



Wowza Media Server® 3

Wowza Media Server 3 for Amazon EC2 Edition

Wowza Media Server 3 for Amazon EC2 Edition



Version 3

Copyright © 2006 – 2012 Wowza Media Systems, Inc.
<http://www.wowza.com>

This document is for informational purposes only and in no way shall be interpreted or construed to create any warranties of any kind, either express or implied, regarding the information contained herein.

Third Party Information

This document contains links to third party websites that are not under the control of Wowza Media Systems, Inc. (“Wowza”) and Wowza is not responsible for the content on any linked site. If you access a third party website mentioned in this document, then you do so at your own risk. Wowza provides these links only as a convenience, and the inclusion of any link does not imply that Wowza endorses or accepts any responsibility for the content on third party sites.

This document also refers to other third party software that is not licensed, sold, distributed or otherwise endorsed by Wowza. Please ensure that any and all use of Wowza® software and third party software is properly licensed.

Trademarks

Wowza, Wowza Media Systems, Wowza Media Server and related logos are either registered trademarks or trademarks of Wowza Media System, Inc. in the United States and/or other countries.

Adobe and Flash are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Microsoft and Silverlight are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

QuickTime, iPhone, iPad and iPod are either registered trademarks or trademarks of Apple, Inc. in the United States and/or other countries.

Other product names, logos, designs, titles, words or phrases mentioned may be third party registered trademarks or trademarks in the United States and/or other countries.

Third party trademarks are used solely to identify and describe third party products as being compatible with Wowza products. Wowza is in no way sponsored, endorsed by or otherwise affiliated with any such third party trademark owners.

Third Party Copyright Notices

Log4j and Mina: Copyright © 2006, The Apache Software Foundation

Java Service Wrapper: Copyright © 1999, 2006, Tanuki Software, Inc.

Silver Egg Technology: Copyright © 2001, Silver Egg Technology

Java ID3 Tag Library and JLayer 1.0 (classic): Copyright © 1991, 1999, Free Software Foundation, Inc.

Bouncy Castle Crypto API: Copyright © 2000 – 2008, The Legion Of The Bouncy Castle

Apache Commons Lang libraries and Modeler libraries: Copyright © 2001-2008, The Apache Software Foundation

WebM VP8 Codec libraries: Copyright © 2010, Google Inc. All rights reserved.

Vorbis/Ogg libraries: Copyright © 2011, Xiph.org Foundation

Libgcc s-4 library and Libstdc++ library: Copyright © 2011, Free Software Foundation, Inc.

Speex Codec: Copyright © 2002-2003, Jean-Marc Valin/Xiph.org Foundation

Table of Contents

What's New	5
Introduction	5
Getting Started	6
Starting an Instance.....	7
Get Public Domain Name of Amazon EC2 Instance.....	9
Stream a Video.....	10
Open a Secure Telnet Session (SSH).....	10
Adding Perpetual or Subscription License Key.....	11
Linux shell command cheat sheet for beginners.....	12
Stopping an Instance.....	12
Performance.....	13
FTP Access.....	13
Wowza Server Configuration	14
Using a Startup Package	14
Startup Package Example	14
Startup Package Basics.....	15
Pre-built Startup Packages	15
Wowza Server Instance Details.....	15
Wowza Server Details	16
Java Management Extension (JMX)	17
Default Startup Package	17
Custom Module Development.....	17
Streaming Media Directly from S3.....	18
Startup Package Reference	21
Command <Install>.....	21
Command <Download>	22
Command <RunScript>	24
Startup Package Debugging	25
Additional Resources	26

What's New

Change	Description	Release Date
Initial Release	Document release	
Doc v3.0.0	Updated for Wowza Media Server 3	November 1, 2011
Doc v3.0.4	Updated for Wowza Media Server 3.0.4	January 26, 2012

Introduction

Amazon Elastic Cloud Computing (Amazon EC2) is a web service from Amazon that provides flexible, easy to provision computing resources. Wowza Media Server 3 (Wowza Server) is a Java based, high-performance, extensible and fully interactive media streaming software platform that provides live and on-demand streaming, chat and remote recording capabilities to a wide variety of media player technologies. Wowza Server can deliver content to many popular media players such as Adobe® Flash® Player, Microsoft Silverlight® player, Apple iPhone®, iPad® and iPod® touch and Apple QuickTime® player, among others. Wowza Media Server 3 includes support for many streaming protocols including the Real-Time Messaging Protocol (RTMP), Microsoft Smooth Streaming, Apple HTTP Live Streaming, Real-Time Streaming Protocol (RTSP), Real-time Transport Protocol (RTP), MPEG2 Transport Streams (MPEG-TS) and more. It is an alternative to the Adobe Flash Media Server products (FMIS and FMSS), Apple Streaming Server (Darwin) and other media servers. This document describes how to install and configure Wowza Media Server 3 on Amazon EC2.

If you are interested in learning more about Wowza Server on Amazon EC2, visit the following page on the Wowza Media Systems website:

<http://www.wowza.com/ec2.html>

This document assumes you have signed up for Amazon EC2 and will provide instructions for starting and stopping Wowza Media Server 3 Amazon EC2 instances via the Amazon AWS Management Console. You can obtain more information about Amazon EC2 here:

<http://www.amazon.com/ec2>

This document also assumes you are familiar with Wowza Server. You can download a free Wowza Server Trial edition license by going to:

<http://www.wowza.com/pricing/trial>

The Wowza Server download includes the Wowza Server software, all premium AddOns, documentation and examples. Once you have your client side and server side applications up and running on your local machine, use this document to learn how to deploy it on Amazon EC2.

