Mac OS X Server

Podcast Producer
Version 10.6 Snow Leopard

019-1419/2009-08-01

Contents

About This Guide

Use this guide to set up and manage Podcast Producer solutions.

Podcast Producer Administration describes how to set up and manage Podcast Producer solutions for publishing podcasts of lectures, training, and other audio and video projects.

What's in This Guide

This guide includes the following chapters:

- Chapter 1, "Overview of Podcast Producer," introduces Podcast Producer and describes its architecture and security model.
- Chapter 2, "Setting Up Podcast Producer," describes how to get Podcast Producer up and running.
- Chapter 3, "Upgrading to Podcast Producer 2," describes how to upgrade your computer running Podcast Producer 1 to Podcast Producer 2.
- Chapter 4, "Setting Up Podcast Producer for High Availability," describes how to configure Podcast Producer for failover.
- Chapter 5, "Podcast Library," describes the Podcast Producer library.
- Chapter 6, "Managing Workflows," provides a high-level overview of workflows and how to manage them using Server Admin.
- Chapter 7, "Managing Cameras," describes how to manage and monitor camera usage using Server Admin.
- Chapter 8, "Managing Feeds," describes how to manage Podcast Library feeds.
- Chapter 9, "Customizing Workflows," describes how to customize workflows.
- Chapter 10, "Deploying Scalable Podcast Producer Solutions," describes how to plan the deployment of Podcast Producer solutions.
- Chapter 11, "Podcast Producer Command-Line Tools," describes the Podcast Producer command-line tools.

- Chapter 12, "Monitoring Podcast Producer," describes how to monitor and troubleshoot Podcast Producer issues.

Note: Because Apple periodically releases new versions and updates to its software, images shown in this book may be different from what you see on your screen.

Using Onscreen Help

You can get task instructions onscreen in Help Viewer while you're managing Mac OS X Server. You can view help on a server, or on an administrator computer. (An administrator computer is a Mac OS X computer with Mac OS X Server administrator software installed on it.)

To get the most recent onscreen help for Mac OS X Server:

- Open Server Admin or Workgroup Manager and then:
  - Use the Help menu to search for a task you want to perform.
  - Choose Help > Server Admin Help or Help > Workgroup Manager Help to browse and search the help topics.

  The onscreen help contains instructions taken from *Advanced Server Administration* and other administration guides.

To see the most recent server help topics:

- Make sure the server or administrator computer is connected to the Internet while you're getting help.

  Help Viewer automatically retrieves and caches the most recent server help topics from the Internet. When not connected to the Internet, Help Viewer displays cached help topics.

Documentation Map

Mac OS X Server has a suite of guides that cover management of individual services. Each service may depend on other services for maximum utility. The documentation map below shows some related guides that you may need in order to to fully configure Podcast Producer Server to your specifications. You can get these guides in PDF format from the Mac OS X Server resources website:

www.apple.com/server/macosx/resources/

Getting Started

Covers basic installation, setup, and management of Podcast Producer service using Server Preferences.

Server Preferences Help

Provides onscreen instructions and answers when you're using Server Preferences to manage Podcast Producer service.

Podcast Producer Workflow Tutorial

Walks you through the process of creating a complete Podcast Producer workflow.

Podcast Producer Administration

Describes advanced options for setting up, configuring, and managing Podcast Producer service.

Podcast Capture Help

Provides onscreen instructions and answers when you're using Podcast Capture to record and submit content for processing by Podcast Producer.

Network Services Administration

Explains how to set up DNS and firewall for use with Podcast Producer.

Introduction to Command-Line Administration

Explains how to use UNIX shell commands to configure and manage servers and services.

Xgrid Administration and High Performance Computing

Explains how to set up Xgrid for use with Podcast Producer.

Podcast Composer

Explains how to create Podcast Producer workflows using Podcast Composer.

Open Directory Administration

Explains how to set up Open Directory for use with Podcast Producer.

Wiki Server Administration

Explains how to set up wiki services for use with Podcast Producer.

Mail Server Administration

Explains how to set up mail service for use with Podcast Producer.

Viewing PDF Guides Onscreen

While reading the PDF version of a guide onscreen:

- Show bookmarks to see the guide's outline, and click a bookmark to jump to the corresponding section.
- Search for a word or phrase to see a list of places where it appears in the document. Click a listed place to see the page where it occurs.
- Click a cross-reference to jump to the referenced section. Click a web link to visit the website in your browser.

Printing PDF Guides

If you want to print a guide, you can take these steps to save paper and ink:

- Save ink or toner by not printing the cover page.
- Save color ink on a color printer by looking in the panes of the Print dialog for an option to print in grays or black and white.
- Reduce the bulk of the printed document and save paper by printing more than one page per sheet of paper. In the Print dialog, change Scale to 115% (155% for *Getting Started*). Then choose Layout from the untitled pop-up menu. If your printer supports two-sided (duplex) printing, select one of the Two-Sided options. Otherwise, choose 2 from the Pages per Sheet pop-up menu, and optionally choose Single Hairline from the Border menu. (If you're using Mac OS X v10.4 or earlier, the Scale setting is in the Page Setup dialog and the Layout settings are in the Print dialog.)

You may want to enlarge the printed pages even if you don't print double sided, because the PDF page size is smaller than standard printer paper. In the Print dialog or Page Setup dialog, try changing Scale to 115% (155% for *Getting Started*, which has CD-size pages).

Getting Documentation Updates

Periodically, Apple posts revised help pages and new editions of guides. Some revised help pages update the latest editions of the guides.

- To view new onscreen help topics for a server application, make sure your server or administrator computer is connected to the Internet and click "Latest help topics" or "Staying current" in the main help page for the application.
- To download the latest guides in PDF format, go to the Mac OS X Server Resources website at: www.apple.com/server/resources/

- An RSS feed listing the latest updates to Mac OS X Server documentation and onscreen help is available. To view the feed, use an RSS reader application such as Safari or Mail and go to: feed://helposx.apple.com/rss/snowleopard/serverdocupdates.xml

Getting Additional Information

For more information, consult these resources:

- *Read Me documents* — get important updates and special information. Look for them on the server discs.
- *Mac OS X Server website* (www.apple.com/server/macosx/)—enter the gateway to extensive product and technology information.
- *Mac OS X Server Support website* (www.apple.com/support/macosxserver/)—access hundreds of articles from Apple's support organization.
- *Apple Discussions website* (discussions.apple.com/)—share questions, knowledge, and advice with other administrators.
- *Apple Mailing Lists website* (www.lists.apple.com/)—subscribe to mailing lists so you can communicate with other administrators using email.
- *Apple Training and Certification website* (www.apple.com/training/)—hone your server administration skills with instructor-led or self-paced training, and differentiate yourself with certification.

Overview of Podcast Producer

1

This chapter introduces Podcast Producer and describes its architecture.

Podcast Producer is a video capture, processing, and publishing system. It is an elegant solution that automates the process of creating and publishing podcasts of lectures, training, or other audio and video projects.

What's New in Podcast Producer

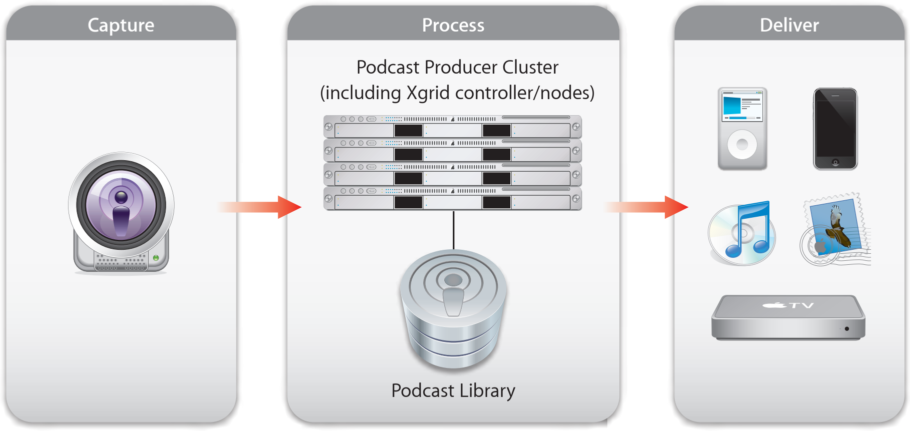
Mac OS X Server v10.6 includes a new version of Podcast Producer, Podcast Producer 2, which offers major enhancements:

- **Podcast Producer Setup Assistant.** This program simplifies the process of setting up Podcast Producer Server. This assistant asks you for the information it needs and takes care of setting up Podcast Producer Server and related services.
- **Podcast Composer.** You can now create workflows effortlessly using this new application (in /Applications/Server/). Podcast Composer graphically leads you through creating workflows that control how Podcast Producer generates and distributes podcasts.
- **Dual video recording.** Podcast Producer 2 lets users record dual video sources using the Podcast Capture application on a Mac or the new Podcast Capture web application on a Mac, iPhone, or Windows computer. Apple provides several picture-in-picture templates, or you can create your own.
- **Podcast Library.** Podcast Library lets your server store podcasts and deliver them to viewers through RSS and Atom feeds. For example, you podcasts can be hosted on your server but accessed through the iTunes U, which can aggregate the feeds, but leaves the content on the server. Atom feeds simplify distributing multiple podcast versions, such as iPod, Apple TV, and audio only, because each Atom episode can contain multiple enclosures.
- **High availability.** Camera agents and Podcast Capture clients can now fail over to other Computers running Podcast Producer Server. Failover is optional and requires Xsan.

- **Compressor and Qmaster support.** Apple Compressor is a powerful tool that gives you complete control over encoding settings and allows you to create production-quality encoders. Apple Qmaster lets you distribute rendering tasks across a Mac OS X network. Podcast Producer 2 supports Compressor and Qmaster.
- **Podcast Capture Web Application.** This is a web version of Podcast Capture, which you can use to remotely capture and upload audio and video QuickTime movies to a Podcast Producer server for encoding and publishing. You can also use this application to upload documents that are compatible with Apple's Quick Look technology. This application runs from any modern browser window on Mac, Windows, and UNIX computers.

How Podcast Producer Works

Podcast Producer does to the production of podcasts what the assembly line did to automobile production. It automates and streamlines the production of podcasts.



Here's how Podcast Producer works:

Step 1: Capture

You use Podcast Capture, Podcast Producer Camera Agent, or Podcast Capture Web Application to record and submit content to the Podcast Producer server. When you submit content, you also specify the workflow to use for processing the content. Podcast Capture (in /Applications/Utilities) is a utility included in Mac OS X and Mac OS X Server v10.5 or later. The Podcast Capture Web Application is a web application that runs in any standards-based web browser.

Step 2: Process

The Podcast Producer server stores the submitted content in the Podcast Library and uses Xgrid to distribute the content processing, encoding, publication, and notification tasks specified in the workflow.

An Xgrid system consists of computing nodes called Xgrid agents. These agents are directed by the Xgrid controller system. When you submit content to a Podcast Producer server, it generates an Xgrid job based on the specified workflow and submits it to the Xgrid controller, which then allocates resources on CPU cores on Xgrid agents.

For example, a workflow might accept a single video as input, add to it introduction and title movies, encode the resulting movie for iPod and iPhone, publish the content to the Podcast Library, and notify a target audience via email.

Step 3: Deliver
Podcast Producer 2 introduces a powerful new way to deliver Atom and RSS feeds with podcast media enclosures: the Podcast Library. The feeds and content from the Podcast Library can be directly accessed by iTunes or a web browser like Safari which directly supports Atom/RSS feeds.

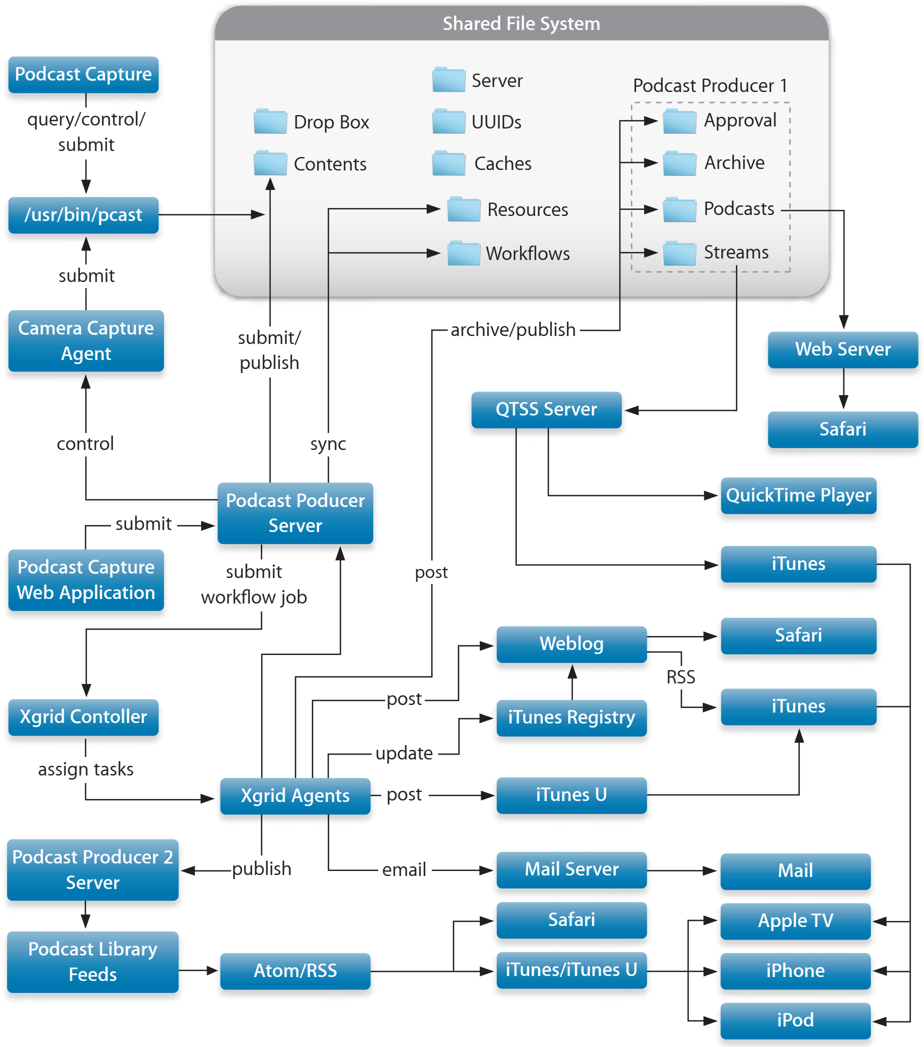
The feeds can also be integrated with other content delivery systems such as iTunes U. Once a client accesses and downloads the content, it can be viewed on the desktop or devices including iPhone, iPod and AppleTV.

A complete Podcast Producer system incorporates these areas: Capture, Process and Deliver. All of these services can run on a single server (an all-in-one configuration). However, because Podcast Producer is a highly scalable solution, you can configure a cluster of servers to distribute processing using Xgrid.

For medium-sized deployments, you can use NFS to share files between Podcast Producer and Xgrid servers. Larger deployments that require a highly scalable system with failover support incorporate Xsan, Apple's fibre channel storage area network, and clustered file system technology.

The Architecture of the Podcast Producer System

The following figures illustrates the architecture of the Podcast Producer system.



The Podcast Producer system consists of the following main components:

- "Podcast Producer Server" on page 18
- "Podcast Library" on page 18
- "Podcast Capture" on page 19
- "Podcast Capture Web Application" on page 20
- "Podcast Composer" on page 20
- "The podcast Command-Line Tool" on page 20

- "Podcast Producer Camera Agent" on page 18
- "Workflows" on page 20
- "Xgrid" on page 21

Podcast Producer Server

The Podcast Producer server (`pcastserverd`) is the central point for the administration of a Podcast Producer solution.

The Podcast Producer server manages camera capture agents, provides access control and centralized management, and accepts QuickTime movies and files compatible with Apple's Quick Look technology to be processed on an Xgrid cluster.

You can use the Server Admin to:

- Specify where the Podcast Library stores content, resources, and metadata associated with workflow submissions.
- Specify the Xgrid controller for processing Xgrid jobs
- Control and monitor access to cameras
- Control and monitor access to workflows
- Customize workflow properties

Podcast Library

Podcast Library is a repository that stores everything that goes through the Podcast Producer server. This includes workflows, job submissions, original content, intermediate files, all versions of published podcasts, metadata, and any other information needed for the successful processing and publishing of podcasts.

Podcast Library uses RSS and Atom feeds to distribute podcasts.

Podcast Library requires a shared file system for storing information and content. The supported shared file systems are Xsan, HFS, and NFS.

CAUTION: Do not modify files in the shared file system directly. For example, to add or remove a workflow, use the `podcast` tool, but do not modify the file system directly.

For more information about Podcast Library, see Chapter 5, "Podcast Library."

Podcast Producer Camera Agent

The Podcast Producer Camera Agent is a daemon controlled by the Podcast Producer server. The agent is responsible for the recording of audio and video.

For example, a Podcast Producer Camera Agent could run on a Mac Mini with an external camera attached to it via Firewire. After you bind the agent to the Podcast Producer server, authorized clients can control the agent locally or remotely using the Podcast Capture application.

After the recording is done, the Podcast Producer agent uses the `podcast` tool to upload the recorded content to the Podcast Producer server for processing and delivery.

Podcast Capture

Podcast Capture (in /Application/Utilities/) is the application you use to:

- Record and submit video for processing by Podcast Producer.

  The video can come from one or two sources. For example, in the case of one source, the video can come from the local iSight camera of an iMac or a remote camera agent configured on a Mac mini. In the case of two sources, either source can be a local or remote camera or a screen recording.

- Record and submit audio for processing by Podcast Producer.

  The audio source can come from a local source or a remote camera.

- Record and submit a screen recording movie for processing by Podcast Producer.

  By default, the recording includes audio from local sources. For example, you can record a Keynote presentation, which might include audio and video, while providing an audio narration.

- Submit documents for processing by Podcast Producer.

  These documents can be movies, images, Keynote presentations, Pages documents, PDF files, Word files, and PowerPoint files.

  If a submitted document is not a movie, Podcast Capture uses Quick Look to extract the pages of the document as images and combines them together into a slide show. To submit multiple documents, you must first combine them into a single zip archive. You then submit the zip archive to the Podcast Producer server.

  The Podcast Producer server uses the Montage workflow to extract files from the archive and assemble them alphabetically based on their filenames into a movie, then encodes the podcast and distributes it using the Podcast Library.

- Share a camera connected to the Mac on which Podcast Capture is running with Podcast Producer.

  This is also referred to as binding a local camera to the Podcast Producer server.

- Configure audio/video and general preferences.

Podcast Capture is available on Macs with Mac OS X and Mac OS X Server v10.5 or later.

Important: If you use Podcast Capture from Macs running Mac OS X v10.5, you can't access the new features of Podcast Capture such as dual source capture.

For more information about how to use Podcast Capture, see its onscreen help.

Podcast Capture Web Application

Podcast Capture Web Application provides a subset of the functionality of Podcast Capture. Podcast Capture Web Application runs in a web browser and allows you to:

- Record and submit audio and video for processing by Podcast Producer.
- Submit documents for processing by Podcast Producer.

However, you cannot use this application to create screen recordings, to bind cameras, or to configure preferences.

You can run Podcast Capture Web Application from any system running Mac OS X, Windows, or UNIX, including iPhone and iPod Touch.

For more information about how to use the Podcast Capture Web Application, see its onscreen help.

Podcast Composer

Podcast Composer (in /Application/Server/) is a graphical workflow editor that leads you through the steps of defining video-based Podcast Producer workflows. You graphically choose the intro, title, and exit videos; specify different transitions and effects between videos; and view real-time titles and effects.

You can add watermarks and overlays to your Podcast content. Your workflow also specifies encoding formats and targets distribution via wiki, iTunes U, or Podcast Library for your finished podcast.

For more information about how to use Podcast Capture, see *Podcast Composer User Guide*.

The podcast Command-Line Tool

The `/usr/bin/podcast` command-line tool enables clients to record and submit QuickTime files. Podcast Capture is a graphical user interface (GUI) that wraps `podcast`.

For more information about `podcast`, see "The podcast Tool" on page 115.

Workflows

Workflows are the heart of the Podcast Producer system. A workflow is a template that defines a set of customizable Xgrid tasks for encoding and publishing podcasts.

When submitting content to the Podcast Producer server for processing, you also specify the workflow to use. Podcast Producer makes the necessary replacements in the workflow based on the configuration information entered in Server Admin and the metadata submitted with the content. Then, Podcast Producer sends the workflow as an Xgrid job to the Xgrid controller for processing.

The Podcast Producer server provides a set of sample workflows that define common encoding and publishing tasks for encoding and publishing QuickTime movies as podcasts. However, you can modify these workflows to suit your needs, or you can create your own workflows.

Each sample workflow defines a set of default properties that you can configure using Server Admin, as described later in this guide. You can also define new properties in your custom workflows and use the Podcast Producer server to configure their values.

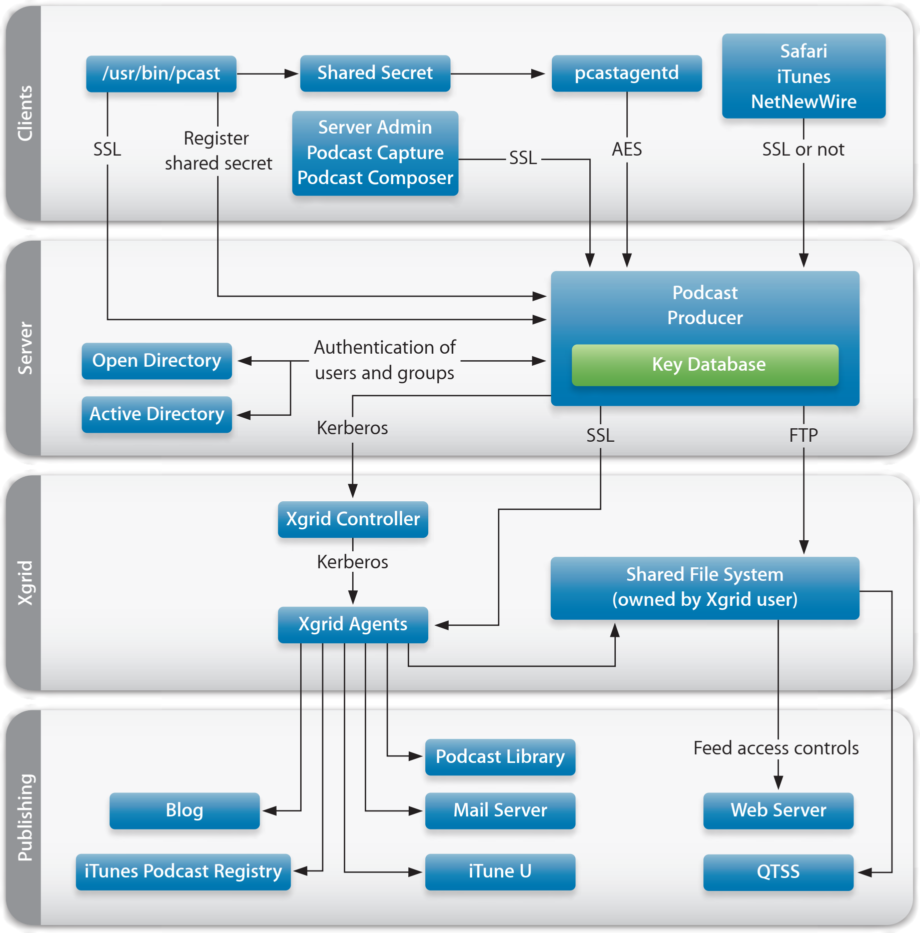
Xgrid

Podcast Producer requires Xgrid for content processing and publishing tasks. With Xgrid, these tasks are distributed across available CPU cores on Xgrid agents. To scale Xgrid technology, you can connect additional servers to the system, which increases system throughput.

Note: Podcast Producer 1 workflows can run on Xgrid agents running Mac OS X Server v10.5 or v10.6. Podcast Producer 2 workflows can only run on Xgrid agents running Mac OS X v10.6 or later.

The Podcast Producer Security Model

Podcast Producer is a secure end-to-end solution, as shown in the following illustration.



Client/Server Communication

To protect sensitive information, the `podcast` command-line tool and Server Admin use the Secure Sockets Layer (SSL) protocol to communicate with the Podcast Producer server. By default, these SSL connections use the server's default self-signed certificate, which comes with the server.

For example, when you specify the passwords for Podcast Producer properties in Server Admin, Server Admin securely passes the passwords to the Podcast Producer server using SSL. The Podcast Producer server encrypts these passwords and stores them in its database.

In addition, the Podcast Producer Camera Agent (`pcastagentd`) uses an Advanced Encryption Standard (AES)-secured tunnel to communicate with the Podcast Producer server. This tunnel allows the Podcast Producer server to control the Podcast Producer Camera Agent at all times.

When you bind a camera system to Podcast Producer, the `podcast` command-line tool creates a shared secret and sends it to the server using SSL. The server encrypts the shared secret, stores it in its database, and sends back information including the shared secret to the `podcast` tool.

The `podcast` tool stores the information it receives from the Podcast Producer server in a property list. When started, the Podcast Producer Camera Agent uses the shared stored in its property list to establish an AES-secured tunnel with the Podcast Producer server.

Authentication

The Podcast Producer server uses Open Directory for user and group authentication The Podcast Producer server also support Active Directory digest authentication.

The Podcast Producer server can support any combination of basic, digest, and Kerberos (Negotiate or SPNEGO) authentication.

The authentication settings for Podcast Producer are stored in the file /Library/Preferences/com.apple.pcastserverd.plist by the `http_auth_type` key. You must have root access to edit this file.

By default, the basic, digest, and Kerberos authentication methods are enabled and the property list file contains this entry:

<key>http_auth_type</key> <array> <string>basic</string> <string>digest</string> <string>kerberos</string> </array>

To disable an authentication method, remove the line containing its `<string>` entry and save the file. Then stop and restart the Podcast server, in Server Admin or by using the command `pcastctl server restart`.

