 2.
- Web server: IBM WebSphere and Apache Tomcat 6.0 (interchangeably).

The table below contains an inventory of elements required to build an exemplary AIX-based HA system:

Configuration elements	
Nodes	Two IBM System p5 505 nodes with the following configuration: • Processors: two 2.1 GHz 64-bit POWER5+ • Main Memory: 16 GB RAM • Internal disk: 300GB 10,000 RPM Ultra320 SCSI • Host Bus Adapter: 4 Gb Single-Port Fibre Channel PCI-X 2.0 DDR Adapter
Network switch	Cisco Catalyst 3750-E Series
Storage	FC Switch: McData Sphereon 4300 Fibre Channel Switch External disk: IBM System Storage DS3400

ⓘ The actual RAM, CPU and storage requirements (sizing of the system) should be estimated based on particular customer requirements – the configurations listed below are to illustrate HA configuration only, and are not to be treated as sizing recommendations.

The topology of the HA configuration is presented in Figure 3 below:

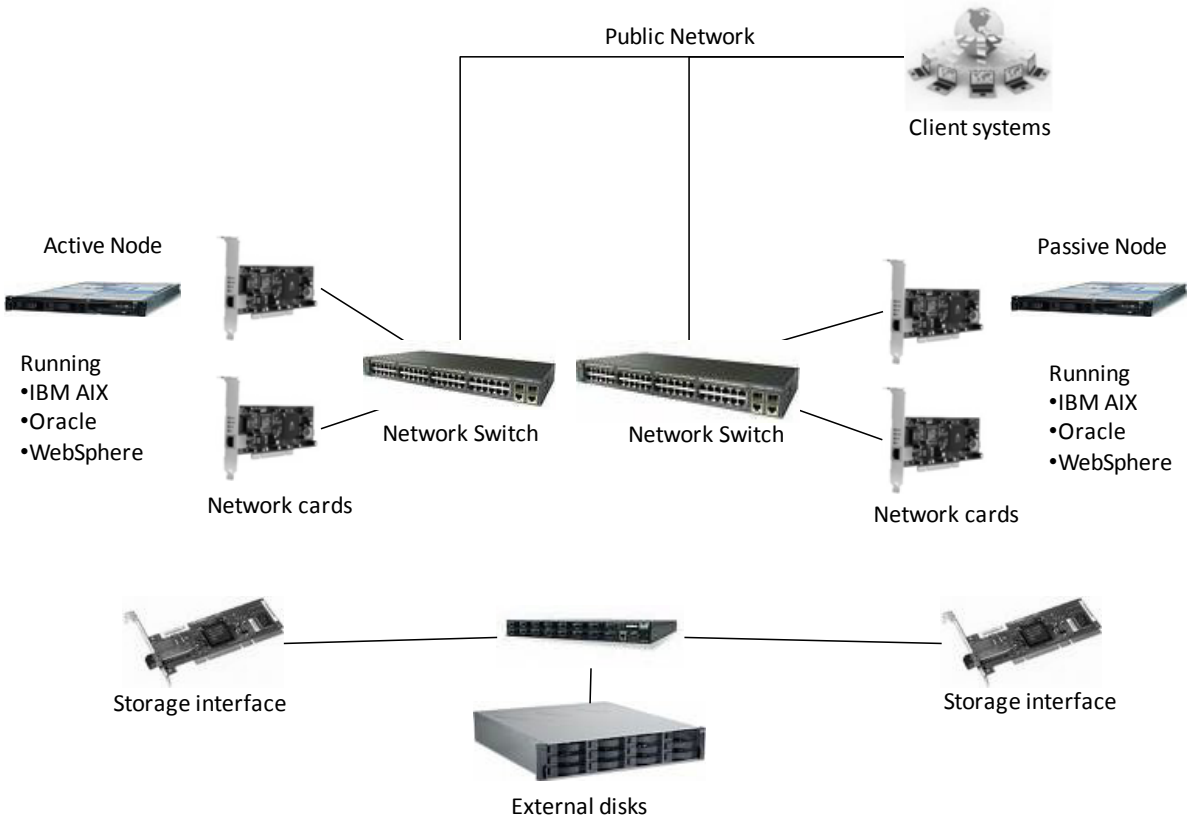


Figure 3: IBM AIX based HA configuration for Verax applications.

How does it work?

If an application failure is detected on an active node, the PowerHA system analyzes, isolates, and identifies the failing component. First, the PowerHA software checks if the cause can be repaired dynamically on the primary node (this takes place through user defined algorithms for software problems, or via standard service practices for hardware problems). If the active node cannot be repaired, the backup node takes the active node's IP address and starts the application (possibly with other, accompanying services such as DNS). When the active node comes on line, this action is reverted (depending on the administrator-defined policy).

3.2 Oracle Solaris

SOLARIS

Verax applications HA configuration on Oracle Solaris is based on the Sun Microsystems' High Availability Cluster product (HAC). Sun HAC is a high availability clustering solution that uses software **agents** which monitor applications' health and take remedy actions if any problem is detected.

Detailed information on the Sun HAC can be found at:

- http://www.sun.com/software/solaris/cluster/index.xml
- http://www.sun.com/software/solaris/howtoguides/twonodecluster.jsp

Sun provides pre-packaged agents for a multitude of applications including:

- Sun's Java applications such as: System Application Server, System Message Queue, Directory Server and others.
- Sun's native applications and services such as: N1 Grid Engine, Sun N1 Service Provisioning System, NFS, DNS, DHCP and others.
- Third party applications (including open source) such as: Apache Web Server, SAMBA, MySQL Apache Tomcat, Oracle (BEA) WebLogic, IBM WebSphere MQ, IBM DB/2, MySQL, Oracle (HA and RAC), Oracle Application Server, SAP and others.

For other applications not supported by Sun directly, custom agents can be created.

3.2.1 Supported platforms

Software. Sun HAC is supported on Solaris version 8 or higher.

ⓘ In order to determine your Solaris system version, use the **`uname -a`** or cat **`/etc/release`** commands.

Hardware. Please refer to the table below for HAC hardware compatibility matrix.

Sun HAC supported platforms list	
Solaris on x86 and x64	Sun Fire V20z and V40z servers Sun Fire X2100 M2, X2200, X4100, X4100 M2, X4140, X4150, X4200, X4200 M2, X4240, X4250, X4440, X4450, X4540, X4600, X4600 M2 servers Sun Blade X6220, X6250, X6450, X8400, X8420, and X8450 server modules Netra X4200, X4250, X4450
Solaris on SPARC	Netra 20, 120, 210, 240, 440, 1280, 1290, T 1120/1125, T1 AC200 & DC200, T 1400/1405, T2000, T5220, T5440, CP3010, CP3060 Sun Enterprise 220R, 250, 420R, 3500, 4500, 5500, 6500, 10000 Sun Enterprise T1000, T2000, T5120, T5140, T5220, T5240, T5440 Sun Enterprise M3000, M4000, M5000, M8000, M9000-32, M9000-64 Sun Fire V120, V125, V210, V215, V240, V245, V250, V440, V445, V480, V490, 280R, V880, V890, V1280, V1290, 3800, 4800/4810, 6800 Sun Fire E2900, E4900, E6900, E12K, E15K, E20K, E25K Sun Fire T1000/T2000 Sun Blade T6300, T6320, T6340
Sun Storage and third party storage	Fiber Channel A3500(FC),A5000, A5100, A5200, T3 2440 RAID, 6140, 6540, 3510 RAID, 3511 RAID, 3910/3960, 6120, 6130, 6320, 6910/6960, 6920, 9910/9960, 9970/9980, 9985/9990, 9985V/9990V SCSI Netra st D130, st A1000, st D1000 arrays S1, D2, A3500 (SCSI), A1000, D1000, 3120 JBOD, 3310 JBOD, 3310 RAID, 3320 JBOD, 3320 RAID, 2530 RAID NAS 5220, 5320 List of third party storage can be found here: http://www.sun.com/software/cluster/osp/index.xml

! For the most up to date compatibility information, please contact your Sun Microsystems representative.

3.2.2 Exemplary configurations

This section contains exemplary HA configurations for Verax applications tested in the Verax Systems' labs.

3.2.2.1 Exemplary configuration

The configuration below presents a simple, cost effective HA configuration for the Solaris/Oracle instance of Verax applications. The following environment has been used:

- Operating system: Solaris 10.
- Database system: Oracle Database 10 release 2.
- Web server: Apache Tomcat 6.0.

The table below contains an inventory of elements required to build an exemplary Solaris-based HA system:

Configuration elements	
Nodes	Two Sun Fire X2200 nodes with the following configuration: • Processors: two AMD Opteron Quad-Core 2.3 GHz • Main Memory:16 GB RAM • Internal disk: 250 GB 7200 rpm 3.5" • Host Bus Adapter: Sun StorageTek 8 Gb Fibre Channel PCIe
Network switch	Cisco Catalyst 3750-E Series
Storage	FC Switch: Sun Storage FC Switch 5802, External disk: Sun StorageTek 6140 Array.

ⓘ The actual RAM, CPU and storage requirements (sizing of the system) should be estimated based on particular customer requirements – the configurations listed below are to illustrate HA configuration only, and are not to be treated as sizing recommendations.

The topology of the HA configuration is presented in Figure 4 below:

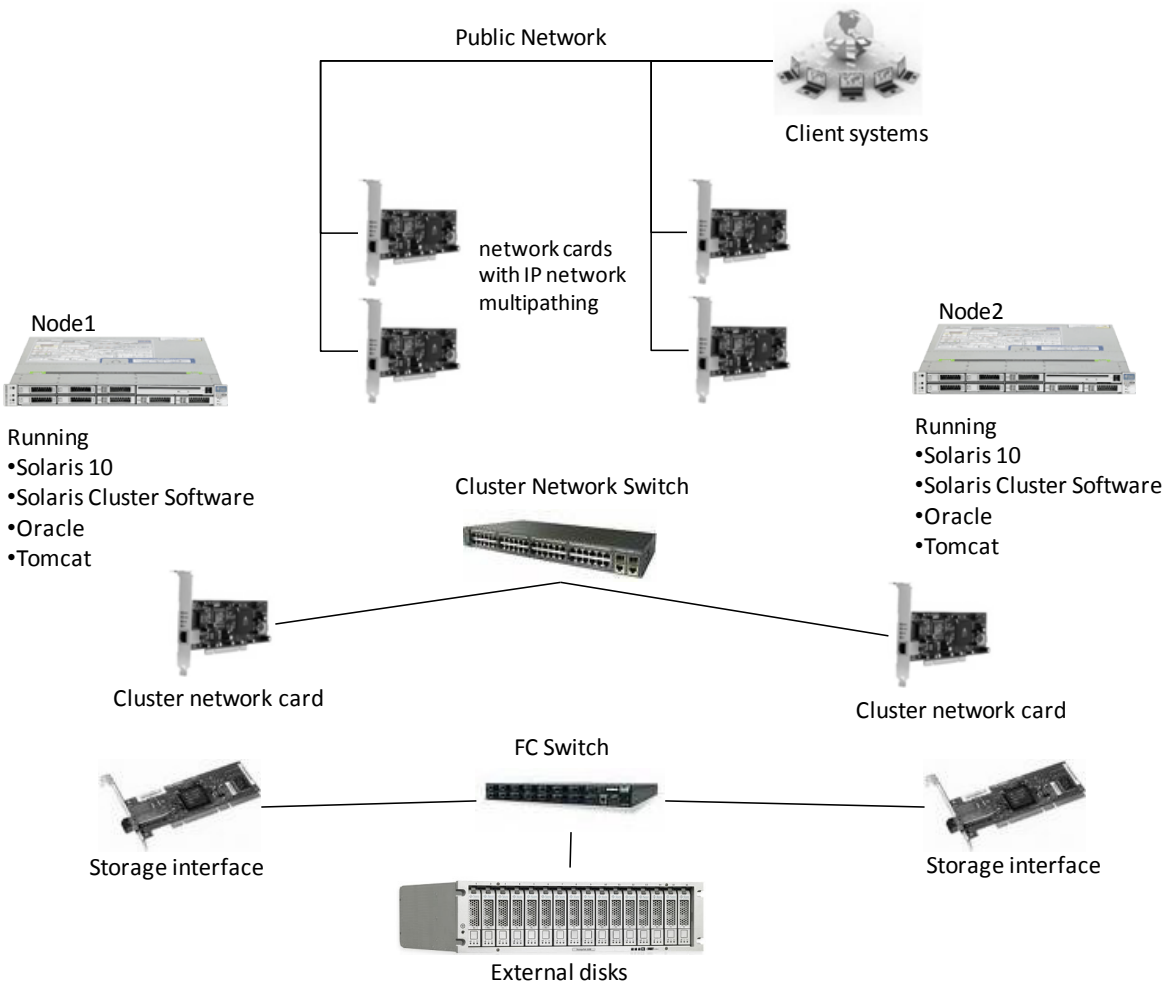


Figure 4: Oracle Solaris based HA configuration for Verax applications.

