



# **Web Server Load Balancing with Genesys Voice Platform and Radware AppDirector**

**Documentation Supplement to  
Genesys Voice Platform: Network Edition 7.2 Deployment Guide  
Radware: AppDirector User Guide, Software Version: 1.06.09DL**

Version 1.1  
March 10, 2009  
Document Status: Final

**CREATED BY:****NOTICE**

Copyright © 2006 Genesys Telecommunications Laboratories, Inc. All rights reserved.

Genesys Telecommunications Laboratories, Inc.  
2001 Junipero Serra Blvd.,  
Daly City, CA 94014

Phone: 1-888-GENESYS  
Fax: 1-650-466-1260

The information contained herein is proprietary and confidential and cannot be disclosed or duplicated without the prior written consent of Genesys Telecommunications Laboratories, Inc.

**ABOUT GENESYS**

Genesys Telecommunications Laboratories, Inc., an independent, wholly owned subsidiary of Paris-based Alcatel, pioneered the field of Computer-Telephony Integration (CTI) and today is the leading provider of infrastructure independent contact center solutions for the enterprise, service provider, and e-business markets. With its ability to integrate interactions across all media types, including the Web and traditional voice, Genesys software helps businesses provide a consistent customer interaction experience.

**TRADEMARKS**

Genesys and the Genesys logo are registered trademarks of Genesys Telecommunications Laboratories, Inc. UNIX is a registered trademark of The Open Group in the United States and other countries. Microsoft, Windows, Windows 2000, Windows NT and Windows XP are registered trademarks of Microsoft Corporation. All other trademarks and trade names referred to in this document are the property of other companies.

## Table of Contents

<b>1. INTRODUCTION .....</b>	<b>4</b>
1.1. PURPOSE.....	4
1.2. INTENDED AUDIENCE .....	4
1.3. REFERENCES.....	4
1.4. GLOSSARY AND ACRONYMS.....	4
<b>2. LOAD BALANCING OVERVIEW.....</b>	<b>5</b>
2.1. METHODS OF LOAD BALANCING .....	5
2.1.1. <i>Software Load Balancing</i> .....	5
2.1.2. <i>Hardware Load Balancing</i> .....	5
2.1.3. <i>Clustering or Session Replication</i> .....	5
2.2. PERSISTENCE .....	5
<b>3. GVP AND LOAD BALANCING .....</b>	<b>7</b>
3.1. PIPELINING .....	7
3.2. IP PERSISTENCE.....	7
3.3. SESSION PERSISTENCE.....	8
<b>4. TEST ENVIRONMENT CONFIGURATION FOR LOAD BALANCING.....</b>	<b>9</b>
4.1. ENVIRONMENT CONFIGURATION.....	9
4.2. GVP ELEMENT MANAGEMENT PROVISIONING SYSTEM .....	9
4.3. WEB SERVER CONFIGURATION .....	9
<b>5. RADWARE APPDIRECTOR CONFIGURATION .....</b>	<b>11</b>
5.1.1. <i>Initial AppDirector Configuration</i> .....	11
5.1.2. <i>IP Configuration</i> .....	12
5.1.3. <i>Farm Configuration</i> .....	13
5.1.4. <i>Create Layer 4 Policy</i> .....	15
5.1.5. <i>Configure Dynamic L7 Persistency</i> .....	16
5.1.6. <i>Adding Servers to the Farm</i> .....	17
5.1.7. <i>Health Monitoring Configuration</i> .....	20
<b>6. RESULTS AND OBSERVATIONS.....</b>	<b>28</b>
6.1. FUNCTIONALITY .....	28
6.2. LATENCY.....	28
6.3. MANAGEMENT OF DYNAMIC TABLE.....	28
6.4. BENEFITS.....	28
<b>7. APPENDIX A – TEST ENVIRONMENT DESCRIPTION .....</b>	<b>29</b>
7.1. HARDWARE AND SOFTWARE ENVIRONMENT.....	29
7.2. ENVIRONMENT HARDWARE LIST.....	29
7.3. SOFTWARE VERSIONS LIST .....	29
<b>8. APPENDIX B – DIAGRAM OF TEST ENVIRONMENT.....</b>	<b>31</b>

# Documentation Supplement

## Web Server Load Balancing with GVP

### 1. Introduction

This document provides a description of the integration of the Radware AppDirector with Genesys Voice Platform. An overview of load balancing principles and implementation details are included. This document is intended as a guide only and is not a substitute for the Radware and Genesys deployment guides. It is a supplement to be used in conjunction with the deployment guides.

#### 1.1. Purpose

This document is a guide to configuring web server load balancing for GVP using the Radware AppDirector.

#### 1.2. Intended Audience

The document is primarily intended for technical teams tasked with deploying web server load balancing for GVP. This includes, but is not limited to: systems integrators, customers, and Genesys and Radware field personnel. This document assumes that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications.
- Web server concepts
- Network design and operation
- Your own network configurations

#### 1.3. References

All related resources listed in the GVP deployment guide and Radware AppDirector User Guide should be referenced as necessary. Consult the third party manuals for the web servers as needed.

#### 1.4. Glossary and Acronyms

Term	Definition
GVP	Genesys Voice Platform
AD	AppDirector
IPCS	Internet Protocol Communication Server
IPCM	Internet Protocol Call Manager
EMPS	Element Management Provisioning System
IIS	Internet Information Services
IVR	Interactive Voice Response
DNS	Domain Name Server
VIP	Virtual IP
ASP	Microsoft Active Server Pages
JSP	JavaServer Pages
EMS	Element Management System
ASR	Automatic Speech Recognition
TTS	Text to Speech
CCAS	Contact Center Activity Simulator (Call Generator)
SIP	Session Initiated Protocol

## 2. Load Balancing Overview

Load Balancing is used to optimize resource utilization and to decrease processing time. For web servers in particular, the number of web pages that can be served concurrently is limited by the network bandwidth and the performance of the web server itself.

In addition, load balancing provides a method of failover support not available when using only one web server. A load balancer monitors the server farm, thus when server failure or planned downtime occurs it will continue to balance the load across the remaining web servers left in the group. Failure of one server does not result in downtime, although it may result in longer processing times for the entire system.

### 2.1. Methods of Load Balancing

A load balancing solution manages the selection of the appropriate physical server in a group. This group of servers may be called a “server farm” or a “server pool”. There are three types of load balancing solutions available: software, hardware, and clustering or session replication.