Amazon EC2 is a cloud computing platform that virtualizes computing resources as virtual machines. A single virtual machine configuration is registered as an item called an AMI (Amazon Machine Image). Wowza Media Systems provides public Amazon AMIs with pre-configured versions of Wowza Media Server 3 that are ready to start through the AWS Management Console. A running virtual machine is called an instance.

Wowza Media Server 3 supports two licensing models on Amazon EC2 with the ability to select the licensing model on a per-instance basis:

- **lickey:** Use a regular license key such as a monthly, daily or standard perpetual edition key. With this option all billing for your Amazon EC2 instance time is handled by Amazon and all billing for the use of the Wowza Server software is handled by Wowza Media Systems.

Note: This option provides access to the complete server functionality and all premium AddOns, including Wowza Transcoder (64-bit instance only), Wowza nDVR and Wowza DRM.

- **devpay:** Use the Amazon EC2 DevPay system. When using this option, the Amazon EC2 instance time, consumed bandwidth and Wowza Server use is billed through the Amazon AWS billing system.

Note: This option does not provide access to the premium AddOns.

There are two sets of AMI IDs listed on the [Wowza Media Server 3 for Amazon EC2 Support](#) page. The AMI ID selection determines which licensing model is used. When using the lickey model you must provide a valid Wowza Media Server license key. The pricing for these options is described here:

<http://www.wowza.com/ec2.html>

This document is meant to help users specifically with Wowza Media Server 3 for Amazon EC2. Once you're familiar with the basic tutorials in this document see the [Wowza Media Server 3 for EC2 Quick Start Guide](#) for additional examples. The [Wowza Media Server 3 User's Guide](#) contains comprehensive documentation for Wowza Media Server 3.

Getting Started

AWS (Amazon Web Services) Management Console is a web interface that allows you to manage your Amazon EC2 account from your web browser. Many users will find the AWS Management Console easier than the command line tools to manage EC2 and the Wowza Media Server 3 AMIs. If you intend to follow the directions for the AWS Management Console this document

assumes that you have a modern web browser installed. We recommend Mozilla Firefox 4 or greater as Amazon has built some specific functionality into the Console that is only supported by this browser. Mozilla Firefox is available for download by going to:

<http://www.mozilla.com/firefox>

The AWS Management Console is available here:

<https://console.aws.amazon.com/ec2/home>

After signing up for Amazon EC2 you will need to get the current Wowza Media Server 3 AMI IDs. You can get these IDs from the [Wowza Media Server 3 for Amazon EC2 Support](#) page. These AMI IDs are what will be used to startup a Wowza Media Server 3 EC2 instance. There are two sets of AMI IDs **lickey** (License Key) and **devpay**. The **lickey** AMI IDs are used when you plan on using a monthly, daily or standard perpetual license key on Amazon EC2. The **devpay** AMI IDs are used when you plan on using the Amazon DevPay billing system. To use the **devpay** AMI IDs you must first register for Wowza Media Server on Amazon EC2. The registration process is described here:

<http://www.wowza.com/ec2-streaming.html>

Starting an Instance

Start the AWS Management Console by opening the following URL in your web browser (Firefox or Chrome are recommended), provide your AWS credentials and sign-in:

<https://console.aws.amazon.com/ec2/home>

The Amazon EC2 Console Dashboard will load. If a different AWS dashboard loads just click the **Amazon EC2** tab along the top of the page. In the **Navigation** pane on the left you will see a drop-down menu under the word **Region**. Confirm that this is set to the region in which you intend to start the instance. Refer to the [Wowza Media Server 3 for Amazon EC2 Support](#) page to see regional pricing differences.

First, we need to open up several TCP and UDP ports for streaming. Do the following:

1. Select **Security Groups** in the **Navigation** pane on the left.
2. Select the **default** security group and select the **InBound** tab.
3. Enter the following values and click the **Add Rule** button to create a new rule to open TCP port 1935 for RTMP streaming.
 - a. Create a new rule: **Custom TCP rule**
 - b. Port range: **1935**
 - c. Source: **0.0.0.0/0**

Repeat the steps above to open the following ports:

- TCP port 80: HTTP streaming
- TCP port 443: HTTP and RTMP streaming
- TCP port 554: RTSP streaming
- UDP ports 0-65535: RTP and MPEG-TS streaming
- TCP port 21: FTP access
- TCP port 22: SSH and telnet access
- TCP port: 8086: Stream Manager
- TCP ports: 8084-8085: JMX Interface

Note: When opening UDP ports, be sure to select **Custom UDP rule** in the **Create a new rule** drop down menu.

Use caution when opening ports on your server. Reference the Amazon security guide here for best practices when using an Amazon AMI:

<http://aws.amazon.com/security/>

Next, we will create a key pair so that we can later log into our Amazon EC2 instance using telnet. Do the following:

1. Select **Key Pairs** in the **Navigation** pane on the left.
2. Click the **Create Key Pair** button.
3. Enter a name for your key pair and click the **Create** button. For this example we name our key pair **wowza-keypair**.
4. You will be prompted to download the private key file **wowza-keypair.pem**. Download and store this file. It will be used later when we connect to the instance using the PuTTY telnet client.