How does it work?

When one of the nodes in the cluster goes offline (the status of the node is determined by HAC application monitoring agents and hardware heartbeat), the applications are started on backup nodes (the IP address is transferred using network adapters with a multipathing feature). This entire process is quick and completely transparent to users of the application. Changes are reverted when the failed node comes back on-line.

3.2.2.2 Using Oracle RAC

In order to fully utilize the power of Solaris Cluster in conjunction with Oracle database, Oracle Real Application Cluster (RAC) should be used. Oracle RAC allows multiple computers to run the Oracle RDBMS software simultaneously while accessing a single database, thus providing a clustered ("virtual") database. The benefits of Oracle RAC include increased scalability, fault tolerance and load balancing of Oracle databases. Running Oracle RAC causes the database to run faster and safer on multiple servers. Additionally, in case the traffic increases, the Oracle RAC system can distribute the load over a cluster of servers.

The topology of the HA configuration (identical to the one presented in section 3.2.2.1) is presented in Figure 5 below:

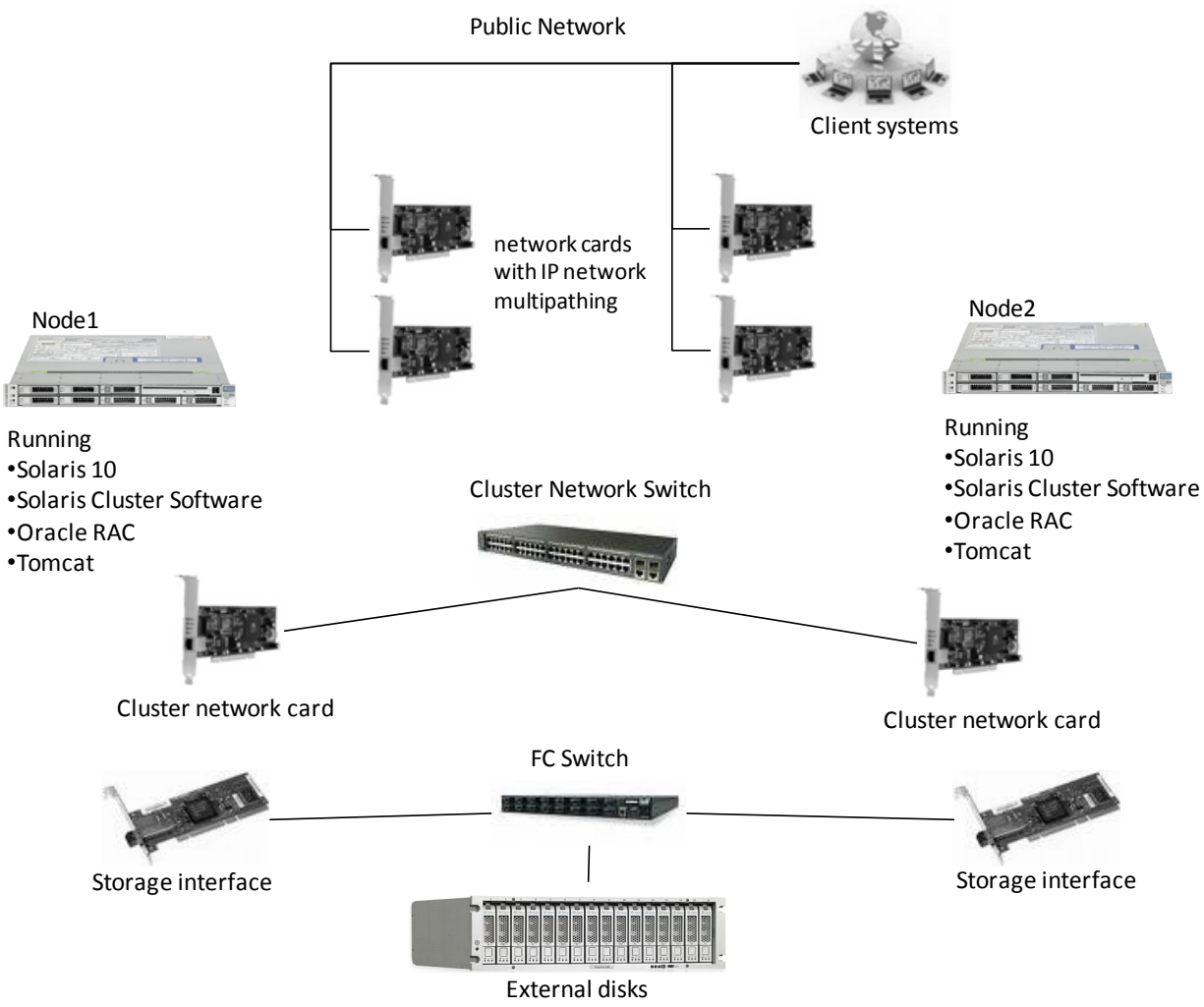


Figure 5: Oracle Solaris based HA configuration using Oracle RAC
(the backup node is used to increase database performance during normal operation).

3.3 Linux

LINUX

There is a multitude of High Availability solutions for Linux and its specific distributions. Two solutions have been successfully tested in the Verax Systems' labs: open-source Linux-HA and commercial IBM Tivoli System Automation.

ⓘ For the most up-to-date information on other HA solutions for Linux, please contact Verax Systems' technical support.

3.3.1 Open source HA

Linux-HA is an open source project providing a Linux-based High Availability framework. It is the most mature and best-tested open source HA solution. Detailed information on Linux-HA can be found at: http://www.linux-ha.org/.

3.3.1.1 Supported platforms

Software. Linux-HA runs on virtually every Linux distribution. In order to check if your distribution is supported, please refer to http://www.linux-ha.org/.

Hardware. Linux-HA runs on many hardware architectures supported by Linux including: i386, x86_64, ia64, IBM pSeries, IBM zSeries mainframes and others. Linux-HA supports all storage devices which are **supported natively** by the running system.

3.3.1.2 Sample configuration

The following environment has been used:

- Operating system: SUSE Linux 11.
- Database software: Oracle Database 10 release 2.
- Application server: Apache Tomcat 6.0.

The table below contains an inventory of elements required to build the configuration:

Configuration elements	
Nodes	Two HP ProLiant DL360 G5 nodes with the following configuration: • Processors: two Intel Xeon Quad-Core 3,33 GHz • Main Memory:16 GB RAM • Internal disk: 300 GB 10000 rpm SAS • Host Bus Adapter: HP Storage Works PCI-e U320
Network switch	Cisco Catalyst 3750-E Series
Storage	External disk: HP Storage Works Disk System 2120

The topology of the HA configuration for Linux using Linux-HA is presented in Figure 6 below:

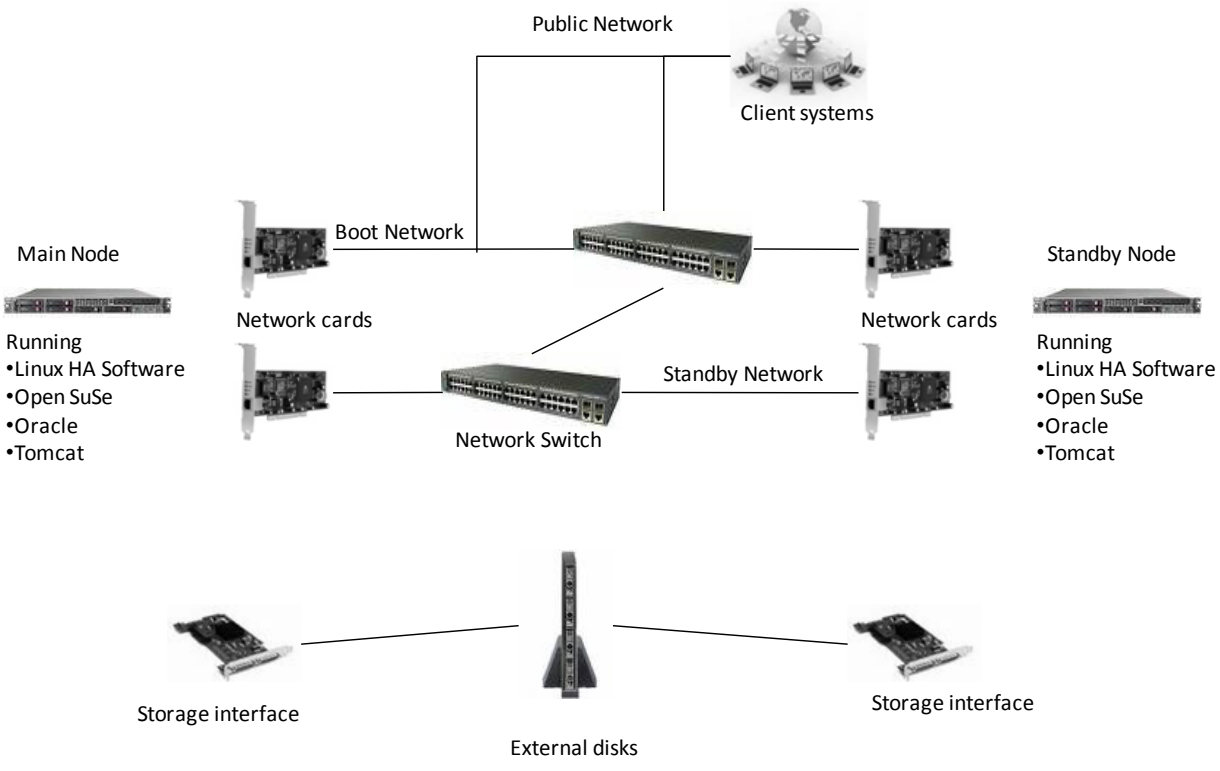


Figure 6: Linux HA configuration.

How does it work?