#### 2.1.1. Software Load Balancing

A software load balancing solution can be loaded on hardware of the user's choice. A very simple example of a software solution would be “DNS Round Robin”. Many web servers and web application servers have load balancing functionality included. Software solutions tend to be less expensive and typically work best for smaller, less complex applications with a lower network complexity. Some software products can provide high complexity configuration and customization; however, they usually require a significant hardware base to run on.

#### 2.1.2. Hardware Load Balancing

Hardware load balancers use virtual IP addresses for a group of servers and rewrite source and destination IP's as they route traffic. There are two types of load balancing solutions that could be classified as hardware load balancers. The first is switch or router based which balances at the network level using layer 2 and 3 functionality. It is typically the most robust; however, it does not provide the ability to direct traffic based on cookies or URLs. The second type is really a software solution that is packaged with specific hardware and sold as a unit. This type of solution provides more functionality and flexibility with switching based on layers 4 through 7, but is more complicated to configure.

#### 2.1.3. Clustering or Session Replication

Clustering or Session Replication can be used as an alternative to hardware load balancing when sticky sessions or persistence is required. It consists of a farm or cluster of web application servers which have the ability to replicate and maintain session state in memory on all servers or using a common database. Traffic can then be directed to any server in the cluster even if persistence is required since all servers have current session state for all sessions. Clustering offers the advantage of high availability and failover support. When one server fails or is taken out of service, the other servers are still aware of session states for sessions on the failed server and are therefore able to seamlessly take over. Clustering does not in actuality replace load balancing. The traffic still has to be directed to the servers in an ordered manner by some process. However it does solve the problem of maintaining session stickiness.

## 2.2. Persistence

Depending on how the state of a client session is managed, persistence of dynamic content may be required. This can be accomplished using a shared database, session replication in memory on the web servers as described above, or by sending all dynamic requests for a session to the same web server. If the latter option is used, it can be realized either by using source IP persistence where all requests from

one client are always serviced by the same web server. Another method of achieving persistence is session persistence, where all parts of a session are identified by a unique identifier and forwarded to the same web server. There are no persistence requirements for static content.

### 3. GVP and Load Balancing

In GVP, the web servers host the web applications that generate VXML pages. The IPCS acts as the client, invoking the VXML pages and receiving content from the web servers. The state of the client transaction is managed on the web servers. Therefore, when software or hardware load balancing is implemented, the same web server must handle all transactions that are part of the same call session. Otherwise, the caller will be unable to progress through the IVR, resulting in an unpredictable response from GVP from the end user's perspective. The caller could be disconnected without warning at any point within the IVR.

GVP can benefit from the implementation of web server load balancing. Without load balancing, one or multiple applications are tied to individual web servers. The load across the servers can be widely varied over time, limiting the performance of GVP.

If load balancing is used, the traffic across the web servers can be optimized. This results in consistently equal utilization of the servers' resources, independent of an individual application's volume of traffic and therefore increased overall system throughput.

#### 3.1. Pipelining

When information is required from the web servers, the IPCS opens a TCP connection to a web server. Instead of tearing down the connection after every request, GVP leaves the connection open, allowing for fewer delays due to the setting up and tearing down of connections and therefore resulting in better performance. Additional TCP connections are opened as dictated by load – a new connection is opened once all current connections are full. For continuous and uniform traffic, this improves efficiency. However, it introduces a level of complexity in the area of load balancing.

Due to this feature, a “many-to-many” ratio exists between call sessions and TCP connections. The requests belonging to one call session can be spread across many TCP connections, and one TCP connection can contain requests from many calls sessions. Therefore, you are never guaranteed to have only one call session or even a complete set of requests for a number of calls sessions, within the same TCP connection.

A common assumption of many load balancing devices is that a “one-to-one” or “one-to-many” ratio of call sessions to TCP connections exists. This is an incorrect supposition with respect to GVP. The intricacies of GVP's method of communication via TCP connections results in specific requirements for load balancing.

The pipelining of requests between the IPCS' and web servers makes it impossible to implement session persistence without breaking the “pipes”. The load balancing device must either be configured for IP persistence or it must persist at a session level by intercepting every request between the IPCS and web server and examining every packet for a unique ID. Assumptions cannot be made based on the TCP socket that is transmitting the request.

#### 3.2. IP Persistence

IP persistence is realized when the load balancer assigns each IPCS to one web server. All TCP connections from one IPCS will be persisted to a single web server. Since all requests for a call session originate from the same IPCS, this ensures that all parts of a call session are served from the same web server where the state of the call is maintained.

One downside to this method of load balancing is that uneven usage of the web servers may occur. If there are three IPCS' and two web servers, one web server will serve two of the ICPS' and the other web server will only serve one IPCS. This results in one web server having twice the traffic of the other one. As well, it will never be possible, using this persistence method, to have more active web servers than IPCS', as each IPCS must be persisted to a single web server and additional web servers would not be

used. Additional web servers could exist as backups, but would not be utilized until a web server failed or was removed from service.

Another negative result of this method of implementing load balancing is that a web server cannot be added dynamically to the farm or pool. Since all IPCS' are already persisted to a web server, there are none to persist to a new web server when it is added. Unless one of the clients ceases to have traffic and becomes unstuck from its web server, allowing it to select a new web server with which to persist when traffic resumes, the web server will remain unused. Removing a web server for maintenance also poses a problem as once a web server is disabled, the load balancer will no longer send new traffic to it, but any TCP connections from the IPCS will remain open and as long as traffic continues to flow, they will not close.

If calls are reasonably balanced across the IPCS', load is relatively small, and there are a proportionally higher number of IPCS' than web servers, then balancing based on client IP address may be sufficient. This is highly dependent on individual GVP configurations.

### **3.3. Session Persistence**

The optimal load balancing mode for GVP 7.2 involves session persistence. Instead of balancing the IPCS' as discrete units by assigning them a web server, each individual client session, or call, should be load balanced. This will result in the most even load distribution across web servers and will allow there to be more web servers than IPCS' if the need should arise.

Session persistence is executed as follows. As each web application request reaches the load balancing interface, it is examined for a unique ID. In the case of IIS, this unique ID is ASPSessionID and for Apache Tomcat it will be JSessionID. The packet is then routed to a web server based on this information. If the packet is the first in a string of requests from a call it will not have an identifier and a web server is chosen according to the load balancing solution's algorithm. The web server will assign a unique identifier to the session and on the return from the web server, the load balancer will store this unique identifier in a dynamic table along with the web server that the session was sent to. Alternatively, if the unique identifier exists, the dynamic table is accessed to determine the web server that it should be sent to and the packet is forwarded to that web server. All web application requests for a single call session are sent to the same web server. The static requests will not contain a unique identifier and will always be load balanced according to the selected load balancing algorithm.

