Aqua Connect Load Balancer User Manual (Linux)

Aqua Connect

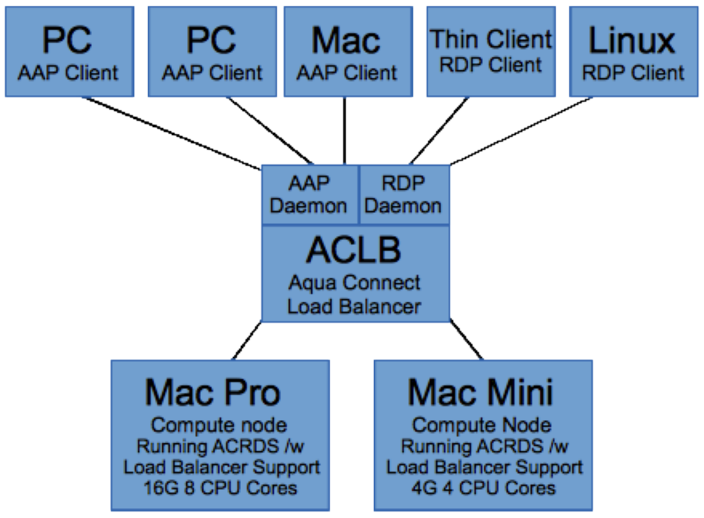
Table of Contents

About Aqua Connect Load Balancer

The Aqua Connect Load Balancer (**ACLB**) will enable your organization to make more efficient use of your Aqua Connect Remote Desktop Services (**ACRDS**) deployment, and can provide an overall better user experience. **ACLB** will intelligently distribute sessions among it's *Compute Nodes* (machines on **ACLB**'s active server list that have **ACRDS with Load Balancer Support** installed).

Additionally, **ACLB** can greatly enhance network security by being an outward facing service allowing for a single TCP/IP address (or multiple addresses in a high availability configuration) and port to be used for all incoming remote desktop services clients.

The intelligent load balancing algorithm takes into account current system load, processing power, and free capacity. Because of this, you can make use of less capable Intel-based Apple Mac OS X hardware within your **ACLB** cluster without balancing issues—although, for a consistent user experience, it is highly recommended that all compute nodes have the same software and Mac OS X version installed.



In the example above, assuming both systems were serving similar tasks, then more **ACRDS** sessions would be assigned to the Mac Pro instead of the Mac Mini because of it's higher capacity. Now, if a user opted to directly connect to the Mac Pro, bypassing the load balancer, to run some very intensive CPU loads, then the load balancer is intelligent enough to see that the Mac Pro is under heavy load and then will start assigning more sessions to the Mac Mini.