Linux-HA configuration depends on the so-called "heartbeats". The heartbeat daemon sends packets across the network. When heartbeat packets are not received, the sending node is assumed to be dead, and any services it was providing are started to the other node. The IP address of the failed node is taken over by the other node. There are three kinds of strategies for a takeover:

- IP address takeover,
- MAC address takeover,
- Dynamic DNS reconfiguration.

Linux-HA is the simplest and most cost-effective of all the solutions described in this manual.

3.3.2 Commercial Linux HA solution – Tivoli System Automation

IBM Tivoli System Automation (TSA) is a high-end HA solution from IBM, allowing management of complex groups of interrelated resources across multiple nodes. TSA provides recovery algorithms and uses the same automation engine which powers IBM's z/OS-run mainframes, used for most critical applications worldwide. TSA allows moving of individual resources to a new cluster node without affecting other resources currently running on the node. It initiates, executes and coordinates starting, stopping and failing over of individual applications. TSA provides a standard toolset supporting failover scenarios involving physical and virtual environments (for instance, various virtualization topologies including physical to physical, virtual to virtual, virtual to physical and physical to virtual failover scenarios). More information about Tivoli System Automation can be found at:

http://www.ibm.com/software/tivoli/products/sys-auto-multi/.

ⓘ The IBM Tivoli System Automation supports not only Linux systems but also Windows and AIX; however, the information presented in this chapter is Linux-specific.

3.3.2.1 Supported platforms

Software. SUSE Linux and Red Hat Enterprise Linux.

Hardware. Please refer to the table below for TSA hardware compatibility matrix.

IBM Tivoli System Automation supported platforms list	
Intel-based	• IBM xSeries BladeCenter HS20 and HS40 • Any 32-bit Intel-based server • Any 64-bit AMD64-based Server
IBM pSeries	Please refer to http://www.ibm.com/systems/p/hardware
IBM pBladeCenter	Please refer to http://www-03.ibm.com/systems/bladecenter/index.html
IBM zSeries	IBM S/390 Parallel Enterprise Server G5/G6, Multiprise 3000 and all IBM System Z processors.

The detailed list of Tivoli System Automation requirements can be found at:

http://www-01.ibm.com/software/tivoli/products/sys-auto-linux/platforms.html?S_CMP=rnav

3.3.2.2 Exemplary configuration

The configuration provided in this section is heterogeneous (i.e. mixed AIX and Linux based) and consists of two nodes:

- Node 1 (Linux):
  - Operating system: SUSE Linux 11.
  - Database software: Oracle Database 10 release 2.
  - Application server: Apache Tomcat 6.0.
- Node 2 (AIX):
  - Operating system: AIX 5.3.
  - Database software: Oracle Database 10 release 2.
  - Application server: Apache Tomcat 6.0.

The table below contains an inventory of elements required to build the configuration:

Configuration elements	
Nodes	Linux node: HP ProLiant DL360 G5 with the following configuration: • Processors: two Intel Xeon Quad-Core 3,33 GHz • Main Memory:16 GB RAM • Internal disk: 300 GB 10000 rpm SAS • Host Bus Adapter: HP Storage Works PCI-e U320 AIX node: IBM System p5 505 with the following configuration: • Processors: two 2.1 GHz 64-bit POWER5+ • Main Memory: 16 GB RAM • Internal disk: 300GB 10,000 RPM Ultra320 SCSI • Host Bus Adapter: IBM Ultra320 SCSI Controller
Network switch	Cisco Catalyst 3750-E Series
Storage	External disk: HP Storage Works Disk System 2120

ⓘ The actual RAM, CPU and storage requirements (sizing of the system) should be estimated based on particular customer requirements – the configurations listed below are to illustrate HA configuration only, and are not to be treated as sizing recommendations.

The topology of the HA configuration for Linux using Tivoli System Automation is presented in Figure 7 below:

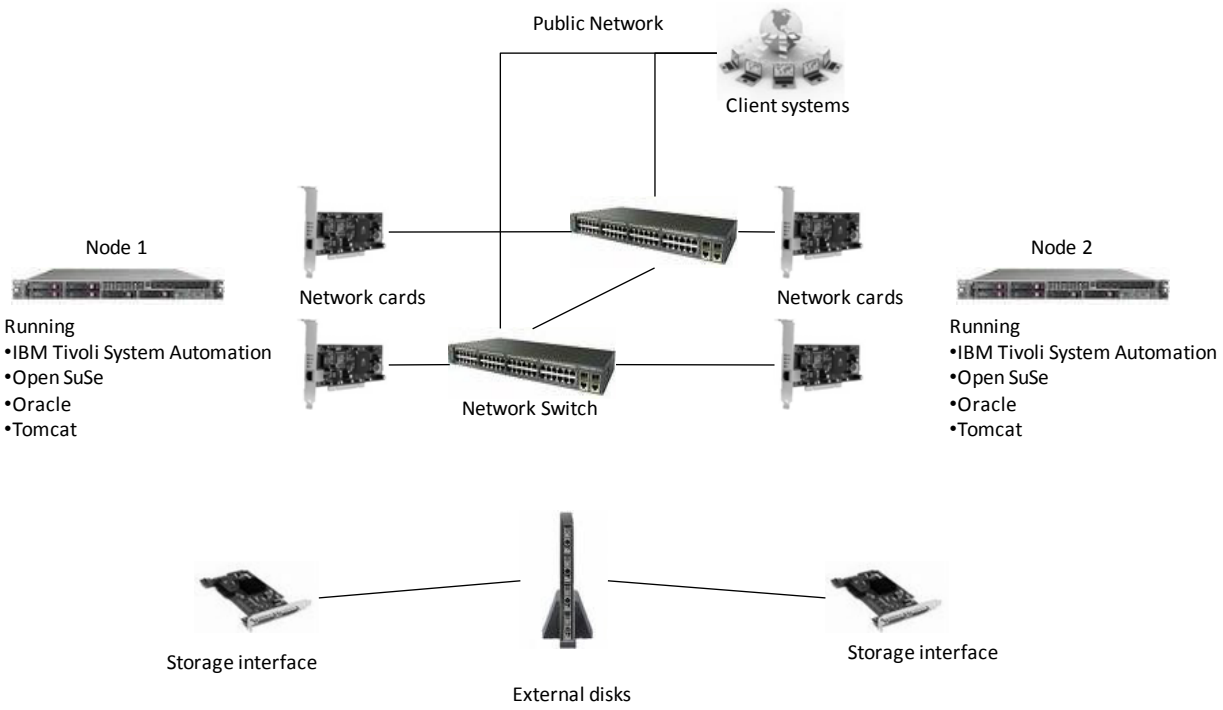


Figure 7: Mixed AIX and Linux HA configuration using IBM Tivoli System Automation software.

How does it work?

When a node is down all of the failed applications migrate to a free node(s). High availability can be provided across heterogeneous operating systems supported by IBM Tivoli System Automation software. The whole process is fully automated: automation software manages the availability of applications using algorithms configured by the system administrator. Depending on the configuration, when a node is down, all applications hosted on the node change the state to offline or are migrated to other nodes. These algorithms allow to use the full potential of hardware similarly to virtualization techniques – when resources are not available, non critical applications can be temporarily disabled. The system also allows for priority-based resource allocation.

3.4 Windows

WIN

Verax System recommends using native Windows Server HA mechanisms to run Verax applications in the Windows environments. The Windows Server 2008 Enterprise and Windows Server 2008 Datacenter environments are frequently used by tier-2 and smaller operators. Before building the cluster, all hardware devices should be checked for compatibility by the validation wizard which performs compatibility checks determining if a given system, storage, and network configuration is suitable for a clustered environment. The following requirements must be met to build a Windows cluster:

- All the servers in a cluster must run the same operating system version and software updates.
- Cluster networks must be redundant.
- Storage elements must support specific SCSI commands.

Detailed information on Microsoft Windows HA solution can be found at:
http://www.microsoft.com/windowsserver2008/en/us/high-availability.aspx.

Exemplary configurations can be found at:
http://www.microsoft.com/windowsserver2008/en/us/high-availability.aspx.

4 Scalability

4.1 Software clustering

Verax applications have a built-in software cluster capability, meaning that each component (e.g. an application or a background task) can be executed on a remote (worker) node (the node may be either a physical host or a virtual machine). The sole purpose of this mechanism is to increase performance (e.g. using a set of low end machines); it does not provide high availability features.

The built-in software clustering installation is described in the Verax Applications Installation Guide. The hardware clustering (increasing both performance and availability) is described in section 4.2.

4.2 Hardware clustering

Hardware clustering (sometimes called operating system clustering) is a hardware-based method of turning multiple servers into a cluster. As a rule, a hardware cluster is created by installing a number of servers on the machine that will control the cluster. Each of the servers functions independently of the others, although they all respond to the same requests. The operating system of the controlling server is responsible for monitoring the cluster and performing administrative tasks, such as deciding when failover is necessary and assigning the load of a failed node to a functioning server.

The hardware clustering was used in section 3 in order to describe HA configurations. Below are sample configurations (similar to the HA ones) increasing availability and performance.

Subsequent sections contain exemplary hardware clustering configurations for major operating environments ordered from the simplest (Linux) to the most complex (AIX).

4.2.1 Linux

LINUX

The configuration described in this section consists of four Linux nodes running a complex tier-2 operator NMS system. Here, IBM Tivoli is used to achieve hardware clustering; however, alternative solutions may also be applied (including open source). The processing is divided into the following nodes:

Node 1 – Runs the system's administrative console (AC) and the network management system (NMS). This part of the system is mainly responsible for presenting the user interface to the Network Operation Center (NOC) personnel. The node runs the following software:

- Operating system: SUSE Linux 11.

- Web server: Apache Tomcat 6.0.
- IBM Tivoli System Automation software.
- Verax resources: Administration Console (AC), Network Management System (NMS).

Node 2 – Runs a business reporting engine. The node runs the following software:

- Operating system: SUSE Linux 11.
- Web server: Apache Tomcat 6.0.
- IBM Tivoli System Automation software.
- Verax resources: Batch Manager (BM) for business reports.

Node 3 – Alarm processing node. The node is responsible for alarm collection and processing logic. Since the processing logic can be very complex (from handling volume data during alarm storms, to implementing alarm correlation, root-cause analysis and other data-intensive operations), it is delegated to a separate node in the cluster. The node runs the following software:

- Operating system: SUSE Linux 11.
- Web server: Apache Tomcat 6.0.
- IBM Tivoli System Automation software.
- Verax resources: Batch Manager (BM) for alarm collection processing.

Node 4 – Database node running the Oracle database.