The downside to this method of load balancing is that opening and examining every packet increases overhead and can cause performance issues and delays. This latency that is introduced may be able to be minimized with proper optimization of the load balancing solution.

Session persistence is however superior to IP persistence when it comes to adding and removing a web server dynamically. When a web server is added the load balancer will immediately begin forwarding call sessions to it according to the load balancing algorithm in place. The load on each web server will quickly become even. When a web server is removed from service, the load balancer will allow all sessions that are stuck to that web server to complete, but will not forward any new sessions to it. This results in the time to remove a web server being equal to or less than the length of one session.



## 4. Test Environment Configuration for Load Balancing

The environment configured for functionality testing contained three IPCS', three web servers and an AppDirector. The full architecture of the test environment is found in Appendix A and a diagram is found in Appendix B.

The choice of three IPCS' and three web servers allowed the testing of the functionality with different combinations of clients and servers.

Multiple AppDirectors could be deployed for redundancy purposes. However, that functionality is not covered in the scope of this document.

### 4.1. Environment Configuration

The load balancing test environment consisted of a GVP NE 7.2 installation. This installation included three IPCS' and three web servers. The web servers were running both Microsoft IIS and Apache Tomcat. All GVP servers in the installation were positioned on one subnet with the web servers situated on a separate subnet. AppDirector was configured with one interface on each of the two subnets.

The traffic flow involving the load balancing components begins when an HTTP request is made through an IPCS to `advip.lsst.genesys.ca` (192.168.43.105), an IP address which is configured on AppDirector to point to the farm/pool of web servers.

AppDirector intercepts the request and inspects the header for a unique ID. If there is no unique ID, the request is load balanced to a web server according to the configured load balancing algorithm, which in this case was Cyclic (Round Robin) and on the return of the request, an entry is made in the dynamic table. If a unique ID is found, the dynamic table is accessed with the unique ID to determine the web server to send the request to.

The default gateways of the web servers were set to be AppDirector's interface on that subnet. This directed the traffic back through AppDirector. Since one web server also housed a database that was used by GVP and the applications, some minor configuration changes had to be made to allow the database traffic to reach the database through the AppDirector. These changes involved the EMPS server and consisted of changing the default gateway of that server to be AppDirector's console interface and adding the IP of the server hosting the database to the Hosts file on the EMPS server.

### 4.2. GVP Element Management Provisioning System

There is only one consideration within GVP's EMPS when provisioning an application on a system that is load balanced. When setting the Primary IVR URL, instead of pointing the application to a specific web server, the application should be provisioned using a Virtual IP address. This virtual address points to the farm/pool of web servers which each contain the application. In the test environment described above, this corresponds to `advip.lsst.genesys.ca`.

### 4.3. Web Server Configuration

The test environment was configured with both Microsoft IIS and Apache Tomcat web servers. This allowed for the testing of both ASP and JSP applications. Both web servers were running on port 80, which meant that only one could be running while testing was taking place. This simplified the management of the web servers as no changes needed to be made within GVP or the AppDirector to accommodate the changing of a port. If different ports were used, the AppDirector would need to perform port multiplexing to transform each request from port 80 to the correct port for the web server.

No specific web server configuration changes need to be performed for load balancing with AppDirector in either IIS or Apache Tomcat. As explained in section 3.3, the implementation of session persistence is dependent on the availability of a unique session identifier. The identifiers are `ASPSessionID` and `JSessionID` for IIS (ASP applications) and Apache Tomcat (JSP applications) respectively. The default configuration of both web servers enables session state persistence for applications, meaning that the

appropriate unique identifier is set as a cookie. It is possible to disable session state persistence at an individual ASP or JSP page level. Session state persistence should not be disabled at the web server level or the ASP/JSP page level. This would result in AppDirector losing the ability to load balance using session persistence. One further note for consideration is that if an application calls an external web application, the unique identifiers will not be the same for the calls to the external application.

## 5. Radware AppDirector Configuration

### 5.1.1. Initial AppDirector Configuration

Using a serial cable and a terminal emulation program, connect to the AppDirector.

The default console port settings are:

- Bits per Second: 19200
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Flow Control: None

1. Assign the following IP address to interface 1 of the AppDirector:

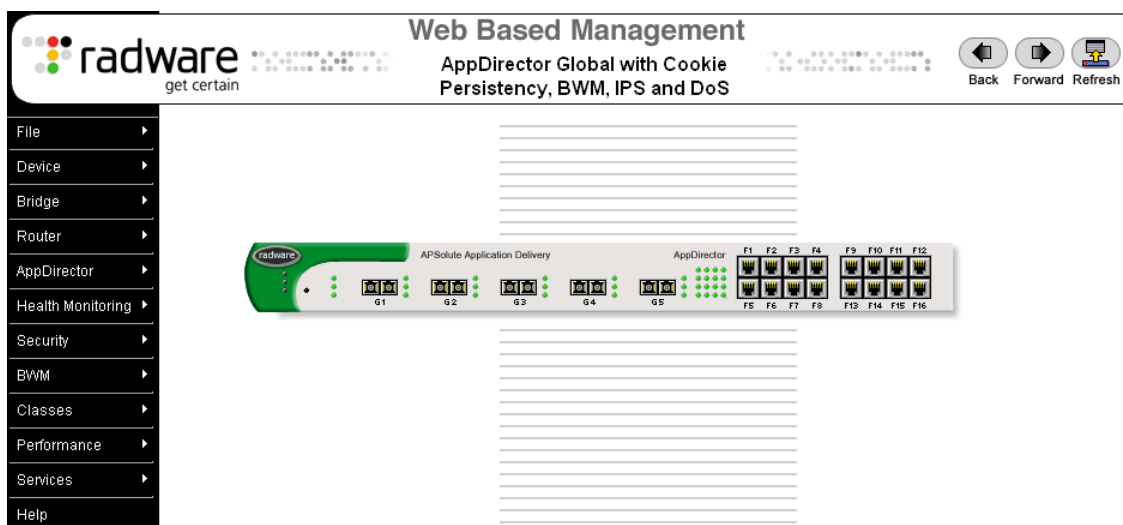
- 192.168.43.106 /25 – IP Address from the GVP subnet

2. Create a default gateway route entry on the AppDirector pointing to 192.168.43.1.

3. Using a browser, connect to the IP Address of the AppDirector (192.168.43.106) via HTTP or HTTPS. The default username and password are “radware” and “radware”.