Next, select an AMI ID from the [Wowza Media Server 3 for Amazon EC2 Support](#) page. There are three factors that go into determining which of the AMI IDs to select: region, license type (**devpay** or **lickey**) and server architecture (32-bit or 64-bit). Select the appropriate AMI ID for your given situation. The document below describes the available instance types and includes server architecture information (32-bit platform or 64-bit platform):

<http://aws.amazon.com/ec2/instance-types>

To launch an Amazon EC2 instance do the following:

1. Select **EC2 Dashboard** in the **Navigation** pane on the left.
2. Click the **Launch Instance** button.
3. Select the **Community AMIs** tab and enter the AMI ID selected above into the field to the right of the **Viewing** drop down list. It may take several seconds for the list of matching AMI IDs to be populated with your selection.
4. Once your AMI ID is found, click the **Select** button to start the **Request Instances Wizard**.
 - a. Select the number of instances you wish to start, the **Instance Type** (be sure the instance type 32-bit or 64-bit matches the AMI ID that is being used) and select the **Availability Zone** (or leave the default as **No Preference**) and click the **Continue** button.
 - b. Select **Advanced Instance Options** or leave default options and click the **Continue** button. If you are using startup packages (described later in this document) this is where you attach the startup package file as **User Data**.
 - c. Add **Tags** to your instance if needed then click the **Continue** button.
 - d. Select the **wowza-keypair** created above and click the **Continue** button.
 - e. Select the **default** security group and click the **Continue** button.
 - f. Click the **Launch** button then the **Close** button to close the wizard.
5. Select **Instances** in the **Navigation** pane on the left to see the status of your EC2 instance. It may take several minutes for your instance to start. Once the instance **Status** changes to **running** the instance is started. It might take an additional minute or two for Wowza Server to initialize and be available for streaming.

Note: Security Groups and **Key Pairs** are regional resources and need to be configured in each of the regions you plan to use for streaming.

Get Public Domain Name of Amazon EC2 Instance

To get the public domain name of your instance, select **Instances** in the **Navigation** pane on the left and select the running instance. Detailed information about your instance will be displayed in the panel below the instance list. The **Public DNS** value is the public domain name of your running instance. You will use this value to access to instance remotely for streaming and to connect to the instance using telnet. In many of the examples below the public domain name of an instance is referred to as **[instance-public-domain]**.

Stream a Video

You can quickly test your running Wowza Media Server 3 instance using the **SimpleVideoStreaming** example that ships with Wowza Server. You will need to install Wowza Server on your local machine or download a set of example files from [here](#). Once the Wowza Server or examples are installed, the **SimpleVideoStreaming** example player can be found here:

```
[install-dir]/examples/SimpleVideoStreaming/client/simplevideostreaming.html
```

Open this HTML file in a web browser and enter the following information and click the **Play** button. The **Big Buck Bunny** video should begin to play.

```
Server: rtmp://[instance-public-domain]/vod
Stream: mp4:sample.mp4
```

Where [instance-public-domain] is the public domain name of your Amazon EC2 instance running Wowza Media Server 3.

Open a Secure Telnet Session (SSH)

To open a secure telnet session to your Amazon EC2 instance using secure shell (SSH), you must first open up TCP port 22. Be sure you have properly followed the **Security Groups** instructions above to open up TCP port 22.

This section will use the Windows telnet client called PuTTY and the key generator called PuTTYgen. PuTTY and PuTTYgen can be downloaded from this website:

<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

Amazon EC2 uses **Key Pairs** (configured above) to connect to running Amazon EC2 instances using telnet. PuTTY requires that these key pairs be converted to PPK format before they can be used with the PuTTY telnet client. To convert the **wowza-keypair.pem** file download above to a PPK file follow these steps:

1. Launch **PuTTYgen.exe**.
2. Click the **Load** button and select the **wowza-keypair.pem** file (You will need to select **All Files *.*** option from the file filter drop down to see the **wowza-keypair.pem** in the file list).
3. Click the **Save private key** button and save the file with the name **wowza-keypair.ppk**.

To open a secure telnet session (SSH) to your Amazon EC2 instance using PuTTY, do the following:

1. Launch **PuTTY.exe**.
2. In the **Host Name (or IP address)** field enter the public domain name of your instance.
3. Select **SSH-Auth** in the **Category** pane on the left.
4. Click the **Browse...** button and open the **wowza-keypair.ppk** file.
5. Click the **Open** button to open the secure telnet session. The first time you connect to your instance you will get a **PuTTY Security Alert** that references the first use the **wowza-keypair**. Click **Yes** to accept the security key.
6. You should now see the **login as:** prompt in the telnet window, enter the username **ec2-user**. You will now be logged into your Amazon EC2 instance. When working with Wowza Server it is best to be logged in as the **root** user. You can switch to the **root** user by entering the following command into the telnet window:

```
sudo su -
```

You can end your SSH session by entering the **exit** command (you may have to enter the **exit** command twice if you are logged in as the **root** user).

Adding Perpetual or Subscription License Key

This section applies to customers that are using the **lickey** AMI IDs. If you are using the **devpay** AMI IDs then there is no need to change the license key.

When using the **lickey** AMI IDs the instance is initially started using a temporary license key. You must replace this key with a valid Wowza Media Server 3 monthly, daily or standard perpetual license key. There are several ways to do this. The first method is to use telnet to connect to your instance, replace the temporary license key with a valid perpetual or subscription license key and restart Wowza Server. Follow these steps:

1. Start the Amazon EC2 instance as instructed above.
2. Once the instance is started, connect to the instance using a secure telnet session.
3. Change directory to the **/usr/local/WowzaMediaServer/conf** folder and edit the file **Server.license** using a text editor such as **vi**.
4. Replace the entire contents of this file with your monthly, daily or standard perpetual license key.
5. Stop and then start Wowza Server to activate the key. You can do this by executing the following commands:

```
service WowzaMediaServer stop  
service WowzaMediaServer start
```

A second option for license key replacement is to include the license key in a startup package. Startup packages are discussed later in this guide. If you include your license key in the following file in your startup package it will replace the temporary license key on instance startup.

```
[package-dir]/wowza/conf/Server.license
```

The third option is to create your Amazon EC2 AMI that includes the license key. We do not have explicit instructions on how to create your own AMI. It is beyond the scope of this document but it is an option when using your own Wowza Media Server 3 license key on Amazon EC2.