The topology of the configuration is presented in Figure 8 below:

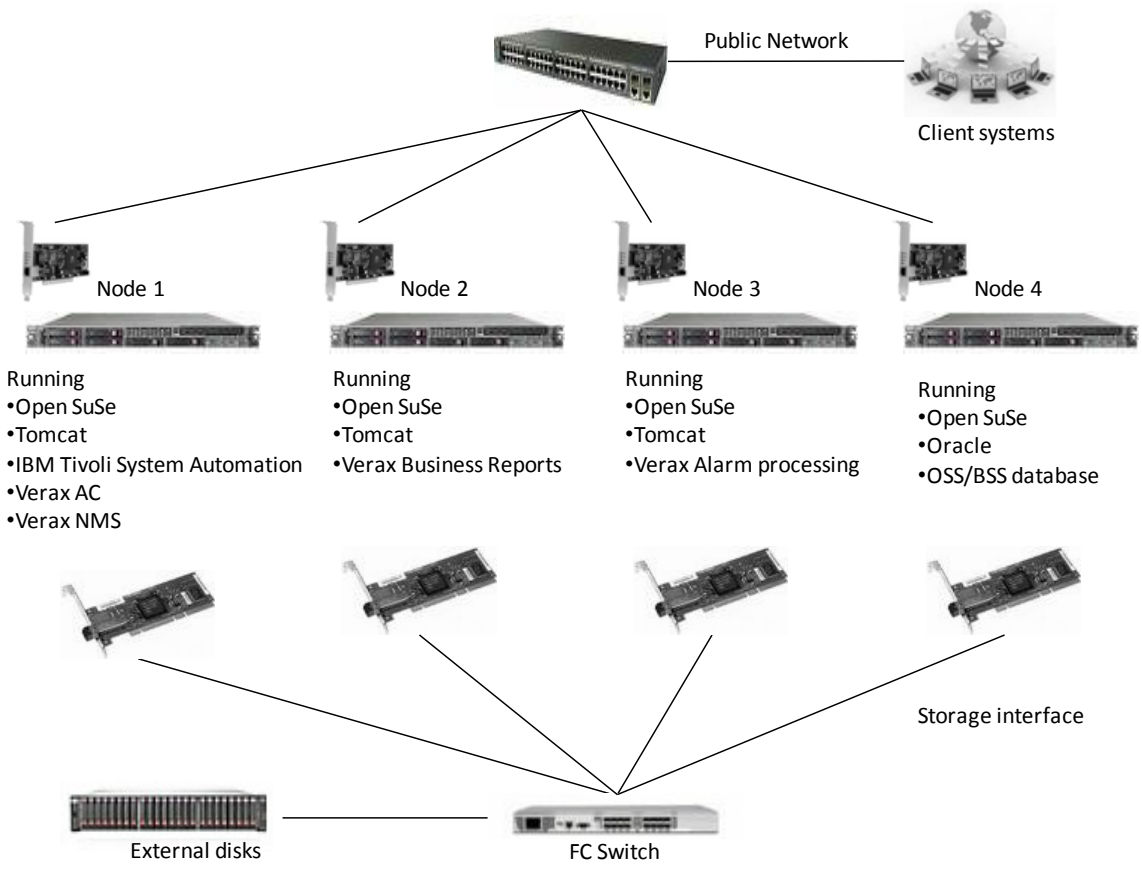


Figure 8: Linux-based hardware cluster configuration for Verax NMS using IBM system Automation.

The table below contains an inventory of elements required to build the configuration:

Configuration elements	
Nodes	Four HP ProLiant DL360 G5 nodes with the following configuration: • Processors: two Intel Xeon Quad-Core 3,33 GHz • Main Memory:16 GB RAM • Internal disk: 300 GB 10000 rpm SAS • Host Bus Adapter: HP Storage Works PCI-e 4GB
Network switch	Cisco Catalyst 3750-E Series
Storage	FC Switch: HP StorageWorks 4/16 SAN Switch External disk: HP StorageWorks 2000fc

4.2.2 Oracle Solaris

SOLARIS

The configuration described is a Oracle Solaris example of a hardware cluster. It is almost the same configuration as the one presented in section 4.2.1 (Linux) with the following exceptions:

- Solaris cluster software is used instead of IBM Tivoli.
- Single node database is replaced with Oracle RAC setup (nodes 4 and 5).

The topology of the configuration is presented in Figure 9 below:

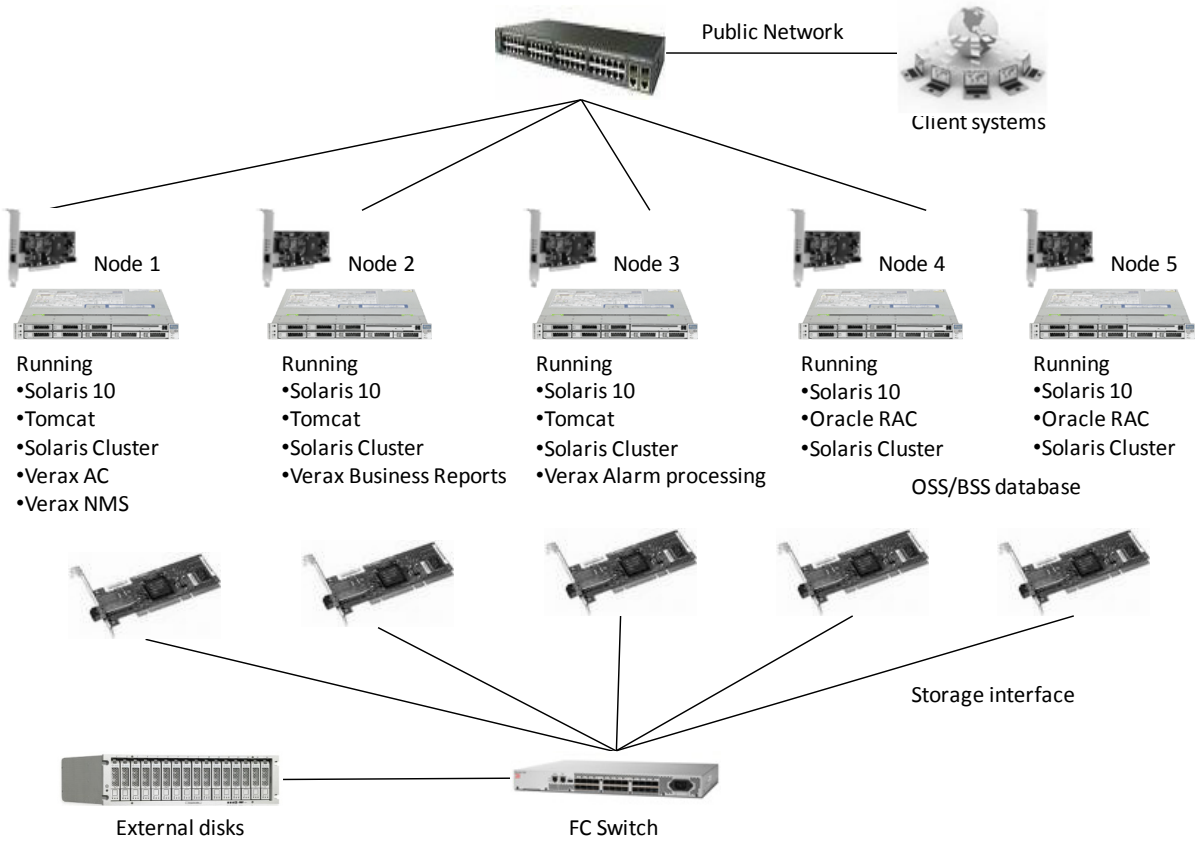


Figure 9: Solaris-based hardware cluster configuration for Verax NMS.

4.2.3 IBM AIX

AIX

The configuration described in this section contains seven nodes running a complex tier-1 operator NMS system managing thousands of network elements. IBM PowerHA is used to enable the clustering functionality. The processing is divided into the following nodes:

Node 1 – Runs the system's administrative console (AC) and the network management system (NMS). This part of the system is mainly responsible for presenting the user interface to the Network Operation Centre (NOC) personnel. The node runs the following software:

- Operating system: AIX 5.3.
- Web server: IBM WebSphere.
- IBM PowerHA.
- Verax applications: Administration Console (AC), Network Management System (NMS).

Node 2 – Runs a business reporting engine. The node runs the following software:

- Operating system: AIX 5.3.
- Web server: IBM WebSphere.
- IBM PowerHA.
- Verax applications: Batch Manager (BM) supervising the business report engine.

Node 3 – Alarm processing node. The node is responsible for alarm collection and processing logic. Since the processing logic can be very complex (from handling volume data during alarm storms, to implementing alarm correlation, root-cause analysis and other data-intensive operations), it is delegated to a separate node in the cluster.

The node runs the following software:

- Operating system: AIX 5.3.
- Web server: IBM WebSphere.
- IBM PowerHA.
- OSS/BSS applications: Batch Manager (BM) with alarm processing logic (combination of Java code, SQL scripts, Unix scripts and JRuby embedded scripts).

Nodes 4 and 5 – Responsible for running the database part of the installation (Oracle RAC in this case). The nodes run the following software (please note that the nodes do not run Verax applications):

- Operating system: AIX 5.3.
- Database system: Oracle RAC Database 10g release 2.
- IBM PowerHA.
- Oracle RAC.

Nodes 6 and 7 – Standby (failover nodes). The nodes run the following software:

- Operating system: AIX 5.3.
- Web server: IBM WebSphere.
- Database system: Oracle Database 10 release 2.
- IBM PowerHA.

The topology of the configuration is presented in Figure 10 below:

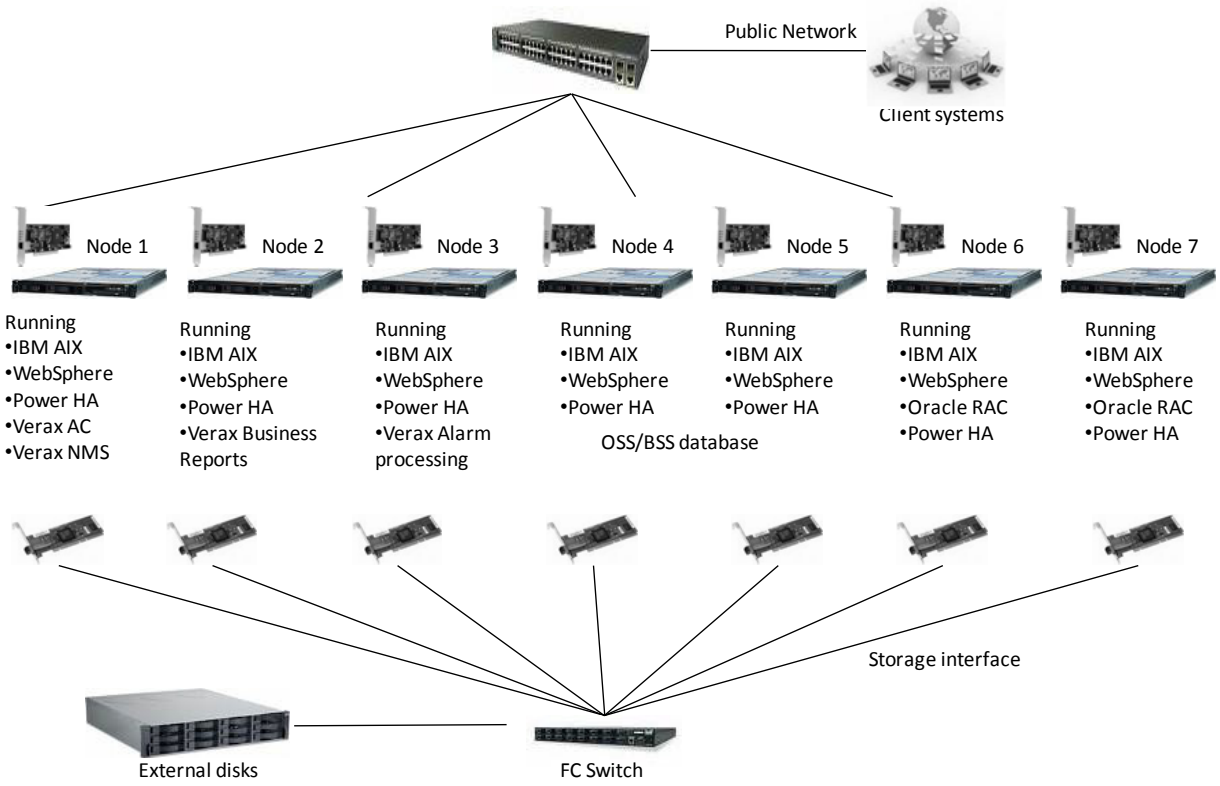


Figure 10: IBM AIX-based hardware cluster configuration for Verax NMS.