Failure to establish a connection may be due to the following:

- Incorrect IP Address in the browser
- Incorrect IP Address or default route configuration in the AppDirector
- Failure to enable Web Based Management or Secure Web Based Management in the AppDirector
- If the AppDirector can be successfully pinged, attempt to connect to it via Telnet or SSH. If the ping or the Telnet/SSH connection are unsuccessful, reconnect to the AppDirector via its console port. Once connected, verify and correct the AppDirector's configuration as needed.<sup>1</sup>



<sup>1</sup> To enable web-based management from the console command line interface, use “manage web status set enable”

### 5.1.2. IP Configuration



- From the menu, select **Router** ⇒ **IP Router** ⇒ **Interface Parameters** to display the **IP Interface Parameters** page similar to the one shown below:

## IP Interface Parameters

[Routing Table](#)   [ARP Table](#)

IP Interface Parameters

IP Address	Network Mask	If Number	VlanTag	✕
<a href="#">192.168.43.106</a>	255.255.255.128	1	0	<input type="checkbox"/>

     
Delete   Create

- Click the **Create** button.
- On the **IP Interface Parameters Create** page, enter the necessary parameters as shown below.<sup>2</sup> This will create the Web Server interface.



## IP Interface Parameters Update

[Routing Table](#)   [ARP Table](#)

**IP Address:**  **Network Mask:**

**If Number:**  **Fwd Broadcast:**

**Broadcast Addr:**  **VlanTag:**

     
Set   Cancel

- Verify that the new entry was created on the **IP Interface Parameters** page:

<sup>2</sup> Items circled in red indicate settings that need to be entered or changed. Items not circled should be left to default settings.

## IP Interface Parameters

[Routing Table](#)   [ARP Table](#)

### IP Interface Parameters

IP Address	Network Mask	If Number	VlanTag	✕
<a href="#">192.168.43.106</a>	255.255.255.128	1	0	<input type="checkbox"/>
<a href="#">192.168.43.203</a>	255.255.255.224	2	0	<input type="checkbox"/>




  
 Delete   Create

### 5.1.3. Farm Configuration

- From the menu, select **AppDirector** ⇒ **Farms** ⇒ **Farm Table** to display the **Farm Table** page similar to the one shown below:

**Farm Table** ? Help

[Extended Farm Parameters](#)   [Layer 4 Policy Table](#)   [Server Table](#)   [DNS Persistence Parameters Table](#)

Farm Name	Admin Status	Aging Time [sec]	Dispatch Method	Connectivity Check Method	Sessions Mode	Operational Status	✕
							  Delete   Create

- Click the **Create** button.
- On the **Farm Table Create** page, enter the necessary parameters as shown below:<sup>3</sup>

<sup>3</sup> Items circled in red indicate settings that need to be entered or changed. Items not circled should be left to default settings.

### Farm Table Update

? Help

[Extended Farm Parameters](#) | [Layer 4 Policy Table](#) | [Server Table](#) | [DNS Persistency Parameters Table](#)

<p><b>Farm Name:</b> <input type="text" value="WebServers"/></p> <p><b>Operational Status:</b> Active</p> <p><b>Dispatch Method:</b> <input type="text" value="Cyclic"/></p> <p><b>Sessions Mode:</b> <input type="text" value="ServerPerSession"/></p> <p><b>Connectivity Check Port:</b> <input type="text" value="HTTP"/></p> <p><b>Connectivity Check Retries:</b> <input type="text" value="5"/></p> <p><b>Home Page:</b> <input type="text"/></p> <p><b>Authorized Password:</b> <input type="text"/></p> <p><b>Distribution Threshold:</b> <input type="text" value="2500"/></p> <p><b>Redirection Mode:</b> <input type="text" value="No Redirection"/></p> <p><b>HTTP Redirection Mode:</b> <input type="text" value="IP Mode"/></p>	<p><b>Admin Status:</b> <input type="text" value="Enabled"/></p> <p><b>Aging Time [sec]:</b> <input type="text" value="60"/></p> <p><b>Connectivity Check Method:</b> <input type="text" value="No Checks"/></p> <p><b>Bandwidth Limit:</b> <input type="text" value="No Limit"/></p> <p><b>Connectivity Check Interval [sec]:</b> <input type="text" value="10"/></p> <p><b>Extended Check Frequency:</b> <input type="text" value="10"/></p> <p><b>Authorized Username:</b> <input type="text"/></p> <p><b>Connection Denials:</b> <input type="text" value="0"/></p> <p><b>Capacity Threshold:</b> <input type="text" value="5000"/></p> <p><b>DNS Redirection Fallback:</b> <input type="text" value="DNS Only"/></p>
---	---

4. Click the **Set** button to save parameters.
5. Verify that the new entry was created on the **Farm Table** page:

### Farm Table

? Help

[Extended Farm Parameters](#) | [Layer 4 Policy Table](#) | [Server Table](#) | [DNS Persistency Parameters Table](#)

Farm Name	Admin Status	Aging Time [sec]	Dispatch Method	Connectivity Check Method	Sessions Mode	Operational Status
<a href="#">WebServers</a>	Enabled	60	Cyclic	No Checks	ServerPerSession	Active

6. Click the **Extended Farm Parameters** URI at the top of the **Farm Table** page.
7. On the **Extended Farm Parameters Table** page, click on the Farm Name **WebServers**.
8. On the **Extended Farm Parameters Update** page, enter the necessary parameters as shown below:

Extended Farm Parameters Update
?

[Farm Table](#)

Farm Name	WebServers	Radius Secret	<input type="text"/>
Connection Limit Exception	<input type="text" value="Disabled"/>	Client NAT Address Range	<input type="text" value="0.0.0.0"/>
Transparent Server Support	<input type="text" value="Disabled"/>	SSL ID Tracking	<input type="text" value="Disabled"/>
Close Session At Aging	<input type="text" value="Disabled"/>	RADIUS Attribute	<input type="text" value="0"/>
Reset Client on Server Failure	<input type="text" value="Disabled"/>	RADIUS Proxy Attribute	<input type="text" value="0"/>
Add X-Forwarded-For to HTTP requests	<input type="text" value="Disabled"/>	Insert Cookie for HTTP Persistency	<input type="text" value="Disabled"/>
Hash Parameter For SIP	<input type="text" value="CallID"/>	SSL ID Aging	<input type="text" value="120"/>
Select Server Per Transaction	<input type="text" value="Enabled"/>		

- Click the **Set** button to save parameters.

