
Amazon DevPay

Developer Guide

Version 2007-12-01



Amazon Web Services

Amazon DevPay: Developer Guide

Amazon Web Services

Copyright © 2013 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

The following are trademarks of Amazon Web Services, Inc.: Amazon, Amazon Web Services Design, AWS, Amazon CloudFront, Cloudfront, Amazon DevPay, DynamoDB, ElastiCache, Amazon EC2, Amazon Elastic Compute Cloud, Amazon Glacier, Kindle, Kindle Fire, AWS Marketplace Design, Mechanical Turk, Amazon Redshift, Amazon Route 53, Amazon S3, Amazon VPC. In addition, Amazon.com graphics, logos, page headers, button icons, scripts, and service names are trademarks, or trade dress of Amazon in the U.S. and/or other countries. Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon.

All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Welcome	1
Introduction to Amazon DevPay	3
What Is Amazon DevPay?	4
What Is Amazon Payments?	4
Ways to Use Amazon DevPay	6
Architectural Overview of Amazon DevPay	6
Customer Experience	8
Customer Access Stored Data	11
Using Amazon DevPay	13
The Business Aspects of DevPay	14
Your Amazon Payments Account	15
Your DevPay Activity Page	17
How Do You Set Your Product's Price?	19
DevPay Fees	20
The Cost of AWS Services	24
Price Components	29
Basic Pricing Components	29
Allowed Usage-Based Charges	30
Tiered Usage-Based Pricing	31
Taxes	36
When and How You Get Paid	37
Sign-Up Payments	37
Monthly Payments	38
If a Customer Doesn't Pay the Monthly Bill	42
E-mails You Receive	46
Customer Billing	48
About Customer Accounts	49
When Customers Are Billed	50
Where Customers Get Information About Their Bills	51
Customer Cancellation	53
When Customers Resubscribe	54
Where Customers Manage the Payment Method	55
When Customers Don't Pay	56
E-mail Contact with Your Customers	56
Customer Refunds and Chargebacks	57
Customer Support	59
Registering Your Product	60
Making Changes to Your Product	63
Changing Pricing	64
Reports	69
Subscription Report	69
Revenue Report	71
Usage Report	76
Recommendations for Product Development	80
Testing and Going Live	82
Advertising Your Product	84
Using DevPay with Your Amazon EC2 AMI	85
Overview of DevPay with Amazon EC2	86
Your Product's Configuration and Price	88
The Product Code and AMI Rebundling	89
If You Have Multiple AMIs to Sell	90
Changing a Product's Configuration	92
Basic Paid AMI Tasks	94
Frequently Asked Questions	97
Web Site Development	98
Verification of a Customer's Subscription Status	99
Supported AMIs	102
Additional Details	103

Using DevPay with Your Amazon S3 Product	104
Overview of DevPay with Amazon S3	105
Development with Amazon S3	107
Quick Reference for Amazon S3 Products	108
Setting Up Desktop Products	112
Prerequisites	112
Overall Authentication Process	112
Desktop Product Activation	114
Making Amazon S3 REST Calls with Desktop Products	117
Query String Authentication with Desktop Products	118
Amazon S3 POST with Desktop Products	118
Desktop Product Exceptions	119
Verification of the Customer's Subscription Status	120
Setting Up Web Products	123
Prerequisites	123
Overall Authentication Process	123
Web Product Activation	125
Making Amazon S3 REST Calls with Web Products	128
Query String Authentication with Web Products	130
Amazon S3 POST with Web Products	132
Web Product Exceptions	135
Verification of the Customer's Subscription Status	135
Using Amazon S3 Copy with DevPay	137
Using Amazon S3 Logs with DevPay	138
Using Amazon S3 Requester Pays with DevPay	142
Using Both a Desktop and Web Product	148
License Service API Reference	149
WSDL Location	149
Requests	150
REST-Query Requests	150
SOAP Requests	151
Request Authentication	153
About Authentication	153
Authentication of REST-Query Requests	155
Authentication of SOAP Requests	162
X.509 Certificates	163
Example Request to Use When Troubleshooting	164
Responses	167
Errors	169
Actions	172
ActivateDesktopProduct	173
ActivateHostedProduct	177
GetActiveSubscriptionsByPid	182
RefreshUserToken	185
VerifyProductSubscriptionByPid	190
VerifyProductSubscriptionByTokens	194
Appendices	198
Appendix: The Customer Purchase Experience	199
The Product Purchase Flow	200
The Application Activation Page	203
The Application Billing Page	204
Appendix: E-mails Sent to Your Customers	206
E-mails Related to Sign-Up or Cancellation	207
E-mails Related to Billing	209
E-mails Related to Price Changes	213
Appendix: Product Registration Flow	217
Amazon Payments Account Page	218
Product Information Page	219

Which Amazon Web Services Page	220
Pricing Overview Page	222
Usage Charges Page	223
One-Time Charge and Monthly Charges Page	227
Review Pricing Page	228
Your Description Page	229
Confirmation Page	230
Preview Order Page	232
Thank You Page	236
Product Details Page	237
Appendix: Example DevPay Activity Pages	239
June	240
July	251
August	259
Appendix: Risks to DevPay Products	261
Amazon DevPay Resources	262
Document History	264
Document Conventions	265
Index	268

Welcome

Amazon DevPay is an easy-to-use online billing and account management service that makes it easy for you to sell an Amazon Elastic Compute Cloud AMI or an application built on the Amazon Simple Storage Service.

This is the *Amazon DevPay Developer Guide*.

For a description of what's new in this release of the *Amazon DevPay Developer Guide*, see [Document History \(p. 264\)](#).

How Do I... ?

How do I?	Relevant Sections
Learn about this changes made to this document since the last release.	<ul style="list-style-type: none">• Document History (p. 264)
Learn the basics of Amazon DevPay	<ul style="list-style-type: none">• Introduction to Amazon DevPay (p. 3)

How do I?	Relevant Sections
Understand the business aspects of Amazon DevPay	<ul style="list-style-type: none">• Your Amazon Payments Account (p. 15)• Your DevPay Activity Page (p. 17)• How Do You Set Your Product's Price? (p. 19)• DevPay Fees (p. 20)• The Cost of AWS Services (p. 24)• Price Components (p. 29)• When and How You Get Paid (p. 37)• Customer Billing (p. 48)• Customer Refunds and Chargebacks (p. 57)• Customer Support (p. 59)• Registering Your Product (p. 60)• Making Changes to Your Product (p. 63)• Changing Pricing (p. 64)• Reports (p. 69)• Recommendations for Product Development (p. 80)• Testing and Going Live (p. 82)• Advertising Your Product (p. 84)
Learn how to use Amazon DevPay with an Amazon EC2 AMI	<ul style="list-style-type: none">• Using DevPay with Your Amazon EC2 AMI (p. 85)
Learn how to use Amazon DevPay with an Amazon S3 desktop product	<ul style="list-style-type: none">• Using DevPay with Your Amazon S3 Product (p. 104)• Setting Up Desktop Products (p. 112)• License Service API Reference (p. 149)
Learn how to use Amazon DevPay with an Amazon S3 web product	<ul style="list-style-type: none">• Using DevPay with Your Amazon S3 Product (p. 104)• Setting Up Web Products (p. 123)• License Service API Reference (p. 149)

Introduction to Amazon DevPay

Topics

- [What Is Amazon DevPay? \(p. 4\)](#)
- [What Is Amazon Payments? \(p. 4\)](#)
- [Ways to Use Amazon DevPay \(p. 6\)](#)
- [Architectural Overview of Amazon DevPay \(p. 6\)](#)
- [Customer Experience \(p. 8\)](#)
- [Customer Access Stored Data \(p. 11\)](#)

This section gives you an introduction to Amazon DevPay.

What Is Amazon DevPay?

Amazon DevPay is a billing and account management service that enables you to get paid for products you build on either Amazon EC2 or Amazon Simple Storage Service. Amazon DevPay creates and manages the order pipeline and billing system for you. Your customers sign up for your product, and we automatically meter their usage of the underlying AWS service, bill them based on the pricing you set, and collect their payments. What else is special about DevPay?

- You can charge customers for your product; the charges can include recurring charges based on the customer's usage of AWS services (Amazon EC2 or Amazon S3), a fixed one-time charge, and a recurring monthly charge.
- Your customers can easily sign up and pay for your product with their trusted Amazon.com accounts; they do not need to sign up for an AWS developer account.
- Your customers are authenticated, thus ensuring they have access only to what they should.
- If your customers don't pay their bills, DevPay turns off their access to your product for you.
- Amazon Payments handles payment processing.