The table below contains an inventory of elements required to build the configuration:

Configuration elements	
Nodes	Seven IBM System p5 505 nodes with the following configuration: • Processors: two 2.1 GHz 64-bit POWER5+ • Main Memory: 16 GB RAM • Internal disk: 300GB 10,000 RPM Ultra320 SCSI • Host Bus Adapter : 4 Gb Single-Port Fibre Channel PCI-X 2.0 DDR Adapter
Network switch	Cisco Catalyst 3750-E Series
Storage	FC Switch: McData Sphereon 4300 Fibre Channel Switch External disk: IBM System Storage DS3400

5 Virtualization

Virtualization can be used to improve the utilization and availability of resources and applications. Typically, computing resources are underutilized under the classic "one server, one application" model. A virtualization platform, allows for responding to resource requirements in a fast and efficient manner. According to VMware, their customers typically "save 50-70% on overall IT costs by consolidating their resource pools and delivering highly available machines with VMware Infrastructure".

The distributed nature of Verax applications lends itself very well for virtualization, allowing our customers to:

- Reduce capital costs by requiring less hardware and lowering operational costs for running the system.
- Build virtualization-based high-availability solutions.
- Build virtualization-based performance solutions (e.g. allowing more processing power for nightly business reports).
- Build virtualization-based business continuity and disaster recovery solutions.

Verax applications have been successfully tested with VMware and IBM virtualization software.

5.1 VMware

Verax Systems recommends using VMware High Availability product for Linux-based installations of Verax applications. More information about the product can be found at:

http://www.vmware.com/products/vi/vc/ha.html

How does it work?

VMware HA monitors all servers in a resource pool and detects server failures. It checks if sufficient resources are available in the resource pool at all times and restarts virtual machines on different physical servers in the event of server failure. The restart of virtual machines is made possible by the clustered Virtual Machine File System which gives multiple ESX Server instances read-write access to the same virtual machine files concurrently. VMware HA is easily configured for a resource pool through the Virtual Center GUI.

A sample VMware configuration is presented in Figure 11 below (please note that it is a virtualized instance of the configuration presented in section 4.2.1):

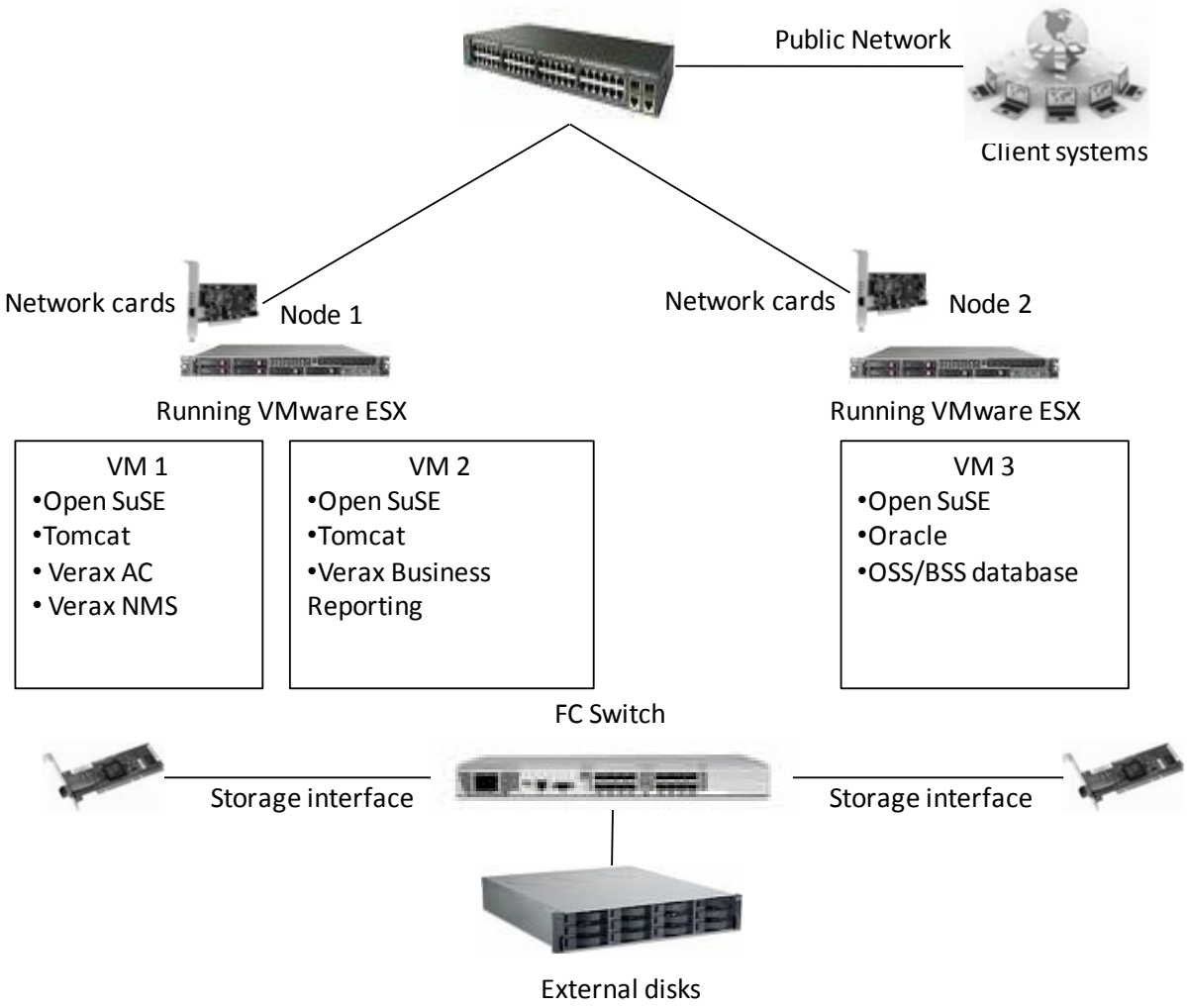


Figure 11: VMware virtualized NMS configuration.

The configuration runs two physical nodes with three virtual machines. The first host prioritizes the "interactive" virtual machine (i.e. the one running the Administrative Console and NMS GUIs) over the "batch" virtual machine (i.e. the one responsible for alarm processing and business reports) in order to ensure maximum CPU utilization. In addition, if any of the hosts fail, the virtual machines are restored on the working host to ensure continuous availability (at the expense of performance degradation).

The table below contains an inventory of elements required to build the configuration:

Configuration elements	
Nodes	Two HP ProLiant DL360 G5 nodes with the following configuration: • Processors: two Intel Xeon Quad-Core 3,33 GHz • Main Memory:16 GB RAM • Internal disk: 300 GB 10000 rpm SAS • Host Bus Adapter: HP Storage Works PCI-e 4GB
Network switch	Cisco Catalyst 3750-E Series
Storage	FC Switch: HP StorageWorks 4/16 SAN Switch External disk: HP StorageWorks 2000fc

5.2 IBM Virtualization Engine for AIX

Verax Systems recommends using IBM Virtualization Engine for AIX-based installations of Verax applications. Detailed information about the product can be found at:

http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp?topic=/veicinfo/eicar_install_config_overview_aix.htm
http://www.redbooks.ibm.com/abstracts/sg247590.html

A Virtualization Engine equivalent to the VMware configuration presented in section 5.1 is presented in Figure 12 below:

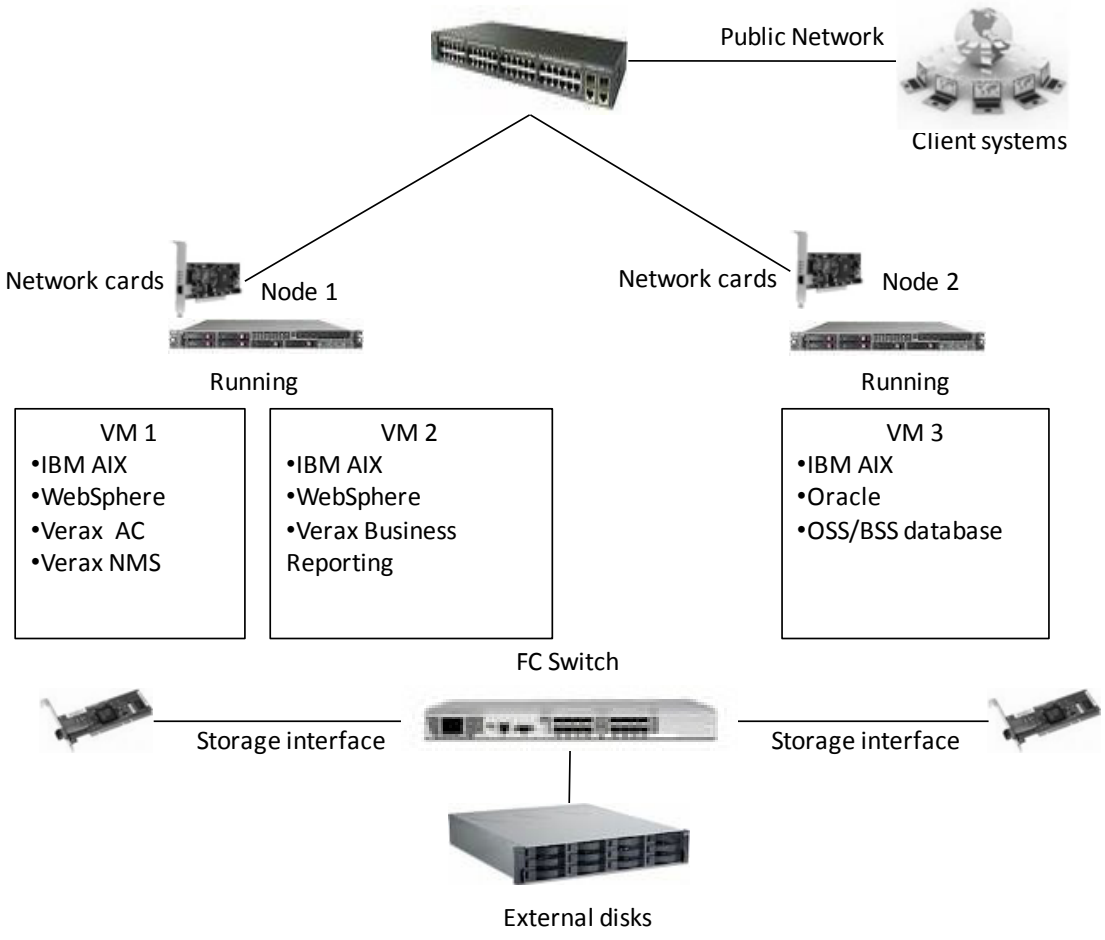


Figure 12: IBM-virtualized NMS configuration on AIX.

6 Failover

Failover is the method of operation that in case of failure, abnormal termination or scheduled down-time of a system component (e.g. active application, server, network, database), involves the transfer of its functions or switching to a redundant or standby secondary system component. Used to make systems more fault-tolerant, failover is typically an integral part of mission-critical systems that must constantly be available.

Failover configurations can be implemented both locally (i.e. within a single data center), as well as in geographically distributed environments, to assure the most seamless fault-tolerant service availability and disaster recovery respectively.