#### 5.1.4. Create Layer 4 Policy

- From the menu, select **AppDirector** ⇒ **Layer 4 Farm Selection** ⇒ **Layer 4 Policy Table** to display the **Layer 4 Policy Table** page similar to the one shown below:

Layer 4 Policy Table
?

[Farm Table](#)   [Layer 7 Policy Table](#)   [Layer 4 Policy Statistics](#)

Virtual IP	L4 Protocol	L4 Port	Source IP From	L4 Policy Name	L7 Policy	Farm Name	
<div style="display: flex; justify-content: center; gap: 20px;"> </div>							

- Click the **Create** button.
- On the **Layer 4 Policy Table Create** page, enter the necessary parameters as shown below.

### Layer 4 Policy Table Create

[Farm Table](#)   [Layer 7 Policy Table](#)   [Layer 4 Policy Statistics](#)   [Help](#)

Virtual IP	<input type="text" value="192.168.43.105"/>	L4 Protocol	<input type="text" value="TCP"/>
L4 Port	<input type="text" value="Any"/>	Source IP From	<input type="text" value="0.0.0.0"/>
L4 Policy Name	<input type="text" value="Web Servers"/>	Source IP To	<input type="text" value="0.0.0.0"/>
Farm Name	<input type="text" value="WebServers"/>	L7 Policy Name	<input type="text" value="None"/>
Application	<input type="text" value="HTTP"/>	Redundancy Status	<input type="text" value="Primary"/>
Backend Encryption Port	<input type="text" value="0"/>	Bytes of Request to Read	<input type="text" value="3584"/>
POST Classification Input	<input type="text" value="Header"/>	HTTP Normalization	<input type="text" value="Disabled"/>
L7 Persistent Switching Mode	<input type="text" value="Overwrite"/>	Segment Name	<input type="text"/>

4. Click the **Set** button to save the parameters.
5. Verify that the new entry was created on the **Layer 4 Policy Table** page:

### Layer 4 Policy Table

[Farm Table](#)   [Layer 7 Policy Table](#)   [Layer 4 Policy Statistics](#)   [Help](#)

Virtual IP	L4 Protocol	L4 Port	Source IP From	L4 Policy Name	L7 Policy	Farm Name
<a href="#">192.168.43.105</a>	TCP	Any	0.0.0.0	WebServers	None	WebServers

### 5.1.5. Configure Dynamic L7 Persistency

1. From the menu, select **AppDirector** ⇒ **L7 Server Persistency** ⇒ **Text Match** to display the **Text Match Session ID Persistency** page similar to the one shown

### Text Match Session ID Persistency

[Pattern Match Session ID Persistency](#)   [Static Session ID Persistency](#)   [Farm Table](#)   [Help](#)

Farm Name	Application Port	L4 Protocol	Persistency Identifier	Learning Direction	Ignore Server Reply	✕
-----------	------------------	-------------	------------------------	--------------------	---------------------	---

2. Click the **Create** button.



- On the **Text Match Session ID Persistency Create** page, enter the necessary parameters as shown below.

Farm Name:	<input type="text" value="WebServers"/>	Application Port:	<input type="text" value="0"/>
L4 Protocol:	<input type="text" value="TCP"/>	Persistency Identifier:	<input type="text" value="ASPSESSIONID"/>
Lookup Mode:	<input type="text" value="Cookie"/>	Identifier Match:	<input type="text" value="Prefix"/>
Learning Direction:	<input type="text" value="Server Reply"/>	Ignore Server Reply:	<input type="text" value="Never"/>
Value Max Length:	<input type="text" value="256"/>	Value Offset:	<input type="text" value="24"/>
Stop Chars:	<input type="text" value=";"/>	Inactivity Timeout:	<input type="text" value="900"/>
Ignore Source IP:	<input type="text" value="Enabled"/>		

- Click the **Set** button to save parameters.
- Verify that the new entry was created on the **Text Match Session ID Persistency** page:

**Text Match Session ID Persistency** ? Help

[Pattern Match Session ID Persistency](#)
[Static Session ID Persistency](#)
[Farm Table](#)

Farm Name	Application Port	L4 Protocol	Persistency Identifier	Learning Direction	Ignore Server Reply
<a href="#">WebServers</a>	0	TCP	SESSIONID	Server Reply	Never

### 5.1.6. Adding Servers to the Farm

- From the menu, select **AppDirector** ⇒ **Servers** ⇒ **Application Servers** to display the **Server Table** page similar to the one shown below:

**Server Table** ? Help

[Farm Table](#)
[Physical Servers](#)
[Static Session ID Persistency](#)

Farm Name	Server Address	Server Port	Server Name	Operational Status	Operation Mode	Admin Status	Attached Users	Peak Load	Frames Load	Type
-----------	----------------	-------------	-------------	--------------------	----------------	--------------	----------------	-----------	-------------	------

2. On the **Server Table Create** page, enter the necessary parameters as shown below:

?
Hel

Server Table Update

[Farm Table](#)
[Physical Servers](#)
[Static Session ID Persistence](#)

**Farm Name:**

**Server Port:**

**Server Description:**

**Operational Status:**

**Operation Mode:**

**Connection Limit:**

**Client NAT:**

**Redirect To:**

**Attached Users:**

**Frames Rate:**

**Client NAT Address Range:**

**Server Address:**

**Server Name:**

**Admin Status:**

**Weight:**

**Type:**

**Response Threshold [ms]:**

**Backup Server Address:**

**Bandwidth Limit:**

**Peak Load:**

**Backup Preemption:**

Set
Cancel

3. Click the **Set** button to save parameters.
4. Verify that the new entry was created on the **Server Table** page:


?
Hel

Server Table

[Farm Table](#)
[Physical Servers](#)
[Static Session ID Persistence](#)

Farm Name	Server Address	Server Port	Server Name	Operational Status	Operation Mode	Admin Status	Attached Users	Peak Load
<a href="#">WebServers</a>	192.168.43.199	None	Eagle	Not In Service	Regular	Enable	0	0



5. Create the second server using the information below:


  
[Help](#)


### Server Table Update

[Farm Table](#)   [Physical Servers](#)   [Static Session ID Persistence](#)