Basic DevPay Flow

1	Your customer uses an Amazon.com account to sign up and pay for your product. The sign-up page indicates that you have teamed up with Amazon Payments to make billing easy and secure.
2	Your customer pays the price you've defined to use your product.
3	DevPay subtracts a fixed transaction fee and pays you the difference.
4	You pay the costs of the AWS services your product consumed, and a percentage-based DevPay fee.

What Is Amazon Payments?

Amazon Payments is the payment processor Amazon DevPay uses to process your customers' payments and disburse money to you. For more information about Amazon Payments, go to <http://payments.amazon.com>. The following sections give you an overview of Amazon Payments and how it's related to Amazon DevPay.

Amazon Payments and You

To use Amazon DevPay, you need an Amazon Payments account (specifically, an *Amazon Payments Business Account*). The money you make from your DevPay customers is deposited into this account. There are a variety of ways to use Amazon Payments, and they're covered on the Amazon Payments

web site (<http://payments.amazon.com>). However, for the purposes of DevPay, you primarily just need to know that your customers' payments go into your Amazon Payments account. For information about how you get this account and access it, see [Your Amazon Payments Account \(p. 15\)](#).

Amazon Payments and Your Customers

Amazon Payments processes your DevPay customers' payments. The only Amazon branding that your customers see in relation to your DevPay product is for Amazon Payments.

Note

The one exception to this is for customers who purchase Amazon EC2 AMI products; they also see AWS branding because they must be signed up for an Amazon EC2 account.

Your site should make it clear that Amazon Payments is handling the billing for your product. Your customers will see the Amazon Payments logo on the pages showing the usage and billing data for your product. We require you to include the Amazon Payments trademark or logo on your site to help make your customers' experience consistent between your site and our site. The trademark, logo, and guidelines for using them with Amazon DevPay are available in the [co-marketing area of the Developer Connection web site](#).

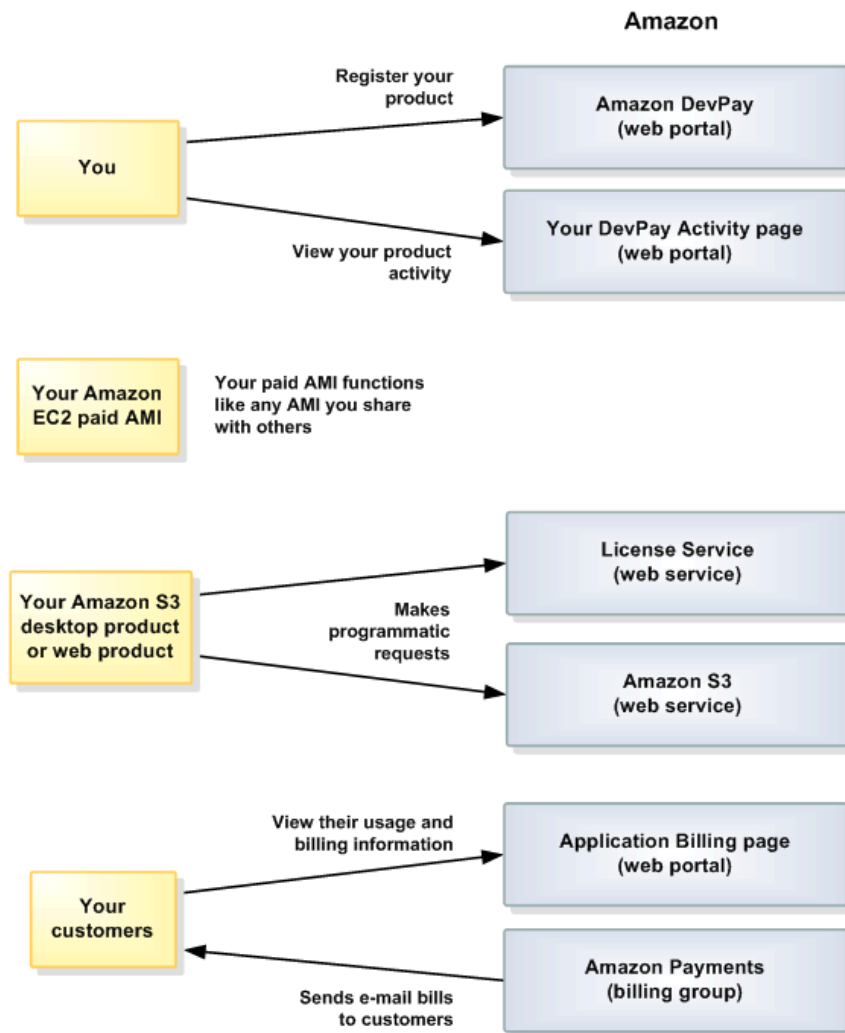
Ways to Use Amazon DevPay

You can use Amazon DevPay these ways:

- With an Amazon EC2 AMI (for more information, see [Using DevPay with Your Amazon EC2 AMI \(p. 85\)](#))
- With a desktop or hosted (web) product that uses Amazon Simple Storage Service (for more information, see [Using DevPay with Your Amazon S3 Product \(p. 104\)](#))

Architectural Overview of Amazon DevPay

The following figure and discussion explain the major components involved with Amazon DevPay.



You, your product, and your customers interact with different parts of Amazon.

You:

- Register your product with DevPay using a web portal
- View your product activity and manage your product and funds using your *DevPay Activity* page

Your Amazon S3 product:

- Makes call to the License Service to obtain required credentials for the customer
- Makes calls to Amazon S3

Your Amazon EC2 paid AMI:

- Functions like any AMI you share with others

Your customers:

- View their usage and billing information on their *Application Billing* page
- Receive billing e-mails from Amazon Payments

Customer Experience

Your customers' experience of your product can be divided into three categories:

- Their experience at sign-up
- Their general experience at any time when using the product
- Their experience monthly when they are billed

Important

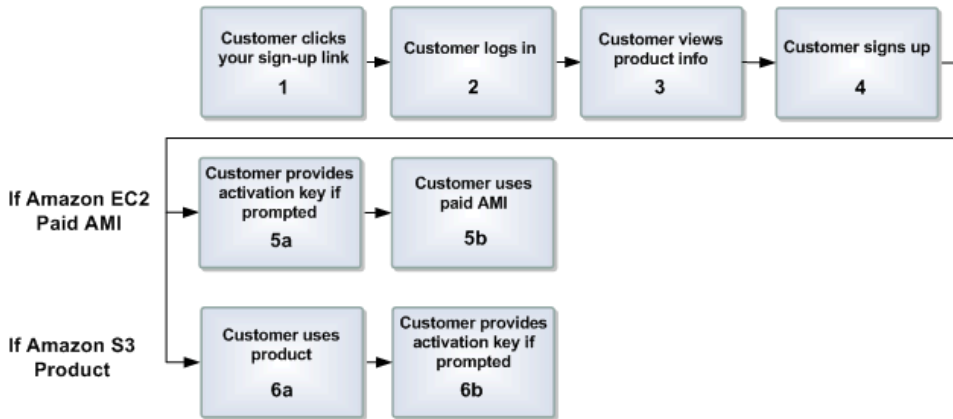
When customers buy your product and are later billed for using it, they deal entirely with *Amazon Payments*, and not *Amazon DevPay*. Your customers receive no direct exposure to Amazon DevPay. Therefore, you don't need to point your customers to the Amazon DevPay documentation or information for any reason.

At Sign Up

At the time of sign-up, there are two aspects of the customer's experience: the sign-up process and billing confirmation (if your product has up-front charges).

Sign-Up Process

The following figure and table describe the customers' experience during sign-up.



Process for Signing Up

1	The customer clicks the sign-up link for your product and is sent to a specific sign-up page provided by DevPay and branded with Amazon Payments. For example pages of this flow, see Appendix: The Customer Purchase Experience (p. 199) .
2	The customer is prompted to log in using an Amazon.com login.
3	The customer views your product's information and pricing.
4	The customer signs up for your product. After the product purchase, the confirmation page displays an activation key. The customers are redirected to your web site (to a URL you specify to DevPay when you register your product) for instructions on using the product.

5	<p>If the product is an Amazon EC2 paid AMI:</p> <ol style="list-style-type: none">Depending on how you implement your system, your system might need the activation key. You can prompt the customer for the activation key or get it programmatically as described later in this guide.The customer can then use the paid AMI.
6	<p>If the product is an Amazon S3 product:</p> <ol style="list-style-type: none">The customer downloads and installs your desktop product or uses your web product.Your Amazon S3 product requires the activation key. The product prompts the customer for the key (for desktop products) or gets the key programmatically (for web products) and then uses the key to activate itself. Once activated, the product is ready to use at any time. For additional information about how customers use your product, see Customer Access Stored Data (p. 11).

Billing Confirmation

If your product has a one-time sign-up fee, a monthly fee, or both, your customers are billed accordingly at the time of sign-up. They each receive an e-mail from Amazon Payments confirming the sign-up charge (for more information, see [When the Customer Signs Up to Use Your Product \(p. 207\)](#)).