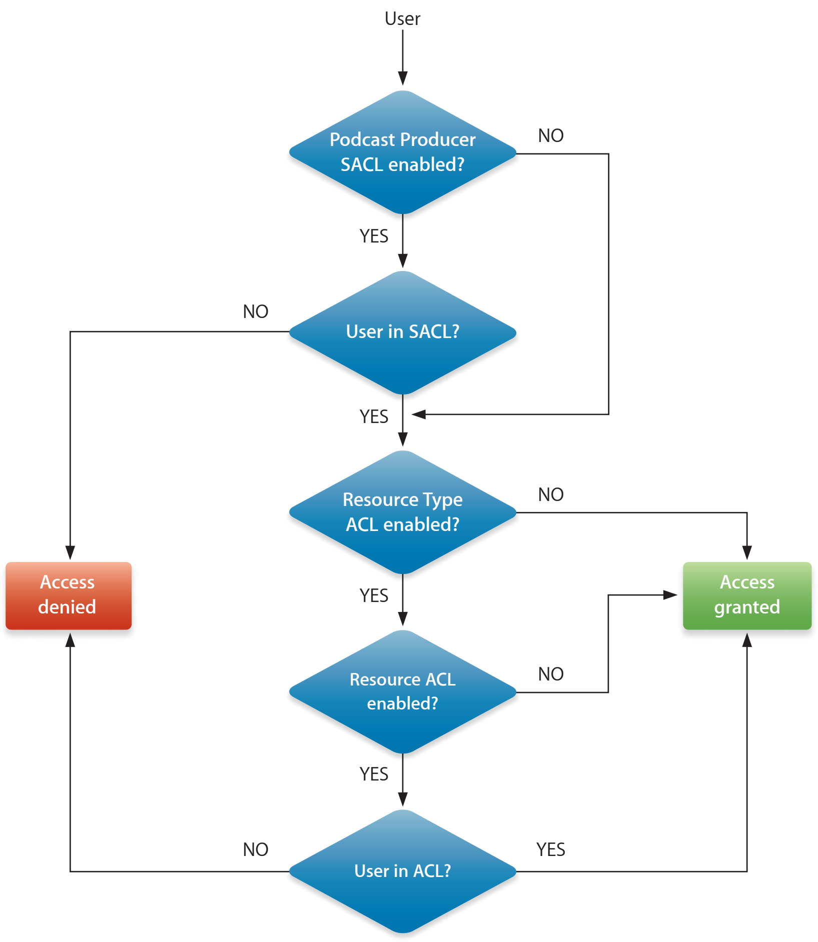
Podcast Producer supports standard HTTP digest authentication for local and Open Directory users and custom digest authentication for Active Directory users.

Access Control

The Podcast Producer server uses Open Directory or Active Directory to authenticate users and groups specified in the Service Access Control List (SACL) in Server Admin and the camera, workflow, and feed Access Control Lists (ACLs) maintained by the Podcast Producer server.

The Podcast Producer server stores the ACLs in the same database it uses to store shared secrets and other sensitive information.

The following diagram illustrates how Podcast Producer uses the Server Admin SACL and the Podcast Producer server ACLs to restrict access to resources: cameras, workflows, and feeds.



Xgrid
The Podcast Producer server uses Kerberos to communicate with the Xgrid controller using the `xgrid` command-line tool.

The Podcast Producer server also uses Kerberos to authenticate itself to the Kerberos Key Distribution Center (KDC). The server uses the standard method to get Kerberos tickets from the KDC using the Xgrid name and password supplied in Server Admin.

The Podcast Producer server uses the ticket it gets from the KDC when sending jobs to the Xgrid controller.

As for the Xgrid agent, it authenticates to the Xgrid controller using Kerberos. However, when communicating with the Podcast Producer server, the Xgrid agent uses SSL. The Xgrid agent calls the Podcast Producer server to obtain property values and responses to challenges.

For example, if an Xgrid agent tries to post to a blog, the agent gets back a 401 HTTP error with a challenge. The agent sends the challenge to the Podcast Producer server and receives a response, which it passes to the blog to be granted access.

The Podcast Producer server and Xgrid controller must belong to the same Kerberos realm. The user designated to run Xgrid jobs must be a valid principal in that Kerberos realm.

Publishing
In the Podcast Producer 1 security model, Xgrid agents have the proper credentials to publish podcasts. If an agent is asked to respond to a challenge, as in the case of a blog, the Xgrid agent can securely obtain the response from the Podcast Producer server and provide the relevant response.

In the Podcast Producer 2 security model, the Podcast Library, which has valid credentials, publishes podcasts.

Podcast Producer Customization
Podcast Producer is an open system that you can customize to meet your needs:

- Workflows are Xgrid jobs, which you can compose using any text editor.
- Podcast Producer leverages Mac OS X technologies like QuickTime, Compressor, and Quartz.

  QuickTime provides a rich set of video/audio codecs and is the premier platform for media data sets. Anything you can do with QuickTime can be done using Podcast Producer. You can create command-line tools that access QuickTime APIs and use these tools in workflows.

  You can create professional codecs using Compressor.

Quartz is a powerful image composition framework for image manipulation. You can create tools for adding movie effects and use these tools in Podcast Producer workflows. An example of such a tool is `qc2movie`, which ships with Mac OS X v10.5 or later. This tool is used by some default Podcast Producer workflows and is described in "Creating and Customizing Workflows" on page 100.

- Hosting technology used by Podcast Producer is based on Internet standards like HTTP, Atom, RSS, Podcast, QTSS, and FTP. You can publish content to any hosting server that supports these standards.
- You can turn virtually any UNIX shell script into an Xgrid job and, therefore, into part of a Podcast Producer workflow.

Supported Audio and Video Formats

Podcast Producer relies on QuickTime and QuickLook to access incoming content for processing. As a result, a wide range of media files and document formats can be submitted to Podcast Producer

For example, media file formats that can be submitted to Podcast Producer for processing include:

- mp3 audio
- aiff or wav audio files with compatible codecs
- QuickTime (.mov) files with compatible codecs
- MPEG-4 files (.mp4, .m4a, .m4v) with compatible codecs

Podcast Producer can encode content using Xgrid or Compressor/Qmaster to distribute media encoding tasks.

With Xgrid, the following codecs are supported:

- Apple TV

  Video: H.264 Video, 1280 x 720, 5 Mbps, 30 fps

  Audio: AAC-LC Music, Stereo, 128 kbps, 44.1 KHz
- Audio

  Video: None

  Audio: AAC-LC Music, Stereo, 128 kbps, 44.1 KHz
- Computer

  Video: H.264 Video, 1920 x 1080, 10 Mbps, 30 fps

  Audio: AAC, Stereo, 256 kbps, 44.1 kHz

- iPod/iPhone

  Video: H.264 Video, 640 x 480, 1.5 Mbps, 30 fps

  Audio: AAC-LC Music, Stereo, 128 kbps, 44.1 KHz
- Mobile

  Video: H.264 Video, 176 x 144, 56 kbps, 15 fps

  Audio: AAC-LC Music, Mono, 24 kbps, 16 KHz

Compressor/Qmaster support a wider range of codecs and media file formats. For more information and available options and settings, see the Compressor documentation.

Setting Up Podcast Producer

2

This chapter describes how to set up the Podcast Producer server.

The instructions in this chapter assume that you have installed and configured Mac OS X Server v10.6. For instructions on setting up Mac OS X Server, see *Getting Started* (included on the Mac OS X Server v10.6 installation disc and downloadable at www.apple.com/server/documentation/).

Hardware and Software Requirements for Podcast Producer

This section describes the hardware and software requirements for providing Podcast Producer services.

For more information about estimating hardware and software requirements, see Chapter 10, "Deploying Scalable Podcast Producer Solutions."

For more information about minimum system requirements and other Podcast Producer topics not covered in this guide, visit the *Mac OS X Server Support* website at www.apple.com/support/macosxserver.

Podcast Capture Requirements

Following is a list of the minimum hardware and software requirements for the Mac used to capture video:

- Mac OS X v10.5 or later
- 20 GB of free disk space
- Network connectivity (100 Mbps)
- A compatible graphics card for some podcast encoding operations
- iSight camera (built-in or external) or FireWire DV camcorder

Podcast Server Computer Requirements

Following is a list of the minimum requirements for the Mac running the Podcast Producer service:

- Mac OS X Server v10.6
- Network connectivity (100 Mbps)
- Xsan for optional cluster file services

Podcast Producer Xgrid Rendering Requirements

Following is a list of the minimum requirements for the Podcast Producer Xgrid system:

- Any Intel-based Macintosh Server or Intel-based desktop Mac (a Mac Pro, for example).
- Mac OS X v10.6 or Mac OS X Server v10.6.
- At least 1 GB of memory (RAM) plus 512 MB of additional RAM per processor core.
- At least 50 GB of available disk space.
- Xsan for optional cluster file services.
- Quartz Extreme-enabled video chipset.

Xgrid Size and Bandwidth Considerations

Following is a list of issues to consider when deciding the size of the Xgrid and the amount of bandwidth needed to process QuickTime movies:

- Number of recording systems and manual submissions
- Type of workflows to be used

  The type of workflow used determines the time needed to complete the processing and publishing of podcasts. Also, the type of workflows used determines how many tasks can be performed in parallel.
- A typical recording day schedule for each recording system and manual submission
- Number of uploading Macs
- Number of compute nodes in the grid

Note: Xgrid agent computers must have a graphics card that supports Quartz Extreme.

Setting Up Podcast Producer

To set up the Podcast Producer server, follow these steps:

Step 1: Plan your deployment

Make sure the systems that make up your Podcast Producer solution meet the minimum system requirements. Make sure that your network can handle the estimated traffic. Make sure the storage space for the Podcast Library is sufficient for the estimated amount of data.

For more information about ensuring proper DNS configuration, see "Plan Your Deployment" on page 31.

Step 2: Ensure proper DNS configuration

Before you can configure the Podcast Producer server, be sure you have access to a properly configured DNS server. The DNS server can be an external server or a server you set up locally on your server.

For more information about ensuring proper DNS configuration, see "Ensuring Proper DNS Configuration" on page 32.

Step 3: Enable Podcast Producer server

Before you can configure and start the Podcast Producer server, you must enable Podcast Producer Server administration in Server Admin. This allows Server Admin to start, stop, and change settings for Podcast Producer Server.

For more information about enabling Podcast Producer server, see "Enabling Podcast Producer Server Administration in Server Admin" on page 33.

Step 4: Configure Podcast Producer server

Use the Podcast Producer Setup Assistant to configure the Podcast Producer server and other required services.

For more information about configuring Podcast Producer server, see "Configuring Podcast Producer Server Using the Podcast Producer Setup Assistant" on page 33.

Step 5: (Optional) Configure default workflow properties

Use the Server Admin to configure the default workflow properties of Podcast Producer.

For more information about configuring the default workflow properties of Podcast Producer server, see "(Optional) Configuring the Default Workflow Settings" on page 39.

Plan Your Deployment

To ensure successful deployment of a Podcast Producer solution, plan ahead by considering the following:

- How much content will be captured?

  Captured content can be very large in size (for example, 5 GB per hour). This has a direct impact on your storage and bandwidth needs.

- What are your branding needs?

  This includes introduction and exit videos, title videos, and watermarking.

- What effects you would like to use in podcasts?

  You can use the default Quartz Composer effects or you can create your own when designing your own workflows. Creating Quartz Composer effects require strong familiarity with Quartz Composer and can be time consuming.

  Note: Using Quartz Composer effects increases processing time.

- How many versions will be encoded?

  The more versions you plan to provide, the more CPU and storage you'll need.

  In addition, if you need to provide your own encodings, which you can create using Compressor, you must look at the impact this has on your resources and whether you must set up or use an existing Apple Qmaster and Compressor distributed processing cluster for processing custom encodings.

  If you plan to use Compressor to process encodings on Xgrid agents, you must install Compressor on all Xgrid agents and restart them, but without logging in.

- How will content be captured?

  You need to take into account many aspects related to capturing content. For example, you might need a camera recording system (a video camera connected to a Mac Mini) for each class and another Mac for presenters to use for recording the Keynote presentations.

- Where will the content be submitted from?

  Video and screen recordings can be very large (5 GB per hour for best quality). For example, to support local users who need to record at the highest quality, you can set up a gigabit network to allow for high-speed uploading of content. However, if your users are remote, you must reduce the quality of the recorded content or find other ways for users to submit content.

- What is the size of the audience?

  The number of people accessing your podcasts can help you determine how much storage, bandwidth, and computing resources are needed to serve the content.

- Where is the audience?

  Depending on where your audience is located, you may need, for example, to provide podcasts in certain encodings to accommodate users with slow Internet access speeds.

- How scalable should your Podcast Producer solution be?

  If you plan to use Podcast Producer on a very small scale, an all-in-one solution might do the job.

  If you need to scale the processing power, but don't need significant storage capacity, an NFS setup might address your needs.

  If you need a highly scalable and redundant solution, you need an Xsan setup.

For more information about issues to consider when planning Podcast Producer deployment, see Chapter 10, "Deploying Scalable Podcast Producer Solutions."

Ensuring Proper DNS Configuration

Whether you're using an existing DNS server or setting up your own, make sure that forward and reverse DNS is working for your server.

For more information about setting up a DNS server, see *Network Services Administration*.

To verify proper DNS configuration:

1 Open Terminal.

2 To test whether forward and reverse DNS is working, enter the following command:

```
sudo changeip -checkhostname
```

3 Verify that the output indicates success and looks like the following example:

```
Primary address     = 10.0.0.80


Current HostName    = pcast.example.com
DNS HostName        = pcast.example.com


The names match. There is nothing to change.
dirserv:success = "success"
```

If this test fails:

- If you're using your own local DNS server, check its configuration.
- If you're using a DNS server on your network, ask your network administrator for help.

You can also use the `nslookup` command to verify proper DNS configuration, as in the following example:

```
$ nslookup pcast.example.com
Server:      192.168.1.100
Address:     192.168.1.100#53

Name:    pcast.example.com
Address: 192.168.1.100

$ nslookup 192.168.1.100
Server:      192.168.1.100
Address:     192.168.1.100#53

100.1.168.192.in-addr.arpa    name = pcast.example.com.
```

Enabling Podcast Producer Server Administration in Server Admin

Before you can configure and start the Podcast Producer server, you must enable Podcast Producer Server administration in Server Admin. This allows Server Admin to start, stop, and change settings for Podcast Producer Server.

To enable Podcast Producer Server for administration:

1. Open Server Admin and connect to the server.
2. Click the Settings button in the toolbar.
3. Click the Services tab.
4. Select the checkbox for Podcast Producer Server.
5. Click Save.

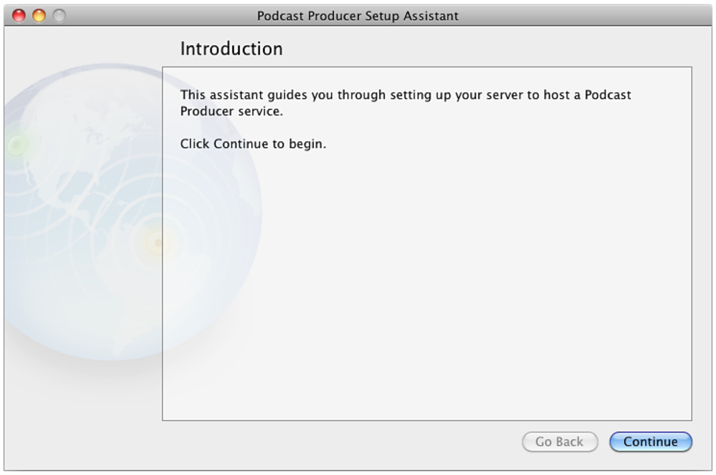
Configuring Podcast Producer Server Using the Podcast Producer Setup Assistant

Podcast Producer Server provides the Podcast Producer Setup Assistant, which leads you through configuring Podcast Producer Server and all other services that are needed by it.

To configure Podcast Producer Server using the Podcast Producer Setup Assistant:

1. Open Server Admin.
2. In the Servers list, select the server that will run Podcast Producer Server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer Server.
4. Click the Overview button in the toolbar.
5. Click Configure Podcast Producer.

6 In the Introduction screen of the Podcast Producer Setup Assistant, click Continue.



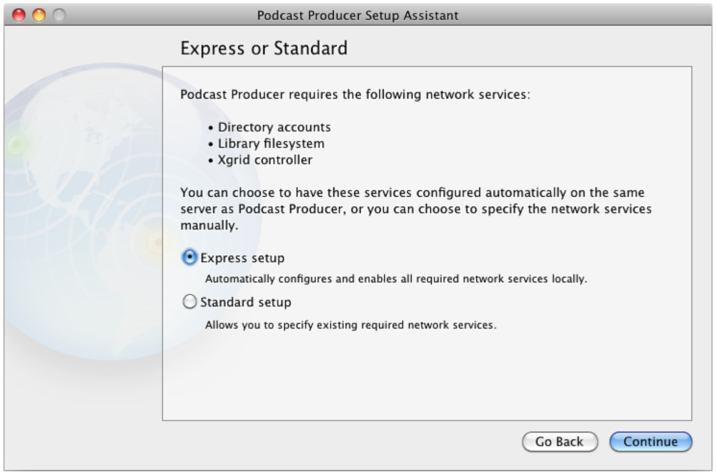
Introduction

This assistant guides you through setting up your server to host a Podcast Producer service.

Click Continue to begin.

7 In the Express or Standard screen, select a setup method and then click Continue:

- Express setup—To automatically configure and enable all required network services locally on the same server as Podcast Producer, select this option. Then proceed to "Using the Express Setup Configuration Option" on page 35 to complete the configuration process.
- Standard setup—To manually specify the network services required by Podcast Producer Server, select this option. Then proceed to "Using the Standard Setup Configuration Option" on page 37 to complete the configuration process.



Express or Standard

Podcast Producer requires the following network services:

- Directory accounts
- Library filesystem
- Xgrid controller

You can choose to have these services configured automatically on the same server as Podcast Producer, or you can choose to specify the network services manually.

[✓ Express setup]
Automatically configures and enables all required network services locally.

Standard setup
Allows you to specify existing required network services.

Important: Using mail or iChat messages for notification requires the Mail and iChat services to be properly configured. For iChat, the iChat server must be running on the Podcast Producer server. The iChat server can be federated with other chat servers for cross-realm chat support.

Using the Express Setup Configuration Option

Use the express setup option to quickly get Podcast Producer Server up and running.

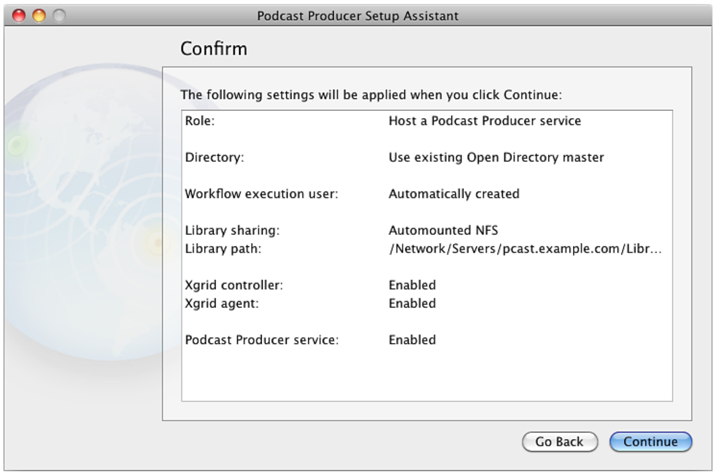
Important: When you set up Mac OS X Server, by default, the short name of the directory administrator is diradmin and the password is the first password you assigned to the first admin user.

To perform an express setup of Podcast Producer Server:

1 In the Directory screen:

- If your server is hosting an Open Directory master, click Continue.
- If your server is bound to a directory and has been Kerberized, click Continue.
- If your server is not hosting an Open Directory master and is not bound to a directory, set up an Open Directory master:
  a In the Name field, enter the name of the directory administrator.
  b In the Short Name field, enter the short name of the directory administrator.
  c In the User ID field, enter a unique ID if you don't want to use the default ID.
  d In the Password and Verify fields, enter the password for the directory administrator.
  e Click Continue.
- If your server is bound to a directory but has not been Kerberized, provide the directory administrator name and password to Kerberize your server:
  a In the Name field, enter the name of the directory administrator.
  b In the Password field, enter the password for the directory administrator.
  The password should be the same as the password you created when you installed Mac OS X Server v10.6.
  c Click Continue.

2 In the Confirm screen, read the summary of the settings that the setup assistant will configure, then click Continue.

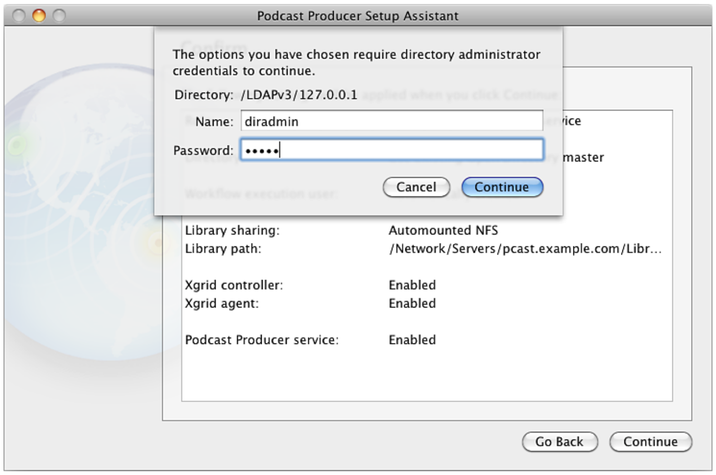

Podcast Producer Setup Assistant

Confirm

The following settings will be applied when you click Continue:

Role:	Host a Podcast Producer service
Directory:	Use existing Open Directory master
Workflow execution user:	Automatically created
Library sharing:	Automounted NFS
Library path:	/Network/Servers/pcast.example.com/Libr…
Xgrid controller:	Enabled
Xgrid agent:	Enabled
Podcast Producer service:	Enabled

3 If prompted, enter the name and password of your Open Directory administrator account, then click Continue.


Podcast Producer Setup Assistant

The options you have chosen require directory administrator credentials to continue.

Directory: /LDAPv3/127.0.0.1

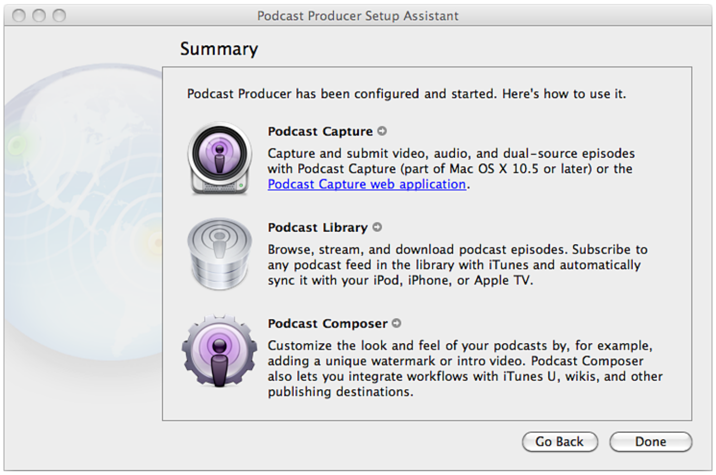
Name: diradmin

Password: •••••

Library sharing:	Automounted NFS
Library path:	/Network/Servers/pcast.example.com/Libr…
Xgrid controller:	Enabled
Xgrid agent:	Enabled
Podcast Producer service:	Enabled

If the express setup succeeds, the setup assistant displays a message informing you that Podcast Producer Server is configured and ready for use. Otherwise, an error message appears describing the problem.

4 If setup succeeds, click Done to dismiss the message.



Podcast Producer has been configured and started. Here's how to use it.

Podcast Capture

Capture and submit video, audio, and dual-source episodes with Podcast Capture (part of Mac OS X 10.5 or later) or the Podcast Capture web application.

Podcast Library

Browse, stream, and download podcast episodes. Subscribe to any podcast feed in the library with iTunes and automatically sync it with your iPod, iPhone, or Apple TV.

Podcast Composer

Customize the look and feel of your podcasts by, for example, adding a unique watermark or intro video. Podcast Composer also lets you integrate workflows with iTunes U, wikis, and other publishing destinations.

5 If setup fails, record the error, click Done, fix the problem, and try again.

6 Using Workgroup Manager, create users and groups if needed.

Using the Standard Setup Configuration Option

Use the standard setup option to specify the network services that Podcast Producer Server requires to run.

Important: When you set up Mac OS X Server, by default, the short name of the directory administrator is diradmin and the password is the first password you assigned to the first admin user.

To perform a standard setup of Podcast Producer Server:

1 In the Directory screen:

- If your server is hosting an Open Directory master, click Continue.
- If your server is bound to a directory and has been Kerberized, click Continue.
- If your server is not hosting an Open Directory master and is not bound to a directory, set up an Open Directory master:
  a In the Name field, enter the name of the directory administrator.
  b In the Short Name field, enter the short name of the directory administrator.
  c In the User ID field, enter a unique ID if you don't want to use the default ID.
  d In the Password and Verify fields, enter the password for the directory administrator.
  e Click Continue.
- If your server is bound to a directory but has not been Kerberized, provide the directory administrator name and password to Kerberize your server:

a In the Name field, enter the name of the directory administrator.

b In the Password field, enter the password for the directory administrator.

The password should be the same as the password you created when you installed Mac OS X Server v10.6.

c Click Continue

2 In the Accounts screen, select one of the following options, then click Continue:

- Create workflow execution user automatically—Select to automatically create a user account that Podcast Producer Server uses to submit jobs to Xgrid.
- Enter credentials for an existing user—Select to specify a user account for Podcast Producer Server to use to submit jobs to Xgrid, then enter the user credentials in the Name and Password fields.

3 In the Library screen:

a Click Choose to specify the path to the Podcast Library or enter the path manually in the Podcast Library field.

Podcast Producer uses this location in the file system to store configuration information, recordings, and podcasts. This location must be in a shared file system if you want to use additional computers as rendering agents. Otherwise, only your computer can process Podcast Producer workflows.

To configure your server for failover, use Xsan for the shared file system.

b To share the Podcast Library using NFS, select "Enable NFS share."

The recommended shared file system is Xsan. However, if the location you choose is not on an Xsan volume, or if not all computers are attached to the Xsan volume, you can share the Podcast Library using NFS.

If you enable "NFS share," the Podcast Producer Setup Assistant configures your NFS automount, but you must bind all Xgrid nodes to the same directory the Podcast Producer server is bound to.

c Click Continue.

4 In the Xgrid screen:

a To set up a local Xgrid controller and agent on your system, select "Set up a local Xgrid controller and agent."

b To use an existing Xgrid controller, select "Use an existing Xgrid controller," enter the address of the controller, and choose one from the list.

c Click Continue.

To set up an Xgrid cluster consisting of a controller and multiple agents, see *Xgrid Administration and High Performance Computing*.

5 In the Confirm screen, read the summary of the settings that the setup assistant will configure, then click Continue.

6 If prompted, enter the name and password of your Open Directory administrator account, then click Continue.

If the standard setup succeeds, the setup assistant displays a message informing you that Podcast Producer Server is configured and ready for use. Otherwise, an error message appears describing the problem.