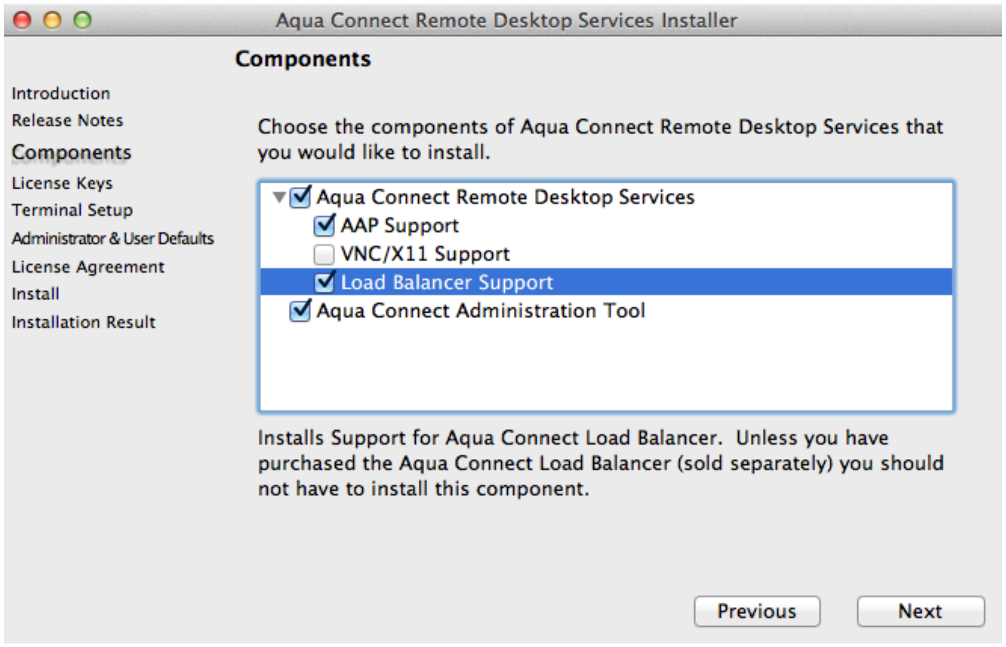
System Requirements

ACLB is only supported for x86 Linux systems. However, the Linux installer ships with a FreeBSD-9.2 (amd64) binary and installer as well. 2 GB of RAM and 20 MB of free disk space is recommended. Ports **310** and **3389** need to be unblocked on both the **ACLB** and **ACRDS** machines; additionally, port **3388** needs to be unblocked on the **ACRDS** compute nodes.

Python 2.6 is required for the Web Console (already installed on most Linux systems, FreeBSD installer will automatically install python as part of the installation process). If Python 2.6 is not installed, or missing some required modules, or /usr/bin/python links to python 3.0 or higher, the web console will not install. In which case, refer to **ACLoad(8)** man page for manual configuration instructions.

Mac OS X ACRDS Server Computer Node Setup

Before **ACLB** can be used, the compute nodes need to be configured with **ACRDS Load Balancer Support** installed. There are also special considerations to take when deciding on how to deal with active, but disconnected sessions.



Choose the components of Aqua Connect Remote Desktop Services that you would like to install.

- [✓] Aqua Connect Remote Desktop Services
  - [✓] AAP Support
  - [ ] VNC/X11 Support
  - [✓] Load Balancer Support
- [✓] Aqua Connect Administration Tool

Installs Support for Aqua Connect Load Balancer. Unless you have purchased the Aqua Connect Load Balancer (sold separately) you should not have to install this component.

If you haven't already done so, install (or reinstall) **ACRDS** with the "**Load Balancer Support**" option checked. Without this package installed, the load balancer will not recognize the systems as being a valid compute node.

You'll also need to ensure port **3389**, **310**, and **3388** are not blocked by your firewall, Apple's Application Firewall, and Mac OS X/Darwin's IPFW (part of Mac OS X Server).

Managing Inactive Sessions

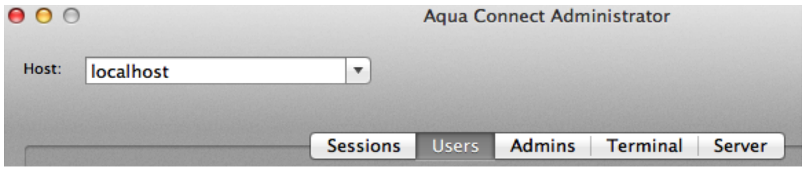
One of the issues that can occur, even though may have 10, 30, or even 250 combined total licensed sessions for all machines in your cluster, you will not be able to utilize all of them if users disconnect from a compute node without logging off by clicking on the Apple logo first. This will leave the session active, and when the user connects to the load balancer again, they may be assigned to a different compute node. If this happens, you now have two active sessions for the same user: one that the user disconnected from and is not actively being used (and counts against your total licensed sessions), and another that the user is actively using.

To resolve this, and ensure that you are getting the most value out of your investment, you can configure each compute node to automatically close sessions when the connection is dropped via the **ACAdminTools**, or use **ACTO** to kill sessions that have been inactive for more than a set period of time (which could allow a user to re-connect and save their work if a connection was dropped; however, the load balancer may not always reconnect them to the same compute node, so they would need to connect directly to the compute node, bypassing the load balancer).