At Any Time

At any time, customers can use the product and also view their usage and billing information. The following figure and table describe the process for the latter.



Process for Viewing Usage and Billing Information

1	The customer goes to your web site and clicks the link you provide for managing the product.
2	The customer is prompted to log in using an Amazon.com login.
3	The customer views the Application Billing page (for more information, see Where Customers Get Information About Their Bills (p. 51)).

Customers can cancel at any time. For more information, see [Customer Cancellation \(p. 53\)](#).

You might decide to change the price of your product at any time. For more information about how this affects customers, see [Your Customers' Experience of a Price Change \(p. 67\)](#).

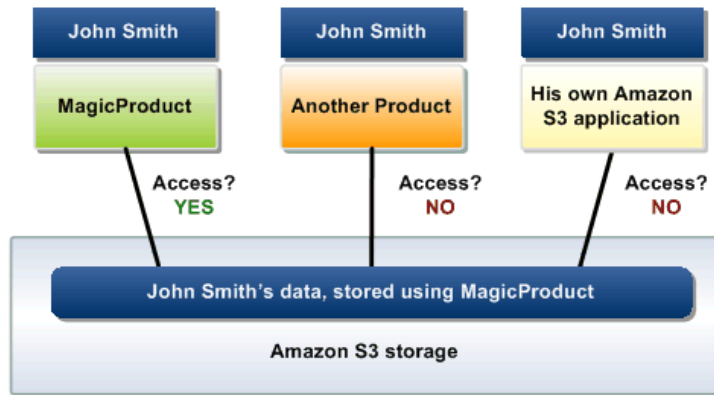
Monthly

Customers are billed monthly (for more information, see [When Customers Are Billed \(p. 50\)](#)) and receive monthly e-mails from Amazon Payments showing the billing amount (for more information, see [When the Customer Is Billed Each Month \(p. 209\)](#)).

Customer Access Stored Data

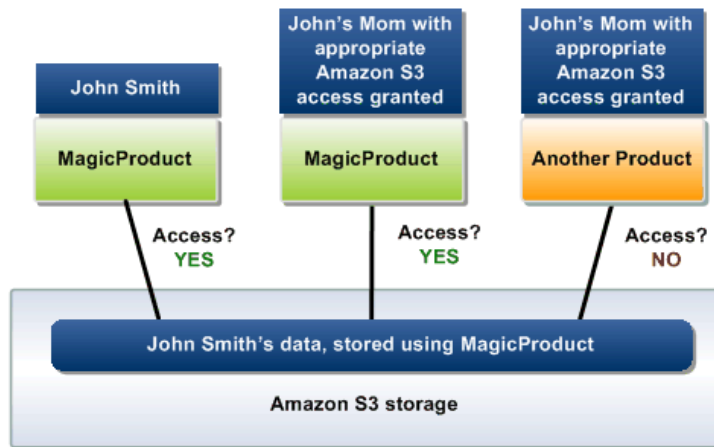
When customers use your Amazon S3 product, they might end up storing data on Amazon servers. For example, you could have a product that stores customers' data in Amazon S3 for them. This section discusses restrictions on access to that data.

Amazon DevPay implements a product-based access control system that is automatically applicable to your product; you don't have to do anything to implement it. Let's say you have a product called MagicProduct that uses Amazon S3 and stores images for customers. If John Smith uses MagicProduct to store his images, he can access those stored images only by using MagicProduct. He can't use some other product that also uses Amazon S3, and he can't get his own AWS developer account and use Amazon S3 directly to access the images.



Although DevPay enforces a product-based access control restriction, you can still use the access control mechanisms implemented by the AWS service your product uses. For example, you can still use the Amazon S3 access control mechanism with your product. The mechanism's operation is independent of the DevPay product-based restriction.

If we go back to our previous example: You could design MagicProduct to use the Amazon S3 access control mechanism so that John Smith could grant his mother access to the images he stored. His mother could then buy MagicProduct herself and access John's images. However, she couldn't access those images by using another product that uses Amazon S3. She also couldn't access those images by getting her own AWS account and using Amazon S3 directly.



Another important implication of the product-based restriction is that John Smith can no longer access his stored data when he cancels his use of MagicProduct. He can't use another product based on Amazon

S3 or use Amazon S3 directly to access his data. If he resubscribes to MagicProduct, his images might or might not be available depending on the Amazon S3 data retention policy.

Important

You should warn your customers that:

- They should retrieve their data before they cancel use of your product
- If they don't pay their bills for your product, Amazon will cancel their access to the product and their data will be unavailable

Using Amazon DevPay

Topics

- [The Business Aspects of DevPay \(p. 14\)](#)
- [Using DevPay with Your Amazon EC2 AMI \(p. 85\)](#)
- [Using DevPay with Your Amazon S3 Product \(p. 104\)](#)

This section explains the business aspects of Amazon DevPay (such as product pricing, how you get paid, etc.), how to use DevPay with an Amazon EC2 AMI, and how to use DevPay with an Amazon S3 application.

The Business Aspects of DevPay

Topics

- [Your Amazon Payments Account \(p. 15\)](#)
- [Your DevPay Activity Page \(p. 17\)](#)
- [How Do You Set Your Product's Price? \(p. 19\)](#)
- [DevPay Fees \(p. 20\)](#)
- [The Cost of AWS Services \(p. 24\)](#)
- [Price Components \(p. 29\)](#)
- [When and How You Get Paid \(p. 37\)](#)
- [Customer Billing \(p. 48\)](#)
- [Customer Refunds and Chargebacks \(p. 57\)](#)
- [Customer Support \(p. 59\)](#)
- [Registering Your Product \(p. 60\)](#)
- [Making Changes to Your Product \(p. 63\)](#)
- [Changing Pricing \(p. 64\)](#)
- [Reports \(p. 69\)](#)
- [Recommendations for Product Development \(p. 80\)](#)
- [Testing and Going Live \(p. 82\)](#)
- [Advertising Your Product \(p. 84\)](#)

This section provides information about the business aspects of Amazon DevPay.

Your Amazon Payments Account

Topics

- [Getting Your Account](#) (p. 15)
- [Adding and Verifying Your Bank Account](#) (p. 15)
- [Your Amazon Payments Balance](#) (p. 16)
- [Your Transaction History](#) (p. 16)

This section discusses your Amazon Payments account, and how you get and prepare it for use with Amazon DevPay. Your Amazon Payments account is the account that receives the payments from your DevPay customers. For more information about Amazon Payments, see [What Is Amazon Payments?](#) (p. 4).

Getting Your Account

Each time you register a product with DevPay, we check to see if the login you used to register the product has an Amazon Payments Business Account associated with it. If it doesn't, we create one for that login during product registration. If the login has some other type of Amazon Payments account associated with it, we upgrade that account to an Amazon Payments Business Account during product registration. Upgrading is a simple process that requires you to answer a few questions about your business.

If we create a new Amazon Payments account for you, Amazon Payments automatically sends you an e-mail message about the account. Before you can use the account or DevPay, you must click the link Amazon Payments provides inside that e-mail to verify your e-mail address.

To verify your e-mail address

1. Check for a message that Amazon Payments has sent to the login e-mail address.
2. Click the link in the e-mail message.

That's all you need to do. Now you can go to Amazon Payments at <http://payments.amazon.com> to access your account. Note that you will receive regular e-mails directly from Amazon Payments about the account.

Adding and Verifying Your Bank Account

This section describes how to add a bank account to your Amazon Payments account and then verify that bank account. You're not required to do this to use DevPay. It's only required if one or both of the following apply:

- You want to withdraw money from your Amazon Payments account
- You expect to receive more than a total of \$10,000 per month from all your DevPay customers (your account has a limit on it until you verify the bank account)

Note that the bank account verification process can take up to 7 days.

Note

If you already had an Amazon Payments account before you started using DevPay, then you might already have a verified bank account associated with your Amazon Payments account.

To add and verify your bank account

1. Go to Amazon Payments at <http://payments.amazon.com>.
2. Log in to your account and click **Add a Bank Account**.

3. Follow the instructions presented to you to add a bank account to your Amazon Payments account.
4. Follow the instructions presented to you to verify the bank account.

For more information about any hold periods or limits that might apply to your Amazon Payments account, refer to the Amazon Payments site.

Your Amazon Payments Balance

You can get your Amazon Payments account balance by going to either Amazon Payments at <http://payments.amazon.com> or your DevPay Activity page at <http://aws.amazon.com/devpayactivity>. For information about what you can do at your DevPay Activity page, see [Your DevPay Activity Page \(p. 17\)](#).