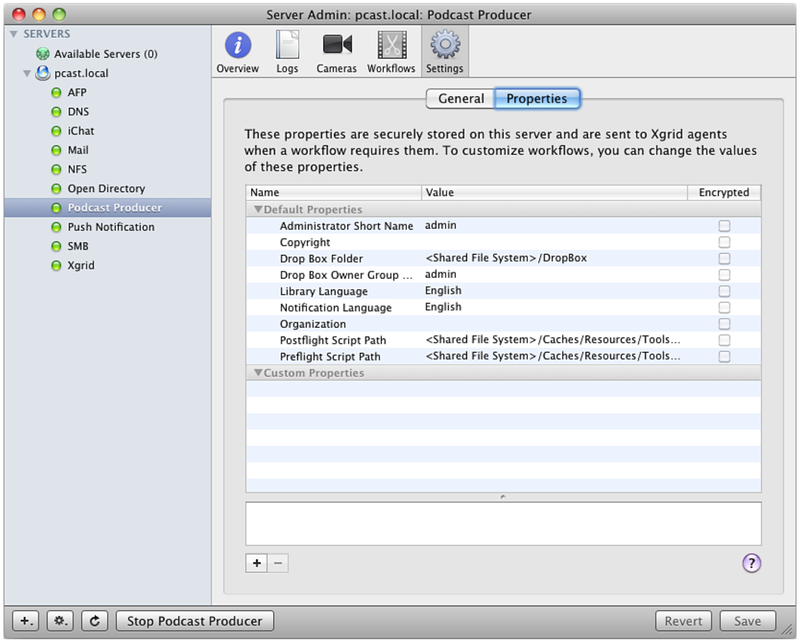
7 If setup succeeds, click Done to dismiss the message.

8 If setup fails, record the error, click Done, fix the problem, and try again.

9 Using Workgroup Manager, create users and groups if needed.

(Optional) Configuring the Default Workflow Settings

You can use Server Admin to configure a set of default workflow properties.



These properties are securely stored on this server and are sent to Xgrid agents when a workflow requires them. To customize workflows, you can change the values of these properties.

Name	Value	Encrypted
▼Default Properties		
Administrator Short Name	admin	☐
Copyright		☐
Drop Box Folder	<Shared File System>/DropBox	☐
Drop Box Owner Group ...	admin	☐
Library Language	English	☐
Notification Language	English	☐
Organization		☐
Postflight Script Path	<Shared File System>/Caches/Resources/Tools...	☐
Preflight Script Path	<Shared File System>/Caches/Resources/Tools...	☐
▼Custom Properties		

To configure default workflow properties:

1 Open Server Admin.

2 In the Servers list, select the server running Podcast Producer, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Podcast Producer Server.

4 Click Settings.

5 Click Properties.

6 Click the triangle next to Default Properties to display the properties.

7 If needed, change the default values of these properties.

Property	Value
Administrator Short Name	The default value is the short name of the administrator account you created when setting up Mac OS X Server v10.6.
Copyright	This field is empty by default. Enter the copyright information you want to add to workflow.
Drop Box Folder	This folder is used by Qmaster when processing Compressor encodings. By default, the DropBox folder is at the root level of the shared filesystem used by Podcast Library.
Drop Box Owner Group Shortname	The default value is the short name of the administrator account you created when setting up your Mac OS X Server v10.6.
Library Language	English is the default language used for the Podcast Library feed catalogs.
Notification Language	English is the default language used for sending notifications.
Organization	This field is empty by default. Enter the name of the organization you want added to podcasts.
Postflight Script Path	The postflight script is the last task that a workflow runs.
Preflight Script Path	The preflight script is the first task that a workflow runs.

If you upgrade to Podcast Producer 2, the Default Properties list includes all default Podcast Producer 1 and Podcast Producer 2 default properties.

For more information about configuring default properties, see "Configuring Default Workflow Properties" on page 64.

Verifying Your Setup

You can verify your setup by uploading a document for processing by your Podcast Producer server.

To verify your setup:

1 On a Mac running Mac OS X v10.6 and connected to the same network that your Podcast Producer server is connected to, launch Podcast Capture (in /Applications/Utilities).

When you run Podcast Capture for the first time, the Podcast Capture Setup Assistant appears.

2 In the Welcome page of the setup assistant, click Continue.

3 In the Account Setup page of the setup assistant, choose or enter the host name or IP address of the Podcast Producer server.

4 Click Continue.

After Podcast Capture establishes a connection to the Podcast Server, the Account Setup page displays the Name and Password fields.

5 In the Name field, enter the username given to you by the Podcast Producer administrator.

6 In the Password field, enter your password.

7 To store your password in your Mac's keychain, select "Remember this password in my keychain."

8 Click Continue.

Podcast Capture sends your credentials to the Podcast Producer server. If the credentials are accepted, the main screen of the Podcast Capture window appears.

9 Click Open an Existing File and choose a short document or small QuickTime movie to send to the Podcast Producer server for encoding and publishing; then click Open.

10 From the Workflow pop-up menu, choose Montage.

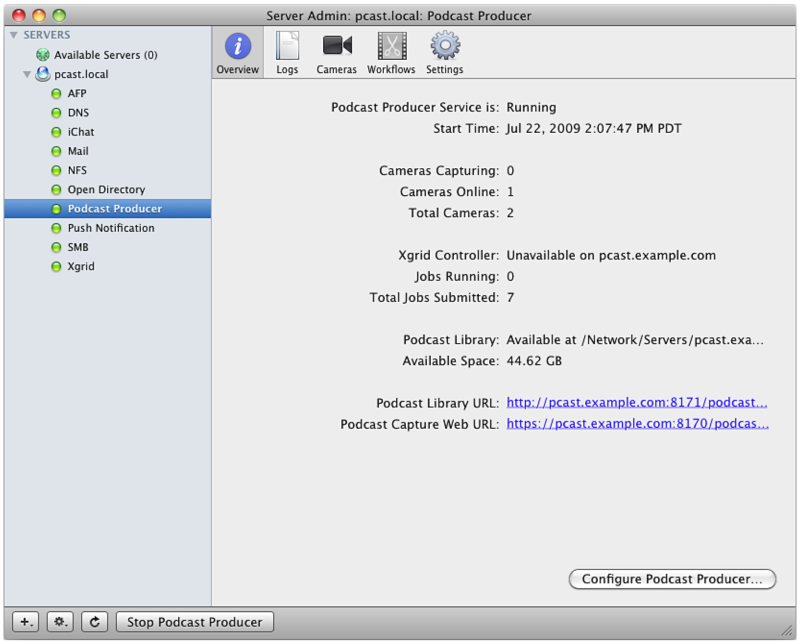
Montage is a default workflow that combines input documents into a movie.

11 In the Episode field, enter the name of your podcast.

12 In the Description field, enter a brief description of your podcast.

13 Click Submit.

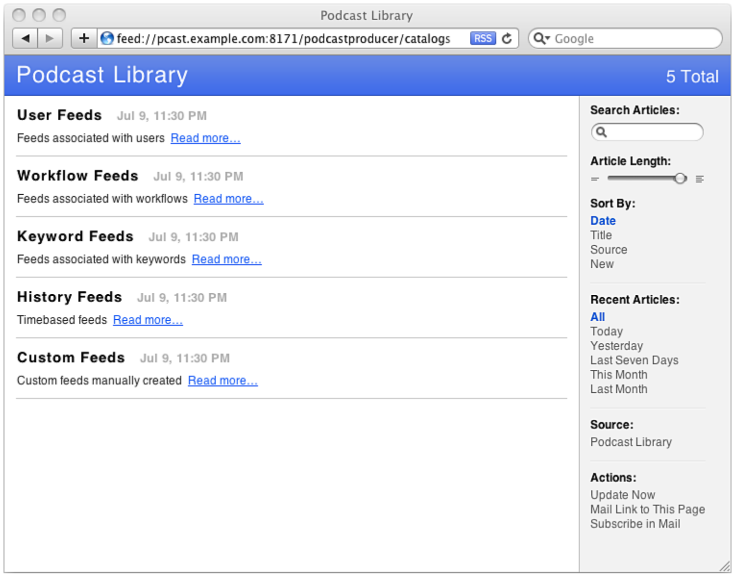
14 Open the Overview pane of the Podcast Producer server in Server Admin.



Podcast Producer Service is:	Running
Start Time:	Jul 22, 2009 2:07:47 PM PDT
Cameras Capturing:	0
Cameras Online:	1
Total Cameras:	2
Xgrid Controller:	Unavailable on pcast.example.com
Jobs Running:	0
Total Jobs Submitted:	7
Podcast Library:	Available at /Network/Servers/pcast.exa…
Available Space:	44.62 GB
Podcast Library URL:	http://pcast.example.com:8171/podcast…
Podcast Capture Web URL:	https://pcast.example.com:8170/podcas…

15 In the Overview pane, click the Podcast Library URL link.

The root catalog of the Podcast Producer Library appears in Safari or your default web browser.



Podcast Library 5 Total

User Feeds Jul 9, 11:30 PM
Feeds associated with users Read more…

Workflow Feeds Jul 9, 11:30 PM
Feeds associated with workflows Read more…

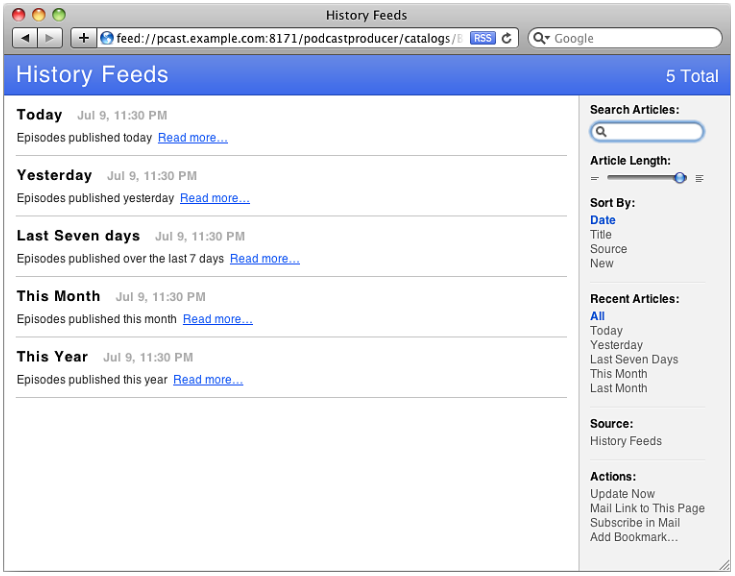
Keyword Feeds Jul 9, 11:30 PM
Feeds associated with keywords Read more…

History Feeds Jul 9, 11:30 PM
Timebased feeds Read more…

Custom Feeds Jul 9, 11:30 PM
Custom feeds manually created Read more…

16 In the main page of the Podcast Library, click History Feeds.

The History Feeds catalog appears.



History Feeds 5 Total

Today Jul 9, 11:30 PM
Episodes published today Read more…

Yesterday Jul 9, 11:30 PM
Episodes published yesterday Read more…

Last Seven days Jul 9, 11:30 PM
Episodes published over the last 7 days Read more…

This Month Jul 9, 11:30 PM
Episodes published this month Read more…

This Year Jul 9, 11:30 PM
Episodes published this year Read more…

Search Articles:

Article Length:

Sort By:
Date
Title
Source
New

Recent Articles:
All
Today
Yesterday
Last Seven Days
This Month
Last Month

Source:
History Feeds

Actions:
Update Now
Mail Link to This Page
Subscribe in Mail
Add Bookmark…

17 In the History Feeds catalog page, click Today.

After processing is done, Podcast Producer publishes your test podcast in the Podcast Library > History Feeds > Today page after Podcast Producer.

Your podcast should also appear in the Podcast Library > Workflow Feeds > Montage and Podcast Library > User Feeds > *username* pages.

From the command line:

```
// Submit file to Podcast Producer server.
podcast -s server -u username --submit --file file_path --workflow
     workflow_name --title "My First Podcast" --description "This is a
     test podcast"

// You can also remotely use the Podcast Producer Camera Agent to record
     and submit a 3-second movie to test the system:
sudo podcast --bind camera-name -u username -p password
sudo /usr/libexec/podcastproducer/pcastagentd -f /tmp/test-movie.com -d 3
sudo podcast --submit --file /tmp/test-movie.mov -w "Single Source"

// Check uploading status.
podcast -s server -u username --list_uploads
```

Binding a Mac to the Podcast Producer Server

After setting up the Podcast Producer server, you can bind the Macs you want to use for remotely recording audio and video to the Podcast Producer server using the Podcast Capture application.

To bind a Mac to the Podcast Producer service:

1. If you are using an external video camera, make sure the camera is connected to your Mac via FireWire and is turned on.
2. Open the Podcast Capture application (in /Applications/Utilities).
3. If the Podcast Capture Setup Assistant appears, complete the following:
   - a. In the Welcome page of the setup assistant, click Continue.
   - b. In the Account Setup page of the setup assistant, choose or enter the host name or IP address of the Podcast Producer server.
   - c. Click Continue.

     After Podcast Capture establishes a connection to the Podcast Server, the Account Setup page displays the Name and Password fields.
   - d. In the Name field, enter the username given to you by the Podcast Producer administrator.
   - e. In the Password field, enter your password.
   - f. To store your password in your Mac's keychain, select "Remember this password in my keychain."
   - g. Click Continue.

     Podcast Capture sends your credentials to the Podcast Producer server. If the credentials are accepted, the main screen of the Podcast Capture window appears.
4. If you've already set up Podcast Capture, complete the following:
   - a. Click Log In.
   - b. In the Server field, choose or enter the host name or IP address of the Podcast Producer server.
   - c. Click Connect.
   - d. If prompted, authenticate using your username and password.
5. Choose Podcast Capture > Preferences.
6. Click Audio/Video.
7. From the Video Source pop-up menu, choose a local camera.

   Verify that you see a preview of the camera's video.

8 From the Microphone pop-up menu, choose the audio source.

Verify that audio works by checking the audio level field below the video preview box.

9 From the Quality pop-up menu, choose an encoding format.

The text below the pop-up menu describes the settings of the selected encoding format.

10 Click Start Sharing.

11 If prompted to authenticate, enter your Mac username and password to allow Podcast Capture to make changes to your system and then click OK.

12 In the sheet that appears, enter the following information:

- Camera Name—Enter the name to assign to the camera. Because Podcast Producer uses this name to identify and control access to the camera, enter a unique, meaningful name to help the Podcast Producer administrator and Podcast Capture users better identify and determine the location of the camera. For example, instead of using a name like Camera7, use a name like Camera007-Eng_Building-Room_304.
- Server—Choose a server from the pop-up menu or enter the host name (for example, pcast_server.example.com) or IP address of the Podcast Producer server.
- Name—Enter the administrative username assigned to you by the Podcast Producer administrator.
- Password—Enter your administrative Podcast Producer password.

13 Click Start Sharing to bind the camera to the Podcast Producer server.

If the binding process is successful, the status indicator of the Remote Camera Sharing field turns green and the value of the field changes to On.

14 Close the Podcast Capture Preferences dialog box.

From the command line:

```
// Bind camera to Podcast Producer server and start the agent process.
sudo podcast -s server -u username --bind camera_name --start_agent

// To bind the camera on a system running Mac OS X v10.5:
sudo podcast -s server -u username --bind camera_name
sudo pcastctl agent start

// Check whether camera is bound.
podcast -s server -u username --isbound

/ To reset the Podcast Producer camera agent:/
sudo pcastctl agent restart
```

```
// To unbind a camera and remove the camera agent's property list:
sudo pcastctl agent stop
sudo pcast --unbind camera_name -u username -p password
sudo rm /Library/Preferences/com.apple.pcastagentd.plist
```

Accessing the Podcast Capture Web Application

The Podcast Capture web application lets you use a web browser to remotely record and submit content to a Podcast Producer server for processing and publishing.

With the Podcast Capture web application, you can:

- Record and submit video from single and dual sources
- Record and submit audio
- Submit files compatible with Apple's Quick Look technology

To access the Podcast Capture web application:

1 In Server Admin, enable the Podcast Capture web application:

  a Open Server Admin.

  b In the Servers list, select the server running Podcast Producer, then click the service disclosure triangle to show the services for administration.

  c In the service list beneath the server, select Podcast Producer Server.

  d Click the Settings button in the toolbar, and then click the General tab.

  e Select "Enable Podcast Capture web application."

  f Click Save.

2 Open a web browser on your system.

For example, open Safari on your Mac, iPhone, or Windows computer.

3 In the Address field, enter https://*server*:8170/podcastproducer/capture.

Replace *server* with the IP address or DNS name of your server (for example, pcastserver.example.com).

You can also launch Podcast Capture Web Application by clicking its URL in the Overview pane of the Podcast Producer server in Server Admin (Server Admin > *server* > Podcast Producer > Overview).

4 In the Name and Password fields, enter valid user credentials.

By default, you can use the administrator account you created when you configured your server for the first time.

5 Click Connect.

To learn how to use the Podcast Capture web application, refer to its onscreen help.

Tip: To simplify access to the web applications for users, consider creating an alias of the URL of the Podcast Capture web application. For example, you can create the https://podcastcapture.myhost.example.com alias, which users can use instead of https://*server*:8170/podcastproducer/capture. For more information on how to create a URL alias, see *Web Technologies Administration*.

Uploading Movies Using QuickTime Player

You can use QuickTime Player to submit audio, video, and screen recording to Podcast Producer for processing and delivery.

To upload movies using QuickTime Player:

1 With the movie open in QuickTime Player, choose Share > Podcast Producer.

This command opens Podcast Capture.

2 If prompted, choose the Podcast Producer server to connect to and authenticate.

3 In the Podcast Information pane of Podcast Capture, complete the following:

a From the Workflow pop-up menu, choose a workflow.

b In the Episode field, enter the name of your podcast.

c In the Description field, enter a brief description of your podcast.

d Click Submit.

Changing the Location of Spooled Recordings

By default, Podcast Capture stores recordings and metadata pending submission in the ~/Library/Application Support/Podcast Capture/ folder. The Podcast Producer Camera Agent uses the /var/pcast/agent/recordings/ folder.

When you use Podcast Capture to record podcasts locally, Podcast Capture stores the recordings in your home folder (~/Library/Application Support/Podcast Capture/), which might reside on a network volume.

To improve performance, consider storing recordings on the local hard disk using the following command:

```
defaults write com.apple.PodcastCapture RecorderDestinationFolder path
```

For example:

```
defaults write com.apple.PodcastCapture RecorderDestinationFolder /tmp/
    recordings/
```

Upgrading to Podcast Producer 2

3

Use this chapter to learn how to upgrade your Podcast Producer server to Podcast Producer 2.

The process of upgrading your Podcast Producer 1 server running on a computer with Mac OS X Server v10.5 to Podcast Producer 2 is fairly simple if you follow the steps in this chapter. Most of the work involved in upgrading the server and migrating the data to the Podcast Library is done automatically.

Before You Upgrade

Before upgrading to Podcast Producer 2, perform a complete backup of your system. You can image the system, use Time Machine, or use third-party backup software.

WARNING: To avoid service interruption, never upgrade a production system without a backup.

After the upgrade, you might need to manually update the workflows that call custom binaries or scripts.

Important: To configure Podcast Producer 2 for failover, you must start with a clean-install, not an upgrade install.

Upgrading Your Podcast Producer Server

During the upgrade process, Podcast Producer clients cannot submit jobs. Perform the upgrade process when there is little or no Podcast Producer activity to minimize the disruption.

To upgrade your Podcast Producer server to Podcast Producer 2:

1 Update your computer to the latest version of Mac OS X Server v10.5 (version 10.5.7 or later).

2 If you're using Xsan to store the shared filesystem, upgrade the computer running the Xsan metadata controller to Mac OS X Server v10.6.

3 Upgrade your computer by installing Mac OS X Server v10.6.

4 In Server Admin, check the status indicator of the Podcast Producer server.

A yellow indicator means that the Podcast Producer data is still being migrated and will be available soon. It can also mean a problem must be fixed before Podcast Producer 2 can run.

If the indicator is green, the upgrade and migration process was successful.

Troubleshooting Upgrade and Migration Issues

This section describes how to troubleshoot Podcast Producer upgrading and migration issues.

- "Adding a Service Principal" on page 49
- "Checking the Startup Log" on page 50

Adding a Service Principal

If a server is Kerberized with versions 10.5.0 to 10.5.5 of Mac OS X Server and is not updated to v10.5.7 or later, the system will not have a pcast service principal and the status indicator of the Podcast Producer server in Server Admin turns yellow.

To fix the problem, add the service principal.

To add the service principal:

1 Open Server Admin.

2 Select the server, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Open Directory.

4 Click the Settings button in the toolbar, then click the General tab.

5 Click Join Kerberos.

From the command line:

```
sso_util configure -r <REALM> -f <node> -a <diradmin> pcast
```

Checking the Startup Log

To check whether there was a problem during the upgrade process, check the Podcast Producer Server Startup log of the Podcast Producer server. This log is stored at /Library/Logs/pcastserverd/startup.log.

To check the Startup log:

1. In the service list beneath the server, select Podcast Producer.
2. Click the Logs button in the toolbar.
3. From the View pop-up menu, choose Podcast Producer Server Startup Log.
4. Check the log for problems.

Setting Up Podcast Producer for High Availability

4

Use this chapter to learn how to set up the Podcast Producer server for high availability.

Podcast Producer 2 supports failover of the Podcast Producer server process, which allows Podcast Producer clients to fail over to other Podcast Producer servers.

Xgrid supports transparent failover for clients and agents; Podcast Producer 2 is one of those clients that are supported.

How Podcast Producer Failover Works

For Podcast Producer failover to work, you must configure the primary Podcast Producer server and the failover servers to be on the same authentication realm and configured to use the same Podcast Library location in the shared file system, which must reside on an Xsan volume.

When a Podcast Producer client (camera agent, Podcast Capture, or Podcast Capture Web Application) logs in to a Podcast Producer server, it receives a list of the IP addresses of the Podcast Producer servers it can fail over to.

This list is cached and keyed using the name of the server the client was connecting to. If the client attempts to connect to that server later and it is unavailable, the client looks in the cached list and attempts to connect to each server in turn.

The client keeps trying until it succeeds in connecting to a server on the list. It then caches the new list of failover hosts based on the response it gets from the new server.

Failover is not supported for web clients, such as Safari accessing the Podcast Library or Podcast Capture Web Application.

In addition, load balancing is not supported for Podcast Producer clients such as `podcast` and Podcast Capture or the camera agents. However, you can use a hardware load balancer or Round-robin DNS for web clients. This can be of particular interest for load balancing content delivery from the Podcast Library. A hardware load balancer might also provide failover support.

If the primary Podcast Producer server or Xgrid controller fails, its clients fail over to a server on the list. Xgrid supports transparent failover for clients and agents, including Podcast Producer 2.

The Xgrid configuration is stored in the shared file system. If you set up a secondary system and point it at the primary system's shared file system location, it uses the same Xgrid controller as the primary system.

Note: If Xgrid is not set up for redundancy and the Xgrid controller fails, Podcast Producer keeps the Xgrid jobs in queue and submits to the Xgrid controller when it becomes available. To set up Xgrid redundancy, see *Xgrid Administration and High Performance Computing*.

Before Setting Up High Availability

Before you configure your Podcast Producer server for failover, note the following:

- The failover server must be bound or Kerberized into the same authentication realm as the primary Podcast Producer server.
- The failover server must not already be Kerberized into some other realm.
- The shared filesystem used by your server's Podcast Library must reside in an Xsan volume.
- The Xsan metadata controller must run on a computer with Mac OS X Server v10.6.

  For more information about setting up Xsan volumes, see *Xsan 2 Administrator's Guide* and related documentation.
- For best results, start with a fresh installation of Mac OS X Server v10.6 on the computer you intend to use as a failover server.
- Don't use the Podcast Producer Setup Assistant to configure failover servers.

Configuring Podcast Producer for Failover

You can set up computers as failover Podcast Producer servers.

Important: Clients (Podcast Capture, Podcast Capture Web Application, and camera agents) must connect successfully to the primary server at least once before they can fail over.

To configure your Podcast Producer server for failover:

1 Install Mac OS X Server v10.6 on the computers you intend to use as failover servers.

2 Bind the failover servers into the same authentication realm as the primary Podcast Producer server.

3 In Server Admin, enable the Podcast Producer server:

a Connect to the server.

b Click Settings and then Services.

c Select the checkbox for Podcast Producer Server.

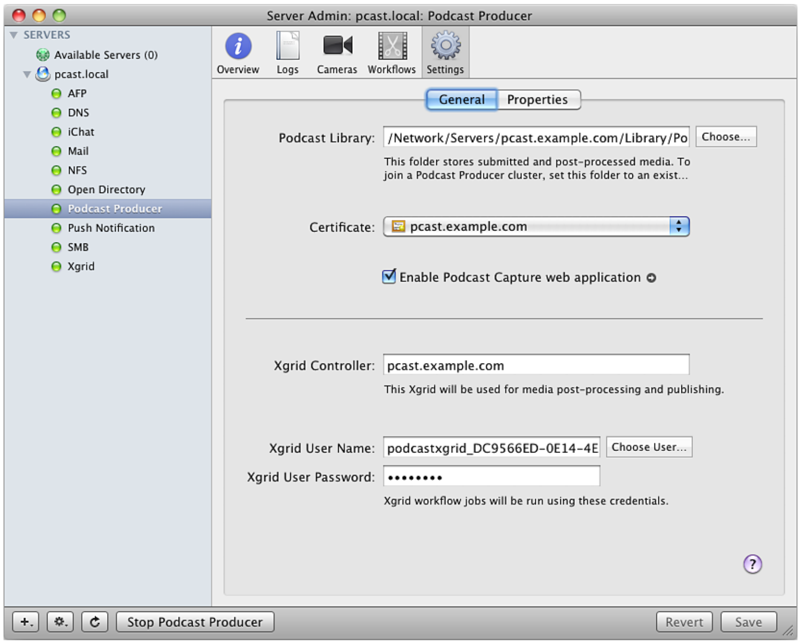
d Click Save.

4 In Server Admin, configure the Podcast Producer server:

a Select the server, then click the service disclosure triangle to show the services for administration.

b In the service list beneath the server, select Podcast Producer.

c Click the Settings button in the toolbar, then click the General tab.



Podcast Library: /Network/Servers/pcast.example.com/Library/Po

This folder stores submitted and post-processed media. To join a Podcast Producer cluster, set this folder to an exist…

Certificate: pcast.example.com

☑ Enable Podcast Capture web application

Xgrid Controller: pcast.example.com

This Xgrid will be used for media post-processing and publishing.

Xgrid User Name: podcastxgrid_DC9566ED-0E14-4E

Xgrid User Password: ••••••••

Xgrid workflow jobs will be run using these credentials.

d In the Podcast Library field, enter the same path used by the primary Podcast Producer server.

You can also set the path to the Podcast Library by setting the value of the `shared_filesystem` key in /Library/Preferences/com.apple.pcastserverd.plist.

For example, from the command line, you can use the following command to set the path to the Podcast Library:

```
sudo defaults write /Library/Preferences/com.apple.pcastserverd shared_
   file_system_path
```

e Click Save.