The section below presents the scenarios of a typical local failover configuration for the Verax applications and the underlying database, assuming it is running on a separate machine (virtual or physical). The subsequent sections provide detailed instructions on how to set up a failover configuration depending on the underlying operating system, including the use of particular tools and their installation steps.

6.1 Failover scenarios

The proposed failover configuration is based on the concept of a floating IP address. Each application and database node in the working environment is duplicated. Each secondary (slave) node performs continuous health checks of the master node and its files are synchronized in real time with the master node as long as the master works. The proposed solution uses an active/passive configuration, which means that services on slave nodes are shutdown as long as the master nodes work.

Once the system is started, the master nodes are used. In case of master node failure, the failover solution will automatically switch to the slave node by stealing the floating IP address and starting the necessary services.

Once the master node is back to the normal operation after a failure, a failback process re-synchronizes and re-starts the master node services and stops the services on the slave node.

6.1.1 Database failover

The diagram below presents the failover scenario for a database running without a vendor-specific failover solution.

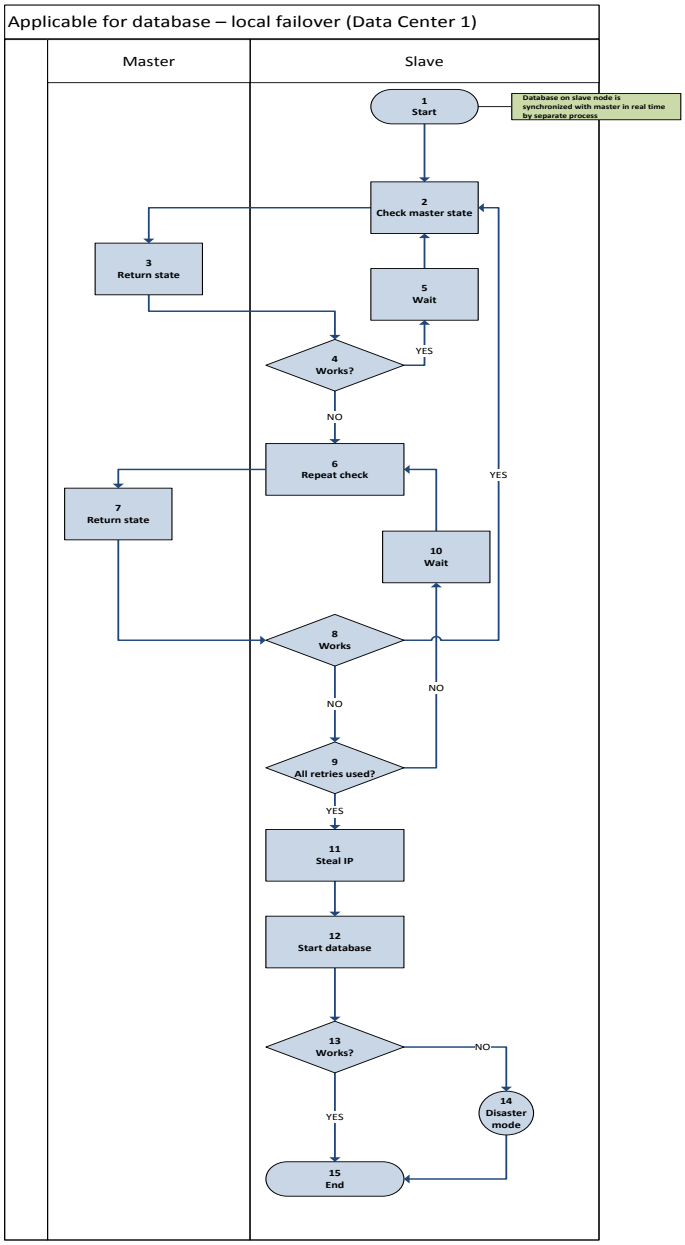


Figure 13: Database failover scenario.

6.1.2 Application failover

The diagram below presents the application failover scenario.

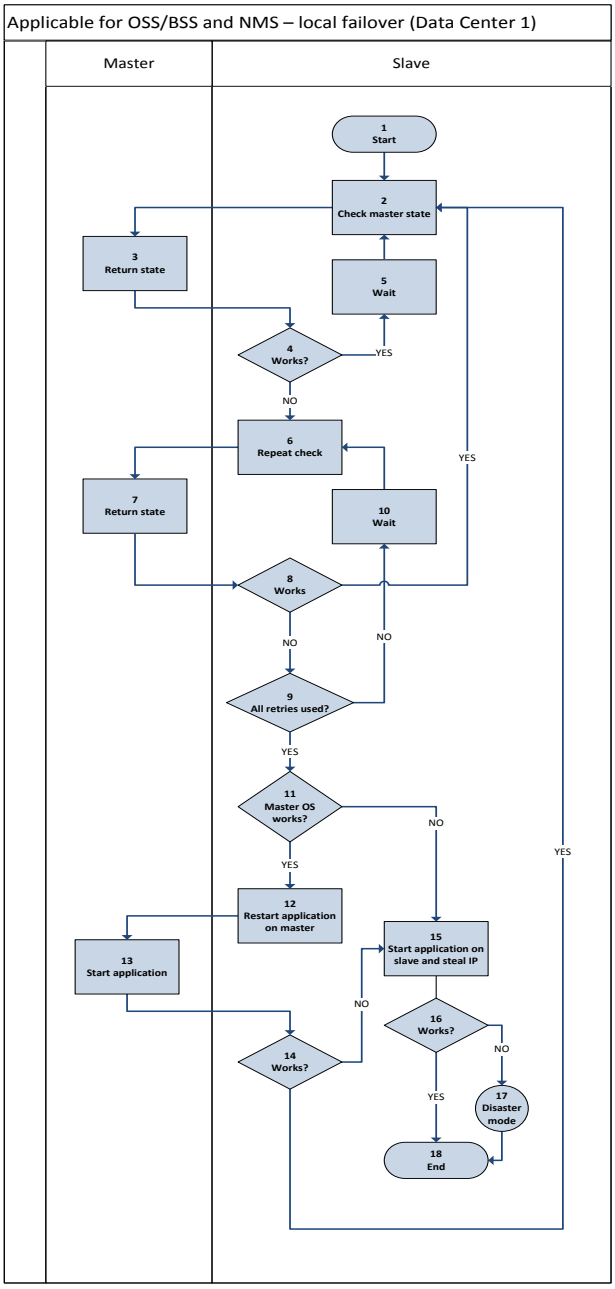


Figure 14: Application failover scenario.

6.1.3 Database failback

The diagram below presents the failback scenario for a database running without a vendor-specific failover solution.

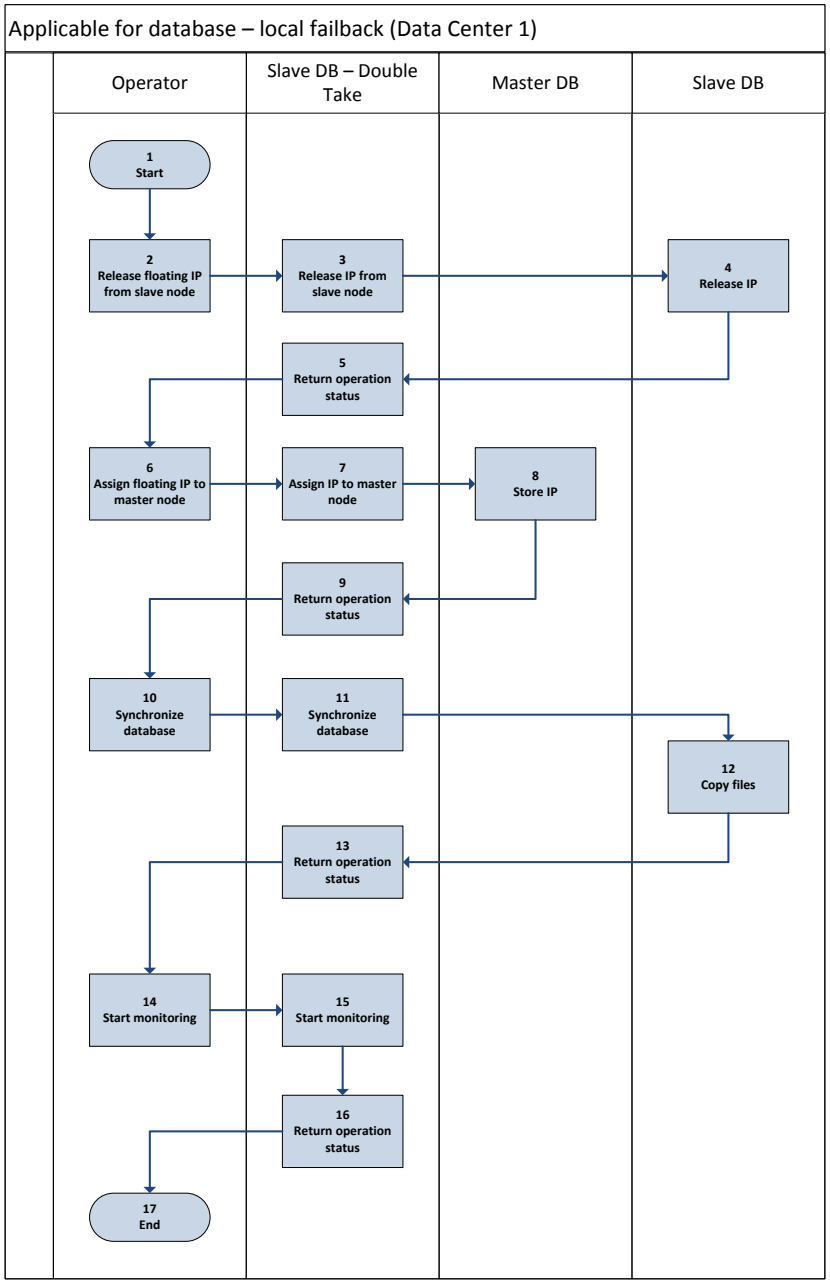


Figure 15: Database failback scenario.

6.1.4 Application failback

The diagram below presents the application failback scenario.