Two different balances are displayed:

- Your total Amazon Payments balance (which includes credit or debit transactions that are not yet completed)
- The amount available for immediate withdrawal

You can withdraw money from your account by going to either Amazon Payments or your Amazon DevPay Activity page. For more information about withdrawing money, see [Your DevPay Activity Page \(p. 17\)](#). Keep in mind that before you can withdraw money from your Amazon Payments account, you must have a verified bank account associated with it (for more information, see [Adding and Verifying Your Bank Account \(p. 15\)](#)).

Your Transaction History

Along with the account balance, Amazon Payments also displays your account's transaction history, which is a list of all the individual transactions involving the account (whether they're related to DevPay or not).

Your Amazon DevPay Activity page has a *DevPay transaction history*, which is different from your Amazon Payments transaction history. The DevPay transaction history includes only the transactions related to your DevPay products, and each day's charges or deposits are rolled up into one charge or deposit per day. For example, if you have ten different deposits from DevPay customers on a given day, the DevPay transaction history shows a single deposit that is the sum of those ten. If you want to see the individual transaction information for each of the ten deposits, you can find it in the Amazon Payments transaction history. For an example of the page and the type of entries you might see in the DevPay transaction history, see [Appendix: Example DevPay Activity Pages \(p. 239\)](#).

The Amazon Payments balance displayed might not reconcile perfectly with the DevPay transaction history. This can be for two reasons:

- You can use the account for reasons other than DevPay
- Some of the transactions in the DevPay transaction history can involve your credit card

Why would your credit card be involved? When we charge you each month (as discussed in subsequent sections), we first try to collect the money from your Amazon Payments account. If the balance is insufficient, we charge the remaining amount to your credit card. The charge to your credit card is included in the DevPay transaction history, but it doesn't affect your Amazon Payments balance.

Note

When we charge you each month, we send you an e-mail. The e-mail doesn't state which account or accounts we charged. We also don't inform you if the Amazon Payments account balance was insufficient. We do, however, contact you if we're unable to charge your credit card.

Your DevPay Activity Page

Your DevPay Activity page is where you manage your DevPay products. It's located at <http://aws.amazon.com/devpayactivity>. When you access the page, you're prompted to log in with the AWS developer login that you used to register the product with DevPay.

From the DevPay Activity page you can:

- View the month-to-date usage of all your DevPay products, the corresponding revenue you expect to receive (assuming your customers pay their bills at the end of the month), and the actual revenue you have received
- View revenue statements for previous months (see the following procedure)
- View your Amazon Payments balance and DevPay transaction history (see the following procedure)
- Withdraw money into your own bank account (see the following procedure)
- View or change DevPay product details (for more information, see [Changing Product Information \(p. 63\)](#))
- Change the pricing of a DevPay product (for more information, see [Changing Pricing \(p. 64\)](#))
- Cancel your product's use of DevPay (for more information, see [Canceling Your Product's Use of DevPay \(p. 63\)](#))

Note

The dates displayed on the DevPay Activity page are in Coordinated Universal Time (Greenwich Mean Time).

Note

During the month, the DevPay Activity page reflects the current month's data up to and through the previous day. Therefore, on the first day of the month, the page has no data to display yet and so shows zeros.

To help you better understand the DevPay Activity page and the contents of other related pages, we've created an example. It shows what the pages would display over a few months when you introduce an imaginary DevPay product. For more information, see [Appendix: Example DevPay Activity Pages \(p. 239\)](#).

To view a previous month's revenue statement

1. Go to <http://aws.amazon.com/devpayactivity>.
2. When prompted, log in with the AWS developer login you used to register your product with DevPay. The **DevPay Activity** page is displayed. By default, it shows your current month-to-date usage and revenue.
3. Select a month from the pull-down menu at the top of the page. The current month-to-date usage and revenue numbers are replaced with the numbers from the month you selected.

To view your Amazon Payments balance and DevPay transaction history

1. Go to <http://aws.amazon.com/devpayactivity>.
2. When prompted, log in with the AWS developer login you used to register your product with DevPay. The **DevPay Activity** page is displayed.
3. At the bottom of the page, click **View DevPay Transaction History**. The **DevPay Transaction History** page is displayed, showing your transaction history and Amazon Payments balance. To understand the amounts that are displayed on this page, see the example discussed in [Appendix: Example DevPay Activity Pages \(p. 239\)](#).

Note

The DevPay transaction history is updated as deposits or withdrawals are made during the day. However, the DevPay Activity page is updated once each day. Because of this, the DevPay transaction history might show entries that are not yet reflected in the DevPay Activity page.

To withdraw money from your Amazon Payments account

1. Go to your **DevPay Transaction History** page as described in the preceding procedure.
2. Click **Withdraw to bank account** (or if you have not yet specified a bank account to use, follow the instructions presented on the page).

Related Topics

- [What Is Amazon Payments? \(p. 4\)](#)
- [Your Amazon Payments Account \(p. 15\)](#)

How Do You Set Your Product's Price?

DevPay allows you to create a product based on an AWS service (Amazon S3 or Amazon EC2) and charge your customers to use that product. What's the appropriate price to charge for your product?

The price you charge is your decision, but we recommend you understand the following before setting the price:

- The fees to use DevPay (for more information, see [DevPay Fees \(p. 20\)](#))
- Your customers' usage and the corresponding cost of the AWS service that your product is based on (for more information, see [The Cost of AWS Services \(p. 24\)](#))
- The price components available to you (for more information, see [Price Components \(p. 29\)](#))
- When and how you get paid (for more information, see [When and How You Get Paid \(p. 37\)](#))

Once you understand the preceding items, you should then think about your product and the price you'd like to use. When you register your product with DevPay, you must provide the final price for the product. If at a later date you want to change the product's price, follow the process to change your price. For more information, see [Changing Pricing \(p. 64\)](#).

Related Topics

- [Registering Your Product \(p. 60\)](#)

DevPay Fees

Topics

- [Quick Summary of the Fees \(p. 20\)](#)
- [Basic Concepts \(p. 20\)](#)
- [When Your Value-Add Is Greater Than or Equal to Zero \(p. 20\)](#)
- [When Your Value-Add Is Less Than Zero \(p. 21\)](#)
- [Examples \(p. 22\)](#)

This section describes the Amazon DevPay fees and the situations in which they are applicable.

Quick Summary of the Fees

You pay:

- 3% of your value-add per customer
- \$0.30 per DevPay product for each customer bill we collect

Note

Your value-add per customer is the amount you charge each customer on top of the cost of the AWS services they used.

When we calculate the 3% fee amount, we use the value-add amount *before* any \$0.30 DevPay fees have been subtracted.

The following sections give more details and examples of the fees.

Basic Concepts

To describe how the fees work, we'll start by describing some basic concepts.

You create a DevPay product and set a price for it. Your customers buy and use the product. At the end of each month, we determine three amounts *for each customer*:

- **Customer revenue**—The total amount your customer owes for using the product that month
- **AWS costs**—The costs of AWS services the customer used that month
- **Your value-add**—The difference between the customer revenue and the AWS costs

Your customer revenue is based on the price you set for your product. This includes any one-time sign-up charge, monthly charge, and usage-based charges.

What you owe AWS for each customer each month varies if your value-add is greater than or equal to zero, or less than zero.

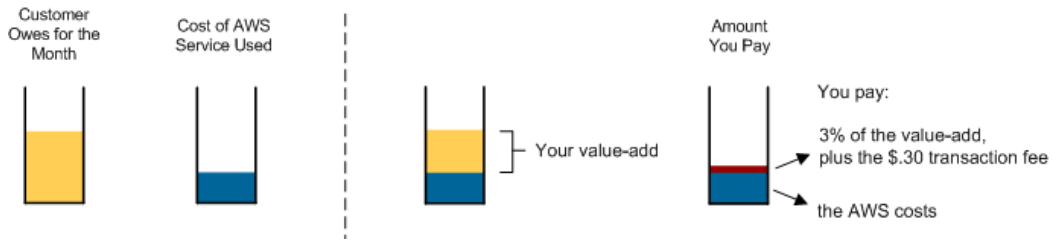
When Your Value-Add Is Greater Than or Equal to Zero

When the revenue expected from the customer is greater than or equal to the AWS costs, and the customer actually pays the money, you pay AWS the items shown in the following list and illustration:

- AWS costs
- DevPay fee: 3% of your value-add
- DevPay fee: a \$0.30 fee per product for each customer bill we collect

Note

If you have multiple products, you are assessed \$0.30 for each product the customer subscribes to.