  The Podcast Producer server uses the information in the Podcast Library of the primary server to configure the other settings needed for failover, such as Xgrid settings.

5 If you're using an SSL certificate on the primary server, in Server Admin, configure the failover server to use its own certificate:

   a Select the server, then click the service disclosure triangle to show the services for administration.

   b In the service list beneath the server, select Podcast Producer.

   c Click the Settings button in the toolbar, then click the General tab.

   d From the Certificate pop-up menu, choose a certificate or create one.

   e Click Save.

6 To verify that your Podcast Producer is setup for failover, run the following command on the primary and failover servers:

```
podcast --listinfo --server hostname
```

This command returns a property list file containing a list (under the `cluster_members` key) of the Podcast Producer servers that a client can fail over to.

Troubleshooting High Availability Issues

This section describes common troubleshooting issues and their solution.

Failover After First Login

Clients (Podcast Capture and camera agents) must connect successfully to the primary server at least once before they can fail over.

Xsan Redundancy

Use standard Xsan techniques to provide failover protection. Set up multiple Xsan metadata controllers for failover.

Failover and Express Setup

Consider the following configuration of a Podcast Producer server:

- The server is set up using the Express setup option of the Podcast Server Assistant.
- The server is bound to a different authentication realm than the realm used by the primary Podcast Producer server. (Express setup creates a standalone Open Directory master and uses it to Kerberize the server.)

To use this server as a failover Podcast Producer server:

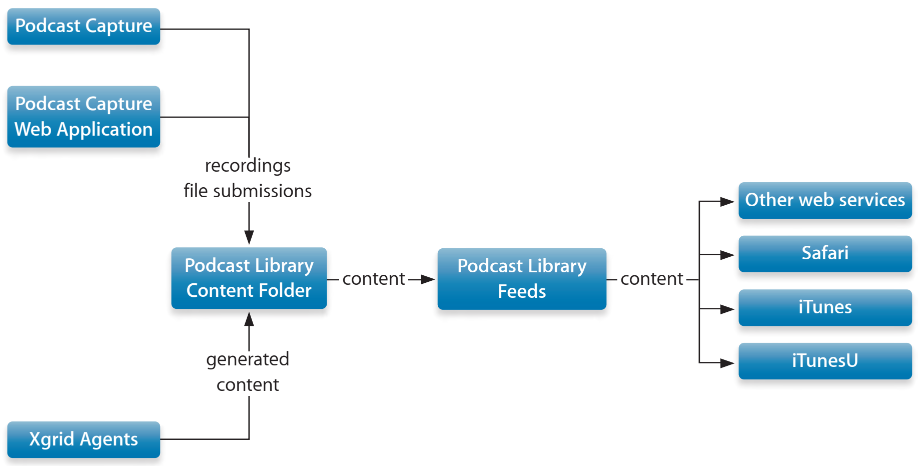
1. Stop the Podcast Producer server.
2. Destroy the Open Directory master on the server.
3. Bind the server to the same directory as the one used by the primary server.
4. Change the Podcast Library location to point to the Podcast Library location used by the primary server.
5. Start the Podcast Producer server.

Podcast Library

5

Use this chapter to learn how the Podcast Library works.

The Podcast Library is central to Podcast Producer. It is responsible for storing, organizing, and serving content to users.



Overview of Podcast Library

The Podcast Library fulfills the following functions:

- Stores information and content

  Podcast Library is a repository that stores everything that goes through the Podcast Producer server. This includes workflows, job submissions, original content, intermediate files, all versions of published podcasts, metadata, and other information needed for the processing and publishing of podcasts.

- Organizes information and content

  Podcast Library organizes information based on the metadata available to it. For example, Podcast Library uses information such as the camera used to record a video, the name of the user who submitted the video, the time of day the recording was submitted, the type of recording, and the name of the workflow that was processed to create folders or categories for organizing videos.

- Serves content

  Podcast Library serves content to users over RSS and Atom feeds. For example, your podcasts can be hosted on your server but accessed through iTunes U, which can aggregate the feeds but leaves the content on the server.

  Atom feeds simplify the distribution of multiple podcast versions, such as iPod, Apple TV, and audio only, because each Atom feed can contain multiple versions and the viewer's playback device picks the best version.

The Podcast Library consists of the following:

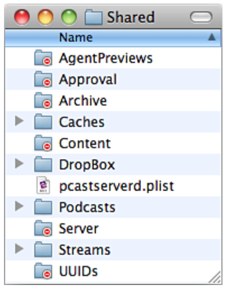
- Code, embedded in pcastserverd, for storing, organizing, and serving content
- A shared file system where information and content is stored

Shared File System

Podcast Library requires a shared file system for storing information and content. The supported shared file systems are Xsan, HFS, and NFS.

Important: Don't modify files in the shared file system directly. Instead, use the `podcast` command-line tool. For example, to add a workflow, use the `podcast --installworkflow` command. To delete a workflow, use the `podcast --deleteworkflow`. For more information about installing and deleting workflows using `podcast`, see "Deploying Workflows" on page 67 and "Removing Workflows" on page 63.

The following diagram shows the contents of the shared file system:


Shared

Name

AgentPreviews
Approval
Archive
Caches
Content
DropBox
pcastserverd.plist
Podcasts
Server
Streams
UUIDs

Podcast Library stores the following in its shared file system:

- Approval, Archive, Podcasts, and Streams folders
  - The Approval folder stores content that requires approval.
  - The Archive folder stores copies of the submitted or processed recordings for archival purposes.
  - The Podcast folder is where Podcast Producer stores finished podcasts.
  - The Streams folder stores QTSS streams produced by Podcast Producer.

  These folders exist to support Podcast Producer 1 workflows. Podcast Producer 2 workflows don't use these folders.
- AgentPreviews

  For every remote camera, this folder stores the last preview frame received.
- Caches

  This folder contains the workflow and resource caches used for processing workflows. This folder also contains caches of the web pages that the Podcast Library has published so that the next time a feed is requested, Podcast Library spends less time processing the request.
- Content

  Stores the submitted and published content.
- DropBox

  This folder allows Podcast Producer to share files with Qmaster when processing Compressor encodings.

  To protect content, Podcast Producer doesn't give Qmaster the file system permissions needed to look into the Content folder. Instead, Podcast Producer moves the files it needs to hand off to Qmaster into the DropBox folder.

  Podcast Producer and Qmaster have the needed permissions to access this folder. When Qmaster finishes processing the Compressor encodings, it saves the output into the DropBox folder. Podcast Producer moves the processed files from this folder into the Published folder of the Content folder.
- pcastserverd.plist

  This property list file defines the identity of the Podcast Library.
- Server

  This folder stores the following:
  - The jobs that will be submitted to Xgrid for processing
  - All resources installed on the server
  - The workflows deployed by the Podcast Producer server

  - Cluster preferences (for example, the Podcast Producer Xgrid settings defined in Server Admin)
  - Member preferences for each member of the Xgrid cluster
- UUIDs

  This folder is an index of the Content folder. It stores aliases to published podcasts.

Accessing Published Content on the Podcast Library

The Podcast Library URL where Podcast Producer publishes content is:

http://*PodcastProducerServer*:8171/podcastproducer/catalogs

or

https://*PodcastProducerServer*:8170/podcastproducer/catalogs

This URL opens the main page of the Podcast Library catalog.

For more information about the Podcast Library catalog, see "Podcast Library Feed Structure" on page 74.

Removing Published Content

To remove feeds and related content from the Podcast Library, remove the relevant content from the Content folder in the shared file system (defined in Server Admin).

This folder organizes content by date. When Podcast Producer receives a request to process and deliver a podcast episode, if needed, Podcast Producer creates a folder in *shared_file_system*/Contents/ and sets the folder's name to the current date (year-month-date). Then, inside the day folder, Podcast Producer creates a .prb bundle and adds to it all files associated with the podcast episode.

Note: Podcast Producer uses UUIDs to name .prb bundles in *shared_file_system/*Contents/.

To remove feeds and related content from the Podcast Library:

1 Archive the day folders or .prb bundles of the podcast episodes you want to delete.

2 Delete the day folders or .prb bundles.

3 Resynchronize the Podcast Library by entering the following command:

```
pcastconfig --sync_library
```

This command removes the references to the deleted podcast episodes from the Podcast Library feed pages.

Important: The `podcast --removefeed` command only removes the feed, but not the content.

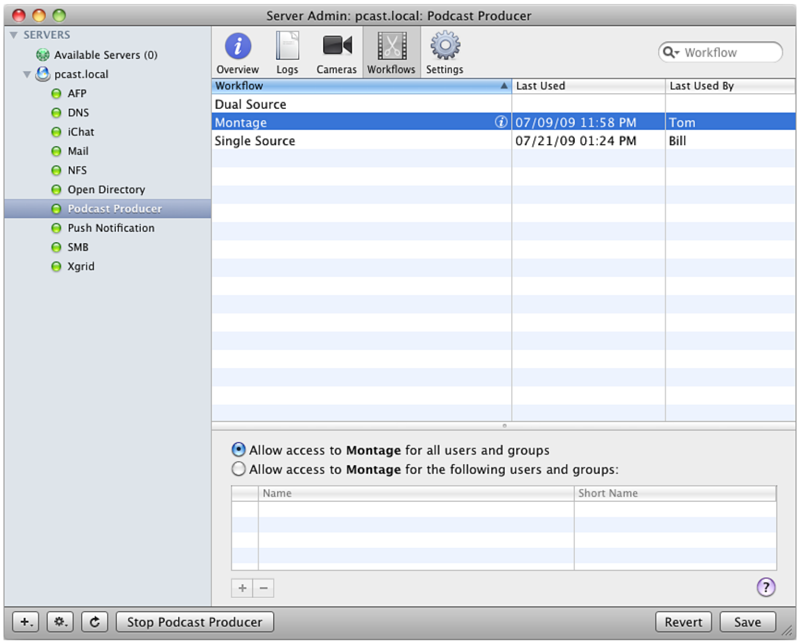
Managing Workflows

6

This chapter describes how to manage workflows and configure their default and custom properties in Server Admin.

Workflows are an essential part of Podcast Producer. Learning how to manage workflows and how to configure their properties allows you to streamline the podcast generation process and make the best use of your computing resources.

You can manage workflows using Server Admin.



Workflow	Last Used	Last Used By
Dual Source		
Montage	07/09/09 11:58 PM	Tom
Single Source	07/21/09 01:24 PM	Bill

[✓ Allow access to **Montage** for all users and groups]
Allow access to **Montage** for the following users and groups:

Name	Short Name

You can also mange workflows using the `podcast` tool.

Controlling Access to Workflows in Podcast Capture

Podcast Producer server lets you specify which workflows users can see (and which are allowed to be used) in Podcast Capture and the `podcast` command-line tool.

To control access to a workflow:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Workflows.
5. Select a workflow in the Workflow list.
6. To restrict access to the workflow, click "Allow access to *workflow name* for the following users and groups."
7. Click the (+) button to add users and groups to the list of users and groups that can access the selected workflow.

   In the Users and Groups window, click Users and drag users to the list.

   In the Users and Groups window, click Groups and drag groups to the list.

   To delete users and groups from the list, select them and click (-).
8. Click Save.

You can also select multiple workflows and configure their access control settings at the same time.

To allow all users and groups to see the selected workflows in Podcast Capture, click "Allow access to *workflow name* for all users and groups."

From the command line:

```
// Enable access control for workflows.
podcast enableacls --resource_type Workflow

// Get the UUID of the workflow from the list of workflows.
podcast --listworkflows

// Enable access control for the workflow
// whose UUID is workflow_uuid.
podcast enableacl --resource_type Workflow --resource_uuid workflow_uuid

// Give a user access to the workflow.
podcast --addaccess --resource_type Workflow --resource_uuid workflow_
     uuid --record_type user --shortname username
```

```
// Give a group access to the workflow.
podcast --addaccess --resource_type workflow --resource_uuid workflow_
     uuid --record_type group --shortname groupname

// Verify that an intended user has an access to a resource
podcast  --checkaccess --resource_type Workflow --resource_uuid workflow_
     uuid --shortname username
```

Monitoring Workflow Usage

You can use the Workflows pane of the Podcast Producer service to see the last time a workflow was used and by whom.

To monitor workflow usage:

1 Open Server Admin.

2 Select the server, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Podcast Producer.

4 Click Workflows.

The Last Used column shows the last time a workflow was used. The Last Used By column shows the short name of the user who last used the workflow.

You can also monitor the processing of workflows as described in "Monitoring Xgrid Job Progress" on page 128.

Filtering Workflows

You can use the search field in the Workflows pane of the Podcast Producer service to specify a search criteria for listing workflows.

To filter workflows:

1 Open Server Admin.

2 Select the server, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Podcast Producer.

4 Click Workflows.

5 To search for workflows by keywords, from the Search pop-up menu, choose Workflow and enter the keywords.

Only workflows whose names contain the specified keywords appear in the list.

6 To search for workflows by filename, from the Search pop-up menu, choose Workflow File Name and enter the name of the workflow file.

Only the workflow whose file name is specified in the Search field appears in the list.

7 To list the workflows that were last used by a user, from the Search pop-up menu, choose Last Used By and enter the short name of the user.

From the command line:

```
// List the workflows available to the
// Podcast Producer server
podcast -s server -u username --listworkflows
```

Removing Workflows

You can use the `podcast` command-line tool to remove workflows from Podcast Producer.

From the command line:

```
// Get the UUID of the workflow
podcast --listworkflows

// Delete workflow UUID.
podcast --deleteworkflow --workflow_uuid UUID
```

Disabling and Enabling Workflows

You can use the `podcast` command-line tool to disable and enable workflows. When you disable a workflow, Podcast Capture and Podcast Capture Web Application users don't see the workflow displayed in the Workflow pop-up menu. However, the workflow is still deployed and you can see it listed in Server Admin.

If you disable or enable workflows, changes don't take effect until Podcast Capture and Podcast Capture Web Application are restarted.

From the command line:

```
// Get the UUID of the workflow
podcast --listworkflows

// Disable workflow UUID.
podcast --disableworkflow --workflow_uuid UUID

// Enable workflow UUID.
podcast --enableworkflow --workflow_uuid UUID
```

Displaying Workflow Information

You can display information about a workflow using its corresponding information button.

To display workflow information:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Workflows.
5. Select a workflow in the Workflow list.
6. Click the information button to the right of the name.

   The Workflow Information window appears. This window displays information about the workflow, including the workflow's filename and location on the Podcast Producer server, a description of what the workflow does, and notes added to the workflow file.

From the command line:

```
// Get the UUID of the workflow from the list of workflows.
podcast --listworkflows

// Display workflow information.
podcast -s server -u bill --infoworkflow --workflow_uuid UUID
```

Configuring Workflow Properties

Server Admin allows you to configure default and custom workflow properties.

- "Configuring Default Workflow Properties" on page 64
- "Configuring Custom Workflow Properties" on page 66

Configuring Default Workflow Properties

Podcast Producer defines a set of default properties that are common to most workflows.

You can use Server Admin to configure the following Podcast Producer default workflow properties.

Property	Value
Administrator Short Name	The default value is the short name of the administrator account you created when setting up your Mac OS X Server v10.6.
Copyright	This field is empty by default. Enter the copyright information you want to add to all workflows.
Drop Box Folder	This folder is used by Qmaster when processing Compressor encodings. By default, the DropBox folder is at the root level of the shared filesystem used by Podcast Library.
Drop Box Owner Group Shortname	The default value is the short name of the administrator account you created when setting up Mac OS X Server v10.6.
Library Language	English is the default language used for the Podcast Library feed catalogs.
Notification Language	English is the default language used for sending notifications.
Organization	This field is empty by default. Enter the name of the organization you want added to podcasts.
Postflight Script Path	The postflight script is the last task that a workflow runs.
Preflight Script Path	The preflight script is the first task that a workflow runs.

To configure default workflow properties:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Settings.
5. Click Properties.
6. Click the triangle next to Default Properties to display the properties.
7. To change the value of a property, double-click the property's value and enter a new value.
8. Click Save.

From the command line:

```
// Change the value of a default property.
pcastconfig --add_property property_name --value value [--protect]
```

Configuring Custom Workflow Properties

You can modify sample Podcast Producer workflows and define customizable properties. You can then add the properties to the list of Custom Properties in Server Admin and change their values as needed.

- "Adding Custom Workflow Properties" on page 66
- "Deleting Custom Workflow Properties" on page 66
- "Modifying Custom Workflow Properties" on page 67

Adding Custom Workflow Properties

You can use Server Admin to add custom workflow properties.

To add a custom workflow property:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Settings.
5. Click Properties.
6. Click the triangle next to Custom Properties to display custom properties.
7. Click the (+) button to add a custom property.
8. Double-click the name field of the property and enter its name.

   If a property (default or custom) having the same name exists, a space character followed by the number 2 is added to the name. Property names must be unique.
9. Double-click the value field of the property and enter its value.
10. Click Save.

From the command line:

```
// Add custom property.
pcastconfig --add_property property_name --value value [--protect]
```

Deleting Custom Workflow Properties

You can use Server Admin to delete custom workflow properties.

To delete custom workflow properties:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Settings.

5 Click Properties.

6 Click the triangle next to Custom Properties to display custom properties.

7 Select the property to delete and click the (-) button.

8 Click Save.

From the command line:

```
// Delete custom property.
pcastconfig --remove_property property_name
```

Modifying Custom Workflow Properties

You can use Server Admin to modify custom workflow properties.

To modify custom workflow properties:

1 Open Server Admin.

2 Select the server, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Podcast Producer.

4 Click Settings.

5 Click Properties.

6 Click the triangle next to Custom Properties to display the custom properties.

7 To change the value of a property, double-click the property's value and enter a value.

8 (Optional) To encrypt a property, click the property's checkbox.

9 Click Save.

From the command line:

```
// Change the value of a custom property.
pcastconfig --add_property property_name --value value [--protect]
```

Deploying Workflows

You can deploy workflows using Podcast Composer and using the `podcast` command-line tool.

For more information about deploying workflows using Podcast Composer, see *Podcast Composer User Guide*.

From the command line:

```
// Install the workflow specified by workflow_bundle_path
// on the Podcast Producer server.
podcast --installworkflow --path workflow_bundle_path [--overwrite]
```

Downloading Workflows

You can download workflows from the Podcast Producer server using Podcast Composer or the `podcast` command-line tool.

For more information about deploying workflows using Podcast Composer, see *Podcast Composer User Guide*.

From the command line:

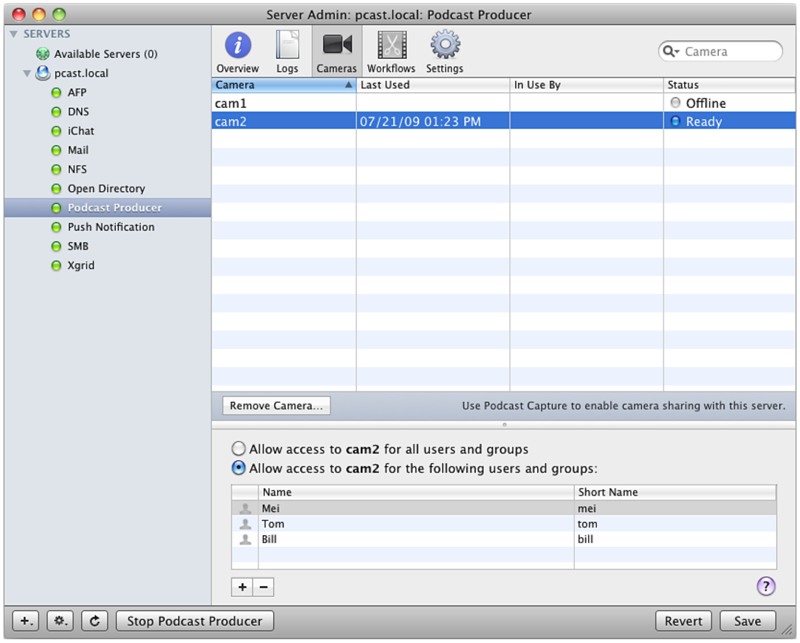
```
// Get the UUID of the workflow
podcast --listworkflows


// Download workflow UUID to the specified
// workflow bundle path.
podcast --downloadworkflow --workflow_uuid UUID --path workflow_bundle_
     path
```

Managing Cameras

7

Learn how to manage and monitor cameras used by Podcast Producer.

You can manage cameras using Server Admin.



Camera	Last Used	In Use By	Status
cam1			Offline
cam2	07/21/09 01:23 PM		Ready

Use Podcast Capture to enable camera sharing with this server.

Allow access to **cam2** for all users and groups
[✓ Allow access to **cam2** for the following users and groups:]

Name	Short Name
Mei	mei
Tom	tom
Bill	bill

You can also mange cameras using the `podcast` and `pcastconfig` tools.

Managing Cameras

The Podcast Producer server allows you to control and monitor camera usage.

- "Controlling Access to Cameras in Podcast Capture" on page 70
- "Removing Cameras" on page 71
- "Filtering Cameras" on page 71

Controlling Access to Cameras in Podcast Capture

The Podcast Producer server lets you specify which cameras users can choose in Podcast Capture.

To control access to a camera:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Cameras.
5. Select a camera in the Cameras list.
6. To restrict access to the camera, click "Allow access to *camera name* for the following users and groups."
7. Click the (+) button to add users and groups to the list of users and groups that can access the selected camera.

   In the Users and Groups window:

   - Click Users and drag users to the list.
   - Click Groups and drag groups to the list.

   To delete users or groups from the list, select them and click (-).
8. Click Save.

You can also select multiple cameras and simultaneously configure their settings.

To allow all users and groups to see the selected camera in Podcast Capture, click "Allow access to *camera name* for all users and groups."

From the command line:

```
// Enable access control for cameras.
podcast enableacls --resource_type Camera
// Get the UUID of the camera.
podcast --listcameras
// Enable access control for the camera
// whose UUID is camera_uuid.
podcast enableacl --resource_type Camera --resource_uuid camera_uuid
// Give a user access to the camera.
podcast --addaccess --resource_type Camera --resource_uuid camera_uuid
     --record_type user --shortname username
// Give a group access to the camera.
podcast --addaccess --resource_type Camera --resource_uuid camera_uuid
     --record_type group --shortname groupname
```

```
// Verify that an intended user has an access to a resource
podcast  --checkaccess --resource_type Camera --resource_uuid camera_uuid
     --shortname username
```

Removing Cameras

The Podcast Producer server lets you remove cameras from the list of cameras in Server Admin.

To remove a camera:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Cameras.
5. Select a camera in the Cameras list.
6. Click Remove Camera.
7. Click OK.

From the command line:

```
// Unbind the camera so it's no longer available to
// Podcast Producer server.
podcast --unbind camera_name
// Hide the camera so users can't see it.
pcastconfig --disable_camera camera_name
// Show the camera so users can see it.
pcastconfig --enable_camera camera_name
```

Filtering Cameras

You can use the search field in the Cameras pane of the Podcast Producer service in Server Admin to specify a filtering or search criteria for listing cameras.

To filter cameras:

1. Open Server Admin.
2. Select the server, then click the service disclosure triangle to show the services for administration.
3. In the service list beneath the server, select Podcast Producer.
4. Click Cameras.
5. To search for cameras by keywords, from the Search pop-up menu, choose Camera and enter the keywords.

   Only the cameras whose names contain the specified keywords appear in the list.

6 To list cameras that were last used by a specific user, from the Search pop-up menu, choose Last Used By and enter the short name of the user.

7 To list cameras based on their status, from the Search pop-up menu, choose Status and enter the name of the status.

From the command line:

```
// List the cameras available to the
// Podcast Producer server
podcast -s server -u username --listcameras
```

Checking Camera Status

You can use Server Admin to check if a camera agent is reachable and listening to Podcast Producer requests.

To check camera status:

1 Open Server Admin.

2 Select the server, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Podcast Producer.

4 Click Cameras.

For every camera in the list, Server Admin lists the following:

- The date the camera was last used
- If in use, the long name of the person using the camera
- The status of the camera: Offline (gray), Ready (blue), Recording (green)

Tip: To ensure that the camera list displays the most up-to-date information, choose View > Refresh.

From the command line:

```
// List the cameras available to the
// Podcast Producer server
podcast -s server -u username --status camera_name
// The status is indicated by the recording_status key in the output, as
     in the following example:
…
    <key>recording_status</key>
    <string>online</string>
…
```

Managing Feeds

8

Use this chapter to learn how to manage Podcast Library feeds.

Podcast Producer 2 serves content using RSS and Atom feeds. The use of feeds greatly simplifies the delivery process and conforms to industry standards.

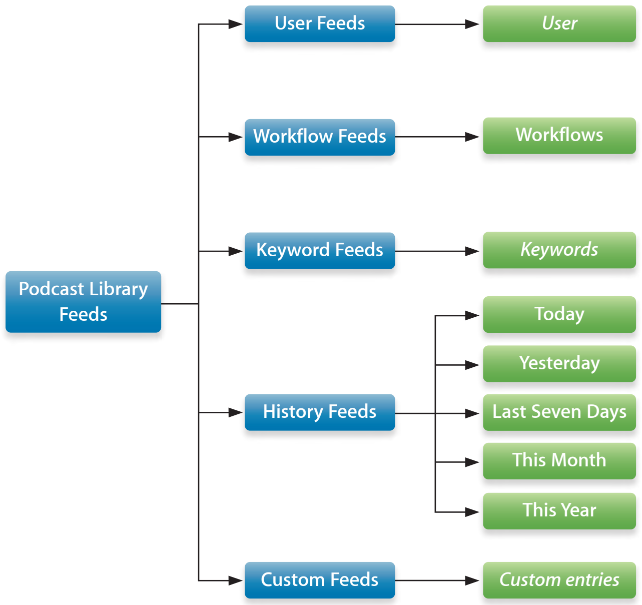
Podcast Library Feeds

Podcast Producer server provides a URL link in its Overview pane in Server Admin that you can click to access the Podcast Library feeds.

By default, the URL is feed://*server_address*:8171/podcastproducer/catalogs.

Podcast Library Feed Structure

The following diagram illustrates the hierarchy of Podcast Library feeds.



In the diagram, blue boxes represent catalogs and green boxes represent feeds.