Linux shell command cheat sheet for beginners

Connecting to an instance with a SSH session makes you a Linux user. Congratulations! If that is not a familiar environment for you here are some simple commands you may find useful. If the command you're looking for is not on this list there are many guides posted on the internet regarding Linux.

```
cd /home/wowza
```

This will change your directory to one that shows the most common user accessed Wowza Media Server directories. From here you can get a list of the Wowza Server sub-directories by typing: `ls`. This will list the files and directories in the Wowza Server directory. To change directories simply type `cd [directory name]` to move into a directory and to move back up a level type `cd ..`.

You can interactively watch the Wowza Server log entries as they are added to the Wowza Server logs by executing the following commands:

```
cd /usr/local/WowzaMediaServer/logs  
tail -f wowzamediaserver_access.log
```

To stop the Wowza Server enter the command:

```
service WowzaMediaServer stop
```

To start the Wowza Server enter the command:

```
service WowzaMediaServer start
```

Stopping an Instance

When you stop an instance you will lose all changes or files you have on the server. If you have anything you don't want to lose save it to Amazon S3 or to a local machine before stopping the instance.

In the **Instances** window of the AWS Management Console select the instance(s) you want to stop running, click the **Instance Actions** button, then **Terminate**. The **State** info will show **shutting-down** and finally **terminated**.

Important: Be aware that Amazon recommends you confirm that the machine reaches a status of **terminated** as they will continue to charge for instances that fail to shut down correctly.

Performance

Below are some performance guidelines when using Wowza Media Server 3 on Amazon EC2. These are total bitrate values that a single instance of a given instance type can handle for outgoing streaming. To calculate concurrent connections divide these numbers by the bitrate of your stream in Kbps.

```
m1.small:    150,000 Kbps
m1.large:    250,000 Kbps
m1.xlarge:   350,000 Kbps
```

For example if you are doing live streaming using a 500Kbps live stream then a given instance type can handle the following number of concurrent connections.

```
m1.small:    300 concurrent connections
m1.large:    500 concurrent connections
m1.xlarge:   700 concurrent connections
```

FTP Access

The Wowza Media Server3 on Amazon EC2 instance comes pre-installed with the **vsftpd** FTP server. A user named **wowza** has been added to the system with the password set the instance id of the Amazon instance (this is done for security reasons). You can get the instance id of a running instance either through the AWS Management Console or if logged into the instance using the following command:

```
wget -q -O - http://169.254.169.254/latest/meta-data/instance-id
```

This account can be used to upload content or configure the server using ftp. Be sure to open up TCP port 21 in your **Security Groups** settings to connect to your instance using FTP.

Also, you will need to setup your FTP client to use the PORT (also known as ACTIVE) communication method (rather than the PASV). Consult your FTP client's documentation for more information.

For convenience most of the Wowza Server folders have been symbolically linked to the /home/wowza folder for easy access using FTP.

You can easily change the password for the default **wowza** user by logging into the instance as the **ec2-user** user and executing the following commands (follow the prompts):

```
sudo passwd wowza
```

Note: For Security reasons we strongly recommend that you change the default password for the **wowza** FTP account for your AMI or that you keep TCP port 21 closed.

Wowza Server Configuration

The getting started section above describes how to start a generic Wowza Media Server 3 instance with most of the example applications installed. You can configure a Wowza Media Server 3 Amazon EC2 instance at startup time by passing in user data in the form of a startup package. A startup package is a zip archive of a folder that contains a startup manifest (startup.xml) along with configuration files and scripts. Amazon EC2 limits the size of a startup package to 16kB (this is the maximum user data size).

Using a Startup Package

When you launch a new instance using the **AWS Management Console** you can load a startup package on the **Advanced Instance Options** step of the **Request Instances Wizard**. In the **User Data** section select the **as file** radio button then browse to your startup package zip file (be sure the **base64 encoded** checkbox is unchecked).

Note: Amazon Management Console support for Startup package functionality is ONLY available in Firefox 4 and greater and Chrome 11.0.696.77 or greater.

Startup Package Example

Below is the file structure of a simple startup package:

```
[mywowzaconfig]
  startup.xml
  [wowza]
    [applications]
      [myapp]
    [conf]
      Server.license
      [myapp]
        Application.xml
  [tuning]
    tune.sh
    MediaCache.xml
    setenv.sh
```

Here are the contents of startup manifest (startup.xml) for the startup package outlined above:

```
<Startup>
  <Commands>
    <Install>
      <Folder>wowza</Folder>
    </Install>
    <RunScript>
      <Script>tuning/tune.sh</Script>
    </RunScript>
  </Commands>
</Startup>
```

You can download the default startup package to see how it is constructed using this URL:

http://wowzamediasystems.s3.amazonaws.com/com/wowza/startup/default_3.0.4.zip

Startup Package Basics

Let's take a quick look at the example startup package.

```
<Install>
  <Folder>wowza</Folder>
</Install>
```

The first command instructs the startup processor to copy the contents of the included **wowza** folder into the **/usr/local/WowzaMediaServer** folder of the running Wowza Server instance. This gives you a chance to create application folders and configuration folders and files as well as inject a **Server.license** file for a **lickey** instance.

```
<RunScript>
  <Script>tuning/tune.sh</Script>
</RunScript>
```

The second command instructs the startup processor to run the included script **tuning/tune.sh**. This script changes configuration parameters based on instance size. This script must be a shell script (not a binary application) and will be executed by the operating environment that is running on the Wowza Server instance. When a script is executed the working directory is set to the root directory of the startup package (the folder that contains the startup.xml file).