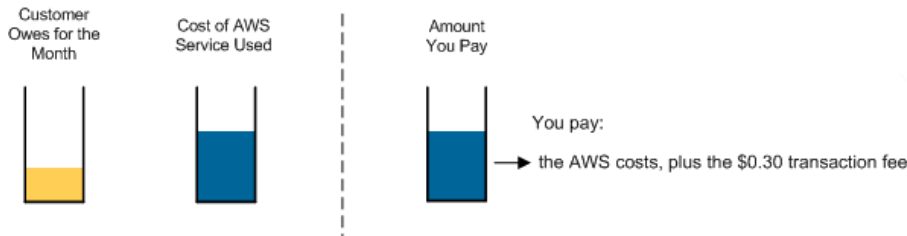


If the customer doesn't pay, you pay AWS nothing. If the customer pays part of what's owed for the month (for example, if the customer pays the monthly charge, but doesn't pay the usage-based charges), then you pay the AWS costs and 3% DevPay fee in proportion to how much the customer has paid. You still pay the \$0.30 transaction fee. For more information, see [If a Customer Doesn't Pay the Monthly Bill \(p. 42\)](#).

When Your Value-Add Is Less Than Zero

Although it will be unlikely, there might be instances when the expected revenue from the customer is less than the AWS costs. If this occurs, and the customer actually pays, you pay AWS the items shown in the following list and illustration:

- AWS costs
- DevPay fee: a \$0.30 fee per product per customer bill collected



If the customer doesn't pay what's owed (and so you receive no revenue from that customer for the month), you pay no DevPay fees. You pay only the AWS costs not covered by the expected revenue amount (as shown in the following illustration). This is equivalent to the absolute value of your value-add amount.

If the customer pays part of what's owed, then you pay the AWS costs in proportion to how much the customer has paid. You still pay the \$0.30 transaction fee. For more information, see [If a Customer Doesn't Pay the Monthly Bill \(p. 42\)](#).



Examples

In the following examples, you have an Amazon EC2 AMI that you're selling through DevPay. A customer uses the AMI during the month of April. The product's pricing scheme includes *only* usage charges; it has no one-time or monthly recurring charges. AWS charges for Amazon EC2 based on the costs described in [Allowed Usage-Based Charges \(p. 30\)](#). However, for simplicity, the examples reflect only some of those possible costs and assume use of only small instances of the AMI.

Note

The Amazon EC2 prices shown in the following examples were valid when this document was originally written. The prices might have changed since then. For the current Amazon EC2 prices, go to [the Amazon EC2 product page](#).

The top table in each example shows your product's price and the corresponding AWS cost for Amazon EC2.

The bottom table shows your customer's usage amounts during April, the corresponding price the customer pays, the corresponding Amazon EC2 cost, and your corresponding value-add.

The equation at the bottom shows the total DevPay fees you owe for that customer for the month of April.

Example DevPay Fees When Your Value-Add Is Greater Than or Equal to Zero

In this example, your price results in a value-add that is greater than zero (the revenue your customer generates in the month is greater than the costs of the AWS service used). In this situation, you pay both the 3% fee and the \$0.30 transaction fee. Note that if your value-add is zero, your 3% DevPay fee is zero.

Price Component	Your Price	Amazon EC2 Cost
Usage Charges	\$0.25 per small instance-hour used	\$0.10
	\$0.30 per GB data transferred in	\$0.10
	\$0.25 per GB data transferred out	\$0.17 (maximum)
One-time Charge	None	N/A
Monthly Charge	None	N/A

EC2 Dimension	Customer's Use in April		
	Usage	Customer Pays	Cost of EC2
Small instance-hours used	50 Hours	\$12.50	\$5.00
GB data transferred in	20 GB	\$6.00	\$2.00
GB data transferred out	10 GB	\$2.50	\$1.70
		\$21.00	\$8.70

→ Value-add = \$21.00 - \$8.70 = \$12.30

3% DevPay fee on value-add	\$ 0.37
Transaction fee	\$ 0.30
Total DevPay fees you owe	\$ 0.67

Example DevPay Fees When Your Value-Add Is Less than Zero

In this example, your price is lower than in the preceding example and results in a value-add that is less than zero (the revenue your customer generates in the month is *less* than the costs of the AWS service used). In this situation, we don't charge the 3% fee. However, we still charge you the \$0.30 transaction fee when we collect from the customer.

Price Component	Your Price	Amazon EC2 Cost
Usage Charges	\$0.10 per small instance-hour used	\$0.10
	\$0.10 per GB data transferred in	\$0.10
	\$0.15 per GB data transferred out	\$0.17 (maximum)
One-time Charge	None	N/A
Monthly Charge	None	N/A

EC2 Dimension	Customer's Use in April		
	Usage	Customer Pays	Cost of EC2
Small instance-hours used	50 Hours	\$5.00	\$5.00
GB data transferred in	20 GB	\$2.00	\$2.00
GB data transferred out	10 GB	\$1.50	\$1.70
		\$8.50	\$8.70

→ Value-add = \$8.50 - \$8.70 = - \$0.20

3% DevPay fee on value-add	N/A
Transaction fee	\$ 0.30
Total DevPay fees you owe	\$ 0.30

Related Topics

- [When and How You Get Paid \(p. 37\)](#)

The Cost of AWS Services

Topics

- [The Basics \(p. 24\)](#)
- [The Cost of Tiered Usage Types \(p. 24\)](#)
- [The Cost of Regional Bandwidth \(p. 27\)](#)
- [Costs You're Not Responsible For \(p. 27\)](#)

This section gives information about the money you're responsible for paying to AWS to cover the cost of AWS services your customers use (e.g., Amazon S3 or Amazon EC2). The first section covers the basics, and the subsequent sections give details about how we handle particular AWS service costs.

The Basics

At the end of each month, you owe AWS the costs of the AWS service (e.g., Amazon S3 or Amazon EC2) each customer used during the month. The next sections cover some important details to understand.

You Pay Only What the Customer Has Already Covered

We charge you for the costs of AWS services *only after the customer pays you first*. This way, you can use your customers' payments to help cover the costs they incur. Also, the amount we collect never exceeds what the customer has actually paid you. In general, this means that if a customer doesn't pay you, we don't charge you for the costs of the AWS services the customer used. For more information about how this works in real situations involving monthly fees, see [If a Customer Doesn't Pay the Monthly Bill \(p. 42\)](#).

You Pay Any Costs Not Covered by Expected Revenue

Typically the revenue a customer generates in a given month will be *greater* than the cost of the AWS services the customer used (this assumes you've priced your product so that you regularly make money and don't lose money). However, depending on how you price your product, you might have situations where the revenue a particular customer generates is *less* than the corresponding cost of the AWS services used. In this case, *you're responsible for the difference, regardless of whether the customer pays*. For example, if your customer owes you \$10, but has used \$15 worth of Amazon S3, you're responsible for paying AWS the \$15. But, if the customer doesn't pay the \$10, then you're responsible only for the \$5 difference.

The Cost of Tiered Usage Types

This section describes how we charge you for usage types that have a tiered cost structure, such as data transfer out and Amazon S3 storage. The section also describes how we determine your value-add *per customer* given that the usage type has a tiered cost structure applicable across *all* your customers of a particular DevPay product.

Note

Although this section focuses specifically on one particular usage type (data transfer out), the concepts apply to any usage type that has a tiered cost structure.

How We Calculate Your Costs

The AWS services have tiered costs for data transfer out (\$0.17 per GB for the first 10 TB/month data downloaded, \$0.13 per GB for the next 40 TB/month, \$0.11 per GB for the next 100 TB/month, and \$0.10 per GB for any additional bandwidth beyond 150 TB).

When we charge you for tiered data transfer out costs, we apply the tiered discount on a *per-product* basis. In other words, if you have more than one DevPay product, we don't sum the data transfer out costs across *all* your DevPay products and then apply the tiered discount. Instead, we sum the costs used by *all* the customers using *one* of your particular DevPay products during the month and then apply the tiered discount to that particular product's data transfer out accordingly. We then repeat that for each DevPay product you have.

For example, if during April, your customers download 8 TB for Product A, and 4 TB for Product B, you are charged \$0.17 per GB for Product A's 8 TB and \$0.17 per GB for Product B's 4 TB; you aren't charged as if you used a total of 12 TB (\$0.17 per GB for 10 TB and then \$0.13 per GB for 2 TB).

Note that you can charge your customers for data transfer out and use a tiered pricing structure like AWS does. For information about setting tiered pricing for your product's usage-based price components, see [Tiered Usage-Based Pricing \(p. 31\)](#).

How We Calculate Your Value-Add

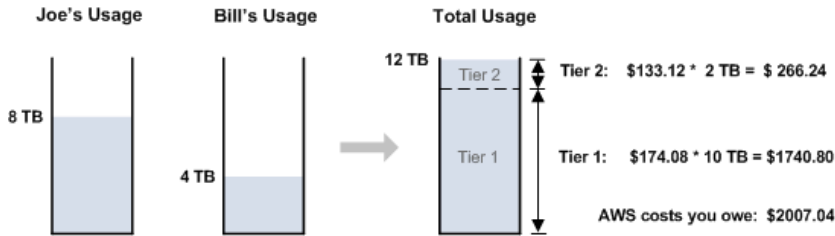
Each month, we charge you 3% of your *value-add per customer* (for more information, see [DevPay Fees \(p. 20\)](#)). Given that we charge you for data transfer out based on the sum total of *all* your customers' use of the product, how do we calculate your value-add *per customer*?