Option 1: ACAdminTools

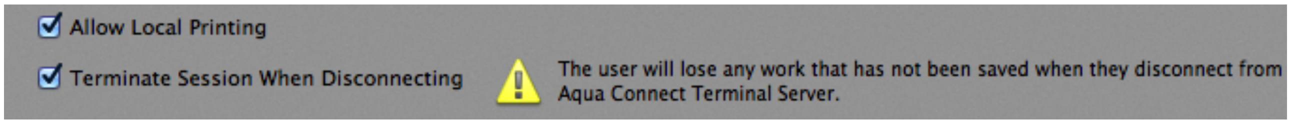
ACAdminTools is found in the Finder under Applications/Server. The default admin login is **acadmin** and password is **pass** (unless you picked a different password when installing ACAdminTools, or changed the login from the “Admins” tab within ACAdminTools—and if you haven't done so already, you should do so now).

To configure an **ACRDS** server to automatically close seasons when a connection ends:



Host: localhost

1. Click on the “Users” tab.
2. Select the users you want to terminate disconnected sessions for (press ⌘A to select all users)
3. And finally, check the “Terminate Session When Disconnecting” box at the bottom of the window.



☑ Allow Local Printing

☑ Terminate Session When Disconnecting

The user will lose any work that has not been saved when they disconnect from Aqua Connect Terminal Server.

You may not want some users to terminate when disconnecting, such as those that have long running computational problems that may take hours, or weeks, to complete. Even though compute nodes are part of a cluster, they can be directly connected to, bypassing the load balancer (provided there are available licensed sessions). Also, the ACAdminTool works on a per-machine basis, so you do not need to disable this feature for a user on all the machines, only the machine they need for the long running process.

Option 2: Installing and Configuring ACTO

As of this writing, the current version of ACTO is available at:
http://www.aquaconnect.net/downloads/ACTO2.dmg

By default **ACTO** will log users off after a time, but applications may ask users if they wish to save their work. This will keep the session open after a user disconnects, but will not solve the potential problem mentioned above. To resolve this, we will need to configure **ACTO** to do a forced logout, which will immediately kill the session after there has been no mouse nor keyboard activity sent to the session for a set period of time (ten minutes by default). Like using the **ACAdminTool** to terminate sessions when disconnecting, any unsaved work will be lost, but the session will be free for others to use later.

To do this, we need to edit the **/etc/ACTO.plist** file and make some changes. Forced logouts will close a user's session if there has been no keyboard nor mouse activity for MaxTimeInterval seconds. Change the **Forced** key from **false** to **true**, and optionally, set a new **MaxTimeInterval** value.

Default /etc/ACTO.plist	**Edited /etc/ACTO.plist**
<?xml version="1.0" encoding="UTF-8"?>	<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"	<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">	"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">	<plist version="1.0">
<dict>	<dict>
<key>Enabled</key>	<key>Enabled</key>
<true/>	<true/>
<key>**Force**</key>	<key>**Force**</key>
<**false**/>	<**true**/>
<key>**MaxTimeInterval**</key>	<key>**MaxTimeInterval**</key>
<string>**600**</string>	<string>**600**</string>
<key>Method</key>	<key>Method</key>
<string>1</string>	<string>1</string>
<key>TimerInterval</key>	<key>TimerInterval</key>
<string>60</string>	<string>60</string>
</dict>	</dict>
</plist>	</plist>

Editing /etc/ACTO.plist file for forced logouts

If a session timeout of 3 minutes (180 seconds) or less is given, the session timeout will be 3 minutes to prevent issues with some users mistaking this field for minutes instead of seconds.

Using ACTO is the easiest way to enforce session logouts on a system, but it offers less control than the **ACAdminTool**. Of course, you could always temporarily set a server as “inactive” in the **ACWebconsole**, then disable ACTO for new sessions by changing the **Enabled** key to **false** for that one machine, if it's needed for a long running process as well.

Linux/FreeBSD Installer

The **.run** file is a self-extracting installer. Simply run it as a Bourne shell script, then it'll attempt to detect your system type, and begin the semi-automated installation process. 64-bit FreeBSD support was tacked on as an added bonus by the resident Unix geek, and is not something we directly support and should be considered “experimental.”