Pre-built Startup Packages

Wowza Media Systems provides several pre-built startup packages that can either be used as is or modified to suite your needs. You can download these packages from the following web site:

<http://wowzamediasystems.s3.amazonaws.com/packagelist.html>

Wowza Server Instance Details

A Wowza Media Server 3 for Amazon EC2 instance is built using the **Amazon Linux AMI** as the base. This AMI is provided by Amazon and is described in detail here:

<http://aws.amazon.com/amazon-linux-ami/>

Other components that are installed:

- Wowza Media Server 3
- Latest Oracle Java JDK for Linux
- FTP Server vsftpd
- S3FS: <http://code.google.com/p/s3fs/>

Note: Wowza Media Server 3 for Amazon EC2 AMI does not include a web server.

Wowza Server Details

Wowza Media Server 3 is installed at its default location:

```
/usr/local/WowzaMediaServer
```

The Wowza Media Server service is running on the following ports:

```
TCP 1935 - RTMP (all variants), RTSP, Smooth and Cupertino Streaming
TCP 80 - RTMP (all variants), RTSP, Smooth and Cupertino Streaming
TCP 443 - RTMP (all variants), RTSP, Smooth and Cupertino Streaming
TCP 554 - RTMP (all variants), RTSP, Smooth and Cupertino Streaming
UDP 0-65535 - RTP and MPEG-TS UDP streaming
```

The Wowza Media Server is managed using the following ports:

```
8084 - JMX/JConsole Management
8085 - JMX/JConsole Management
8086 - Stream Manager and Administration
21 - FTP access
22 - SSH access
```

The Administration TCP port 8086 is configured to return load balancing information over HTTP. What this means is if you open a web browser and enter the URL:

```
http://[instance-public-domain]:8086/connectioninfo
```

where [instance-public-domain] is the public domain name for the instance.

Wowza Server will return load information for the instance. Example output is:

```
server=864
```

Where **864** is the current number of client connections to the instance.

Java Management Extension (JMX)

The JMX/JConsole interface (as described in the **Server Management Console and Monitoring** chapter of the [Wowza Media Server 3 User's Guide](#)) to your instance is preconfigured to listen to connections on TCP ports 8084 and 8085 using the public domain name. You will need to open these ports to TCP traffic to be able to successfully connect to your EC2 instance.

The JMX url is:

```
service:jmx:rmi://[instance-public-domain]:8084/jndi/rmi://[ instance-public-domain]:8085/jmxrmi
```

From most JMX tools such as JConsole you should be able to connect using the address:

```
[instance-public-domain]:8085
```

Where [instance-public-domain] is the public domain name of the instance. The default username is **admin** and password is **admin**. User access is managed in the following two files and is described in the [Wowza Media Server 3 User's Guide](#):

```
/usr/local/WowzaMediaServer/conf/jmxremote.access  
/usr/local/WowzaMediaServer/conf/jmxremote.password
```

Default Startup Package

If you start the Wowza Server EC2 AMI without specifying a startup package the default startup package will be used. If you want to take a look at the default startup package it can be downloaded at:

http://wowzamediasystems.s3.amazonaws.com/com/wowza/startup/default_3.0.4.zip

The following application names are configured in the default startup package: live, rtplive, shoutcast, videochat, vod, vods3

If you supply your own startup package the default startup package will not be used. Your startup package must provide all the application configurations that are needed for streaming.

Custom Module Development

There are several System level properties that are available when developing custom server side modules. These properties describe the current running instance. You can get the value of one of these system properties by executing the Java method:

```
String value = System.getProperty("com.wowza.amazonaws.ec2.AWSEC2_METADATA_INSTANCE_ID");
```

There are several System level properties that are available when developing custom server side modules. These properties describe the current running instance. You can get the value of one of these system properties by executing the Java method:

The available properties are:

com.wowza.amazonaws.ec2.AWSEC2_METADATA_INSTANCE_ID	- Amazon instance id
com.wowza.amazonaws.ec2.AWSEC2_METADATA_SECURITY_GROUPS	- Security group
com.wowza.amazonaws.ec2.AWSEC2_METADATA_LOCAL_IPV4	- Local IP address
com.wowza.amazonaws.ec2.AWSEC2_METADATA_AMI_LAUNCH_INDEX	- Launch index
com.wowza.amazonaws.ec2.AWSEC2_METADATA_PUBLIC_HOSTNAME	- Public host name
com.wowza.amazonaws.ec2.AWSEC2_METADATA_PRODUCT_CODES	- DevPay product code
com.wowza.amazonaws.ec2.AWSEC2_METADATA_INSTANCE_TYPE	- instance type (m1-small, m1-large, m1-xlarge)
com.wowza.amazonaws.ec2.AWSEC2_METADATA_HOSTNAME	- Public host name
com.wowza.amazonaws.ec2.AWSEC2_METADATA_LOCAL_HOSTNAME	- Local host name
com.wowza.amazonaws.ec2.AWSEC2_METADATA_PUBLIC_IPV4	- Public IP address
com.wowza.amazonaws.ec2.AWSEC2_METADATA_AMI_MANIFEST_PATH	- S3 manifest path
com.wowza.amazonaws.ec2.AWSEC2_METADATA_RESERVATION_ID	- Instance reservation ID
com.wowza.amazonaws.ec2.AWSEC2_METADATA_AMI_ID	- AMI ID
com.wowza.amazonaws.ec2.AWSEC2_METADATA_ANCESTOR_AMI_IDS	- List of ancestor IDs
com.wowza.amazonaws.ec2.AWSEC2_METADATA_KERNEL_ID	- Kernel ID
com.wowza.amazonaws.ec2.AWSEC2_METADATA_AVAILABILITY_ZONE	- Availability zone
com.wowza.amazonaws.ec2.AWSEC2_METADATA_PUBLIC_KEYS	- Public keys
com.wowza.amazonaws.ec2.AWSEC2_METADATA_RAMDISK_ID	- RAM disk ID
com.wowza.amazonaws.ec2.AWSEC2_METADATA_BLOCK_DEVICE_MAPPING	- block mapping