We first calculate the *cost-per-unit* of data transfer out across all your customers and then apply that value to determine each individual customer's cost for data transfer out. The value-add for a particular customer is then the difference between the amount the customer pays you for data transfer out (based on your product's price) and the calculated cost for data transfer out for that customer. Following is a simple (although unlikely) example that illustrates the point.

Example Value-Add Calculation

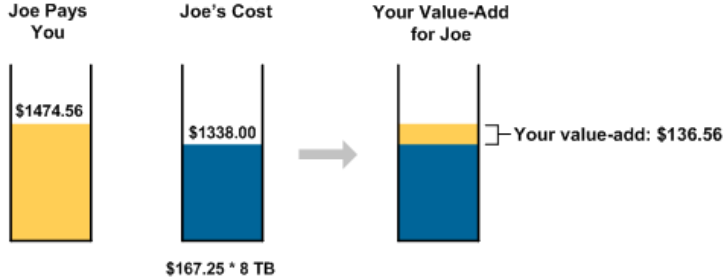
In this example, you have only two customers: Joe and Bill. Joe uses 8 TB of data transfer out during the month, and Bill uses 4 TB (for a total of 12 TB used).

In turn, AWS charges you \$0.17 per GB for the first 10 TB of data transfer out used, and \$0.13 per GB for the next 40 TB used. This translates into \$174.08 per TB for the first 10 TB, and \$133.12 per TB for the next 40 TB (remember that 1 TB = 1024 GB). This means for the 12 TB Joe and Bill have used, the AWS costs you owe are: $(\$174.08 * 10 \text{ TB}) + (\$133.12 * 2 \text{ TB}) = \$1740.80 + \$266.24 = \2007.04 (as shown in the following figure).

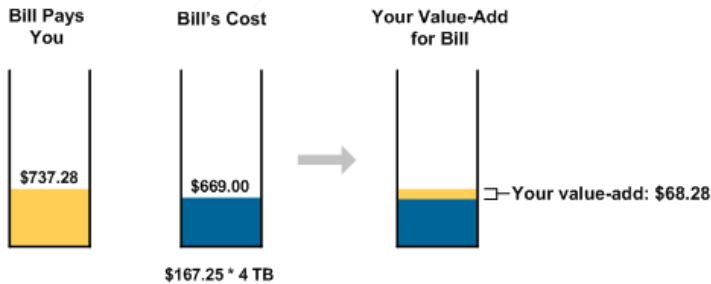


The cost-per-unit of data transfer out for the month is therefore $\$2007.04 / 12 \text{ TB} = \167.25 per TB.

You charge your customers \$0.18 per GB data transfer out (or \$184.32 per TB). Therefore, Joe pays you $\$184.32 * 8 \text{ TB} = \1474.56 , as shown in the following figure. Your value-add for Joe is the price he pays you, minus the per-unit cost of data transfer out multiplied by his usage: $\$1474.56 - (\$167.25 * 8 \text{ TB}) = \$1474.56 - \$1338.00 = \136.56 .



Bill pays you $\$184.32 * 4 \text{ TB} = \737.28 , as shown in the following figure. Your value-add for Bill is $\$737.28 - (\$167.25 * 4 \text{ TB}) = \$737.28 - \$669.00 = \68.28 .



The Cost of Regional Bandwidth

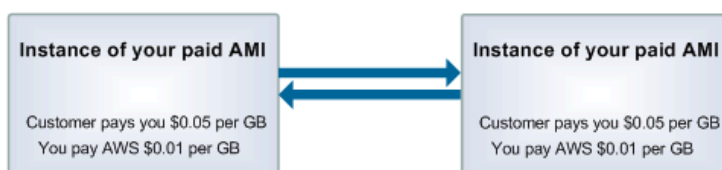
One of the costs of using Amazon EC2 is regional data transfer. You can charge your customers for their use of regional data transfer by including it in the price of your paid and supported AMIs.

Regional data transfer covers two items:

- Data transferred in and out between instances in different availability zones in the same region (for information about availability zones, go to the *Amazon Elastic Compute Cloud Developer Guide*)
- Data transferred in and out between instances inside the Amazon EC2 network using public IP addresses or elastic IP addresses, even if the instances are in the same availability zone

AWS charges for both regional data transfer *in* and regional data transfer *out* of Amazon EC2 instances. This means that if both instances involved in the transfer are instances of your paid AMI, the customer pays you for your regional data transfer price at *both* ends of the data transfer. You also pay the corresponding AWS cost for regional data transfer at both ends of the data transfer.

For example, let's say AWS charges \$0.01 per GB for regional data transfer, and you charge your customers \$0.05 per GB. The following figure shows the price the customer pays you and the corresponding cost you pay AWS for data transfer in and out of the instances.



By contrast, what if the transfer is between an instance of your paid AMI and an instance of the customer's personal AMI? The following figure shows this situation.



Costs You're Not Responsible For

You're responsible for paying the costs of the AWS services your customers use. There are several exceptions to this:

- Amazon EC2 elastic IP addresses
- Amazon Elastic Block Store (Amazon EBS)
- Elastic Load Balancing
- Auto Scaling
- Amazon CloudWatch

Important

Currently you can't use Elastic Load Balancing (either by itself or in conjunction with Auto Scaling) with instances of paid or supported AMIs.

You're not responsible for the AWS costs your customers incur for these features, and you can't include charges for the features in the price of your DevPay product. Why not?

You can't because AWS bills Amazon EC2 users directly for their use of the features. We do this because customers can use the features with any instances they launch (or that Auto Scaling launches on their behalf). That usage is associated with the customer's Amazon EC2 account, not the instance itself or the owner of the AMI. The instance can be of your paid AMI, someone else's paid AMI, or the customer's own AMI. Because of this separation, you can't charge for the features in your product's price. For elastic IP addresses, this applies to the charges for having—but not using—elastic IP addresses and for remapping elastic IP addresses beyond the number of free mappings allowed each month. For all of the other features, this applies to all charges related to the feature's use. For a list of the usage-based charges you can include in your product's price, see [Allowed Usage-Based Charges \(p. 30\)](#).

Related Topics

- [Price Components \(p. 29\)](#)
- [When and How You Get Paid \(p. 37\)](#)

Price Components

Topics

- [Basic Pricing Components \(p. 29\)](#)
- [Allowed Usage-Based Charges \(p. 30\)](#)
- [Tiered Usage-Based Pricing \(p. 31\)](#)
- [Taxes \(p. 36\)](#)

This section describes the different components you can use as part of your product's price.

Basic Pricing Components

Your product's price can include three optional parts listed in the following table.

Price Component	Description
One-time charge	This is a fixed, one-time charge assessed when your customer signs up to use your product. If a customer cancels use of your product and then later resubscribes, we bill that customer the one-time charge again.
Monthly recurring charge	This is a fixed, monthly charge for use of your product. For information about how this is billed, see When Customers Are Billed (p. 50) . If you include a monthly charge that is small, you should be aware of the implications. For more information, see When You Have a Small Monthly Charge (p. 38) .
Usage-based charges	These are charges based on the usage dimensions of the AWS service itself. For a list of the usage-based charges for Amazon EC2 and Amazon S3 and whether you can charge your customers for each of them, see Allowed Usage-Based Charges (p. 30) . For an example of a price scheme that includes usage-based charges, see DevPay Fees (p. 20) . For information about using a tiered pricing structure for any of the usage-based charges, see Tiered Usage-Based Pricing (p. 31) .

You can include all, some, or none of the preceding price components when you set your product's price. For example, you could:

- Include only usage-based charges, so the amount your customers pay is based solely on how much they use the AWS service

Note

You can include one, some, or all of the possible usage-based charges for the particular AWS service. For example, if your product uses Amazon S3, you could charge your customers for GB-Month data stored only. Or, you could also include other charges such as the bandwidth for data uploaded to Amazon S3. Which of the usage-based charges you include in your product's price is your choice. Regardless, you are responsible for the cost of your customers' usage for all the dimensions.

- Include only fixed charges (a monthly charge, a one-time charge, or both), so that the amount your customers pay is not related to how much they use the AWS service
- Use a combination of usage-based charges and fixed charges
- Charge nothing for your product and be entirely responsible for all costs and DevPay fees

Allowed Usage-Based Charges

You can include usage-based charges in your product's price. However, not all of the available usage-based charges can be part of your product's price. The following tables show each of the usage-based costs AWS charges for and whether you can include a corresponding charge in the price of your DevPay product. For more information about the usage-based costs, go to the [Amazon EC2 detail page](#) and the [Amazon S3 detail page](#).