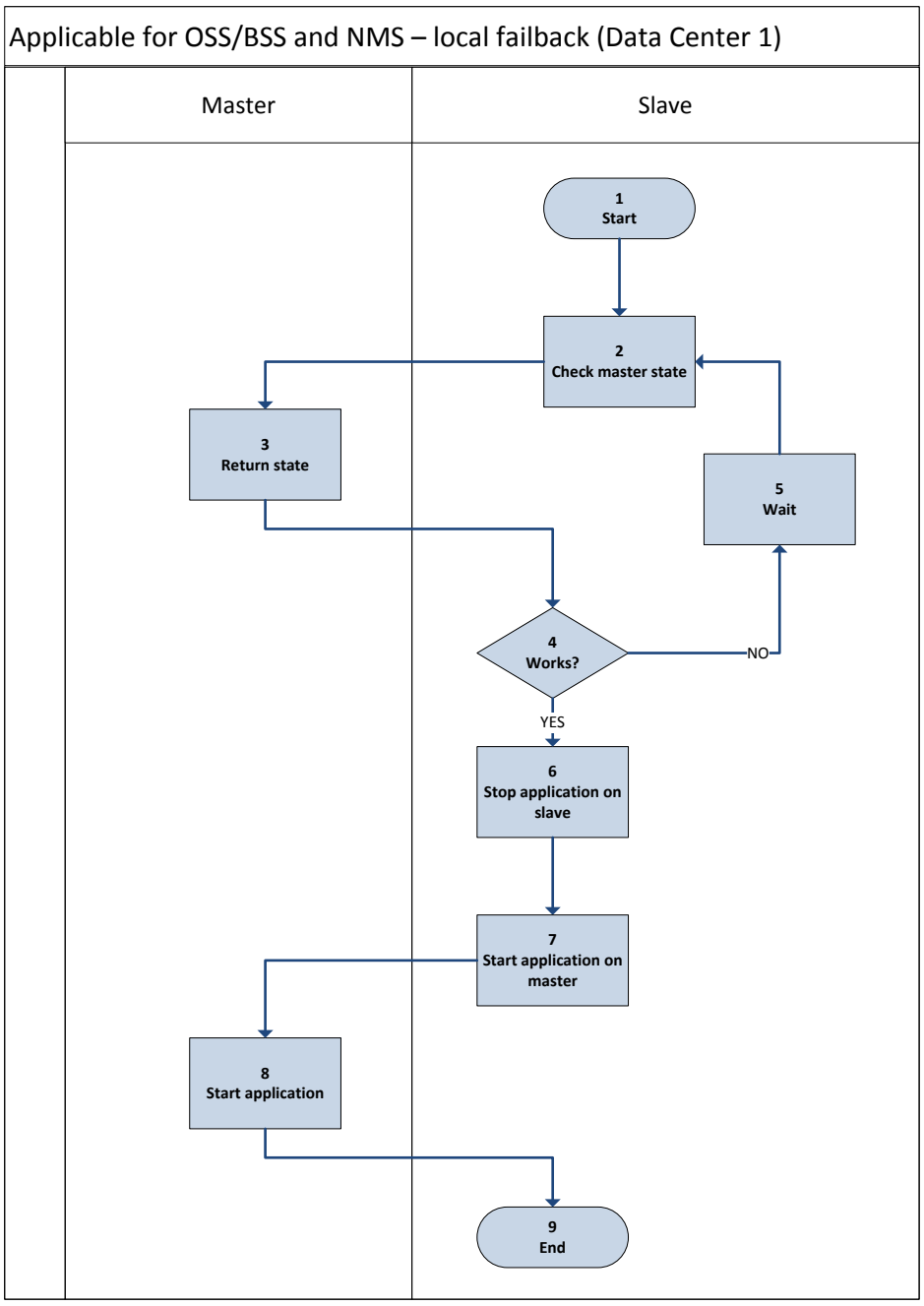


Figure 16: Application failback scenario.

6.2 Linux

LINUX

For Linux, Verax Systems recommends the following software:

- **Pacemaker** – an open source, High Availability resource manager. It automatically initiates recovery, making sure the application is available from one of the remaining machines in the cluster.
  For more details, see http://www.clusterlabs.org.
- **Corosync** – a High Availability framework used by Pacemaker for reliable cluster communications. It is a configuration and statistics in-memory database that provides the ability to set, retrieve, and receive information change notifications.
  For more details, see http://www.corosync.org.

Pacemaker constitutes the superior element of a cluster infrastructure being responsible for the high level control, whereas Corosync provides low level inter-node communication and command framework.

Supported Linux distributions include:

- Fedora
- openSUSE
- EPEL-compatible distributions (Debian, RHEL, Centos, Scientific Linux, etc.)

6.2.1 Installation process

ⓘ Prerequisites:

- Both servers set up in the same network
- Linux compatible with *epel* repository. In this guide, CentOS 5.6 was used
- Configured ssh keys
- Firewall disabled
- Text editor, e.g. *vim*

Installation process:

1. Login as *root* on the master node
2. Install *epel* and *clusterlabs* repositories
   - To install *clusterlabs,* execute:
     ```
     # wget –O /etc/yum.repos.d/pacemaker.repo
     http://clusterlabs.org/rpm/epel-5/clusterlabs.repo
     ```
   - To install *epel,* execute:
     ```
     # wget http://download.fedora.redhat.com/pub/epel/5/i386/epel-
     release-5-4.noarch.rpm
     # rpm -Uvh epel-release-5.4.noarch.rpm
     ```
3. Install *corosync* and *pacemaker* with all dependencies:
   ```
   # yum install –y corosync pacemaker
   ```
4. Configure *corosync*
   - Create *corosync.conf* file – for this, use the command:
     ```
     # touch /etc/corosync/corosync.conf
     ```
   - Edit *corosync.conf* in Vim
     ```
     # vim /etc/corosync/corosync.conf
     ```
   - Press "i" and copy & paste the example configuration from
     ```
     http://www.clusterlabs.org/wiki/Initial_Configuration
     ```
   - Press "Esc" and enter ":wq". Confirm by pressing "Enter"
5. Instruct *corosync* to start *pacemaker*

- Create *pcmk* file using the command
  `# touch /etc/corosync/service.d/pcmk`
- Edit *pcmk* file in Vim
  `# vim /etc/corosync/corosync.conf`
- Press "i" and copy & paste the following:

```
service {
        # Load the Pacemaker Cluster Resource Manager
        name: pacemaker
        ver:  0
}
```

6. Login as *root* on the slave node and repeat the steps b-e
7. Run *corosync* on the master node
   `# /etc/init.d/corosync start`
8. Verify if *corosync* has been stared successfully – the following message should be displayed:
   `Starting Corosync Cluster Engine (corosync): [ OK ]`
9. Run *corosync* on the slave node
   `# /etc/init.d/corosync start`
10. Verify if *corosync* has been stared successfully – the following message should be displayed:
    `Starting Corosync Cluster Engine (corosync): [ OK ]`
11. Display the cluster status using `crm_mon`. The command can be run on the master as well as on the slave node. If the cluster has been created successfully, a message similar to the one below should appear:

```
[root@pcmk-1 ~]# crm_mon
============
Last updated: Thu Aug 27 16:54:55 2009
Stack: openais
Current DC: pcmk-1 - partition with quorum
Version: 1.1.5-bdd89e69ba545404d02445be1f3d72e6a203ba2f
2 Nodes configured, 2 expected votes
0 Resources configured.
============

Online: [ pcmk-1 pcmk-2 ]
```

12. Configure the cluster resource by executing the following steps on the master node:
    - Create *conf* file
      `# touch /root/conf`

- Edit *conf* in Vim

```
# vim /root/conf
```

- Press "i" and copy & paste the following:

```
primitive APPLICATION ocf:heartbeat:tomcat \
params catalina_home="catalina_home" catalina_opts="catalina_opts" \
java_opts="java_opts" statusurl="statusurl" java_home="/usr" \
        op start interval="0s" timeout="160s" \
        op monitor interval="10s" timeout="60s" \
        op stop interval="0s" timeout="120s" \
        meta target-role="Started"
primitive ClusterGateway ocf:heartbeat:IPaddr2 \
        params ip="FLOATING_IP_ADDRESS" cidr_netmask="24" \
        op monitor interval="10s" \
        meta target-role="Started"
primitive ping_gate ocf:pacemaker:pingd \
        params host_list="GATEWAY_IP_ADDRESS" multipler="100" \
        op monitor interval="15s" timeout="5s"
location my_server failover \
        rule $id="my_server-rule" -inf: not_defined pingd or pingd
lte 0
group failover ClusterGateway APPLICATION
location prefer-cluster failover  50: pcmk-1
property $id="cib-bootstrap-options" \
        dc-version="1.0.12-unknown" \
        cluster-infrastructure="openais" \
        expected-quorum-votes="2" \
        stonith-enabled="false" \
        no-quorum-policy="ignore" \
        last-lrm-refresh="1323093765"
rsc_defaults $id="rsc-options" \
        resource-stickiness="100"
```

- Replace the FLOATING_IP_ADDRESS and GATEWAY_IP_ADDRESS with valid IP addresses. The other variables (marked in bold font) should be replaced with the following values:

Variable values		
APPLICATION	Parameter	Value
NMS	catalina_home	/opt/verax/tomcat-nms/ or other path to directory with app
	catalina_opts	-Xms500M -Xmx2048M -Dcom.sun.management.jmxremote.port=9408 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false
	java_opts	-Xms500M -Xmx2048M -XX:PermSize=64m -XX:MaxPermSize=256m -XX:+HeapDumpOnOutOfMemoryError
	statusurl	http://(hostname_or_ip):9400/enetworkmanagementsystem-fds/eNetworkManagementSystem/index.jsp
Administrator Console	catalina_home	/opt/verax/tomcat-ac/ or other path to directory with app
	catalina_opts	-Xms500M –Xmx1024M -Dcom.sun.management.jmxremote.port=9208 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false
	java_opts	-Xms500M -Xmx1024M -XX:PermSize=64m -XX:MaxPermSize=128m -XX:+HeapDumpOnOutOfMemoryError
	statusurl	http://(hostname_or_ip):9200/eadministratorconsole-fds/eAdministratorConsole/index.html
Service Desk	catalina_home	/opt/verax/tomcat-ett/ or other path to directory with app
	catalina_opts	-Xms500M –Xmx1024M -Dcom.sun.management.jmxremote.port=9258 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false

Variable values		
	java_opts	-Xms500M –Xmx1024M -XX:PermSize=64m -XX:MaxPermSize=128m -XX:+HeapDumpOnOutOfMemoryError
	statusurl	http://(hostname_or_ip):9400/enetworkmanagementsystem-fds/eNetworkManagementSystem/index.jsp
Customer Care & Billing	catalina_home	/opt/verax/tomcat-ccb/ or other path to directory with app
	catalina_opts	-Xms500M -Xmx1024M -Dcom.sun.management.jmxremote.port=9408 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false
	java_opts	-Xms500M -Xmx1024M -XX:PermSize=64m -XX:MaxPermSize=128m -XX:+HeapDumpOnOutOfMemoryError
	statusurl	http://(hostname_or_ip): 9040/eCustomerCare-fds/eCustomerCare/
Provisioning Inventory, Workflow Manager	catalina_home	/opt/verax/tomcat-ePI/ or other path to directory with app
	catalina_opts	-Xms500M -Xmx1024M -Dcom.sun.management.jmxremote.port=19208 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false
	java_opts	-Xms500M -Xmx1024M -XX:PermSize=64m -XX:MaxPermSize=128m -XX:+HeapDumpOnOutOfMemoryError
	statusurl	http://(hostname_or_ip):19200/eProvisioningInventory-fds/eProvisioningInventory/index.jsp
Provisioning	catalina_home	/opt/verax/tomcat-provisioning/ or other path to directory with app
	catalina_opts	-Xms500M -Xmx1024M -Dcom.sun.management.jmxremote.port=19108 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false
	java_opts	-Xms500M -Xmx1024M -XX:PermSize=64m -XX:MaxPermSize=128m -XX:+HeapDumpOnOutOfMemoryError

Variable values		
	statusurl	http://(hostname_or_ip): 19100/provisioning/index.jsp
Batch Manager, Batch Runner, SSO server, HTML Gateway	catalina_home	/opt/verax/tomcat-extras/ or other path to directory with app
	catalina_opts	-Xms500M -Xmx1024M -Dcom.sun.management.jmxremote.port=9708 -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false
	java_opts	-Xms500M -Xmx1024M -XX:PermSize=64m -XX:MaxPermSize=128m -XX:+HeapDumpOnOutOfMemoryError
	statusurl	http://(hostname_or_ip):9700/eBatchManager-web/

- Press "Esc" and enter ":wq". Confirm by pressing "Enter"

13. Add the configured resource to the cluster

```
# crm configure
crm(live)configure# load replace /root/conf
crm(live)configure# commit
crm(live)configure# exit
```

14. Enter `#crm configure show` command. Verify if the value displayed is exactly the same as entered into the */root/conf* file
15. Add *corosync* to *autostart*

```
# chkconfig –level 345 corosync on
```

6.2.2 Failover script

The table below presents the recommended failover actions for master and slave nodes.