Note: To learn more about Wowza Media Server 3 module development download and install the Wowza IDE and read the included User’s Guide:

<http://www.wowza.com/ide.html>

Streaming Media Directly from S3

You can use Wowza Media Server 3 for Amazon EC2 to stream media directly from the Amazon Simple Storage Service (S3). Wowza Server employs a caching system improve the performance of streaming media from Amazon S3. The system uses a technology called MediaCache. The MediaCache AddOn package is a read through caching mechanism for video on demand streaming that can pull content from an HTTP or network attached storage source. The MediaCache system is tuned on a per-instance basis as a part of the Wowza Media Server provided startup packages. You can see the tuning in the **tuning/tune.sh** of any of the provided startup packages.

This feature is included in the default startup package for Wowza Media Server 3 for Amazon EC2 edition and is available when using the **vods3** application.

To stream content, use stream names in the following form. The amazons3/ portion of the stream identifies the stream as being sourced from Amazon S3:

[media-type]:amazons3/[s3-bucket-name]/[path-to-content-in-s3]

For example to play the file **mycoolvideo.m4v** that is stored in S3 bucket **mybucket** at the path **videos/** the stream URLs are:

To play using Adobe Flash player (RTMP):

```
Server: rtmp://[instance-public-domain]/vods3
Stream: mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v
```

To play using Adobe Flash player (RTMP single URL):

```
rtmp://[instance-public-domain]/vods3/_definst_/
mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v
```

To play using Adobe Flash player (San Jose/Flash HTTP)

```
http://[instance-public-domain]/vods3/_definst_/
mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v/manifest.f4m
```

To play using an Apple iOS device (Cupertino/Apple HTTP Live Streaming)

```
http://[instance-public-domain]/vods3/_definst_/
mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v/playlist.m3u8
```

To play using Microsoft Silverlight (Smooth Streaming)

```
http://[instance-public-domain]/vods3/_definst_/
mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v/Manifest
```

To play using RTSP/RTP player or device

```
rtsp://[instance-public-domain]/vods3/_definst_/
mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v
HH
```

Note: When streaming using a stream name that includes path elements such as our example above **mp4:amazons3/mybucket/videos/coolvideos/mycoolvideo.m4v** you must supply both the **application** and **applicationInstance** name as part of the URL. In the example above we are using the default **applicationInstance** name **_definst_**.

By default the **vods3** feature is configured with S3 authorization turned off. This means that all content must be publicly available. To stream non-publicly available content, unzip the **default_3.0.4.zip** startup package and modify the two system properties **awsAccessKeyId** and **awsSecretAccessKey** in **tuning/MediaCache.xml**. Set these two values to the **Access Key ID** and **Secret Access Key** for the user in which you would like to authorize access and zip up the default folder. Use this new zip archive as your startup package. For example:

```
<Property>
  <Name>awsAccessKeyId</Name>
  <Value>11DV8PNKTHN1234732</Value>
</Property>
<Property>
  <Name>awsSecretAccessKey</Name>
  <Value>p0fsdFIE1Nofyx5Sfe+CmuQi0uXt7ygrD8Xxz+</Value>
</Property>
```

The default setup is such that any content in S3 that is publicly available can be re-streamed through your instance. To limit the content to specific buckets you can use the stream alias system to add aliases for the buckets from which you would like to stream. The stream alias

package has been included with the **default_3.0.4.zip** startup package. To modify the default configuration edit the file **wowza/conf/aliasmap.play.txt** that is included in the **default_3.0.4.zip** startup package and create a new startup package with the modifications. To do this, unzip the **default_3.0.4.zip** startup package, add wildcard entries for the buckets from which you would like to stream content and comment out the default stream alias rule. For example, to limit streaming to content only coming from **mybucket** the **wowza/conf/aliasmap.play.txt** file should look like this:

```
mybucket/*=amazons3/mybucket/${Wildcard.Match1}
flv:mybucket/*=flv:amazons3/mybucket/${Wildcard.Match1}
mp3:mybucket/*=mp3:amazons3/mybucket/${Wildcard.Match1}
mp4:mybucket/*=mp4:amazons3/mybucket/${Wildcard.Match1}
# commented out *=${Stream.Name}
```

With this in place to play the file **mycoolvideo.m4v** that is stored in S3 bucket **mybucket** at the path **videos/coolvideos** the streaming URLs are:

To play using Adobe Flash player (RTMP):

```
Server: rtmp://[instance-public-domain]/vods3
Stream: mp4:mybucket/videos/coolvideos/mycoolvideo.m4v
```

To play using Adobe Flash player (RTMP single URL):

```
rtmp://[instance-public-domain]/
vods3/_definst_/mp4:mybucket/videos/coolvideos/mycoolvideo.m4v
```

To play using Adobe Flash player (San Jose/Flash HTTP)

```
http://[instance-public-domain]/vods3/_definst_/
mp4:mybucket/videos/coolvideos/mycoolvideo.m4v/manifest.f4m
```