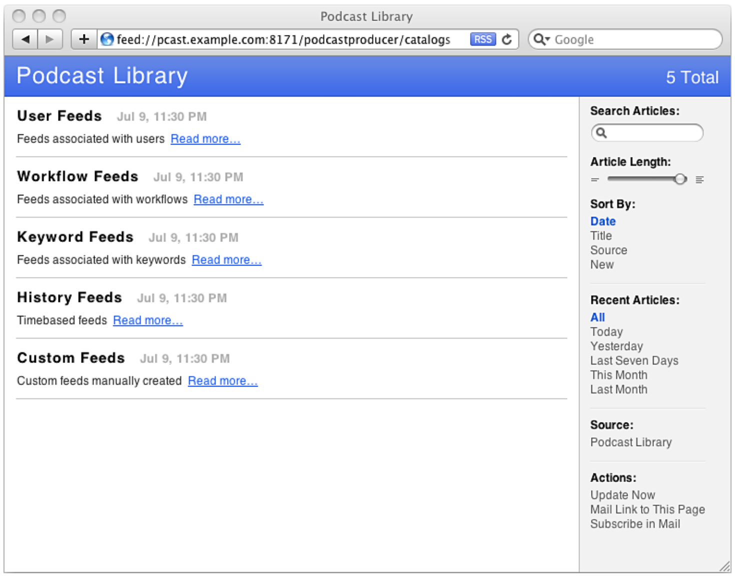
The Users, Workflows, Keywords, and Custom entries boxes can represent multiple feeds.

For example, by default, the Workflow Feeds page contains links to these pages: Single Source, Montage, and Dual Source. Each page stores the podcast episodes generated using the corresponding workflow. Every time you deploy a workflow, Podcast Producer creates a page and adds a link to it in the Workflow Feeds page.

Note: Safari displays the catalog and feed pages using its default RSS view.

Podcast Library Catalogs and Feeds

The main page, Podcast Library, is a super catalog of all feeds. It groups feeds into the following catalogs.


Podcast Library

feed://pcast.example.com:8171/podcastproducer/catalogs

Podcast Library 5 Total

User Feeds Jul 9, 11:30 PM
Feeds associated with users Read more…

Workflow Feeds Jul 9, 11:30 PM
Feeds associated with workflows Read more…

Keyword Feeds Jul 9, 11:30 PM
Feeds associated with keywords Read more…

History Feeds Jul 9, 11:30 PM
Timebased feeds Read more…

Custom Feeds Jul 9, 11:30 PM
Custom feeds manually created Read more…

These catalogs are:

- User Feeds

  Each feed in this catalog represents a user and links to another page that lists the podcast episodes submitted by the user. Podcast Library creates a user feed in this catalog after a user submits content to Podcast Producer for the first time.

- Workflow Feeds

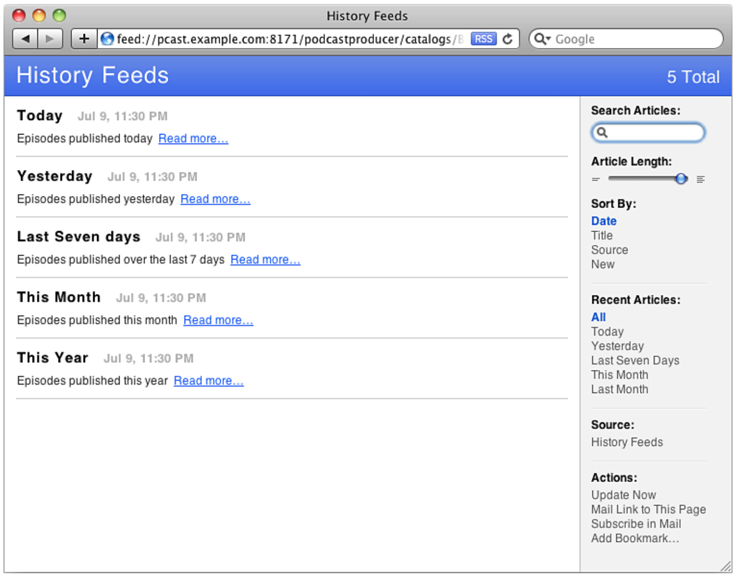
  Each feed in this catalog represents a workflow and links to another page that lists the podcast episodes generated by the workflow. Podcast Library adds workflow feeds to this catalog after workflows are deployed on the Podcast Producer server.

- Keyword Feeds

  Each feed in this catalog represents a keyword associated with a workflow and links to another page that lists the podcast episodes associated with the keyword. Podcast Library adds keyword feeds to this catalog after workflows with keywords defined in them are deployed on the Podcast Producer server.

- History Feeds

This catalog groups feeds based on when podcast episodes were submitted. This catalog links to the following feeds:


History Feeds

feed://pcast.example.com:8171/podcastproducer/catalogs/E

History Feeds 5 Total

Today Jul 9, 11:30 PM
Episodes published today Read more…

Yesterday Jul 9, 11:30 PM
Episodes published yesterday Read more…

Last Seven days Jul 9, 11:30 PM
Episodes published over the last 7 days Read more…

This Month Jul 9, 11:30 PM
Episodes published this month Read more…

This Year Jul 9, 11:30 PM
Episodes published this year Read more…

Today Lists podcast episodes published today

Yesterday Lists podcast episodes published a day earlier

Last Seven Days Lists podcast episodes published in the last seven days

This Month Lists podcast episodes published this month

This Year Lists podcast episodes published this year

- Custom Feeds

  Each feed in this catalog represents a custom grouping of podcast episodes. For example, you can use the `podcast` command to create a feed that lists all podcast episodes submitted by College of Medicine faculty.

Catalog and Feed URLs

The following are examples of catalog and feed URLs:

- Atom Root catalog URL:

  feed://pcast.example.com:8171/podcastproducer/catalogs

- Atom Catalog URL:

  feed://pcast.example.com:8171/podcastproducer/catalogs/13B819AF-1E82-4F7A-8F66-7B1E436949B4

- Atom Feed URL:

  feed://pcast.example.com:8171/podcastproducer/atom_feeds/3F747A75-562C-4025-A342-CA627D4794D4

- RSS Feed URL:

  feed://pcast.example.com:8171/podcastproducer/rss_feeds/3F747A75-562C-4025-A342-CA627D4794D4

- Shortcut Atom feed URLs:

  feed://pcast.example.com:8171/podcastproducer/user_atom_feeds/diradmin

  feed://pcast.example.com:8171/podcastproducer/workflow_atom_feeds/Single%20 Source

  feed://pcast.example.com:8171/podcastproducer/keyword_atom_feeds/hot

Note: The SSL port is 8170. The non-SSL port is 8171.

In Atom catalog URLs, the UUID of the catalog is the last part of the URL, as indicated by the bold text in the following example:

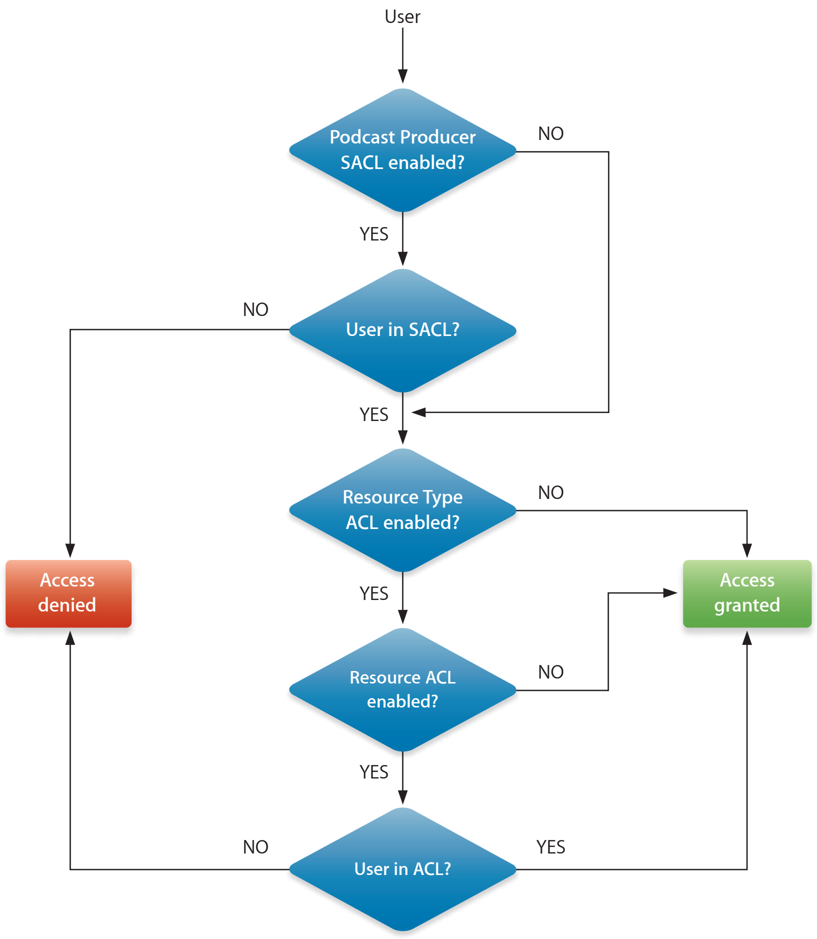
feed://pcast.example.com:8171/podcastproducer/catalogs/**`13B819AF-1E82-4F7A-8F66-7B1E436949B4`**

In Atom feed URLs, the UUID of the feed is the last part of the URL, as indicated by the bold text in the following example:

feed://pcast.example.com:8171/podcastproducer/atom_feeds/**`3F747A75-562C-4025-A342-CA627D4794D4`**

Controlling Access to Feeds

Podcast Producer provides several levels of access control to resources (cameras, workflows, and feeds) as shown in the following diagram.



Anyone can access Podcast Library feeds. By default, the following access controls are disabled:

- Podcast Producer SACL
- The use of ACLs for the feed resource type

In Podcast Producer, you control access to feeds using the `podcast` command. For more information about `podcast`, see its man page.

Enabling and Configuring the Podcast Producer SACL

You enable and configure the Podcast Producer SACL in Server Admin.

To enable Podcast Producer Server for administration:

1. Open Server Admin and connect to the server.
2. Click the Access button in the toolbar.
3. Click Services.
4. Click "For selected services below."
5. Select Podcast Producer.
6. Click "Allow only users and groups below."
7. Click the Add (+) button and add users or groups:
   a. Click Users.
   b. Drag users into the list.
   c. Click Groups.
   d. Drag groups into the list.
8. Click Save.

Enabling Access Control for Feeds

Whether SACL is enabled for Podcast Producer or not, you can enable access control for feeds using the `podcast` command.

To enable access control for feeds:

```
podcast --enableacls --resource_type Feed
```

When you enable access control for feeds, users with administrative privileges can access Podcast Library feeds. Other users can't access feeds unless you add them to the ACL of every feed you want to grant them access to.

Note: By default, when Podcast Producer creates a user feed, it adds the user to the ACL of the feed.

To disable access control for feeds:

```
podcast --disableacls --resource_type Feed
```

If you disable access control for feeds, any user in the same directory domain can access the feeds unless access is restricted in the Podcast Producer SACL.

Enabling and Configuring a Feed's ACL

You can control access to a feed by enabling its ACL. However, for this to have an effect, you must first enable access control for feeds, as described in "Enabling Access Control for Feeds" on page 79.

To enable and configure a feed's ACL:

1 Get the UUID of the feed.

```
podcast --listfeeds
```

In the property list generated by this command, the UUID is the value of the uuid key (<key>uuid</key>).

You can also get a feed's UUID from the feed's URL. The UUID is the last element of the URL.

2 Enable ACL for the feed.

```
podcast --enableacl --resource_type Feed --resource_uuid UUID
```

3 Add users and groups to the feed's ACL.

```
podcast --addaccess --resource_type Feed --resource_uuid UUID --record_
     type {group | user} --shortname name
```

To remove a user or group from the ACL:

```
podcast --removeaccess --resource_type Feed --resource_uuid UUID
     --record_type {group | user} --shortname name
```

4 Verify that the feed's ACL has all of your entries.

```
podcast --showacl --resource_type Feed --resource_uuid UUID
```

5 Check whether users have access to the feed.

```
podcast --checkaccess --resource_type Feed --resource_uuid UUID
     --shortname name
```

To disable the feed's ACL:

```
podcast --disableacl --resource_type Feed --resource_uuid UUID
```

Using Catalog Administration Commands

To manage catalogs, use the catalog administration commands of `podcast`.

These commands let you set the logo for the catalog and configure its title, subtitle, author shortname, and copyright properties.

To set the logo for a catalog:

1 Get the UUID of the catalog.

```
podcast --listcatalogs
```

In the property list generated by this command, the UUID is the value of the uuid key (<key>uuid</key>).

2 Set the logo for the catalog.

Important: Even if the image is in the Atom feed, you Atom reader might not display the image, like in the case of Safari.

```
podcast --setcatalogimage --catalog_uuid UUID --path PATH_TO_IMAGE
```

To set the title, subtitle, author shortname, and copyright properties for a catalog:

1 Get the UUID of the catalog.

```
podcast --listcatalogs
```

2 Set the title property for the catalog.

```
podcast --setcatalogproperty --catalog_uuid UUID --property_name title
     --value value
```

3 Set the subtitle property for the catalog.

```
podcast --setcatalogproperty --catalog_uuid UUID --property_name subtitle
     --value value
```

4 Set the author shortname property for the catalog.

```
podcast --setcatalogproperty --catalog_uuid UUID --property_name author_
     shortname --value value
```

5 Set the copyright property for the catalog.

```
podcast --setcatalogproperty --catalog_uuid UUID --property_name
     copyright --value value
```

For more information about catalog administration commands, see the man page of `podcast`.

Using Feed Administration Commands

To manage feeds, use the feed administration commands of `podcast`.

For more information about these commands, see the man page of `podcast`.

Enabling and Disabling Feeds

By default, all feeds are enabled. However, you can prevent access to a feed by disabling it.

When you disable a feed, it disappears from the catalog. Users won't know it's there. When you enable the feed, it becomes visible and accessible again.

To disable a feed:

1 Get the UUID of the feed.

```
podcast --listfeeds
```

2 Disable the feed.

```
podcast --disablefeed --feed_uuid UUID
```

To enable the feed:

```
podcast --enablefeed --feed_uuid UUID
```

Adding and Removing Feeds

You can add feeds to the User Feeds, Keyword Feeds, and Custom Feeds catalogs.

To add a feed:

- To add a feed to the User Feeds catalog:

  ```
  podcast --addfeed --shortname name
  ```

  For example, the following command creates the feed Tom Clark, which contains all podcasts produced by Tom whose shortname is tom:

  ```
  podcast --addfeed --shortname tom
  ```

- To add a feed to the Keyword Feeds catalog:

  ```
  podcast --addfeed --keyword keyword_string
  ```

  For example, the following command creates the feed Physics, which contains all podcasts tagged with the keyword Physics:

  ```
  podcast --addfeed --keyword "Physics"
  ```

- To add a feed to the Custom Feeds catalog:

  ```
  podcast --addfeed --query query_string --feedname name --description
       description_string
  ```

  For example, the following command creates the custom feed Smart Feed, which contains all podcasts produced by users Bill and Tom:

  ```
  podcast --addfeed --query "episode.author_shortname = 'bill' or 'tom'"
       --feed_name "Smart Feed" --description "All podcasts by Bill and
       Tom"
  ```

  For more examples of queries, check the value query keys (<key>query</key>) in the property list returned by the `podcast --listfeeds` command. Also, check the Command Examples section of the `podcast` man page.

  Important: The query string uses SQLite syntax. You can use episode and attachment objects to build queries.

To remove a feed:

1 Get the UUID of the feed.

```
podcast --listfeeds
```

2 Remove the feed.

```
podcast --removefeed --feed_uuid UUID
```

Setting Feed Image

You can use the `podcast` command to set the logo for the feed. The image must not be bigger than 500 KB.

To set the feed image:

1 Get the UUID of the feed.

```
podcast --listfeeds
```

2 Set the feed image.

```
podcast --setfeedimage --feed_uuid UUID --path PATH_TO_IMAGE
```

Tagging Feeds

You can use the `podcast` command to tag feeds as containing explicit material.

To tag a feed as explicit:

1 Get the UUID of the feed.

```
podcast --listfeeds
```

2 Set the feed as explicit.

```
podcast --makefeedexplicit --feed_uuid UUID
```

To tag a feed as nonexplicit:

```
podcast --makefeednonexplicit --feed_uuid UUID
```

Setting Feed Properties

You can set the name, description, author shortname, and copyright properties for a feed.

To set the name, description, author shortname, and copyright properties for a feed:

1 Get the UUID of the feed.

```
podcast --listfeeds
```

2 Set the title property for the feed.

```
podcast --setfeedproperty --feed_uuid UUID --property_name name --value
     value
```

3 Set the subtitle property for the feed.

```
podcast --setfeedproperty --feed_uuid UUID --property_name description
     --value value
```

4 Set the author shortname property for the catalog.

```
podcast --setfeedproperty --feed_uuid UUID --property_name author_
     shortname --value value
```

5 Set the copyright property for the catalog.

```
podcast --setfeedproperty --feed_uuid UUID --property_name copyright
     --value value
```

Managing Catalogs

The Podcast Library catalog provide commands for searching catalogs when you open the catalog in Safari. These include commands for sorting catalog entries (also referred to as articles), displaying articles based on when podcasts were published, subscribing feeds in Mail and iTunes, and mailing links to catalog pages.

Searching Articles

You can use the Search Articles field to filter the contents of a page. For example, to list podcast episodes that contain the word Physics in the Workflow Feeds > Montage page, enter Physics in the search field.

This search feature is very useful when you have a long list and need to narrow your search for podcast episodes.

Controlling Article Length

You can use the Article Length slider to determine how much data is displayed per article or entry. If you move the slider all the way to the left, one line is displayed per entry, which allows you display more articles per page than when the slider is all the way to the right.

Sorting Articles

You can sort articles by date, title, source, or whether a new podcast has been added.

To sort articles:

- Under Sort By, click an option.

  For example, to list articles updated this month only, under Recent Articles, click This Month.

Displaying Articles According to Publishing Date

You can list articles based on when podcast episodes were published.

To list articles based on when podcast episodes were published:

- Under Recent Articles, click an option.

  For example, to list articles updated this month only, under Recent Articles, click This Month.

Updating the Podcast Library Catalog

To update the Podcast Library catalog:

- Under Actions, click Update Now.

Mailing a Link to a Podcast Library Catalog Page

To mail a link to a Podcast Library catalog page:

- Under Actions, click Mail Link to This Page.

  This command opens Mail and creates a message with a link to the Podcast Library catalog page.

Subscribing Feeds in Mail

You can subscribe feeds in Mail. Subscriptions appear under RSS in Mail.

To subscribe feeds in Mail:

- Under Actions, click Subscribe in Mail.

  This command opens Mail and adds a feed subscription under RSS. Use the feed management commands in Mail to rename, delete, archive, and update feeds.

Subscribing Feeds in iTunes

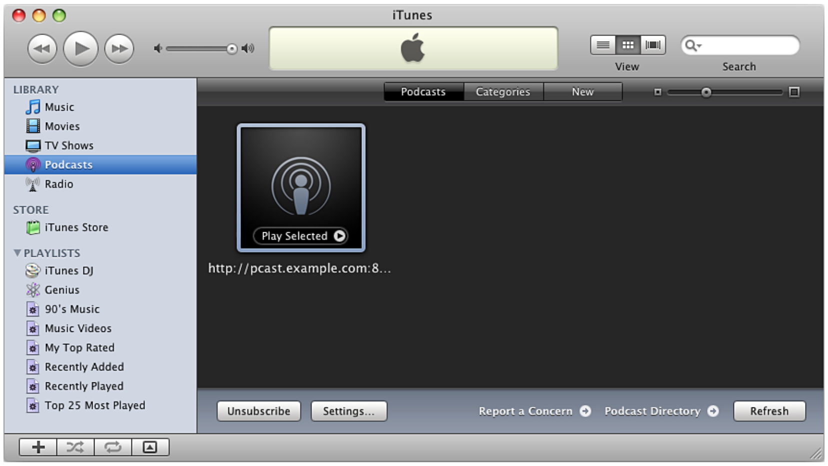
You can subscribe feeds in iTunes. Subscriptions appear under Podcasts in iTunes.

To subscribe feeds in iTunes:

1 Open the Podcast Library catalog page containing the feed's podcast episodes.

2 Under Actions, click Subscribe in iTunes.

This command opens iTunes and adds a feed subscription in the Podcasts library. Use the feed management commands in iTunes to rename, unsubscribe, delete, archive, and update feeds.



Podcast Library Feed Specification

The Podcast Library feed specification is based on the Atom Syndication Format, also referred to as Atom, which uses XML to define web feeds. However, the Podcast Library feed specification also uses iTune's RSS tags and extends or adds new elements.

A Podcast Library feed consists of general tags that apply to the entire feed (such as feed title, author, and rights) and `<entry>` tags. Every entry represents an article in the Podcast Library feed page. An entry contains podcast versions.

For more information about Atom and iTunes RSS extensions, see the relevant documentation available over the Internet.

Retrieving Feed XML

To retrieve a feed's XML, use the `curl` command.

To retrieve the XML of a catalog:

```
curl -k [-u username] feed://server_address:8171/podcastproducer/
     catalogs/uuid
```

To retrieve the XML of a feed:

```
curl -k [-u username] http://server_address:8171/podcastproducer/atom_
     feeds/uuid
```

Extensions of the link Element

The Podcast Library feed specification provides extensions to `link` element as indicated by the bold text in the following examples:

```
<link type="application/atom+xml" pcp:feedtype="root_catalog"
     href="http://pcast.example.com:8171/podcastproducer/admin/catalogs"
     title="Podcast Library" rel="root"/>


<link type="application/atom+xml" pcp:feedtype="user" href="http://pcast.
     example.com:8171/podcastproducer/admin/atom_feeds/338C4291-53E5-
     4A8E-9FF8-E5144F3623CE" title="Tom" rel="self"/>


<link type="video/x-m4v" rel="enclosure" length="834578" title="iPod /
     iPhone"
          href="http://machine.example.com:8171/podcastproducer/
     attachments/BDE1371D-8952-4A05-8C68-6A62F8373F65.m4v"
          pcp:format="com.apple.lc"
          pcp:duration="00:00:04"
          pcp:hasAudio="yes" pcp:audioCodec="AAC"
          pcp:audioDataRate="120.25"
          pcp:audioChannels="2"
          pcp:audioSampleRate="48.00"
          pcp:hasVideo="yes" pcp:videoCodec="H.264"
          pcp:videoDataRate="1366.93"
          pcp:videoFrameRate="24"
          pcp:isVideoInterlaced="no"
          pcp:width="640" pcp:height="480"
          pcp:hasChapters="no"
          pcp:isDefault="yes"
         pcp:hasClosedCaptions="no"
         pcp:isStreaming="no"
          pcp:keywords=""/>
```

```
<link type="image/png" rel="related" length="155849" …
          pcp:role="image" … />
```

pcp:feedtype

The `pcp:feedtype` extension of `link` specifies the feed document type as root catalog, catalog of feeds, or a feed of podcast episodes.

The `pcp:feedtype` extension supports the following values:

- `root_catalog`
- `user_feeds_catalog`
- `workflow_feeds_catalog`
- `keyword_feeds_catalog`
- `history_feeds_catalog`
- `custom_feeds_catalog`
- `user`
- `workflow`
- `keyword`
- `timebased`
- `custom`

The following link opens the main page of the Podcast Library catalog:

```
<link type="application/atom+xml" pcp:feedtype="root_catalog"
     href="http://pcast.example.com:8171/podcastproducer/admin/catalogs"
     title="Podcast Library" rel="root"/>
```

The following link opens the feed page for user Tom:

```
<link type="application/atom+xml" pcp:feedtype="user" href="http://pcast.
     example.com:8171/podcastproducer/admin/atom_feeds/338C4291-53E5-
     4A8E-9FF8-E5144F3623CE" title="Tom" rel="self"/>
```

pcp:format

The `pcp:format` extension of `link` specifies the format of attachment the link points to. For example:

```
pcp:format="com.apple.3g"
```

The Podcast Library feed specification uses reverse DNS name to refer to the podcast formats. By default, the Podcast Library feed specification supports the following formats:

- Low Complexity (`com.apple.lc`)
- Standard Definition (`com.apple.sd`)
- High Definition (`com.apple.hd`)
- 3rd Generation (`com.apple.3g`)
- Audio (`com.apple.audio`)
- Other formats (`com.apple.custom`)

pcp:width and pcp:height

The `pcp:width` and `pcp:height` extensions of `link` specify the height and width of images and video in pixels. For example:

```
pcp:width="640" pcp:height="480"
```

pcp:videoDataRate and pcp:audioDataRate

The `pcp:videoDataRate` and `pcp:audioDataRate` extensions of `link` specify the audio and video data rates in kbits/s. For example:

```
pcp:audioDataRate="120.25" pcp:videoDataRate="1366.93"
```

pcp:videoFrameRate

The `pcp:videoFrameRate` extension of `link` specifies the video frame rate in frames per second. For example:

```
pcp:videoFrameRate="24"
```

pcp:duration

The `pcp:duration` extension of `link` specifies the podcast duration (HH:MM:SS). For example:

```
pcp:duration="00:00:04"
```

pcp:hasAudio, pcp:hasVideo, and pcp:hasChapters

The `pcp:hasAudio`, `pcp:hasVideo`, and `pcp:hasChapters` extensions of `link` specify whether the enclosure or linked podcast has audio, video, and chapters. For example:

```
pcp:hasAudio="yes" pcp:hasVideo="yes" pcp:hasVideo="no"
```

pcp:isVideoInterlaced

The `pcp:isVideoInterlaced` extension of `link` specifies whether the video is interlaced. For example:

```
pcp:isVideoInterlaced="no"
```

pcp:audioCodec and pcp:videoCodec

The `pcp:audioCodec` and `pcp:videoCodec` extensions of `link` specify the audio and video encoders used to encode the linked podcast. For example:

```
pcp:audioCodec="AAC" pcp:videoCodec="H.264"
```

The supported audio codec values are AAC and AAC (protected). The supported video codec values are: "H.264, MPEG-4 video, and AVC0 Media.

pcp:audioChannels

The `pcp:audioChannels` extension of `link` specifies the number of audio channels in the linked podcast. For example:

```
pcp:audioChannels="2"
```

pcp:audioSampleRate

The `pcp:audioSampleRate` extension of `link` specifies the audio sample rate in KHz. For example:

```
pcp:audioSampleRate="48.00"
```

pcp:isDefault

The `pcp:isDefault` extension of `link` tells the application reading the feed to download the linked file. This element should be used one time per entry (an entry contains one or more links). If more than one element is used in an entry, the application uses the first instance. For example:

```
pcp:isDefault="yes"
```

pcp:role

The `pcp:role` extension of `link` defines a link as a poster image or preview video to be associated with the feed. For example:

```
<link type="image/png" rel="related" length="155849" …
          pcp:role="image" … />

<link type="image/png" rel="related" length="155849" …
          pcp:role="preview" … />
```

pcp:keywords

The `pcp:keywords` extension of `link` specifies keywords associated with the linked podcast. For example:

```
pcp:keywords="Physics, science"
```

Other Extensions

The Podcast Library feed specification defines additional elements and extensions.

pcp:entries

The `pcp:entries` element informs applications whether the feed has multiple enclosures and what the formats of these enclosures are.