<b>Farm Name:</b> WebServers <b>Server Port:</b> None <b>Server Description:</b> GVP WebServers <b>Operational Status:</b> Not In Service <b>Operation Mode:</b> Regular ▾ <b>Connection Limit:</b> 0 <b>Client NAT:</b> Disabled ▾ <b>Redirect To:</b> <input type="text"/> <b>Attached Users:</b> 0 <b>Frames Rate:</b> 0 <b>Client NAT Address Range:</b> 0.0.0.0 ▾	<b>Server Address:</b> 192.168.43.200 <b>Server Name:</b> Hornet <b>Admin Status:</b> Enable ▾ <b>Weight:</b> 1 <b>Type:</b> Regular ▾ <b>Response Threshold [ms]:</b> 0 <b>Backup Server Address:</b> 0.0.0.0 ▾ <b>Bandwidth Limit:</b> No Limit ▾ <b>Peak Load:</b> 0 <b>Backup Preemption:</b> Enable ▾
--	---

   
  
 Set   Cancel


6. Click the **Set** button to save parameters.
7. Create the third server using the information below:


  
[Help](#)

### Server Table Update

[Farm Table](#)   [Physical Servers](#)   [Static Session ID Persistence](#)

<b>Farm Name:</b> WebServers <b>Server Port:</b> None <b>Server Description:</b> GVP WebServers <b>Operational Status:</b> Not In Service <b>Operation Mode:</b> Regular ▾ <b>Connection Limit:</b> 0 <b>Client NAT:</b> Disabled ▾ <b>Redirect To:</b> <input type="text"/> <b>Attached Users:</b> 0 <b>Frames Rate:</b> 0 <b>Client NAT Address Range:</b> 0.0.0.0 ▾	<b>Server Address:</b> 192.168.43.204 <b>Server Name:</b> Tomcat <b>Admin Status:</b> Enable ▾ <b>Weight:</b> 1 <b>Type:</b> Regular ▾ <b>Response Threshold [ms]:</b> 0 <b>Backup Server Address:</b> 0.0.0.0 ▾ <b>Bandwidth Limit:</b> No Limit ▾ <b>Peak Load:</b> 0 <b>Backup Preemption:</b> Enable ▾
--	---

   
  
 Set   Cancel

8. Click the **Set** button to save parameters.

### 5.1.7. Health Monitoring Configuration

1. From the menu, select **Health Monitoring** ⇒ **Global Parameters** to display the **Health Monitoring Global Parameters** page.
2. On the **Health Monitoring Global Parameters** page, change the parameters as shown below:

**Health Monitoring Global Parameters**

[Check Table](#)    [Binding Table](#)    [HM Server Table](#)

**Health Monitoring Status:**

**Response Level Samples:**

**SSL Certificate File:**



**SSL Private Key File:**

  
**Set**

3. Click the **Set** button to save parameters.
4. Create the Health Monitoring Check for the First Server.
5. From the menu, select **Health Monitoring** ⇒ **Check Table** to display the **Health Monitoring Check Table** page similar to the one shown below:

**Health Monitoring Check Table**

[Binding Table](#)    [Packet Sequence Table](#)    [Health Monitoring Global Parameters](#)

Check Name	Check ID	Method	Status	Dest Host	
<div style="display: flex; justify-content: center; gap: 20px;"> <div style="text-align: center;">   <b>Delete</b> </div> <div style="text-align: center;">   <b>Create</b> </div> </div>					

6. Click the **Create** button.
7. On the HM Check Table Create page, enter the necessary parameters as shown below:

## HM Check Table Update

[Binding Table](#)   
 [Packet Sequence Table](#)   
 [Health Monitoring Global Parameters](#)

<b>Check Name:</b>	Eagle_WebServers	<b>Method:</b>	HTTP
<b>Dest Host:</b>	192.168.43.199	<b>Next Hop:</b>	0.0.0.0
<b>Destination Port:</b>	0	<b>Arguments:</b>	PATH=GVP HOST=192... <span style="font-size: small;">...</span>
<b>Interval:</b>	20	<b>Retries:</b>	3
<b>Timeout:</b>	1	<b>No New Session Timeout:</b>	0
<b>Measure Response Time:</b>	Disabled <span style="font-size: small;">v</span>	<b>Response Level:</b>	0
<b>Check ID:</b>	0	<b>Status:</b>	Failed
<b>Dest IP:</b>	0.0.0.0	<b>Reverse Check Result:</b>	disable <span style="font-size: small;">v</span>
<b>Uptime %:</b>	0	<b>Success Counter:</b>	0
<b>Failure Counter:</b>	4218	<b>Average Duration:</b>	0

8. Before clicking the Set button, choose the button next to Arguments ... to populate the specific settings for the rest of this check.
9. Enter the information below:

### Arguments for HTTP Method

<b>Path:</b>	GVP
<b>Hostname:</b>	192.168.43.199
<b>HTTP Method:</b>	GET <span style="font-size: small;">v</span>
<b>Proxy HTTP:</b>	No <span style="font-size: small;">v</span>
<b>Pragma Nocache:</b>	Yes <span style="font-size: small;">v</span>
<b>Username:</b>	<input type="text"/>
<b>Password:</b>	<input type="text"/>
<b>Match search string:</b>	<input type="text"/>
<b>Match mode:</b>	<input type="text"/> <span style="font-size: x-small;">Lookup stri</span>
<b>HTTP return code:</b>	200
<b>HTTP return code:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text"/>

10. Click the **Set** button for the Method Arguments and click the **Set** button again in the HM Check Table Create window.
11. You should have a single entry in the Health Monitoring Check Table:

## Health Monitoring Check Table

[Binding Table](#)    [Packet Sequence Table](#)    [Health Monitoring Global Parameters](#)



Check Name	Check ID	Method	Status	Dest Host	✕
<a href="#">Eagle WebServers</a>	0	HTTP	Failed	192.168.43.199	<input type="checkbox"/>

Note: The status of this check may display “Unknown” until the server replies successfully to the AppDirector’s check.