You can pass the **.run** file parameters:

• oscheck	- Checks if the installer is compatible with your system
• uninstall	- uninstalls the software
• -h	- help

For most Linux (and 64-bit FreeBSD-9.2 users), all you need to do is run the following command as root:

```
sh ACLoad-installer.run
```

After installation, the load balancer and web console (if your system met the python requirements) will have installed and started, and will be waiting for you to visit **http://localhost:3300** with either a text or graphical web browser to start adding compute nodes to your new **ACLB** cluster.

Linux/FreeBSD Installer Details

The Universal Linux/FreeBSD Installer has been tested with with Ubuntu 12.04, CentOS 6.4, OpenMandriva LX 2013.0 Beta, Fedora 19, Debian 7, Slackware 14, and FreeBSD-9.2 (amd64 only). The installer can detect the different startup methods used by modern, and increasingly complex, Linux distributions. So, even if your Linux distribution isn't one we tested against, you may still be able to install on your system, provided it is a x86 or x86_64 system.

For FreeBSD 9.2, a separate installer launches to work around a bug with FreeBSD's sh and pkg_add programs that causes the shell to cease accepting user input after pkg_add is run. The FreeBSD installer will install python 2.7, if it's not currently present on your system, and add the standard /usr/bin/python link. It is largely an automatic process, although the initial admin user creation for the web console is left for the end user.

The Linux installer **does not** install python. Python was installed as a standard component on all Linux distributions we tested against. If your system lacks Python, the load balancer will still install and work, but you'll lack the web console interface. The **ACLoad(8)** man page provides detailed instructions on how to setup the load balancer if you did not install the web console. The man page will always be up to date with your current installed version.

For the web console to install **/usr/bin/python** must exist, and be version 2.6 and above, but not version 3.0 and above. All standard python 2.6 modules should also be installed.

AC Web Console

The **ACWebconsole** is a python-based HTTP server that listends on port **3300** by default. This can be changed by editing the Config.py file (see the **ACWebconsole(8)** man page for more more information).

Some forms have “Update form” and “Save Configuration” buttons. The “Update” button doesn't save your configuration, it merely cleans up the web form and adds an extra entry field below (for text web browsers, those that use graphical browser will have rows added for them automatically, unless Java/ECMAScript is disabled).

User Accounts

The web console supports two user classes: admins, and users. Users are only able to view cluster status. Admin users, on the other hand, can add, remove, and mark servers as inactive, as well as add/remove users, and set IP address restrictions for the web console interface.

Should you ever forget your password, you can change it with the command line tool as root: **/usr/local/sbin/Acwebcon/addedituser.py**

Users can change their passwords from the “Profile” page.

Manage Cluster

The cluster manager can take either IP addresses or domain names. There are two check boxes, one to delete the entry, and another to mark the cluster member as active or not. Marking a server as inactive allows you to temporarily remove a machine from the cluster, while keeping it's configuration saved.

Status

The status page shows the current status of the machines on your cluster. It automatically updates after a period of time using AJAX requests, as such, the status page does not work in a text-based browser.

The percentage fields show the amount of resources free, so 100% means no utilization, and 0% means you're capping out the system resources.

Inactive servers will appear at the bottom of the status page, with a grayed out field and the words “inactive” behind them. Stats will still be pulled from these machines if they are online, otherwise the fields will be blank if they did not respond to requests made on port 3388, or if the **ACRDS Load Balancer Support** was not installed.

Allowed IP Range

This page allows you to restrict which IP addresses can use the web console. It won't allow you to save the IP list if it'll block your current IP from accessing the machine.