To play using an Apple iOS device (Cupertino/Apple HTTP Live Streaming)

```
http://[instance-public-domain]/vods3/_definst_/
mp4:mybucket/videos/coolvideos/mycoolvideo.m4v/playlist.m3u8
```

To play using Microsoft Silverlight (Smooth Streaming)

```
http://[instance-public-domain]/vods3/_definst_/
mp4:mybucket/videos/coolvideos/mycoolvideo.m4v/Manifest
```

To play using RTSP/RTP player or device

```
rtsp://[instance-public-domain]/vods3/_definst_/
mp4:mybucket/videos/coolvideos/mycoolvideo.m4v
```

We have a publicly available sample MP4 file you can use for vods3 testing at the following stream name:

```
mp4:amazons3/wowzamediacache/sample/sample.mp4
```

To play using Adobe Flash player (RTMP):

Server: rtmp://[instance-public-domain]/vods3
Stream: mp4:amazons3/wowzamediacache/sample/sample.mp4

To play using Adobe Flash player (RTMP single URL):

```
rtmp://[instance-public-domain]/vods3/_definst_  
mp4:amazons3/wowzamediacache/sample/sample.mp4
```

To play using Adobe Flash player (San Jose/Flash HTTP)

```
http://[instance-public-domain]/vods3/_definst_  
mp4:amazons3/wowzamediacache/sample/sample.mp4/manifest.f4m
```

To play using an Apple iOS device (Cupertino/Apple HTTP Live Streaming)

```
http://[instance-public-domain]/vods3/_definst_  
mp4:amazons3/wowzamediacache/sample/sample.mp4/playlist.m3u8
```

To play using Microsoft Silverlight (Smooth Streaming)

```
http://[instance-public-domain]/vods3/_definst_  
mp4:amazons3/wowzamediacache/sample/sample.mp4/Manifest
```

To play using RTSP/RTP player or device

```
http://[instance-public-domain]/vods3/_definst_  
mp4:amazons3/wowzamediacache/sample/sample.mp4
```

Startup Package Reference

This section describes in detail each of the commands that can appear in a startup manifest file (startup.xml). The three commands are; **<Install>**, **<Download>**, **<RunScript>**.

Command **<Install>**

The **<Install>** command will copy the contents of a folder that is contained within a startup package into the Wowza Server installation folder **/usr/local/WowzaMediaServer**.

```
<Install>  
  <Folder>[relative-directory-path]</Folder>  
</Install>
```

Element **<Install>/<Folder>**

The **<Folder>** element specifies a folder within the startup package that will be copied into the Wowza Server installation folder. The directory structure of the folder should mirror that of the Wowza Media Server 3 installation folder. The directory specified is relative to the root of the startup package.

For example, if you have a startup package with the following structure:

```
[startup-package]
  startup.xml
  [wowza]
    [applications]
      [myapp]
    [conf]
      Server.license
      [myapp]
        Application.xml
  [tuning]
    tune.sh
    MediaCache.xml
    setenv.sh
```

And the contents of **startup.xml** is:

```
<Startup>
  <Commands>
    <Install>
      <Folder>wowza</Folder>
    </Install>
    <RunScript>
      <Script>tuning/tune.sh</Script>
    </RunScript>
  </Commands>
</Startup>
```

The contents of the **wowza** folder will be copied into the **/usr/local/WowzaMediaServer** folder of the running Wowza Server instance. The result in this case will be the creation of the application **myapp** and this application will use the **Application.xml** configuration file **wowza/conf/myapp/Application.xml**.

Note: Startup packages must be 16kB or less in size once compressed into a zip file. The **<Install>** method above works great if you have a small number of configuration files. If you have a more extensive configuration that pushes the size of the startup package beyond the 16kB then use the **<Download>** command described in the next section.

Command **<Download>**

The **<Download>** command will download content from a web server and save it to the local Amazon instance. The **<Download>** command includes the following elements:

```
<Download>
  <URL>[URL]</URL>
  <Data>[data]</Data>
  <Header><Name>[key-name]</Name><Value>[value]</Value></Header>
  <Header><Name>[key-name]</Name><Value>[value]</Value></Header>
  <Destination>[relative-or-absolute-file-path]</Destination>
  <Action>[UNZIP, INSTALL]</Action>
</Download>
```

The only two required elements are **<URL>** and **<Destination>**. To download a file from the url **http://www.mycompany.com/myfile.zip**, save it to the local machine at the location **/opt/myfile.zip** and unzip the file after download, the command is:

```
<Download>
  <URL>http://www.mycompany.com/myfile.zip</URL>
  <Destination>/opt/myfile.zip</Destination>
  <Action>UNZIP</ Action >
</Download>
```

When completed, the contents of the zip archive are located at **/opt/myfile**.

One use of the **<Download>** command is to work around the 16kB startup package size limitation. For example, if you have a more extensive configuration or you need to add several **.jar** files into the Wowza Server **lib** folder and these files push your startup package size over the 16kB limit, you might package these files into a separate zip archive. You can then host this zip archive on a web server and use the **<Download>** command to install the files into the Wowza Server **lib** folder.