12. Create the Health Monitoring Check for the Second Server
13. If the **Health Monitoring Check Table** page is not already displayed from the previous step, select **Health Monitoring** ⇒ **Check Table** from the menu.
14. Click the **Create** button.
15. On the **HM Check Table Create** page, enter the necessary parameters as shown below:

## HM Check Table Update



[Binding Table](#)    [Packet Sequence Table](#)    [Health Monitoring Global Parameters](#)

<b>Check Name:</b>	Hornet_WebServers	<b>Method:</b>	HTTP
<b>Dest Host:</b>	192.168.43.200	<b>Next Hop:</b>	0.0.0.0
<b>Destination Port:</b>	0	<b>Arguments:</b>	PATH=GVP HOST=192 ...
<b>Interval:</b>	20	<b>Retries:</b>	3
<b>Timeout:</b>	1	<b>No New Session Timeout:</b>	0
<b>Measure Response Time:</b>	Disabled ▾	<b>Response Level:</b>	0
<b>Check ID:</b>	1	<b>Status:</b>	Failed
<b>Dest IP:</b>	0.0.0.0	<b>Reverse Check Result:</b>	disable ▾
<b>Uptime %:</b>	0	<b>Success Counter:</b>	0
<b>Failure Counter:</b>	4238	<b>Average Duration:</b>	0
 			
Set		Cancel	

16. Click the button next to the **Arguments**  text box to configure the check specific arguments:

**Arguments for HTTP Method**

<b>Path:</b>	<input type="text" value="GVP"/>
<b>Hostname:</b>	<input type="text" value="192.168.43.200"/>
<b>HTTP Method:</b>	<input type="text" value="GET"/>
<b>Proxy HTTP:</b>	<input type="text" value="No"/>
<b>Pragma Nocache:</b>	<input type="text" value="Yes"/>
<b>Username:</b>	<input type="text"/>
<b>Password:</b>	<input type="text"/>
<b>Match search string:</b>	<input type="text"/>
<b>Match mode:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text" value="200"/>
<b>HTTP return code:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text"/>

   
Set Cancel

17. Click the **Set** button to save the Method Argument parameters.
18. Verify that the Arguments text box has been populated.
19. Click the **Set** button to save the Health Check.
20. Create the Health Monitoring Check for the Third Server
21. If the **Health Monitoring Check Table** page is not already displayed from the previous step, select **Health Monitoring** ⇒ **Check Table** from the menu.
22. Click the **Create** button.
23. On the **HM Check Table Create** page, enter the necessary parameters as shown below:

## HM Check Table Update

[Binding Table](#)   
 [Packet Sequence Table](#)   
 [Health Monitoring Global Parameters](#)

<b>Check Name:</b>	Tomcat_WebServers	<b>Method:</b>	HTTP
<b>Dest Host:</b>	192.168.43.204	<b>Next Hop:</b>	0.0.0.0
<b>Destination Port:</b>	0	<b>Arguments:</b>	PATH=GVP HOST=192 ...
<b>Interval:</b>	20	<b>Retries:</b>	3
<b>Timeout:</b>	1	<b>No New Session Timeout:</b>	0
<b>Measure Response Time:</b>	Disabled ▾	<b>Response Level:</b>	0
<b>Check ID:</b>	2	<b>Status:</b>	Failed
<b>Dest IP:</b>	0.0.0.0	<b>Reverse Check Result:</b>	disable ▾
<b>Uptime %:</b>	0	<b>Success Counter:</b>	0
<b>Failure Counter:</b>	4249	<b>Average Duration:</b>	0

24. Click the button next to the **Arguments** ... text box to configure the check specific arguments:

### Arguments for HTTP Method

<b>Path:</b>	GVP
<b>Hostname:</b>	192.168.43.204
<b>HTTP Method:</b>	GET ▾
<b>Proxy HTTP:</b>	No ▾
<b>Pragma Nocache:</b>	Yes ▾
<b>Username:</b>	<input type="text"/>
<b>Password:</b>	<input type="text"/>
<b>Match search string:</b>	<input type="text"/>
<b>Match mode:</b>	▾
<b>HTTP return code:</b>	200
<b>HTTP return code:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text"/>
<b>HTTP return code:</b>	<input type="text"/>





25. Click the **Set** button to save the Method Argument parameters.
26. Verify that the Arguments text box has been populated.
27. Click the **Set** button to save the Health Check.
28. Binding Health Checks to Servers
29. Create the Health Monitoring Binding for the First Server
30. From the menu, select **Health Monitoring** ⇒ **Binding Table** to display the **Health Monitoring Binding Table** page similar to the one shown below:

## Health Monitoring Binding Table

[Check Table](#)   [HM Server Table](#)   [Health Monitoring Global Parameters](#)

Check	Server/NHR/Report	Group	Mandatory	X
-------	-------------------	-------	-----------	---



  
Delete   Create



31. Click the **Create** button.
32. On the **HM Binding Table Create** page, enter the necessary parameters as shown below:

## HM Binding Table Update

[Check Table](#)   [HM Server Table](#)   [Health Monitoring Global Parameters](#)

**Check:**    **Server/NHR/Report:**

**Group:**    **Mandatory:**



  
Set   Cancel

33. Click the **Set** button to save parameters.
34. Verify that the new entry was created on the **Health Monitoring Table** page:

## Health Monitoring Binding Table

[Check Table](#)[HM Server Table](#)[Health Monitoring Global Parameters](#)


Check	Server/NHR/Report	Group	Mandatory	✕
<a href="#">Eagle WebServers</a>	Farm WebServers - 19 2.168.43.199 - 0	0	Mandatory	<input type="checkbox"/>

35. Create the Health Monitoring Binding for the Second Server
36. If the **Health Monitoring Binding Table** page is not already displayed from the previous step, select **Health Monitoring** ⇒ **Binding Table** from the menu.
37. Click the **Create** button.
38. On the **HM Binding Table Create** page, enter the necessary parameters as shown below:

### HM Binding Table Update

[Check Table](#)   [HM Server Table](#)   [Health Monitoring Global Parameters](#)

<b>Check:</b> <span style="border: 1px solid red; border-radius: 5px; padding: 2px;">Hornet_WebServers</span>	<b>Server/NHR/Report:</b> <span style="border: 1px solid red; border-radius: 5px; padding: 2px;">Farm WebServers - 192.168.43.200 - 0</span>
<b>Group:</b> <input style="width: 150px;" type="text" value="0"/>	<b>Mandatory:</b> <span style="border: 1px solid #ccc; padding: 2px;">Mandatory</span> ▼



   
**Set**   **Cancel**

39. Click the **Set** button to save parameters.
40. Create the Health Monitoring Binding for the Third Server
41. If the **Health Monitoring Binding Table** page is not already displayed from the previous step, select **Health Monitoring** ⇒ **Binding Table** from the menu.
42. Click the **Create** button.
43. On the **HM Binding Table Create** page, enter the necessary parameters as shown below:

### HM Binding Table Update

[Check Table](#)   [HM Server Table](#)   [Health Monitoring Global Parameters](#)

<b>Check:</b> <span style="border: 1px solid red; border-radius: 5px; padding: 2px;">Tomcat_WebServers</span>	<b>Server/NHR/Report:</b> <span style="border: 1px solid red; border-radius: 5px; padding: 2px;">Farm WebServers - 192.168.43.204 - 0</span>
<b>Group:</b> <input style="width: 150px;" type="text" value="0"/>	<b>Mandatory:</b> <span style="border: 1px solid #ccc; padding: 2px;">Mandatory</span> ▼

   
**Set**   **Cancel**

44. Click the **Set** button to save parameters.

This completes the AppDirector Configuration.