The following example tells the application, for example iTunes or Safari, to expect multiple enclosures and at least six podcast versions:

```
<pcp:entries hasMultipleEnclosures="yes">
    <pcp:enclosedHints>
       <pcp:format>com.apple.lc</pcp:format>
       <pcp:format>com.apple.sd</pcp:format>
       <pcp:format>com.apple.hd</pcp:format>
       <pcp:format>com.apple.3g</pcp:format>
       <pcp:format>com.apple.audio</pcp:format>
       <pcp:format>com.apple.custom</pcp:format>
    <pcp:enclosedHints>
</pcp:entries>
```

Third-Party Attachments

The following is an example of a link to a podcast encoded using Compressor to work with third-party software:

```
<link type="video/x-ms-wmv" rel="enclosure" length="1834578"
      title="Windows Media Video"
      href="http://machine.example.com:8171/podcastproducer/
     attachments/42BD6551-8A7F-419A-984A-3CA5508E3D18.wmv"
/>

<link type="video/flv" rel="enclosure" length="1834578"
      title="Flash Video"
      href="http://machine.example.com:8171/podcastproducer/
     attachments/39FB69BC-DC2D-44A5-8A07-54C8C6FC02D0.flv"
/>
```

Customizing Workflows

9

This chapter describes workflows and shows you how to customize or create your own.

The workflows that ship with Mac OS X Server v10.6 are examples of the type of back-end processing you can perform on input files.

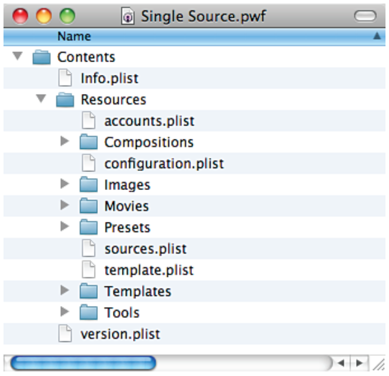
Important: Podcast Composer (in /Applications/Server/) is the primary tool for creating, editing, and modifying workflows. You can also manually create and customize workflows, but that requires experience with the command line and scripting technologies, and can be involved and time-consuming.

For more information about Podcast Composer, see *Podcast Composer User Guide*. For more information about customizing workflows, see *Podcast Producer Workflow Tutorial*.

The Structure of a Workflow Bundle

A workflow is a self-contained bundle that stores all files needed by the workflow.

The following figure shows the contents of a workflow bundle:



```
Single Source.pwf
Name
▼ Contents
    Info.plist
  ▼ Resources
      accounts.plist
    ▶ Compositions
      configuration.plist
    ▶ Images
    ▶ Movies
    ▶ Presets
      sources.plist
      template.plist
    ▶ Templates
    ▶ Tools
    version.plist
```

The following table describes the contents of a workflow bundle:

Element	Description
Info.plist	Contains information about the workflow bundle, including the name of the workflow as it appears in Server Admin and in the workflow's description. (Name and description strings are localizable.)
Resources	(Optional) Contains local resources used by the workflow. Global workflow resources are stored in /System/Library/PodcastProducer/Resources.
version.plist	Contains workflow version information.

The following table describes the default contents of the Resources folder:

Element	Description
accounts.plist	Stores credential information used by Podcast Composer.
Compositions	(Optional) Stores the Quartz compositions referenced by the workflow.
configuration.plist	Stores the configurations of the Import, Edit, Export, Notify, and Publish stages of Podcast Composer.
Images	(Optional) Contains images used by the workflow.
Movies	(Optional) Contains movies used by the workflow.
Presets	(Optional) Contains property lists that specify the encoding presets used by the workflow.
sources.plist	(Optional) Specifies the type of content processed by the workflow.
template.plist	Specifies the tasks to execute and the order of execution.
Templates	(Optional) Contains templates used by the workflow. By default, this folder contains a mail template and a template for posting to blogs. If your custom workflows use special templates, store them in this folder.
Tools	(Optional) Contains command-line tools used by the workflow.

In addition to the default contents of the Resources folder, Podcast Composer uses additional property lists: accounts.plist, crypto.plist, and configuration.plist.

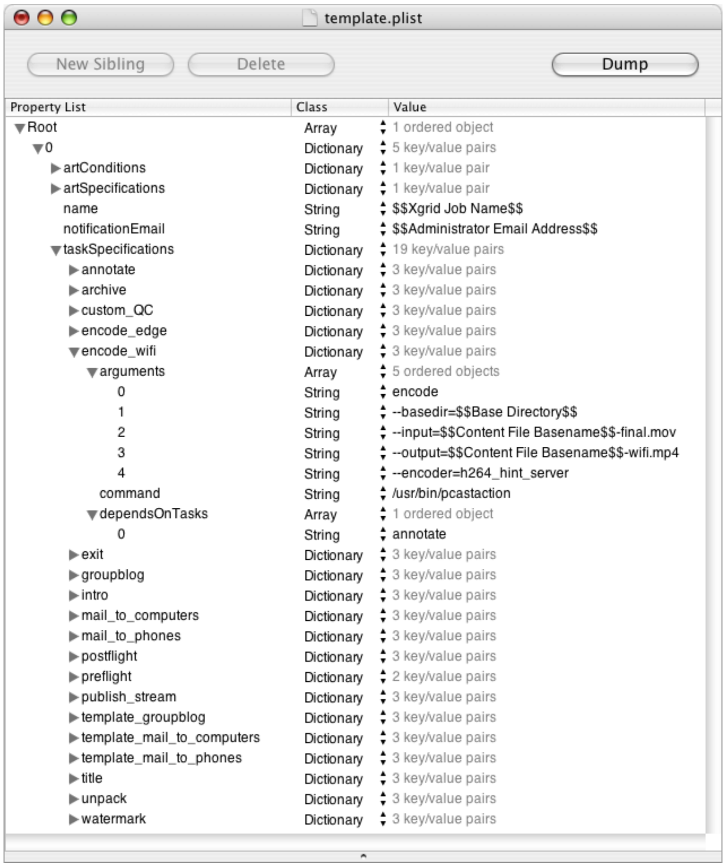
The Structure of a Workflow

A workflow template is a property list (plist) file containing tasks that Xgrid agents must execute. A workflow consists of the following first-level elements:

Element	Description
artConditions artSpecifications	These agent ranking tools (ART) allow you to score Xgrid jobs. You can tell the Xgrid controller to run certain jobs on certain Xgrid nodes and to prefer certain Xgrid nodes. For more information about these elements and how to configure them, see the *Xgrid Administration and High Performance Computing* or the `xgrid` man page.
name	This entry specifies the name of the Xgrid job. When the Podcast Producer server receives a job submission, it replaces the value of this key ($$Xgrid Job Name$$) in the corresponding workflow with a name it assigns to the job before sending it to the Xgrid controller.
notificationEmail	This key specifies the mail address the Podcast Server uses to notify the administrator about the status of workflow jobs submitted to the Xgrid controller. Podcast Producer replaces the value of this key ($$Administrator Email Address$$) with the administrator's mail address before sending the job to the Xgrid controller.
taskSpecifications	This is the most important entry. It specifies the tasks to be executed by Xgrid agents.

Workflow Task Specifications

The task-Specifications entry consists of task specifications, as shown in the following example. In this example, the taskSpecifications section lists 20 tasks. The name of the first task is annotate, the name of the second task is archive, and so on.



Property List	Class	Value
▼Root	Array	1 ordered object
▼0	Dictionary	5 key/value pairs
▶artConditions	Dictionary	1 key/value pair
▶artSpecifications	Dictionary	1 key/value pair
name	String	$$Xgrid Job Name$$
notificationEmail	String	$$Administrator Email Address$$
▼taskSpecifications	Dictionary	19 key/value pairs
▶annotate	Dictionary	3 key/value pairs
▶archive	Dictionary	3 key/value pairs
▶custom_QC	Dictionary	3 key/value pairs
▶encode_edge	Dictionary	3 key/value pairs
▼encode_wifi	Dictionary	3 key/value pairs
▼arguments	Array	5 ordered objects
0	String	encode
1	String	--basedir=$$Base Directory$$
2	String	--input=$$Content File Basename$$-final.mov
3	String	--output=$$Content File Basename$$-wifi.mp4
4	String	--encoder=h264_hint_server
command	String	/usr/bin/pcastaction
▼dependsOnTasks	Array	1 ordered object
0	String	annotate
▶exit	Dictionary	3 key/value pairs
▶groupblog	Dictionary	3 key/value pairs
▶intro	Dictionary	3 key/value pairs
▶mail_to_computers	Dictionary	3 key/value pairs
▶mail_to_phones	Dictionary	3 key/value pairs
▶postflight	Dictionary	3 key/value pairs
▶preflight	Dictionary	2 key/value pairs
▶publish_stream	Dictionary	3 key/value pairs
▶template_groupblog	Dictionary	3 key/value pairs
▶template_mail_to_computers	Dictionary	3 key/value pairs
▶template_mail_to_phones	Dictionary	3 key/value pairs
▶title	Dictionary	3 key/value pairs
▶unpack	Dictionary	3 key/value pairs
▶watermark	Dictionary	3 key/value pairs

Each task represents a UNIX shell script command and consists of the following elements:

Element	Description
arguments	Lists the arguments required by the command.
command	Specifies the command to run (including the path).
dependsOnTasks	(Optional) Lists the tasks that must be completed before this command runs.

For example, the encode_wifi task runs the following command after the preflight task is completed:

```
/usr/libexec/podcastproducer/tasks/pcastaction encode
--basedir=$$Base Directory$$
--input=$$Content File Basename$$-final.mov
--output=$$Content File Basename$$-wifi.mp4
--encoder=h264_hint_server
```

When Podcast Producer receives a job submission, it replaces the $$ property keys in the workflow template with the corresponding values before sending the job to the Xgrid controller.

Property Keys

Workflow templates use keys (strings enclosed by "$$") to represent default and custom workflow properties.

Podcast Producer defines a set of default properties used by workflows that ship with the product. In addition, Podcast Producer allows you to define custom properties for use in custom workflows. You can modify the values of properties and create properties in Server Admin.

When Podcast Producer receives a job submission from Podcast Capture, the Podcast Producer server replaces the property keys in the specified workflow with relevant values defined in Server Admin before submitting the Xgrid job for processing.

For example, the Podcast Producer server replaces the $$Group Short Name$$ key with the value of the Group Short Name default property you defined in Server Admin.

Important: Podcast Producer also uses protected property keys (strings enclosed by "##") in some workflows. For example, Podcast Producer replaces ##Groups Administrator Username:Groups Administrator Password## with a one-time password for authentication challenges. Protected property keys are intended for use by Podcast Producer and should not be modified.

Default Property Keys

To represent a default property in workflow templates, Podcast Producer encloses the name of the property as it appears in Server Admin by "$$" characters. For example, the Archive Root property appears as $$Archive Root$$ in workflows.

Podcast Producer does not store passwords in the workflow template because the workflow jobs are XML files and are not encrypted.

Instead, the Podcast Producer server encrypts and stores the passwords you enter in Server Admin in a special database for maximum protection, as described in "The Podcast Producer Security Model" on page 22.

Server-Generated Property Keys

The Podcast Producer server replaces the following property keys with the relevant values.

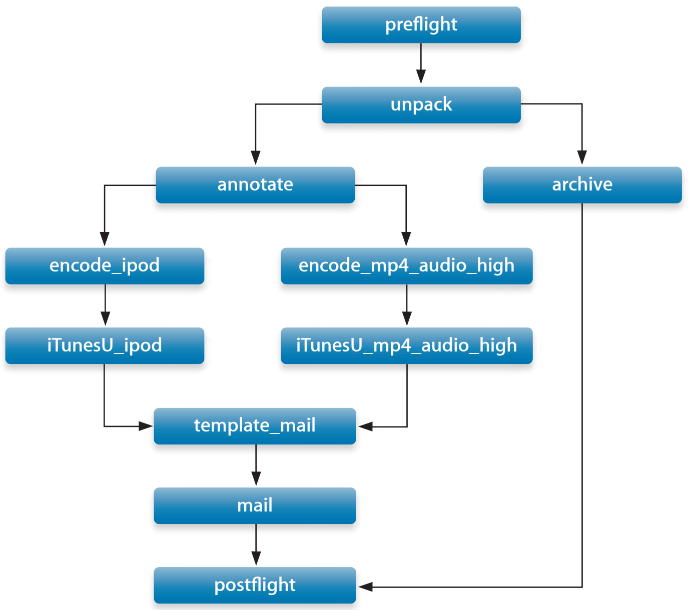
Property Key	Matching Property Name
$$Administrator Email Address$$	The mail address of the Podcast Producer administrator.
$$Administrator Full Name$$	The full name of the Podcast Producer administrator.
$$Base Directory$$	The base directory that Podcast Producer creates for storing submitted recordings (for example, <Shared_File_System>/Recordings/<UUID>).
$$Content File Basename$$	The name of the submitted recording.
$$Content File Extension$$	The name of the file extension of the submitted recording.
$$Content File Name$$	The combination of Basename and Extension. For example, if Content File Basename is MyMovie and Content File Extension is .mov, Content File Name is MyMovie.mov.
$$Date_YYYY-MM-DD$$	The date of the workflow submission.
$$Global Resource Path$$	The same as $$Workflow Resource Path$$ except that it refers to <*Shared File System*>/Caches/Resources.
$$Library Bundle Path$$	The path to the root level of the Podcast Library on the shared file system.
$$Podcast Producer URL$$	The URL of the Podcast Producer server.
$$Podcast Producer Wiki Content URL$$	The URL of the wiki.
$$Podcast Producer Wiki Content Document Root$$	The path to the wiki's root level folder.
$$Recording UUID$$	The UUID of the recording.
$$Server UUID$$	The UUID of the Podcast Producer server.
$$Shared Filesystem$$	The top level of the shared file system.
$$User Email Address$$	The mail address of the user who submitted the job.
$$User Full Name$$	The full name of the user who submitted the job.
$$User Home Directory$$	The home directory path of the user submitting the job.

Property Key	Matching Property Name
$$User ID$$	The ID of the user who submitted the job.
$$User Short Name$$	The short name of the user who submitted the job.
$$Workflow Bundle Path$$	The path to the top level of the workflow bundle on the shared file system (for example, <*Shared File System*>/Caches/Workflows/Blog.pwf).
$$Workflow Resource Path$$	The path to the workflow's Resources folder.
$$Xgrid Job Name$$	The name that Podcast Producer assigns to the job it submits to the Xgrid controller.
$$Active Podcast Producer Cluster Members$$	The Podcast Producer Xgrid cluster members.

Task Dependencies

A workflow task specification can specify tasks as dependencies. This helps the Xgrid controller determine the order of task execution.

The following illustration shows an example of task dependencies.



In this example, The unpack task won't run until the preflight task completes successfully. Also, the template_mail task won't run until iTunesU_ipod and iTunesU_mp4_audio_high tasks complete successfully.

In addition to specifying the order of task execution, task dependencies help the Xgrid controller determine which tasks can run in parallel. In the above example, the tasks at the same level (for example, the encode_ipod and encode_mp4_audio_high tasks) can be run in parallel if two Xgrid agents are available.

Workflow Commands

Podcast Producer provides a rich set of workflow commands or tasks to use in custom workflows. However, you can always write commands or use commands from third parties.

For a listing of the `pcastaction` commands, see "The pcastaction Tool" on page 123.

Podcast Producer Default Workflows

Podcast Producer ships with three sample workflows. These workflows were created using Podcast Composer.

To modify these workflows, you can use Podcast Composer or you can modify them manually. However, if you modify them manually, you can't edit them again using Podcast Composer.

To explore and modify these workflows, open them using Podcast Composer.

To open a default workflow using Podcast Composer:

1 Open Podcast Composer.

2 Choose File > Open Remote.

3 In the dialog sheet that appears:

   a In the Server field, enter the address of your Podcast Producer server.

   b In the Name and Password fields, enter the user name and password assigned to you by the Podcast Producer administrator.

   c Click Connect.

4 In the Open Remote Workflow window, complete the following:

   a Select one of the default workflows.

   b Click Open.

   c (Optional) In the Save As field, enter a different name.

   d Select a folder in which to store the workflow.

   e Click Save.

5 Explore and modify the workflow as needed.

6 Save the workflow (File > Save).

7 To Deploy the workflow to the server, choose File > Deploy to Server.

8 In the dialog sheet that appears, complete the following:

   a In the Server field, enter the address of your Podcast Producer server.

   b In the Name and Password fields, enter the user name and password assigned to you by the Podcast Producer administrator.

   c Click Deploy.

9 If prompted, click Overwrite to replace the existing workflow or Duplicate to create a new workflow.

   Every workflow has a unique identifier (UUID). Even if you change the name of the workflow in Podcast Composer, the workflow retains its UUID, unless you instruct Podcast Composer to deploy a new workflow with a new UUID.

   Note: Saving a copy of this workflow using the Saving As command generates a UUID.

10 When prompted, click OK.

Single Source

This workflow:

- Takes as input one QuickTime movie (video recording, audio-only recording, or screen recording)
- Adds introduction, title, and exit movies to the recording
- Adds a watermark to the recording
- Generates three versions of the podcast (iPod/iPhone, Apple TV (SD), and Audio)
- Publishes the podcast versions to the Podcast Library

Dual Source

This workflow:

- Takes as input two QuickTime movies (two video recordings or a video recording and a screen recording)
- Combines the two movies using the Overlay Quartz composition
- Adds introduction, title, and exit movies to the combined movie
- Adds a watermark to the combined movie
- Generates three versions of the podcast (iPod/iPhone, Apple TV (SD), and Audio)
- Publishes the podcast versions to the Podcast Library

Montage

This workflow:

- Takes as input documents (video recording, audio-only recording, or screen recording)
- Uses Quick Look to convert all pages to images
- Combines all images into a master QuickTime movie
- Adds introduction, title, and exit movies to the master movie
- Adds a watermark to the master movie
- Generates 3 versions of the podcast (iPod/iPhone and Apple TV (SD))
- Publishes the podcast versions to the Podcast Library

Creating and Customizing Workflows

There are several ways to create and customize workflows:

- Use Podcast Composer to create or modify a workflow.

  To modify a workflow, it must have been created using Podcast Composer and must not have been manually modified afterwards.

  For more information about Podcast Composer, see *Podcast Composer User Guide*.
- Manually modify workflows by manually adding, removing, or modifying workflow tasks.

Important: If you manually modify the tasks of a workflow created using Podcast Composer, you can't open it or edit it using Podcast Composer.

Tools for Creating and Editing Workflows

The primary tool for editing workflows is Podcast Composer. However, to manually modify a workflow, you can use any text editor or Property List Editor.

Manually Customizing Workflow Tasks

To add, delete, and modify workflow tasks manually, you start by duplicating an existing workflow and then make the necessary modifications.

For more information about manually customizing workflows, see *Podcast Producer Workflow Tutorial*.

Deploying Scalable Podcast Producer Solutions

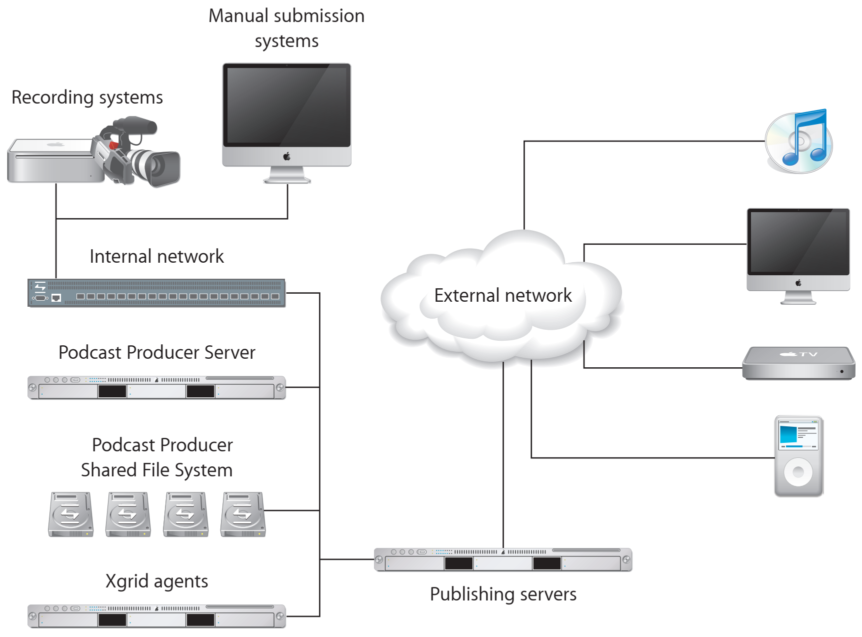
10

This chapter describes how to plan the deployment of Scalable Podcast Producer solutions.

Podcast Producer is designed for scalability. However, several factors determine how easy it is to scale your system and whether it is feasible. This chapter discusses scalability aspects and provides planning tips.

Resource Planning

Depending on your application, setting up Podcast Producer can require a serious investment in computing, storage, and network resources, as shown in the following illustration.



Manual Submission Systems

You use manual submission systems to upload QuickTime movies using Podcast Capture or the `podcast` command-line tool.

These systems do not need to be dedicated systems because Podcast Capture is available in Mac OS X v10.5 or later. Users with systems running Mac OS X v10.5 or later can upload video content using their systems.

Any system capable of running Mac OS X v10.5 or later with enough hard disk space can be used for a manual submission system.

Video Recording Systems

Video recording systems are dedicated systems running Mac OS X v10.5 or later with a video camera connected to them. A typical video recording system is a headless Mac Mini with 40 to 60 GB of free hard disk space. These systems are remotely controlled by other systems using Podcast Capture or `podcast`.

For example, a professor could use Podcast Capture to start and stop recording on the video-recording Mac from the podium computer, which could be a MacBook. You can even write a small web application to non-Mac podium systems to control the video-recording Macs.

The number of video-recording systems depends on your needs. For example, a school might have a requirement that every classroom be equipped with a video-recording system.

Although deciding the number of systems might be a matter of policy, keep in mind the cost of acquiring and maintaining these systems. In addition, consider the impact on your network when all these systems start uploading recorded content.

Recording Quality

An important factor to consider when planning a Podcast Producer deployment is the recording quality.

The recording quality you choose has an impact on the following:

- Storage requirements for the recording system
- Storage requirements for Podcast Producer's shared file system
- Network traffic
- Processing power

Although you can use Podcast Capture to specify recording quality, you can also use the `podcast --setconfig` command. For more information about using `podcast` to specify recording quality, see "The podcast Tool" on page 115.

Note: The recording quality presets are defined defined in /usr/lib/podcastproducer/agent/agent_config.rb.

Recording at the Best Quality

Recording audio, video, or screen activity at the Best quality results in large QuickTime files, which require more resources to store, upload, and process.

Recording video at the Best quality (H.264 High Quality video, Apple Lossless audio) generates 1 GB/h.

For example, a 2-hour recording at the Best quality requires more than 2 GB of free hard disk space on the recording system. In addition, while the first recorded movie is being uploaded, your recording system must have enough disk space to store additional recordings. Otherwise, you can't use the recording system until the first movie has uploaded successfully.

To overcome this limitation, you can customize your recording systems so that recordings are directly stored on an Xsan system, which provides greater storage capacity that is scalable.

Also, if the uploaded movies must be archived on the Podcast Producer's shared file system for an extended period, your storage needs will increase significantly as the number of submissions increases.

Another factor to consider is network traffic. Consider the impact on the network if you have several systems uploading large files at the same time.

Recording at Better and Good Quality

As shown in the previous section, recording at the Best quality can require a significant expenditure in computing and networking resources. This is why many organizations prefer recording at a lower quality.

By default, Podcast Capture and `podcast` record at Better quality (H.264 SD video, AAC High Quality audio), which generates 841 MB/h.

The advantage of recording at this quality is that it significantly improves the efficiency of your system without compromising quality. The Better quality is almost identical to the Best quality, but the difference in size is significant, especially if you plan to use multiple recording systems on a daily basis.

If quality is not an issue, you can record at the Good quality, which results in even smaller files.

To record 320 x 240 video, record at the Good quality (H.264 320 x 240 video, AAC High Quality audio), which generates 274 MB/h.

Network Bandwidth

When planning Podcast Producer deployment, consider using a private 1 GB/s network for submitting QuickTime movies to the Podcast Producer server. Doing so provides faster upload speed and shields the main network from traffic slowdowns when multiple systems are uploading content at the same time.

Also consider controlling the upload bandwidth at the switch level to prevent the network from being overwhelmed.

Publishing Systems

Although you can use one server to provide Podcast Producer services, including web, Mail, and other services for users to access podcasts, consider using dedicated servers for publishing podcasts for increased reliability and better performance.

Also consider using proxy servers (for example, a proxy server for every building's network) to improve the scalability of your system.

Take into account the size of the podcasts or movies that your workflows will generate. The size of the podcasts helps you determine the number of servers you need and how much network bandwidth to allocate.

For example, the average file size for 1 hour of video encoded for iPod is 250 MB and for Apple TV is 800 MB. If you plan to serve iPod and Apple TV podcasts at an average of 400 MB per user to 1,000 users per day, you'll need servers to handle about 390 GB/day of throughput. You also need to factor in the cost per GB.

Storage

Ideally, you should use RAID arrays and Xsan to provide a scalable high-availability, high-performance storage solution for your Podcast Producer system.

In very small deployments, you can use the Podcast Producer server's hard disks for storing podcasts. However, in medium to large deployments, your storage needs can grow very large, requiring terabytes of hard disk space.

Also, the more Xgrid servers you have in your Podcast Producer system, the more data communication bandwidth you'll need. This is why Xsan is an ideal solution because it provides the necessary bandwidth to process the data.

Xgrid Agents

When planning your Podcast Producer solution, you'll need to figure out how many Xgrid agents you should deploy to provide adequate computing power and to maximize efficiency.

Grid Size

Podcast Producer workflows can vary greatly in complexity and the maximum number of parallel tasks.

When using Podcast Composer to create and edit workflows, you can obtain some information on workflow complexity by using the Workflow Inspector in Podcast Composer.

However, the best way to determine the architecture for your Xgrid environment is by benchmarking with typical workflows and the content you use.

For example, a typical workflow might take a single-source video capture and add to it introduction and title videos. Then, the workflow encodes the resulting movie to iPod video and audio-only, and publishes the podcasts to the Podcast Library.

Typical content for the workflow might be one-hour classroom lectures. A typical capture schedule might involve a maximum of five lectures simultaneously submitted to Podcast Producer.

To test this configuration, deploy the workflow and submit five one-hour lectures simultaneously. Monitor the job execution with Xgrid Admin and measure how long it takes for all five submissions to complete.

To minimize job queuing on the grid, add additional Xgrid agents until all simultaneous jobs submitted begin executing without the need to wait in queue.

If the need arrives for additional throughput, additional Xgrid agents can be added to increase the number of CPU cores available for processing.