For this example let's say we have two **.jar** files **wms-plugin-modulea.jar** and **wms-plugin-moduleb.jar** and you wish to setup two applications **live** and **vod**. Let's also assume you are using the lickey licensing option so you need to supply your own **Server.license** file. First, create the following directory structure:

```
[wowzmodules]
  [applications]
    [live]
    [vod]
  [conf]
    Server.license
    [live]
      Application.xml
    [vod]
      Application.xml
  [lib]
    wms-plugin-modulea.jar
    wms-plugin-moduleb.jar
```

Next, zip up the **[wowzmodules]** folder into a zip archive named **wowzmodules.zip** and copy it to your company's web server. Let's assume this file is now available at **http://www.mycompany/modules/wowzmodules.zip**. The **<Download>** command to install this package into the Wowza Server lib folder is:

```
<Download>
  <URL>http://www.mycompany/modules/wowzmodules.zip</URL>
  <Destination>/opt/wowzmodules.zip</Destination>
  <Action>INSTALL</Action>
</Download>
```

Element **<Download>/<URL>**

The **<URL>** is the URL of the file to be downloaded. The download can be performed over SSL by starting the url with **https://** rather than **http://**. The URL can also contain query parameters. The file will be downloaded using the GET method unless **<Data>** is specified.

Element <Download>/<Data>

The **<Data>** element is text data that will be included as part of the body of the HTTP request. You can use post data to send user name and password information to your web server so you can protect your content.

Element <Download>/<Header>: <Name> and <Value>

The **<Header>** elements are name value pairs added to the header part of the HTTP request. An example would be:

```
<Header>
  <Name>Content-type</Name>
  <Value>text/plain</Value>
</Header>
```

You can use **<Header>** data to protect your content. For example, you can use the header values to specify a username and password using BASIC authentication:

```
<Header>
  <Name>Authorization</Name>
  <Value>Basic dXNlcm5hbWU6cGFzc3dvcmQ= </Value>
</Header>
```

Element <Download>/<Destination>

The **<Destination>** element is the path to which the file will be saved (including the filename). This path can be relative or absolute. The base directory when calculating a relative file path, is the root directory of the startup package (the folder that contains the startup.xml file).

Element <Download>/<Action>

The **<Action>** element is the action performed after the file is downloaded. The action can either be **UNZIP** or **INSTALL**. If the action is **UNZIP** the downloaded file will be unzipped using the unzip command. If the action is **INSTALL** the downloaded file will be unzipped and the contents of the folder will be installed (copied) into the Wowza Server installation folder **/usr/local/WowzaMediaServer**.

Command <RunScript>

The **<RunScript>** command will execute a script on a running Amazon instance.

```
<RunScript>
  <Script>[relative-or-absolute-file-path]</Script>
  <Param>[parameter]</Param>
  <Param>[parameter]</Param>
</RunScript>
```

Element <RunScript>/<Script>

The <Script> element is the path to the script file to be executed. This path can be relative or absolute. The base directory when calculating a relative file path, is the root directory of the startup package (the folder that contains the startup.xml file).

Element <RunScript>/<Param>

The <Param> elements are parameters that will be passed to the running script. For example the following <RunScript> command:

```
<RunScript>
  <Script>scripts/copyfile.sh</Script>
  <Param>filea.txt</Param>
  <Param>fileb.txt</Param>
</RunScript>
```

Would be the equivalent of executing the command:

```
./scripts/copyfile.sh filea.txt fileb.txt
```

Before a script is executed, the startup processor initializes several environment variables with information that describes the current Amazon instance. These variables are:

AWSEC2_METADATA_INSTANCE_ID	- Amazon instance id
AWSEC2_METADATA_SECURITY_GROUPS	- Security group
AWSEC2_METADATA_LOCAL_IPV4	- Local IP address
AWSEC2_METADATA_AMI_LAUNCH_INDEX	- Launch index
AWSEC2_METADATA_PUBLIC_HOSTNAME	- Public host name
AWSEC2_METADATA_PRODUCT_CODES	- DevPay product code
AWSEC2_METADATA_INSTANCE_TYPE	- instance type (m1-small, m1-large, m1-xlarge)
AWSEC2_METADATA_HOSTNAME	- Public host name
AWSEC2_METADATA_LOCAL_HOSTNAME	- Local host name
AWSEC2_METADATA_PUBLIC_IPV4	- Public IP address
AWSEC2_METADATA_AMI_MANIFEST_PATH	- S3 manifest path
AWSEC2_METADATA_RESERVATION_ID	- Instance reservation ID
AWSEC2_METADATA_AMI_ID	- AMI ID
AWSEC2_METADATA_AVAILABILITY_ZONE	- Availability zone
AWSEC2_METADATA_PUBLIC_KEYS	- Public keys
AWSEC2_METADATA_ANCESTOR_AMI_IDS	- Ancestor AMI IDs
AWSEC2_METADATA_RAMDISK_ID	- RAM disk ID
AWSEC2_METADATA_BLOCK_DEVICE_MAPPING	- Block device mapping
AWSEC2_METADATA_KERNEL_ID	- Kernel ID

Startup Package Debugging

The best way to debug a startup package is to just try to launch an AMI using it then look in the startup log file to look for errors or warnings. The startup log file is written to instance in the following file:

```
/usr/local/WowzaMediaServer/logs/wowzamediaserver_startup.log
```

The log is quite extensive and should provide ample information to help debug startup package issues.

Additional Resources

Wowza Media Systems Amazon EC2 Support page:

<http://www.wowza.com/ec2support.html>

Amazon Web Services: Articles & Tutorials:

<http://aws.amazon.com/articles>

Amazon Web Services: Resources:

<http://aws.amazon.com/resources>

Amazon Web Services: Discussion Forums:

<https://forums.aws.amazon.com/index.jspa>

Amazon EC2 Overview:

<http://aws.amazon.com/ec2>

Starting Amazon EC2 with Mac OS X (from Robert Sosinski):

<http://www.robertsosinski.com/2008/01/26/starting-amazon-ec2-with-mac-os-x/>