Note

The list of usage-based costs changes as Amazon EC2 and Amazon S3 release new features. The pricing examples, fee examples, and DevPay Activity page examples presented in this guide might not reflect all the currently available usage-based costs the services have. However, the concepts presented by the examples are still valid.

Amazon EC2

Important

Currently you can't use Elastic Load Balancing (either by itself or in conjunction with Auto Scaling) with instances of paid or supported AMIs.

Usage-based Cost	Can You Include in Your DevPay Product's Price?
Instance-hours: all types/sizes, environments, and regions that Amazon EC2 makes available	Yes (you can charge a different price for each instance type, environment, region, etc.; for more information, see Your Product's Configuration and Price (p. 88)).
Data transfer in	Yes
Data transfer out (the cost is tiered according to the usage amount; for information about how you are charged for tiered data transfer out, see The Cost of Tiered Usage Types (p. 24))	Yes
Regional data transfer (for more information about regional data transfer, see The Cost of Regional Bandwidth (p. 27))	Yes
Elastic IP addresses: cost for non-use of an elastic IP address	No (for information about why you can't charge for it, see Costs You're Not Responsible For (p. 27))
Elastic IP addresses: cost for additional re-mappings above the number allowed each month	No
Amazon Elastic Block Store: cost for storage and requests for Amazon EBS volumes	No (for information about why you can't charge for it, see Costs You're Not Responsible For (p. 27))
Amazon Elastic Block Store: cost for storage and requests for Amazon EBS snapshots	No
Auto Scaling	No (for information about why you can't charge for it, see Costs You're Not Responsible For (p. 27))
Amazon CloudWatch	No (for information about why you can't charge for it, see Costs You're Not Responsible For (p. 27))

Amazon S3

Usage-Based Cost	Can You Include in Your DevPay Product's Price?
Storage: all types (the cost is tiered according to the usage amount; for information about how you are charged for tiered storage, see The Cost of Tiered Usage Types (p. 24))	Yes
Data transfer in	Yes
Data transfer out (the cost is tiered according to the usage amount; for information about how you are charged for tiered data transfer out, see The Cost of Tiered Usage Types (p. 24))	Yes
Requests: PUT or LIST	Yes
Requests: GET and all other requests (no charge for DELETE requests)	Yes

Related Topics

- [Tiered Usage-Based Pricing \(p. 31\)](#)

Tiered Usage-Based Pricing

Topics

- [Monthly Free Usage Model \(p. 31\)](#)
- [Tiered AWS Pricing \(p. 32\)](#)
- [Cell Phone Pricing Model \(p. 32\)](#)
- [Price Changes with Tiered Pricing \(p. 33\)](#)
- [Cancellation and Resubscription in the Same Month \(p. 36\)](#)

You can specify an unlimited number of tiers for any of the usage-based components of your product's price. With tiered usage-based pricing, you can charge different rates according to how much your customers use. For example, if you have a product that uses Amazon S3, you could set your price so that customers pay \$1 per GB-Month for the first 10 GB-Month they use, and they pay \$0.90 per GB-Month for all usage above 10 GB-Month. This section discusses some of the supported pricing models and how a price change works with tiered pricing.

Note

Even if you set tiered usage-based pricing for your product, the amount you pay AWS for your customers' usage of AWS services does not change based on their usage. Exception: usage types with a tiered cost structure (for more information, see [The Cost of Tiered Usage Types \(p. 24\)](#)).

Monthly Free Usage Model

With this model, you create a pricing plan that lets you give your customers free use of a particular usage dimension each month, up to a limit you specify. This means you include no monthly or sign-up fee, and you price the first X units of the particular usage dimension at \$0 per unit. Your customers still must

provide a credit card number when they sign up for your product. Once they exceed the initial free usage limit during the month, they are charged the price you've set for usage above that limit.

Important

You still must pay the associated AWS costs for your customers' monthly free usage.

For example, perhaps you have an Amazon S3 product and you want to give customers 5 GB-Month free each month and then charge \$3 per GB-Month for storage above that amount. The following table shows how you would set the tiers.

Price Component		Your Price
Monthly fee		\$0
Sign-up fee		\$0
Tiers	1. Per GB-Month of storage used (first 5 GB-Month)	\$0 per GB-Month
	2. Per GB-Month of storage used (over 5 GB-Month)	\$3 per GB-Month

Note

This model doesn't support time-based free usage (for example, where the first *month* of usage is free).

A customer who signs up on the last day of the month still gets the entire amount of free usage for that first month. For example, if the customer signs up on June 30, the customer gets the free X units of free usage in June, and then gets X units of free usage again in July.

Tiered AWS Pricing

AWS uses tiered pricing for some usage types (for example, the data transfer out for Amazon S3 and Amazon EC2). We charge a certain amount for the first X TB, then a lower amount for the next Y TB, and so on.

You might want to pass on some of the AWS tiering discounts to your customers. With tiered usage-based pricing, you have the option to set your product's price to follow the AWS pricing tiers.

Important

How you can charge your customers for their use tiered usage types is different from how AWS charges you for your customers' use of them. You charge each customer based on their individual usage with your DevPay product, whereas AWS charges you based on the total usage across *all* the customers of your DevPay product. For more information, see [The Cost of Tiered Usage Types](#) (p. 24).

Cell Phone Pricing Model

You can use tiered usage-based pricing to create a pricing plan that is similar to common cell phone pricing plans. You have a fixed monthly charge, and with that comes a certain amount of free usage. For example, perhaps you have an Amazon S3 product and you want to charge a \$10 monthly fee that comes with 5 GB-Month storage. Anything above the first 5 GB-Month costs \$1 per GB-Month. Therefore, you would set the monthly fee to \$10 and set the first tier's price at \$0 per GB-Month and its boundary at 5 GB-Month. You would then set the second tier's price at \$1 per GB-Month for anything over 5 GB-Month. The following table shows how you would set the tiers.

Price Component	Your Price
Monthly fee	\$10

Price Component		Your Price
Sign-up fee		\$10
Tiers	1. Per GB-Month of storage used (first 5 GB-Month)	\$0 per GB-Month
	2. Per GB-Month of storage used (over 5 GB-Month)	\$1 per GB-Month

Important

At sign-up, we prorate the monthly fee based on when the customer signs up during the month. Also, if a customer cancels, we refund any unused portion of the monthly fee. Be aware that if a customer signs up near the end of the month, the prorated amount might be small, but the customer can still use the first tier for free (the 5 GB-Month of storage in the preceding example). We recommend you include a one-time sign-up charge (which is non-refundable) to protect against the loss that could result if a customer signed up late in the month, quickly used the first tier for free, and then canceled use of the product before the end of the month.

You're not limited to two tiers. For example, you could still have the \$10 monthly fee and the first tier set the same way as in the preceding example. But instead you could set a boundary for the second tier at 10 GB-Month, and then have a third tier where you charge \$0.75 per GB-Month for any usage above 10 GB-Month. The following table shows how you would set the tiers.

Price Component		Your Price
Monthly fee		\$10
Sign-up fee		\$10
Tiers	1. Per GB-Month of storage used (first 5 GB-Month)	\$0 per GB-Month
	2. Per GB-Month of storage used (over 5 GB-Month)	\$1 per GB-Month
	3. Per GB-Month of storage used (over 10 GB-Month)	\$0.75 per GB-Month

Note

This model doesn't support a usage limit. For example, you can't set the monthly fee at \$10 and then have AWS limit your customers to using only 20 GB-Month. Also, this model doesn't support tiered *monthly* pricing based on usage. For example, you can't charge \$10 per month if the customer uses 5 GB-Month storage, and then \$20 per month if the customer goes above 5 GB-Month or reaches some higher limit.

Price Changes with Tiered Pricing

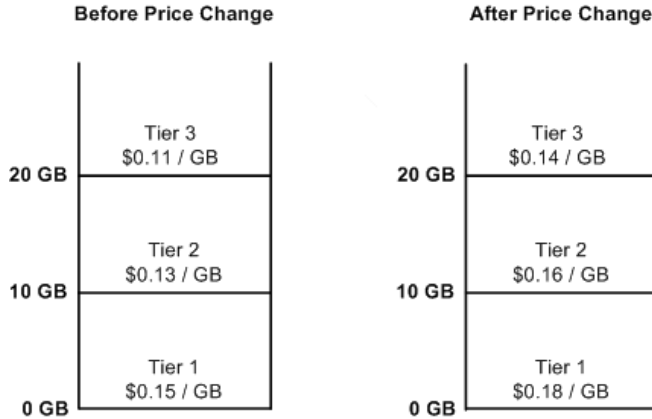
When you do a price change for your product, you could change the prices for the existing tiers, or you could change the number of tiers. The following example shows how we charge a customer after you change the prices for the existing tiers. For information about changing your product's price, see [Changing Pricing \(p. 64\)](#).