MODE	STAGE	MASTER	SLAVE
MONITORING	Determining server status	• check gateway connection • check application status	• check connection to master server • check gateway connection
	Making decision to failover	• slave node is offline • ping to gateway is not responding • when server cannot start application over specified time	• master node is offline • ping to gateway is responding
FAILOVER	Application Failover	• restart application • if an application still has an issue then: ○ turn application off ○ release IP	• if application has issue after being restarted then: ○ switch IP (Floating IP) ○ run application if IP is switched
	Server failover	• turn application off • release IP	• the same actions as in case of Application failover
FAILBACK	Making decision to failback	• both nodes are online • ping to gateway is responding	
	Failback	• switch IP • run application if IP is switched • when server cannot start application over specified time go to failover status	• turn application off • go to monitoring mode

6.3 Windows

WIN

For Windows, Verax Systems recommends the use of Double-Take Availability, which provides real-time high availability and immediate disaster recovery. It performs the following operations:

- Mirroring - The initial copy or subsequent resynchronization of selected data,
- Restoration - A mirror of selected data from the target back to the source,
- Failure monitoring service using native or custom script written in Visual Basic, Power Shell or Java Script,
- Automatic failover to target server when source server is down or there are issues concerning the monitored application/service,
- Manual failback,
- Running batch script on target server when the failover/failback process is complete and running batch script on Source server,
- Sending e-mail notification.

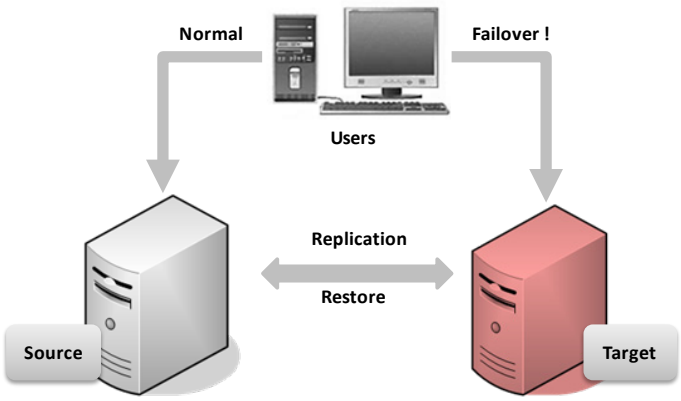


Figure 17: Failover on Windows

Double-Take Availability uses WMI (Windows Management Instrumentation) service for communication between servers and management application/service.

Supported Microsoft Windows platforms include:

- Windows Server 2003 / 2008

  Standard, Enterprise and Datacenter Editions (32-bit / 64-bit)
- Windows XP

For more details, see http://www.visionsolutions.com.

6.3.1 Installation process

ⓘ Prerequisites:

- Supported version of Windows installed on both servers
- Double-Take Availability installed on both servers
- WMI service installed and enabled on both servers
- Servers are configured with static IP addresses
- Master Server is configured with a Virtual IP, which will be switched
- User account belonging to Double-Take Admin or Double-Take Monitor group on both servers

Installation process:

1. Run Double-Take console application and add two servers with Verax applications. To add servers with Verax applications, execute the following steps:
   a. Select **Get Started** from the toolbar.
   b. Select **Add servers** and click **Next**.
   c. On the **Manual Entry** tab, specify the server information.
      - Server – IP address of the server to be added to the console.
      - User name – Specify a user that is a member of the Double-Take Admin or Double-Take Monitors security group on the server.
      - Password – Specify the password associated with the User name you entered.

d. Once you have specified the server information, click **Add**.

2. Setting e-mail notifications
    a. Select **Manage Servers.**
    b. Select the desired server and click **Server Details.**
    c. Select **Edit Server Properties** from the right hand side .
    d. Go to the **E-Mail Notification** section.
    e. Check **Enable e-mail notification** and provide e-mail server settings such as: e-mail server address, login credentials, recipients e-mail address.
3. Configure application protection. To configure application protection, execute the following steps:
    a. Select **Get Started** from the toolbar in Double Take console.
    b. Select **Double-Take Availability** and click **Next**.
    c. Select **Protect an application** and click **Next**.
    d. A **Double-Take Availability Application Manager** window will pop up. Go to **Tools** - **Options** and tick the **Display advanced options** checkbox.
    e. Click **Protect File Server.**
    f. Double Take will automatically detect the root domain. Change this value to a valid one if necessary.
    g. Add the servers. To add servers, execute the following steps:
        - Click **Advanced find**.
        - In the popup window, enter the master server name and click **Add**.
        - Repeat the above for the slave server.
        - When completed, click **OK**.
    h. Select the master server.
    i. Enter user credentials (username and password) in the popup window.
    j. Select the slave server.
    k. Enter user credentials (username and password) in the popup window.
    l. Now click **Configure**

m. Double-Take will try to connect to the DNS server. As it is not necessary at this stage, ignore any warning messages that appear.

n. In a new window, select **Identity Failover** as the **Failover Type** and click **Configure**.

o. All IP addresses of the master server will be displayed. Identify the desired master server IP address and select the slave server interface for it. Then, click **Target NIC**.

p. In the section **Items to Failover** select the **IP Address** option only and click **OK**.

q. In the **Services** section add the Verax application service(s).

Variable values	
APPLICATION	Windows service name
NMS	VeraxNMS
AC	VeraxAC
Service Desk	VeraxTT
Customer Care & Billing	VeraxCCB
Provisioning Inventory, Workflow Manager	VeraxProvisioningFrontend
Provisioning	VeraxProvisioningEngine
Batch Manager, Batch Runner, SSO server, HTML Gateway	VeraxExtras

r. In the **Monitor Settings** section set:

- Method to Monitor for Failover to **Application Monitoring**.
- Failover trigger to **One monitored IP address fail**.
- Monitor interval to **20**.
- Failure Count to **3**.

- In **Monitored IP** tick only **Virtual IP** of the source server.
- Provide the credentials.
- In **Monitoring Option** select **Custom Monitoring Script** and enter the path to the script delivered by Verax Systems.

s. At this stage, the files to be protected should be added. In the section **File Shares,** select the path to the Verax application installation directory (default C:\Program Files\Verax Systems\ApplicationName\) and C:\Windows\etc\verax.d\

t. In the section **Mirroring Settings** select **Checksum** and, optionally, **Enable compression.**

4. Close the opened windows by clicking **OK**.
5. In Double-Take Availability Application Manager click **Validate**. The following warning messages are expected and can be ignored:
   a. Identity Failover is not recommended.
   b. Target server does not have permission to update SPN's for master on failover.
   c. The server (NETBIOS) name is not selected for failover.
   d. Reverse look up is not enable.
6. Click **Enable Protection**.
7. Double-Take starts mirroring the files and monitors the applications.

For more details, see:

http://download.doubletake.com/_download/dt52/DT521/docs/Double-Take%20Availability%20User%27s%20Guide.pdf.

6.3.2 Failover script

The table below presents actions performed by DT. In the case of Windows, most operations are done on the slave server side.

MODE	STAGE	MASTER	SLAVE
MONITORING	Determining server status	• no actions	• check connection to master server using ping • check service status • if there are changes in the protected file, mirror will be created
	Making decision to failover	• no actions	• detect connection issue to master server • application service is still down after restart • application URL is still not responding after restarting the service
FAILOVER	Application Failover	• turn application off	• switch IP • run failover scripts, e.g. NMS service • send email • remain in failover status
	Server failover	• no action	• the same actions as in the case of Application failover
FAILBACK	Making decision to failback	• decision to start failback process is left to the Administrator	
	Failback	• restore IP	• release IP • restore Files (Full Or Differential)

6.4 Failover on Solaris

SOLARIS

Verax Systems developed custom failover scripts for the Sun Solaris platform. The scripts support the following functionality:

- Automatic failover to slave node,
- Manual failback to master node,
- Network connection detection,
- Power and application issue,
- Write activity to log,
- HA block on demand,
- E-mail notification.

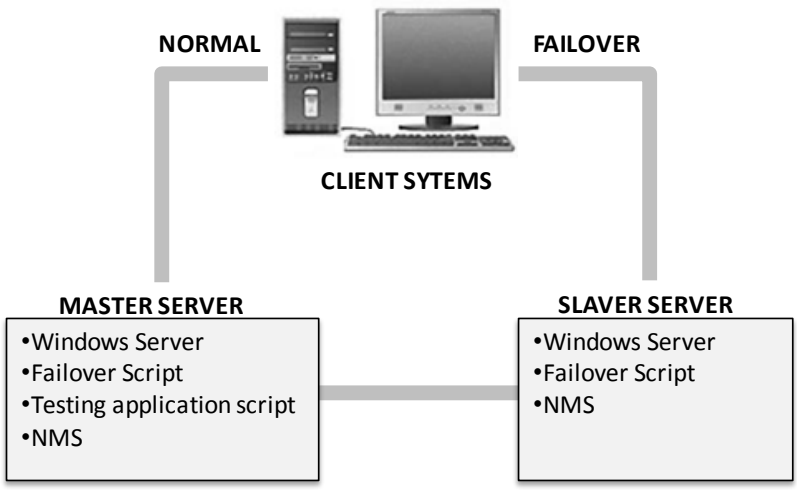


Figure 18: Failover on Solaris

6.4.1 Installation process

1. Create the `failover.config` file in the specified directory on both servers.

   e.g. `etc/verax.d/`

2. Edit and configure the `failover.config` file:

   - Path to scripts provided by Verax (scripts may be located in the same directory as the `failover.config` file)
   - Virtual interface that will be created
   - Source and target server IP addresses
   - Floating IP
   - Gateway IP address
   - Path to wget and the application to be monitored
   - Node definition
     - For source server, set node=master
     - For target server, set node=slave

6.4.2 Failover script

The following table shows the actions performed by scripts.

MODE	STAGE	MASTER	SLAVE
MONITORING	Determining server status	• connection to slave server and gateway • application testing: ○ connection to database ○ URL application ○ Writing/saving status + timestamp to log file	• checking connection to master server and gateway • checking status in log file using ssh
	Making decision to failover	• FAILED status in log file • no connection to gateway and slave server	• FAILED status in log file • Master server is not responding / gateway is responding
FAILOVER	Application Failover	• turn application testing script off • turn application off • turn NIC (floating IP) off • send e-mail notification • disable failover scripts	• ping IP master server (Floating IP) at user-defined time intervals till its response. **Output:** IP is not responding: ○ switch IP ○ run services ○ send email notification ○ go to stand by status No IP response for longer than one minute: ○ send alert email ○ check IP again ○ if not responding switch IP ○ run services ○ send e-mail notification

MODE	STAGE	MASTER	SLAVE
			○ disable failover scripts
	Server failover	• the same actions as in the case of Application failover	• the same actions as in case of Application failover
FAILBACK	Making decision to failback	• decision to start failback script is left to the Administrator	
	Failback	• restore IP (Floating IP) • run services • go to monitoring status	• release IP (Floating IP) • stop services • go to monitoring status

Index