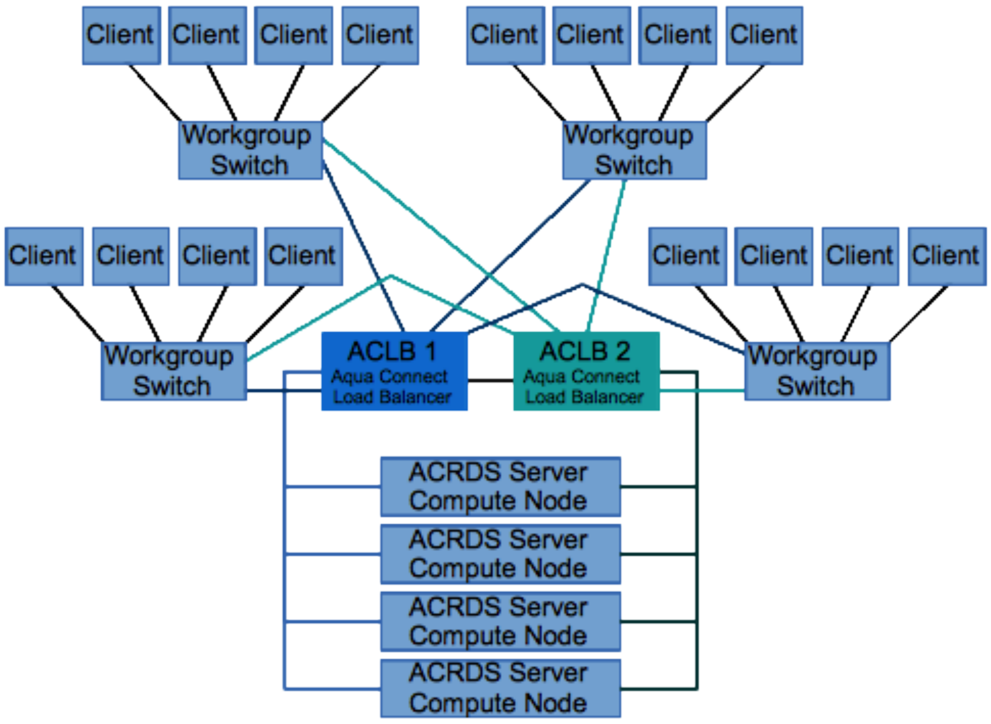
If you inadvertently locked yourself out of the web console system, delete **/etc/ACT-LB/http/data/iplist.dat** and the system will revert to the default settings, which is to allow all connections.

Cluster Design

For best performance, the load balancer should have at least two network interfaces, with one dedicated for connecting to the compute nodes, and the other for client connections. Further performance increases can be achieved by installing multi-port Ethernet cards. Intel-branded cards seem to perform better that most other brands.

The example diagram below is for a fault-tolerant **ACLB**, provided that the Linux machines are configured to operate in a HA/failover capacity. There are many articles online on how to do this. Seeking assistance from a Network Engineer and System Administrator is highly recommended when designing and implementing a large and/or fault-tolerant cluster.

Using Activity Monitor on Macs, and iftop on Linux systems, can help determine if you are saturating your network interface, as well as SNMP and associated monitoring software for managed switches that support the protocol.


Client
Workgroup Switch
ACLB 1
Aqua Connect Load Balancer
ACLB 2
Aqua Connect Load Balancer
ACRDS Server Compute Node

Aqua Connect Load Balancer Technical Details

The load balancer works in a similar way to a proxy, accepting external connections, and then forwarding those connections to other machines. If your network is properly designed, this can serve to act as a buffer between your internal network, and the external world. How incoming connections are routed is determined by a proprietary algorithm that finds which systems are the least busy.

The default setup launches two ACLoad processes: one to handle RDP, and a second to process AAP requests (when the startup scripts/definitions are installed, these are given the name of ACLoad-aap and ACLoad-rdp). Each of these can accept a total of 250 connections, so 500 connections in total. This is currently a hard-coded limit, although it will be made configurable in future versions. In the *ACLB High-Availability Cluster Example* diagram, this setup would be able to accept 1000 (500 AAP, 500 RDP) total connections with the default configuration.

The web console will work on any HTML5 capable browser, and will work with text browsers as well– although Chromium and Chrome display consistently across all platforms. The web console is a single-thread application and is not designed, nor intended, for high traffic.