For more details, see "Workflow Benchmarking" on page 106.

Scheduling

In Mac OS X Server v10.6, Xgrid distributes jobs in a round-robin fashion across the grid. This improves performance, particularly on grids with available cores.

Xgrid selects the fasted CPUs first. For this reason, the simple best practice is to use homogeneous systems as grid nodes.

Memory

Memory is important on Xgrid nodes. You need enough memory to support workflow complexity.

On every Xgrid node, you should have at least 1 GB of memory (RAM) in addition to 512 MB of additional RAM per processor core.

An Xserve featuring the Quad-Core or 8-Core Intel Xeon Nehalem processor presents each processor core as two cores to Mac OS X Server.

For example, an 8-Core Xserve requires at least 1 GB + (16 x 512) or 9 GB of RAM. Based on the memory architecture of this server, round up to the next highest memory configuration, which is 12 GB.

Workflows

Workflows are important in planning the deployment of a Podcast Producer solution. They dictate what resources are needed and whether a deployment is feasible.

Workflows determine how submitted QuickTime movies are processed. The more processing the movies require, the more computing power and resources you'll need.

Although workflows allow you to perform batch processing for QuickTime movies, you'll need to plan your workflows based on the resources available to you.

For example, you might choose to create simplified workflows that produce only audio podcasts of conference presentations. Also, you can create workflows that do not archive the submitted QuickTime movies to save storage.

Workflow Benchmarking

Benchmarking your workflows is highly recommended. In your deployment, you want to anticipate peak processing requirements.

When designing and deploying new workflows, testing the workflow and establishing benchmarks helps you optimize your deployment.

The following criteria help you maximize the use of your Xgrid environment:

- Benchmark the workflows that you plan to use in your production environment.
- Use content that is typical.

  Benchmarks from processing a two-minute video cannot be extrapolated for a one-hour lecture.
- Submit content that represents your primary usage patterns.

  For example, if you expect 20 one-hour lectures submitted per day, use that schedule to benchmark your Xgrid configuration.
- Design your workflows to maximize the use of Xgrid agents and increase efficiency.

  For example, decrease dependencies in workflows so more workflow tasks can run in parallel.

- When benchmarking workflows, identify the bottleneck tasks, those that consume most of the CPU cycles available to the workflow.

  Optimizing these tasks will provide the largest benefit in lowering workflow execution time.
- You can simulate real-world content submission using command-line scripts.
- Submit content on your network to determine the impact on the network infrastructure for content upload.
- Monitor disk and CPU usage on your grid using Server Admin.

  Collect and analyze this data, and make adjustments to your computing grid, shared file system, and network configuration to optimize system performance.
- After you deploy your system, continue to monitor performance using Server Admin and other Mac OS X tools.

Deployment Scenarios

This section discusses three common deployment scenarios and describes their scalability.

The following table compares these scenarios.

	All-in-One	NFS	Xsan
Single Server Configuration	√		
Express Setup	√	√	
Uses simple DAS Setup	√	√	
Scalable Xgrid		√	√
High Availability			√
Storage Pools and Affinities			√
Growable File System			√
Fail Over Support			√
Video Workgroups			√

All-In-One Deployment

An all-in-one deployment is suitable for testing and workflow development or for small organizations with limited computing resources and limited podcasting needs.

In an all-in-one deployment, your Podcast Producer solution can consist of the following:

- One or more recording systems
- One Mac (ideally an Xserve) running Podcast Producer server and Xgrid

An Xserve with a 128-GB solid-state boot drive and three 1 TB drives set up with RAID 5/Hardware RAID makes a powerful all-in-one system. If you anticipate a large archive of content, or your storage needs grow, you can add high-performance Direct-attached Storage (DAS).

You can easily migrate an all-in-one configuration to an NFS configuration if the locally attached storage is used for the shared file system.

However, to host the shared file system on an external NFS server, you must migrate the data and reconfigure Podcast Producer server. In this case, migration to NFS for the shared file system requires:

- Reconfiguring Podcast Producer settings
- Migrating data from the old shared file system to the new one
- Possibly updating your workflows to take advantage of additional processing power
- Possibly reconfiguring your publishing services

When making a configuration change to your Podcast Producer system, other factors for consideration include:

- Updating your workflows to take advantage of additional processing power
- Reconfiguring your publishing services

NFS Deployment

The minimum configuration for a scalable NFS deployment is the following (assuming that DNS, Mail, Open Directory, and Web services are already available):

- 1 Xserve with Podcast Producer
- 1 Xserve running Xgrid (controller and agent)
- 1 high-performance DAS for NFS

The Podcast Producer Setup Assistant configures your NFS automount.

To scale your system, add your Xgrid nodes, bind them to the same directory that Podcast Producer is bound to, and they'll mount the NFS file system in the relevant path for Podcast Producer.

You might also consider binding systems to support custom publishing and delivery, such as the Leopard wiki server or Quicktime Streaming.

Although the NFS deployment offers significant degree of processing power scalability, the storage element in this configuration does not scale because the metadata is set up within the data storage pool. This configuration optimizes the maximum usage of storage space on one RAID array but does not provide the highest scaling flexibility.

To scale the storage element of this configuration, migrate the data from the old RAID array to a new system where the storage area network (SAN) metadata pool and the SAN data pool are separate. Then, add logical unit numbers (LUNs) to the storage pool to increase the storage size and get better performance. You'll also need to reconfigure your Podcast Producer shared file system settings.

NFS is a great solution for medium Podcast Producer deployments. If your Xgrid has eight or fewer nodes, consider NFS, particularly if the fail-over features enabled by Xsan aren't required.

Xsan Deployment

The perfect minimum setup that gives you maximum flexibility for scaling the computing and storage element for your Podcast Producer solution is the following (assuming that DNS, Mail, Open Directory, and Web services are already available):

- 1 Xserve with Podcast Producer
- 1 Xserve running Xgrid (controller and agent)
- 1 MDC and 2 RAID arrays for the Xsan system

In this configuration you set up the metadata separately from the data storage pool using RAID 1, and you use RAID 5 for the data storage pool with three drive modules as hot spares.

To scale your solution, add Xgrid and RAID systems as needed.

Xsan provides advanced features for mission-critical deployments (for example high availability capabilities supported by meta-data fail-over and Fibre Channel multipathing).

Affinities allow allocation of classes of storage for specific purposes. Seamless to the user, affinities help ensure that an application or task that requires speed or extra protection always stores its files in a suitably fast or protected pool.

Xsan enables you to integrate video workgroups and Compressor clusters into a Podcast Producer environment. With this capability, Final Cut workflows can access and incorporate capabilities such as the Podcast Library for content delivery, and Podcast Producer workflows can use Compressor clusters.

Podcast Producer workflows can call Compressor for media encoding. The same Compressor cluster might also be available to a Final Cut Studio workflow environment. If you deploy Compressor and Qmaster, keep it separate from the Xgrid cluster.

Case Study

To better understand the importance of planning and the resource demands of a Podcast Producer solution, this section discusses a sample deployment.

This case study takes the following into consideration:

- Network bandwidth
- Content uploading
- Typical recording-day schedule with workflow benchmarks

In addition, this case study provides charts to illustrate the results of the study.

Note: The charts were generated by a special tool developed by Apple engineers. The numbers in these charts are based on established benchmarks for a set of custom workflows. These numbers may not apply to your setup.

Recording System Configuration

The Podcast Producer system discussed in this case study has the following configuration for recording video:

- 4 recording systems (no manual submission systems) to record and upload content
- A default network bandwidth of 100 Mbit/s from the recording systems to the Podcast Producer server
- A recording quality of H.264 (1 GB/h)

Workflow Benchmarks

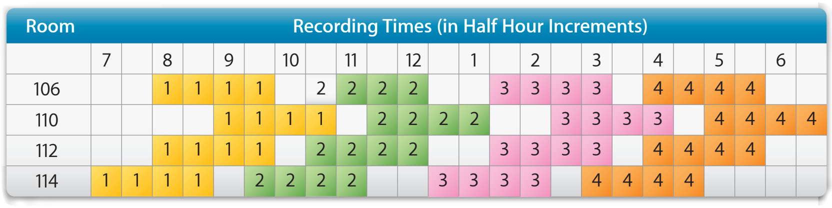
This sample deployment uses four workflows. The following table lists the benchmarks established for these workflows after testing them:

Workflow	Description	Ratio	CPUs	In (MB/h)	Out (MB/h)
1	This workflow takes in a 720x576 movie (less than 90 minutes) and produces three podcasts: • AAC Audio • iPod video (H264) • High-quality video (H264/native resolution)	100%	2	250	500
2	This workflow is the same as workflow 1 except that it takes in a movie more than 90 minutes.	150%	2	250	500
3	This workflow takes the same input and generates the same output as workflow 1, but also generates additional podcasts.	100%	4	250	1,050
4	This workflow is the same as workflow 3 except that it takes in a movie more than 90 minutes.	150%	4	250	1,050

The Ratio column in the table lists the ratio of recording time to processing time. For example, if the recording time is 1 hour, a ratio of 150% means that it takes 1.5 hours to produce and publish the podcast. The CPUs column lists the number of CPUs (not systems) required by the workflow for best performance. The In column lists the size per hour of the recorded movie and the Out column lists the per-hour size of the resulting podcasts.

Recording Schedule

A typical recording day (from 7 a.m. to 7:00 p.m.) for this case study is illustrated below.



Room	Recording Times (in Half Hour Increments)																							
	7		8		9		10		11		12		1		2		3		4		5		6	
106			1	1	1	1		2	2	2	2			3	3	3	3		4	4	4	4		
110					1	1	1	1		2	2	2	2			3	3	3	3		4	4	4	4
112			1	1	1	1		2	2	2	2			3	3	3	3		4	4	4	4		
114	1	1	1	1		2	2	2	2			3	3	3	3		4	4	4	4				

The numbers in the colored cells represent the workflow being used. In this example, workflow 1 is used to record the first sessions in all rooms, workflow 2 is used for recording the following sessions in all four rooms, and so on.

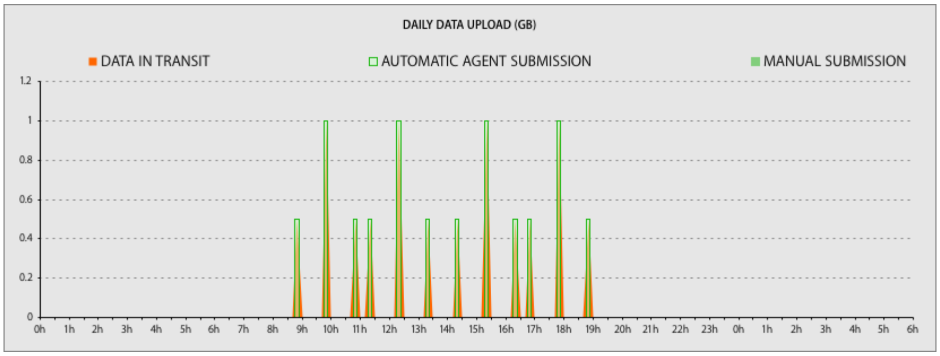
The colored cells help you see the recording pattern.

Performance

With one system for uploading the recorded movies to the shared file system and three Xgrid nodes, this section describes the performance of the system.

Daily Data Upload

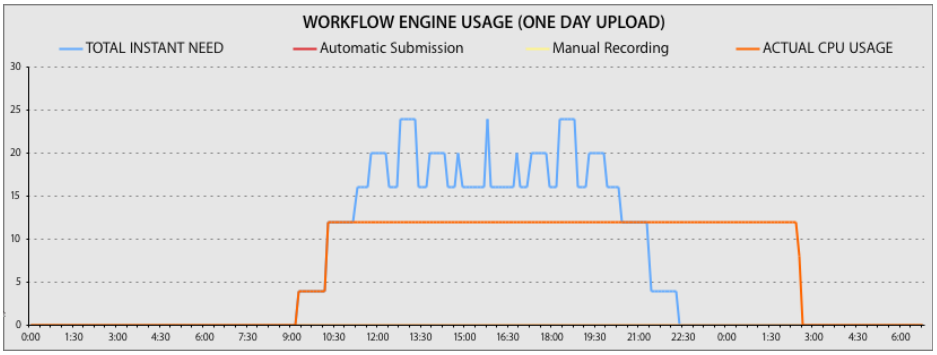
The following chart illustrates the daily upload times.



In this case, the size of recorded movies and the spacing of recording times allows for a smooth uploading pattern of 15 minutes per submission.

CPU Usage

The following chart illustrates the CPU usage pattern when using three Xgrid nodes.

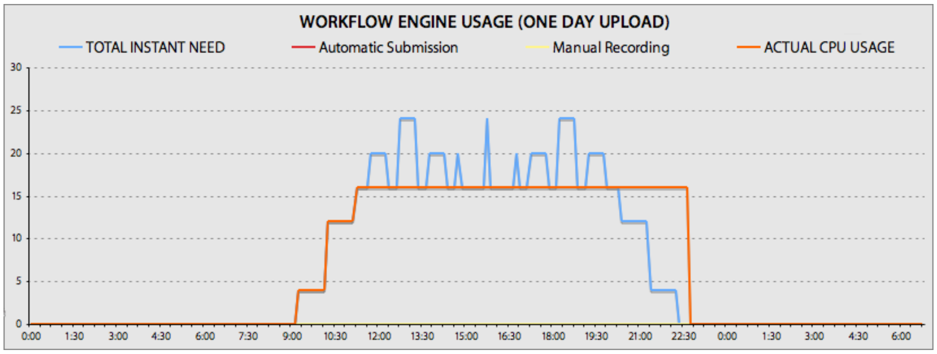


In the chart, the blue line shows the instant computing need if unlimited computing resources were available. The orange line shows CPU usage based on the number of Xgrid nodes available. In this case, there are three Xgrid nodes, each with four CPUs.

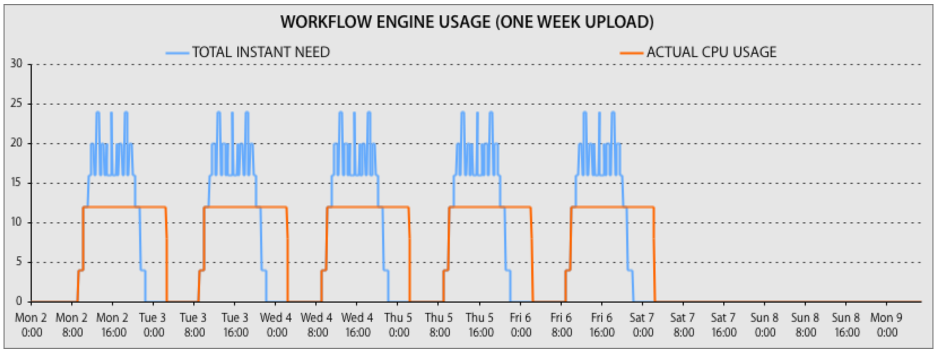
As shown in the chart above, at 3:00 in the morning, all workflows were completed and all content was published.

Note: From a mathematical point of view, the area under the blue line is equal to the area under the orange line.

If the number of Xgrid nodes is now four (16 CPUs), the performance improves as shown in the following chart.



However, using only two Xgrid nodes will do the job if fast delivery of content is not required, as shown in the following chart.



As indicated in the chart, all podcasts will be available six days after the recording started.

The choice of how many Xgrid nodes to use depends on delivery time requirements and cost.

Storage Usage

The following shows the required Xsan setup for an academic year (40 weeks) to accommodate the recording schedule used in this case study:

- Uploaded data per day: 8,500 MB
- Processed data per day: 35,700 MB per day

The estimated total storage need for 40 weeks is 8,840,000 MB, or about 8.6 TB.

Summary

Based on this case study, the following factors play a big role in planning and deploying a Podcast Producer solution:

- Number of recording and manual submission systems
- Type of workflows to be used
- Typical recording-day schedule for each recording and manual submission system
- Estimated time to complete a workflow job under best conditions
- Number of Xgrid nodes and CPUs that can be used in parallel
- Required delivery time and system optimization needed to meet requirements
- Required Xsan storage
- Required Xsan bandwidth per Xgrid agent to ensure that tasks execute at full speed
- Required network bandwidth

Podcast Producer Command-Line Tools

11

This chapter describes Podcast Producer command-line tools.

Podcast Producer command-line tools provide the flexibility you need to customize Podcast Producer. This chapter provide an overview of these commands.

For more information about Podcast Producer command-line tools, see the corresponding man page.

The podcast Tool

The `/usr/bin/podcast` tool allows you to interface with Podcast Producer server. For example, you can use this tool or command to get status information from the server, bind a local device to the server, and control the capture and submission of media for processing by Podcast Producer.

It allows you to perform any function that Podcast Capture offers, and more. For example, you can use `podcast` for listing and setting audio, video, and screen recording quality.

Tip: The `podcast` tool is a Ruby script. To see how it works, open it in a text editor. For example, to open `podcast` in TextEdit, enter `open -a TextEdit /usr/bin/podcast`.

The following are ways in which you can take advantage of the `podcast` tool:

- Wrapping the `podcast` tool.

  You can wrap the `podcast` tool to add a layer of functionality.

  For example, you can write a GUI application that authenticates users and provides only one option for starting and stopping screen recording.

- Creating Podcast Producer widgets.

  For example, you can create a widget for viewing Podcast Producer presets.

- Scheduling recording jobs.

  You can create `cron` jobs to automate the recording and submission of audio and video content to the Podcast Producer server.

For example, you can write `cron` jobs for submitting Podcast Producer jobs at 11:00 p.m. every day.

Syntax

```
podcast [-s server] [-u username] [-p password] [--auth {Password |
     Kerberos}] [--checksslcert] [--timeout seconds] command [command-
     options]
```

Command Options

The `podcast` command provides the following categories of commands:

- "Information Commands" on page 116
- "Agent Commands" on page 117
- "Camera Control Commands" on page 117
- "Submission Commands" on page 118
- "Workflow Administration Commands" on page 118
- "Access Control List Commands" on page 119
- "Feed Administration Commands" on page 119
- "Catalog Administration Commands" on page 120

The following sections provide a brief overview of these commands. For more information about `podcast` and its commands, see its man page or enter `podcast --help`.

Information Commands

The following describes `podcast` information commands.

Command	Description
`--listcameras`	Lists available cameras.
`--listworkflows`	Lists available workflows.
`--listfeeds`	Lists available feeds.
`--listcatalogs`	Lists available catalogs.
`--listservers`	Lists available servers (Bonjour discovery).
`--listinfo`	Lists information for a server (does not require authentication).

Agent Commands

The following describes `podcast` agent commands.

Command	Description
`--isbound`	Checks if the local camera agent is bound to a server (returns `1` if bound, `0` if unbound).
`--bind` *`camera_name`* `[--lockdown]`	Binds the local camera with the name *`camera_name`* to the Podcast Producer server. The lockdown flag restricts the camera to the Administrator group (80).
`--unbind` *`camera_name`*	Removes binding for camera *`camera_name`*.
`--presets`	Lists available capture presets for camera agent configuration.
`--devices`	Lists available audio/video devices for Podcast Producer agent configuration.
`--getconfig`	Lists agent configuration settings.
`--setconfig` *`key=value`*`[;`*`key=value`* `...]`	Writes an agent configuration setting (requires root privilege).

Camera Control Commands

The following describes `podcast` camera control commands.

Command	Description
`--status camera_name [--update_preview_image [--include_preview_image]]`	Gets status of a specific remote camera. Optionally, asks for the preview image to be updated.
`--start camera_name [--audio_only] [--delay seconds]`	Starts capture on a specific remote camera. Optionally, starts recording with audio-only mode or with delay.
`--pause` *`camera_name`*	Pauses capture on a specific remote camera.
`--resume` *`camera_name`*	Resumes capture on a specific remote camera.
`--stop` *`camera_name`* `{--workflow` *`workflow_name`* `| --workflow_uuid` *`UUID`* `| --submission_uuid` *`UUID`*`}` `[--user_metadata` *`metadata_file_path`*`]` `[--title` *`title`*`]` `[--description` *`description`*`]`	Stops capture on a specific remote camera and submits footage for processing with the workflow specified by *`UUID`* or as the source for the precreated submission specified by *`UUID`*.
`--cancel camera_name`	Cancels capture for a specific remote camera.

Submission Commands

The following describes `podcast` submission commands.

Command	Description
`--create_recording --workflow_uuid` *`UUID`*	Creates an empty recording container for the workflow specified by *`UUID`*.
`--submit --file` *`file_path`* `[--file` *`file_path`* `...] {--workflow_uuid` *`UUID`* `| --submission_uuid` *`UUID`*`} [--user_metadata` *`user_metadata_file_path`*`] [--recording_metadata` *`recording_metadata_file_path`*`] [--title` *`title`*`] [--description` *`description`*`]`	Submits file(s) specified by the `file_path` arguments for processing with the workflow specified by *`UUID`* or as the source for the precreated submission specified by *`UUID`*.
`--run_uploader`	Starts the uploader process. This command option is used by the --submit and --stop command options for uploading content.
`--list_uploads`	Lists pending uploads and progress status.
`--clear_completed_uploads`	Clears completed uploads from list.

Workflow Administration Commands

The following describes `podcast` workflow administration commands.

Command	Description
`--installworkflow --path` *`workflow_bundle_path`* `[--overwrite] [--master_password` *`password`*`]`	Installs the workflow at *`workflow_bundle_path`*.
`--enableworkflow --workflow_uuid` *`UUID`*	Enables the workflow specified by *`UUID`*.
`--disableworkflow --workflow_uuid` *`UUID`*	Disables the workflow specified by *`UUID`*.
`--downloadworkflow --workflow_uuid` *`UUID`* `--path` *`workflow_bundle_path`*	Downloads the workflow bundle for the workflow specified by *`UUID`* to the path specified by *`workflow_bundle_path`*.
`--infoworkflow --workflow_uuid` *`UUID`*	Gets information about the workflow specified by *`UUID`*.
`--deleteworkflow --workflow_uuid` *`UUID`*	Deletes the workflow specified by *`UUID`* from the server.

Access Control List Commands

The following describes `podcast` ACL commands.

Command	Description
`--addaccess --resource_type {Camera	Workflow	Feed} --resource_uuid` *`UUID`* `--record_type {group	user} --shortname` *`name`*	Grants access to a resource for a user or group. Requires access controls to be enabled via the `--enableacls` command.
`--removeaccess --resource_type {Camera	Workflow	Feed} --resource_uuid` *`UUID`* `--record_type {group	user} --shortname` *`name`*	Removes access for a user or group to a resource. Requires access controls to be enabled via the `--enableacls` command.
`--showacl --resource_type {Camera	Workflow	Feed} --resource_uuid` *`UUID`*	Lists all access control entries for a resource.
`--enableacl --resource_type {Camera	Workflow	Feed} --resource_uuid` *`UUID`*	Enables access controls for a resource.
`--enableacls --resource_type {Camera	Workflow	Feed}`	Enables access controls for a resource type.
`--disableacl --resource_type {Camera	Workflow	Feed} --resource_uuid` *`UUID`*	Disables access controls for a resource.
`--disableacls --resource_type {Camera	Workflow	Feed}`	Disables access controls for a resource type.
`--checkaccess --resource_type {Camera	Workflow	Feed} --resource_uuid` *`UUID`* `--shortname` *`username`*	Verifies that a user has access to a resource.

Feed Administration Commands

The following describes `podcast` feed administration commands.

Command	Description
`--addfeed {--shortname` *`name`* `| --keyword` *`keyword_string`* `| --query` *`query_string`* `--feed_name` *`name`* `--description` *`description_string`*`}`	Creates a feed on the server based on a specific username, a keyword search of podcast metadata, or a query string.
`--enablefeed --feed_uuid` *`UUID`*	Enables the feed specified by *`UUID`*.
`--disablefeed --feed_uuid` *`UUID`*	Disables the feed specified by *`UUID`*.

Command	Description
`--removefeed --feed_uuid UUID`	Removes the feed specified by *UUID* from the server.
`--setfeedimage --feed_uuid UUID --path PATH_TO_IMAGE`	Sets the logo for the feed specified by *UUID* to the image at `PATH_TO_IMAGE`. ***Note:*** Not all feed reader support this feature.
`--makefeedexplicit --feed_uuid UUID`	Tags the feed specified by *UUID* as explicit.
`--makefeednonexplicit --feed_uuid UUID`	Tags the feed specified by *UUID* as nonexplicit.
`--setfeedproperty --feed_uuid UUID --property_name PROPERTY --value VALUE`	Sets the property to the specified value for the feed specified by *UUID*. You can set the following properties: *name*, `description`, `author_shortname`, *copyright*.

Catalog Administration Commands

The following describes `podcast` catalog administration commands.

Command	Description
`--setcatalogimage --catalog_uuid UUID --path PATH_TO_IMAGE`	Sets the logo for the catalog specified by *UUID* to the image at *PATH_TO_IMAGE*. ***Note:*** Not all feed reader support this feature.
`--setcatalogproperty --catalog_uuid UUID --property_name PROPERTY --value VALUE`	Sets the property to the specified value for the catalog specified by *UUID*. You can set the following properties: *title*, `subtitle`, `author_shortname`, *copyright*.

The pcastconfig Tool

You can use the `/usr/bin/pcastconfig` tool to configure the Podcast Producer server at the command line, instead of using Server Admin.

Syntax

```
pcastconfig command [command-options]
```

Commands

The `pcastconfig` command provides the following categories of commands:

- "Information Commands" on page 116
- "Agent Commands" on page 117
- "Camera Control Commands" on page 117

- "Submission Commands" on page 118
- "Workflow Administration Commands" on page 118

The following sections provide an overview of these commands. For more information about `pcastconfig` and its commands, see its man page or enter `pcastconfig --help`.

Workflow Commands

The following describes the `pcastconfig` workflow commands.

Command	Description
`--enable_workflow` *`workflow_name`*	Enables the workflow named *`workflow_name`*.
`--disable_workflow` *`workflow_name`*	Disables (hides) the workflow named *`workflow_name`*.
`--validate_workflow` *`workflow_name`*	Validates the contents of the workflow named *`workflow_name`*. The workflow must be in the database.
`--validate_workflow_at_path` *`workflow_path`*	Validates the contents of the workflow at the path *`workflow_path`*. The workflow does not need to be in the database.
`--cache_workflow` *`workflow_name`*	Caches the workflow named *`workflow_name`* in the shared file system.
`--validate_all_workflows`	Validates the contents of all workflows in the database.
`--update_workflows_in_db`	Makes the database current with workflows in: • /System/Library/PodcastProducer/Workflows • /Library/PodcastProducer/Workflows

Camera Commands

The following describes `pcastconfig` camera commands.

Command	Description
`--enable_camera` *`camera_name`*	Enables the camera named *`camera_name`*.
`--disable_camera` *`camera_name`*	Disables (hides) the camera named *`camera_name`*.