## **6. Results and Observations**

### **6.1. Functionality**

Extensive testing of AppDirector with GVP applications utilizing DTMF and voice inputs resulted in confirmation that load balancing with session persistence functions correctly. All requests belonging to a single session are load balanced to the same web server and load is evenly divided among web servers. The dynamic addition and removal of web servers and IPCS' can be achieved with no interruption to service. Failure of a web server does result in some calls being lost, as expected, but recovery time is short and recovery is complete. A moderate load was directed to the GVP applications in order that pipelining would be initiated and session persistence proven. However, further testing would be warranted to expose any issues that might arise under a heavy load of calls.

### **6.2. Latency**

One downside of persisting at the session level is that inevitably, a certain amount of latency is introduced. This is the result of the load balancer having to inspect every packet and make a decision based on a lookup in the dynamic table containing sessions. The overhead introduced in the testing was noticeable, but did not seem to have a detrimental effect on the sessions. Again, further testing is necessary to determine the level of latency introduced when the system is under load.

### **6.3. Management of Dynamic Table**

Careful consideration needs to be given to the management of the size of the dynamic session table. This table contains an entry for every session detailing the unique session id and the server that is servicing that call session. An inactivity timeout exists whereby if no traffic for a session is received during the length of the timeout period, the session entry is removed from the table. This timeout value should be set greater than the longest holding time possible within the GVP applications. It should not however be set to an arbitrarily large value as this may cause the table to grow to a point where performance would be impacted either by causing longer lookup times or by the table running out of space.

The entries in this table could be kept to a minimum by having the server set a cookie on each session which contains the server name. This cookie would be set by the web server on the first request of a session. GVP then propagates the cookie through all dynamic requests for the session. If the load balancer were to use this identifier to establish and maintain persistence for a session, then the dynamic table would consist of only as many entries as there are web servers. The size of the table would not be an issue and aging of the table could be set at a maximum value. However, this method of implementation requires some maintenance at the web server or application level that is not required when using the unique session identifiers.

### **6.4. Benefits**

There are additional benefits that can be realized from load balancing aside from the obvious benefit of having more evenly balanced load on the web servers. Health monitoring can be implemented to detect problems with the web servers. As described above, how the load balancer deals with any detected failures will depend on the persistence method implemented. Session persistence will provide the most efficient handling of failures. As well, without load balancing, GVP allows a primary and backup web server per application. Load Balancing can provide redundancy capabilities that are more flexible than are possible without its implementation. To eliminate a single point of failure however, more than one load balancer would need to be deployed.

## 7. Appendix A – Test Environment Description

### 7.1. Hardware and Software Environment

Please see the table below for the distribution of processes on the LSST lab hardware. For each machine there is a list of the software configured and the OS installed. Sections 7.2 and 7.3 in Appendix A provide detailed information about the machine and software versions respectively.

<i>Host Name</i>	<i>(See Section 7.3 for software versions)</i> <b>Software Configuration</b>	<i>(See Section 7.2 for HW specs)</i> <b>Operating System</b>
Blackbird	1 IPCS	Windows 2003 Server SP1
Chengdu	1 IPCS	Windows 2003 Server SP1
Draken	1 IPCS	Windows 2003 Server SP1
Eagle	Microsoft IIS web server, Apache Tomcat web server, SQL Server (EMPS and Application dB)	Windows 2003 Server SP1
Falcon	EMPS	Windows 2003 Server SP1
Flanker	EMS: Policy Manager and Bandwidth Manager	Windows 2003 Server SP1
Fulcrum	IPCM: Resource Manager and SIP Session Manager	Windows 2003 Server SP1
Gripen	ASR/TTS Server	Windows 2003 Server SP1
Hornet	Microsoft IIS web server, Apache Tomcat web server	Windows 2003 Server SP1
Nighthawk	StreamManager, CCAS	Windows 2003 Server SP1
Thunderbolt	StreamManager, CCAS	Windows 2003 Server SP1
Tomcat	Microsoft IIS web server, Apache Tomcat web server	Windows 2003 Server SP1
AppDirector	Software Version – 1.02.05DL	APSolute OS

### 7.2. Environment Hardware List

<i>Description</i>	<i>Host Name</i>	<i>RAM</i>	<i>CPU</i>	<i>Hard Drives</i>
Sun Sunfire V60x	Blackbird	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Chengdu	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Draken	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Eagle	2GB	Dual 3.0 GHz Xeon	73 GB
Sun Sunfire V60x	Falcon	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Flanker	2GB	Dual 3.0 GHz Xeon	73 GB
Sun Sunfire V60x	Fulcrum	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Gripen	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Hornet	2GB	Dual 3.0 GHz Xeon	73 GB
Sun Sunfire V60x	Nighthawk	2GB	Dual 3.0 GHz Xeon	73 GB
Sun Sunfire V60x	Thunderbolt	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Sun Sunfire V60x	Tomcat	2GB	Dual 3.0 GHz Xeon	Dual 73 GB
Application Server 2	AppDirector			

### 7.3. Software Versions List

<i>Software</i>	<i>Windows 2003 Server SP1</i>	<i>APSolute OS</i>
GVP Network Edition	7.2.001.00	-
Common Components	7.2.015.23	-
IPCS	7.2.015.20	-
Bandwidth Manager	7.2.015.09	-
EMPS	7.2.001.05	-
Policy Manager	7.2.015.04	-

Scripts for LDAP	7.2.000.07	-
Resource Manager	7.2.015.12	-
SIP Session Manager	7.2.015.09	-
Text To Speech Server	7.2.015.04	-
Studio	7.2.012.10	-
Radware Application Switch 2		AppDirector 1.06.09DL(56)
Apache Tomcat	5.0	-
Microsoft Internet Information Services Manager	6.0	-
Nuance OSR		
Apache HTTP Server	1.3.33	-
Open Speech Recognizer	3.0.8	-
RealSpeak	4.0.8	-
SpeechWorks Media Server	3.1.9	-

## 8. Appendix B – Diagram of Test Environment