Upload Node Commands

The following describes `pcastconfig` upload node commands.

Command	Description
`--create_upload_node` *`shared_filesystem_path`*	Sets up the Podcast Producer to be an upload-only node. Runs only apache and the HTTPS upload CGI.

Property Commands

The following describes `pcastconfig` property commands.

Command	Description
`--add_property` *`property_name`* `--value` *`value`* `[--protect]`	Creates a global system property in the Podcast Producer server database.
`--remove_property` *`property_name`*	Creates a global system property in the Podcast Producer server database.

Property Access Commands

The following describes `pcastconfig` property access commands.

Command	Description
`--add_access` *`access_group`* `--properties` *`property_list`*	Creates a one-time access key for a colon-separated list of properties in the Podcast Producer server database. Associates the access key with an *`access_group`*.
`--remove_access` *`access_group`*	Revokes access keys for property accesses associated with a specific *`access_group`*.

The pcastctl Tool

Use the `/usr/sbin/pcastctl` tool to start, stop, and restart the Podcast Producer server or agent. Also use this tool to display the status of running daemons.

For more information about `pcastctl`, see its man page.

The pcastaction Tool

The `/usr/bin/pcastaction` tool is used in workflows and provides a rich set of commands for processing and producing audio and video podcasts.

For example, the `pcastaction watermark` command imposes a watermark image on the input video and the `pcastaction encode` command outputs an encoded version of the input file.

For a list of the commands of `pcastaction`, see its man page or enter `pcastaction help`.

Syntax

`pcastaction` *`command`* `[`*`command-options`*`]`

Commands

The following describes `pcastaction` commands you can use in workflows.

For more information about these commands and their command options, enter `pcastaction help` *`command`*.

Command	Description
`annotate`	Adds annotations to the input movie.
`addchapter`	Adds a chapter to the input file at the specified time.
`addtracks`	Adds tracks to the input file with an option offset and layer.
`approval`	Submits content for approval.
`archive`	Archives the input movie at the specified location.
`chapterize`	Adds chapters to an input movie by detecting scene changes.
`compressor`	Submits an encoding job via Compressor.
`deletetracks`	Deletes tracks of the specified type from the input file.
`documents2movie`	Takes a file, a folder of documents or a zipped archive of documents and generates a chaptered movie with the content of these files.
`encode`	Encodes the input movie using the specified codec.
`extracttracks`	Extracts tracks of the specified type from the input file.
`flatten`	Flattens the input file and saves the result into the specified output file.

Command	Description
`getposterimage`	Takes a movie and generates a poster image (PNG) by grabbing a frame from a specific time frame.
`getpreviewmovie`	Generates a preview movie from the input movie.
`groupblog`	Posts the content to the specified group blog.
`iTunes`	Instructs the iTunes Store to check the specified RSS feed for new episodes.
`iTunesU`	Posts the input video at the specified iTunes U tab.
`jabber`	Sends a message via Jabber.
`join`	Joins both input files into an output file with an optional gap in-between.
`mail`	Sends a notification mail to the specified user using the mail template in the workflow's Resources/Templates folder.
`merge`	Merges two movies with a fade transition between them.
`notify_itunesu`	Notifies iTunes U that new content is available.
`pip`	Takes 2 movies (main and secondary) and creates a Picture-in-Picture reference movie using a Quartz Composer composition.
`postflight`	Runs the postflight script (System/Library/PodcastProducer/Resources/Tools/postflight_script) with the specified arguments.
`preflight`	Runs the preflight script (System/Library/PodcastProducer/Resources/Tools/preflight_script) with the specified arguments.
`publish`	Publishes the input file to a web or QTSS server.
`publish2finalcutserver`	Publishes files to a folder watched by Final Cut Server.
`publish2folder`	Publishes files to a folder.
`publish2library`	Publishes files to the Podcast Library.
`qceffect`	Applies a Quartz Composer effect (composition) to a movie.
`qtimport`	Prepares a QuickTime movie.

Command	Description
`qtinfo`	Retrieves QuickTime information from the input file.
`shell`	Runs the specified shell script with the specified arguments.
`split`	Splits the input file at the specified time into two output files.
`template`	Processes a web or mail template into a localized content block to be used in mail or web postings.
`title`	Adds the supplied title to the input video.
`trim`	Extracts content from the input file as specified by the start and end times and stores the extracted content in a new file.
`unpack`	Unpacks folder archives before running the main part of a workflow.
`upload`	Submits content for approval.
`userblog`	Uploads podcast to a userblog.
`watermark`	Superimposes the specified image as a watermark over the input video.
`wikiserver`	Posts the content to the specified posting URL destination.
`workflow`	Submits content file to another workflow.

Monitoring Podcast Producer 12

This chapter describes how to monitor and troubleshoot Podcast Producer issues.

Podcast Producer provides several ways for monitoring Podcast Producer activity.

Viewing Podcast Producer Logs

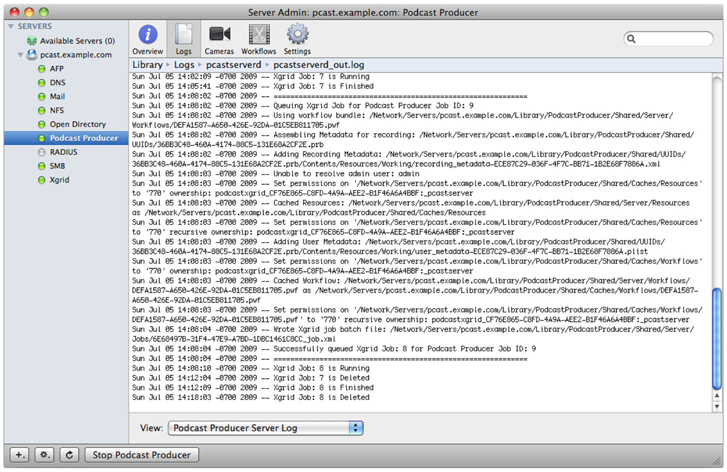
You can use the Logs pane of the Podcast Producer server to view Podcast Producer logs.

The logs help you monitor and troubleshoot Podcast Producer issues. You can even write scripts that look for certain log entries to alert you of possible issues.

Podcast Producer provides the following logs:

Log	Description
Podcast Producer Server Log	Records Podcast Producer server (`pcastserverd`) activity.
Podcast Producer Server Error Log	Records error messages generated by the Podcast Producer server.
Podcast Producer Server Startup Log	Records error messages generated by the Podcast Producer server during startup.

Log	Description
Podcast Producer HTTP Access Log	Records requests processed by the Apache server instance (`httpd`) used by Podcast Producer.
Podcast Producer HTTP Error Log	Records HTTP error messages generated by Podcast Producer's `httpd` instance.
Podcast Producer Application Log	Records external HTTP requests to the Podcast Producer server.



```
Sun Jul 05 14:02:09 -0700 2009 -- Xgrid Job: 7 is Running
Sun Jul 05 14:05:41 -0700 2009 -- Xgrid Job: 7 is Finished
Sun Jul 05 14:08:02 -0700 2009 -- ==============================================================
Sun Jul 05 14:08:02 -0700 2009 -- Queuing Xgrid Job for Podcast Producer Job ID: 9
Sun Jul 05 14:08:02 -0700 2009 -- Using workflow bundle: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Server/
Workflows/DEFA1587-A650-426E-92DA-01C5EB811705.pwf
Sun Jul 05 14:08:02 -0700 2009 -- Assembling Metadata for recording: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/
UUIDs/36BB3C48-460A-4174-88C5-131E68A2CF2E.prb
Sun Jul 05 14:08:02 -0700 2009 -- Adding Recording Metadata: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/UUIDs/
36BB3C48-460A-4174-88C5-131E68A2CF2E.prb/Contents/Resources/Working/recording_metadata-ECE87C29-036F-4F7C-BB71-1B2E68F7886A.xml
Sun Jul 05 14:08:03 -0700 2009 -- Unable to resolve admin user: admin
Sun Jul 05 14:08:03 -0700 2009 -- Set permissions on '/Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Resources'
to '770' ownership: podcastxgrid_CF76E865-C8FD-4A9A-AEE2-B1F46A6A4BBF:_pcastserver
Sun Jul 05 14:08:03 -0700 2009 -- Cached Resources: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Server/Resources
as /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Resources
Sun Jul 05 14:08:03 -0700 2009 -- Set permissions on '/Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Resources'
to '770' recursive ownership: podcastxgrid_CF76E865-C8FD-4A9A-AEE2-B1F46A6A4BBF:_pcastserver
Sun Jul 05 14:08:03 -0700 2009 -- Adding User Metadata: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/UUIDs/
36BB3C48-460A-4174-88C5-131E68A2CF2E.prb/Contents/Resources/Working/user_metadata-ECE87C29-036F-4F7C-BB71-1B2E68F7886A.plist
Sun Jul 05 14:08:03 -0700 2009 -- Set permissions on '/Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Workflows'
to '770' ownership: podcastxgrid_CF76E865-C8FD-4A9A-AEE2-B1F46A6A4BBF:_pcastserver
Sun Jul 05 14:08:03 -0700 2009 -- Cached Workflow: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Server/Workflows/
DEFA1587-A650-426E-92DA-01C5EB811705.pwf as /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Workflows/DEFA1587-
A650-426E-92DA-01C5EB811705.pwf
Sun Jul 05 14:08:03 -0700 2009 -- Set permissions on '/Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Workflows/
DEFA1587-A650-426E-92DA-01C5EB811705.pwf' to '770' recursive ownership: podcastxgrid_CF76E865-C8FD-4A9A-AEE2-B1F46A6A4BBF:_pcastserver
Sun Jul 05 14:08:04 -0700 2009 -- Wrote Xgrid job batch file: /Network/Servers/pcast.example.com/Library/PodcastProducer/Shared/Server/
Jobs/6E68497B-31F4-47E9-A7BD-1DBC1461C8CC_job.xml
Sun Jul 05 14:08:04 -0700 2009 -- Successfully queued Xgrid Job: 8 for Podcast Producer Job ID: 9
Sun Jul 05 14:08:04 -0700 2009 -- ==============================================================
Sun Jul 05 14:08:10 -0700 2009 -- Xgrid Job: 8 is Running
Sun Jul 05 14:12:04 -0700 2009 -- Xgrid Job: 7 is Deleted
Sun Jul 05 14:12:09 -0700 2009 -- Xgrid Job: 8 is Finished
Sun Jul 05 14:18:03 -0700 2009 -- Xgrid Job: 8 is Deleted
```

View: Podcast Producer Server Log

In addition, Podcast Producer server stores additional error logs in /Library/Logs/pcastserverd/DiagnosticReports/. Anytime a workflow fails, Podcast Producer server adds an error log to this folder.

Important: The DiagnosticReports folder exists on the computer running Podcast Producer server.

To view Podcast Producer logs using Server Admin:

1 Open Server Admin.

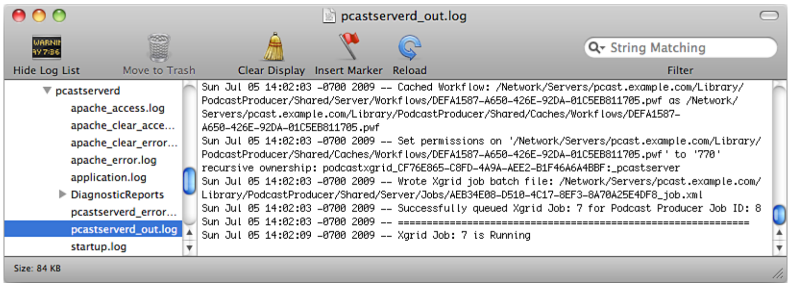
2 Select the server, then click the service disclosure triangle to show the services for administration.

3 In the service list beneath the server, select Podcast Producer.

4 Click Logs.

5 From the View pop-up menu, choose the log to view.

You can also view the Podcast Producer logs (LOG FILES > /Library/Logs > pcastserverd) and the system log using Console (in /Applications/Utilities/).



```
Sun Jul 05 14:02:03 -0700 2009 -- Cached Workflow: /Network/Servers/pcast.example.com/Library/
PodcastProducer/Shared/Server/Workflows/DEFA1587-A650-426E-92DA-01C5EB811705.pwf as /Network/
Servers/pcast.example.com/Library/PodcastProducer/Shared/Caches/Workflows/DEFA1587-
A650-426E-92DA-01C5EB811705.pwf
Sun Jul 05 14:02:03 -0700 2009 -- Set permissions on '/Network/Servers/pcast.example.com/Library/
PodcastProducer/Shared/Caches/Workflows/DEFA1587-A650-426E-92DA-01C5EB811705.pwf' to '770'
recursive ownership: podcastxgrid_CF76E865-C8FD-4A9A-AEE2-B1F46A6A4BBF:_pcastserver
Sun Jul 05 14:02:03 -0700 2009 -- Wrote Xgrid job batch file: /Network/Servers/pcast.example.com/
Library/PodcastProducer/Shared/Server/Jobs/AEB34E08-D510-4C17-8EF3-8A70A25E4DF8_job.xml
Sun Jul 05 14:02:03 -0700 2009 -- Successfully queued Xgrid Job: 7 for Podcast Producer Job ID: 8
Sun Jul 05 14:02:03 -0700 2009 -- ==============================================================
Sun Jul 05 14:02:09 -0700 2009 -- Xgrid Job: 7 is Running
```

Tip: Camera binding failures appear in system.log. The full log is at /Library/Logs/pcastagentd.log.

Monitoring Movie Transfers

Podcast Capture users can monitor the progress of movie submissions to the Podcast Producer server using the Transfers window, as described in the onscreen help for Podcast Capture.

You can also look at the system log on the Podcast Producer server for Podcast Producer uploader entries.

Monitoring Xgrid Job Progress

The Podcast Producer server sends a notification message when a Podcast Producer Xgrid job completes successfully, but you may want to actively monitor the progress of an Xgrid job.

Using Xgrid Admin

To actively monitor Xgrid job progress, use Xgrid Admin. Xgrid Admin shows you the progress of Xgrid jobs and whether they succeeded or failed.

To monitor Xgrid jobs using Xgrid Admin:

1 Launch Xgrid Admin (in /Application/Server/).

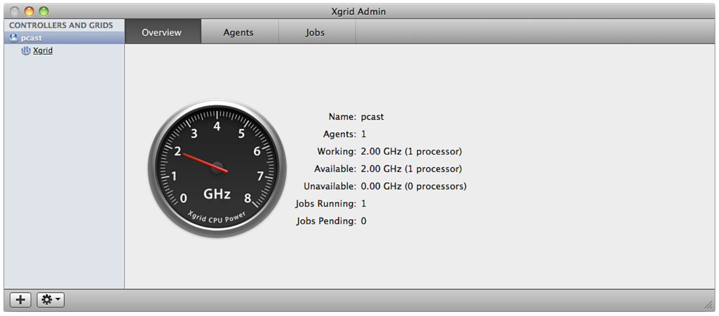
2 If prompted, provide your administrator credentials to authenticate.

Authenticating as an administrator user allows you to monitor and manage Xgrid jobs.

3 In the Controllers and Grids list, select the controller.

4 Click Overview.

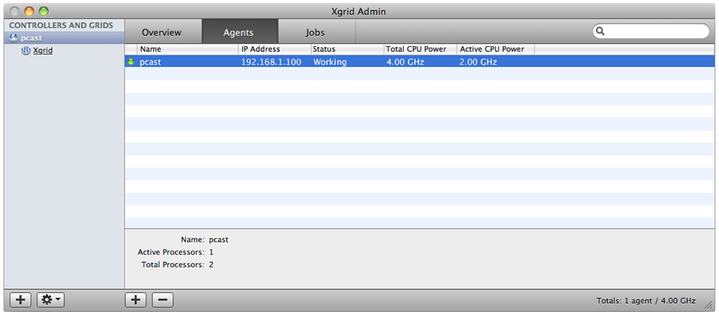
The Overview pane shows you how much CPU power your Xgrid cluster is consuming while processing Podcast Producer workflows.



Name: pcast
Agents: 1
Working: 2.00 GHz (1 processor)
Available: 2.00 GHz (1 processor)
Unavailable: 0.00 GHz (0 processors)
Jobs Running: 1
Jobs Pending: 0

5 Click Agents.

The list of agents lists all Xgrid agents in your Xgrid cluster. A green indicator next to an agent means that the agent is executing workflow tasks.



Name	IP Address	Status	Total CPU Power	Active CPU Power
pcast	192.168.1.100	Working	4.00 GHz	2.00 GHz

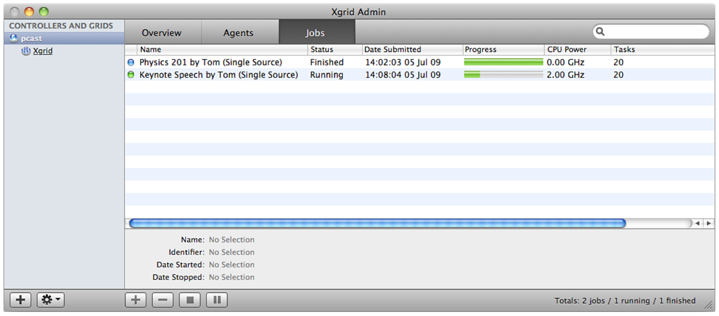
Name: pcast
Active Processors: 1
Total Processors: 2

Totals: 1 agent / 4.00 GHz

You can add and remove agents from the cluster using the Add (+) and Remove (-) buttons.

6 Click Jobs.

Jobs that are running appear in the list of jobs. After a job completes running successfully, it is removed from the list. However, if a job fails, it remains on the list and a red indicator appears next to the job.



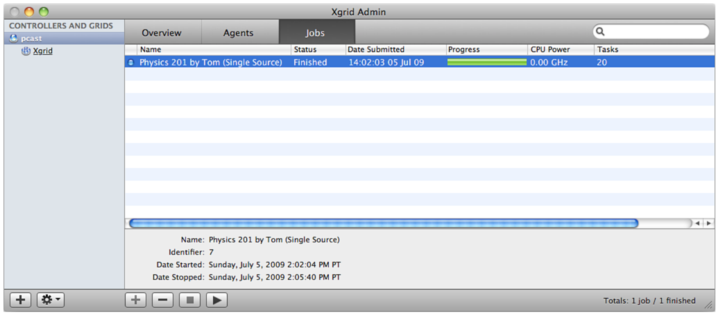
Name	Status	Date Submitted	Progress	CPU Power	Tasks
Physics 201 by Tom (Single Source)	Finished	14:02:03 05 Jul 09		0.00 GHz	20
Keynote Speech by Tom (Single Source)	Running	14:08:04 05 Jul 09		2.00 GHz	20

Name: No Selection
Identifier: No Selection
Date Started: No Selection
Date Stopped: No Selection

Totals: 2 jobs / 1 running / 1 finished

Tip: You can use Xgrid Admin to pause, resume, stop, and restart a job. You can also remove jobs.

7 To see the details of a job, select it.



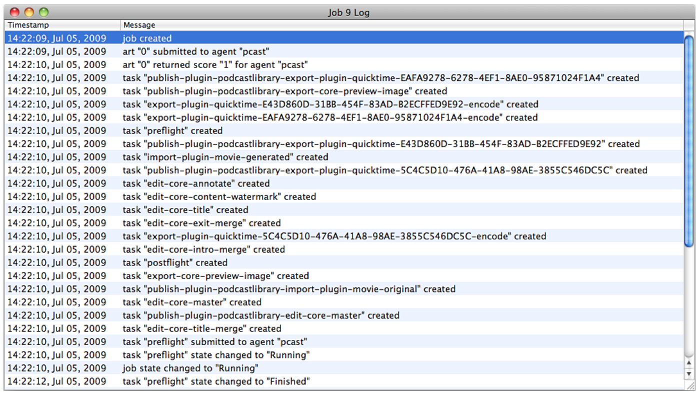
Name	Status	Date Submitted	Progress	CPU Power	Tasks
Physics 201 by Tom (Single Source)	Finished	14:02:03 05 Jul 09		0.00 GHz	20

Name: Physics 201 by Tom (Single Source)
Identifier: 7
Date Started: Sunday, July 5, 2009 2:02:04 PM PT
Date Stopped: Sunday, July 5, 2009 2:05:40 PM PT

Totals: 1 job / 1 finished

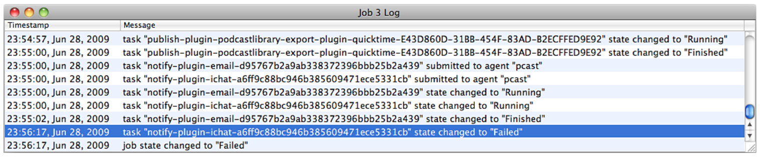
8 To find more information about the status of a job's tasks, double-click the job.

A window displays the job's log. This log lists the tasks that were processed and provides details about each task.



```
Timestamp                Message
14:22:09, Jul 05, 2009   job created
14:22:09, Jul 05, 2009   art "0" submitted to agent "pcast"
14:22:10, Jul 05, 2009   art "0" returned score "1" for agent "pcast"
14:22:10, Jul 05, 2009   task "publish-plugin-podcastlibrary-export-plugin-quicktime-EAFA9278-6278-4EF1-8AE0-95871024F1A4" created
14:22:10, Jul 05, 2009   task "publish-plugin-podcastlibrary-export-core-preview-image" created
14:22:10, Jul 05, 2009   task "export-plugin-quicktime-E43D860D-31BB-454F-83AD-B2ECFFED9E92-encode" created
14:22:10, Jul 05, 2009   task "export-plugin-quicktime-EAFA9278-6278-4EF1-8AE0-95871024F1A4-encode" created
14:22:10, Jul 05, 2009   task "preflight" created
14:22:10, Jul 05, 2009   task "publish-plugin-podcastlibrary-export-plugin-quicktime-E43D860D-31BB-454F-83AD-B2ECFFED9E92" created
14:22:10, Jul 05, 2009   task "import-plugin-movie-generated" created
14:22:10, Jul 05, 2009   task "publish-plugin-podcastlibrary-export-plugin-quicktime-5C4C5D10-476A-41A8-98AE-3855C546DC5C" created
14:22:10, Jul 05, 2009   task "edit-core-annotate" created
14:22:10, Jul 05, 2009   task "edit-core-content-watermark" created
14:22:10, Jul 05, 2009   task "edit-core-title" created
14:22:10, Jul 05, 2009   task "edit-core-exit-merge" created
14:22:10, Jul 05, 2009   task "export-plugin-quicktime-5C4C5D10-476A-41A8-98AE-3855C546DC5C-encode" created
14:22:10, Jul 05, 2009   task "edit-core-intro-merge" created
14:22:10, Jul 05, 2009   task "postflight" created
14:22:10, Jul 05, 2009   task "export-core-preview-image" created
14:22:10, Jul 05, 2009   task "publish-plugin-podcastlibrary-import-plugin-movie-original" created
14:22:10, Jul 05, 2009   task "edit-core-master" created
14:22:10, Jul 05, 2009   task "publish-plugin-podcastlibrary-edit-core-master" created
14:22:10, Jul 05, 2009   task "edit-core-title-merge" created
14:22:10, Jul 05, 2009   task "preflight" submitted to agent "pcast"
14:22:10, Jul 05, 2009   task "preflight" state changed to "Running"
14:22:10, Jul 05, 2009   job state changed to "Running"
14:22:12, Jul 05, 2009   task "preflight" state changed to "Finished"
```

The job's log is useful in troubleshooting workflow problems. It helps you quickly determine which task has failed.



```
Timestamp                Message
23:54:57, Jun 28, 2009   task "publish-plugin-podcastlibrary-export-plugin-quicktime-E43D860D-31BB-454F-83AD-B2ECFFED9E92" state changed to "Running"
23:55:00, Jun 28, 2009   task "publish-plugin-podcastlibrary-export-plugin-quicktime-E43D860D-31BB-454F-83AD-B2ECFFED9E92" state changed to "Finished"
23:55:00, Jun 28, 2009   task "notify-plugin-email-d95767b2a9ab338372396bbb25b2a439" submitted to agent "pcast"
23:55:00, Jun 28, 2009   task "notify-plugin-ichat-a6ff9c88bc946b385609471ece5331cb" submitted to agent "pcast"
23:55:00, Jun 28, 2009   task "notify-plugin-email-d95767b2a9ab338372396bbb25b2a439" state changed to "Running"
23:55:00, Jun 28, 2009   task "notify-plugin-ichat-a6ff9c88bc946b385609471ece5331cb" state changed to "Running"
23:55:02, Jun 28, 2009   task "notify-plugin-email-d95767b2a9ab338372396bbb25b2a439" state changed to "Finished"
23:56:17, Jun 28, 2009   task "notify-plugin-ichat-a6ff9c88bc946b385609471ece5331cb" state changed to "Failed"
23:56:17, Jun 28, 2009   job state changed to "Failed"
```

For more information about Xgrid Admin, see *Xgrid Administration and High Performance Computing.*

Using the Command Line

You can use the `xgrid` command-line tool to monitor Podcast Producer Xgrid job progress.

However, before you can use the `xgrid` command-line tool to monitor an Xgrid job, you need to get the job's ID from the Podcast Producer Server Log.

For example, when the Podcast Producer server submits a job to the Xgrid controller, entries similar to the following appear in the Podcast Producer Server Log:

```
Sun Jul 05 13:03:05 -0700 2009 -- Wrote Xgrid job batch file: /Network/
     Servers/pcast.example.com/Library/PodcastProducer/Shared/Server/
     Jobs/B48F2BC8-E01C-4721-B60B-883CC0179A87_job.xml

Sun Jul 05 13:03:05 -0700 2009 -- Successfully queued Xgrid Job: 4 for
     Podcast Producer Job ID: 5

Sun Jul 05 13:03:05 -0700 2009 -- ========================================
     ======================

Sun Jul 05 13:03:11 -0700 2009 -- Xgrid Job: 4 is Running
```

The first entry tells you where the Xgrid job that the Podcast Producer created is stored. The next entry displays the ID of the Xgrid job and the last entry indicates that the job is running.

To find out the status of an Xgrid job, run the `xgrid` command and supply the ID of the Xgrid job as input, as in the following example:

```
$ xgrid -h pcast.example.com -auth Kerberos -job results -id 4
```

You can also view the attributes of an Xgrid job, as in the following example:

```
$ xgrid -h pcast.example.com -auth Kerberos -job attributes -id 5
{
    jobAttributes =     {
        activeCPUPower = 2000;
        dateNow = "2009-07-05 13:32:42 -0700";
        dateStarted = "2009-07-05 13:30:47 -0700";
        dateSubmitted = "2009-07-05 13:30:45 -0700";
        jobStatus = Running;
        name = "Business Report Q1 by Tom (Montage)";
        percentDone = "44.68087005615234";
        taskCount = 18;
        undoneTaskCount = 10;
    };
}
```

For more information about using the `xgrid` command-line tool, see its man page.

Index