Note

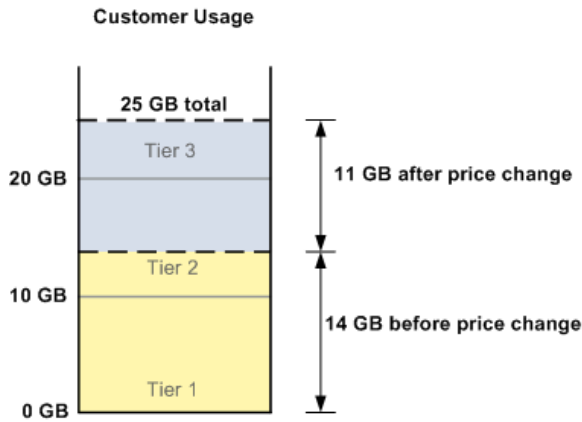
Regardless of whether you change the pricing for the tiers or change the number of tiers, the costs you pay AWS are not affected by your price change.

In this example, you have a product that has tiered pricing for data transferred in. The boundaries between tiers occur at 10 GB and 20 GB of data transferred in. The following figure shows the tiered pricing before and after the price change.

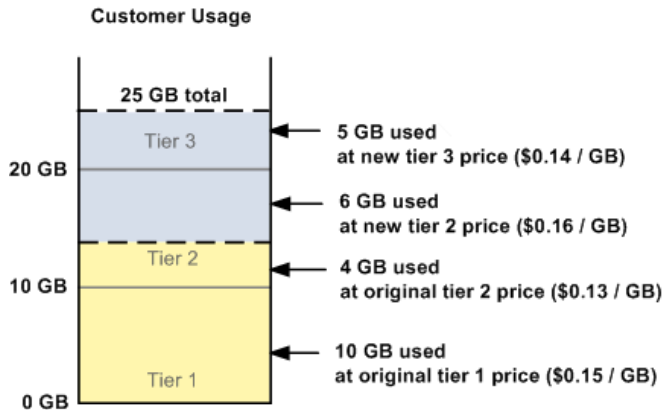
**Amazon DevPay Developer Guide
Price Components**



A customer uses 25 GB total: 14 GB *before* the price change, and 11 GB *after* the price change, as shown in the following image.



How do we charge the customer for the usage, given the tiered pricing structure and the price change? The following figure illustrate this.

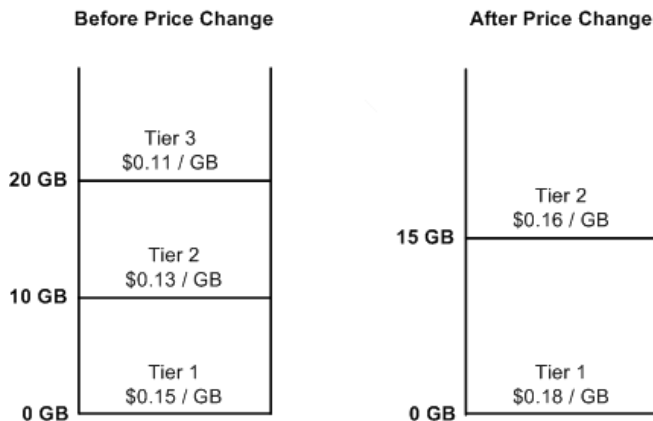


Amazon DevPay Developer Guide Price Components

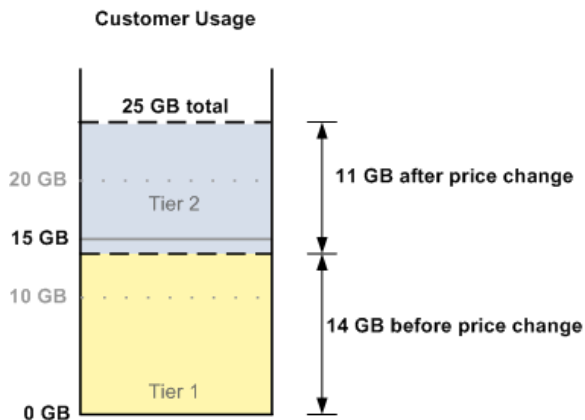
We charge the original tier 1 price (\$0.15 per GB) for the initial 10 GB transferred in and the original tier 2 price (\$0.13 per GB) for the next 4 GB. This covers the 14 GB used before the price change. After the price change, we continue with the tiered measurement, but at the new price. So we charge the new second tier price (\$0.16 per GB) for 6 GB (to make the 20 GB total for the second tier), and then the new third tier price (\$0.14 per GB) for the remaining 5 GB.

The customer's billing statement for the month shows usage before the price change separately from usage after the price change. For the period before the price change, the statement shows 10 GB of first-tier usage and 4 GB of second-tier usage. For the period after the price change, the statement only shows 6 GB of second-tier usage, and 5 GB of third-tier usage. Nothing is shown for the first tier.

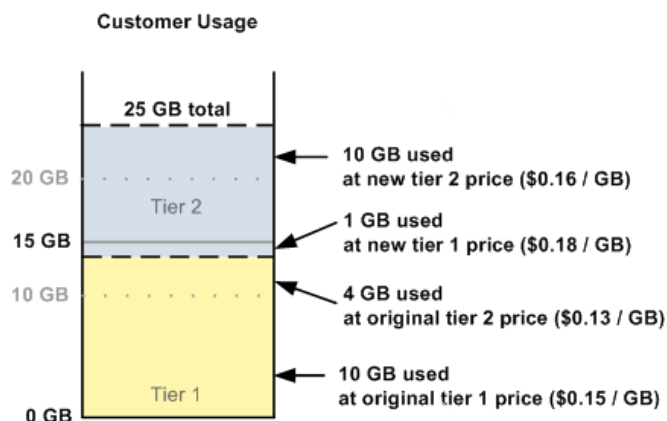
What if your price change involves changing the number of tiers? We charge according to the old pricing scheme until the date of the change. Thereafter, we continue measuring usage, but apply the new pricing as appropriate to the usage accrued after the price change occurred. For example, let's say you change the product from having three tiers for data transferred in to having two tiers, as shown in the following figure.



As in the preceding example, the customer uses 25 GB total: 14 GB *before* the price change, and 11 GB *after* the price change, as shown in the following image. The image also shows the original pricing scheme's tier boundaries at 10 GB and 20 GB.



The following figure shows how we charge the customer.



We charge the original tier 1 price (\$0.15 per GB) for the initial 10 GB transferred in and the original tier 2 price (\$0.13 per GB) for the next 4 GB. This covers the 14 GB used before the price change. With the old pricing scheme, the 14 GB of usage puts the customer in the second tier. But with the new pricing scheme, the customer is still in the first tier. So after the price change, we charge the new first tier price (\$0.18 per GB) for 1 GB (to make the 15 GB total for the new first tier boundary), and then the new second tier price (\$0.16 per GB) for the remaining 10 GB.

Cancellation and Resubscription in the Same Month

If a customer cancels use of your product and then resubscribes in the same month, we don't restart the customer's usage measurements after the resubscription. Instead, we continue the usage measurements where they left off when the customer canceled. Although we consider it a separate subscription (the customer must pay any one-time sign-up fee again, etc.), for the purposes of calculating the customer's usage for the month, we treat it as the same subscription.

Related Topics

- [Allowed Usage-Based Charges \(p. 30\)](#)
- [The Cost of AWS Services \(p. 24\)](#)
- [When and How You Get Paid \(p. 37\)](#)
- [Changing Pricing \(p. 64\)](#)

Taxes

Amazon does not provide tax collection services for you. You are responsible for paying the appropriate taxes (local, state, federal, etc.).

When and How You Get Paid

Topics

- [Sign-Up Payments \(p. 37\)](#)
- [Monthly Payments \(p. 38\)](#)
- [If a Customer Doesn't Pay the Monthly Bill \(p. 42\)](#)
- [E-mails You Receive \(p. 46\)](#)

This section explains when you get paid and the monthly billing cycle we use to pay you.

You can receive payments at two times:

- When a customer signs up (if you have a sign-up charge, a monthly charge, or both)
- At the beginning of the month, for any usage the previous month (if your product has usage-based charges) and for the current month's monthly charge (if your price includes a monthly charge)

Important

If you think you'll make more than a total of \$10,000 each month from all your customers, you should make sure you have a verified bank account associated with your Amazon Payments account as soon as possible. You also need a verified bank account if you want to withdraw money from your Amazon Payments account. For more information, see [Adding and Verifying Your Bank Account \(p. 15\)](#).

Sign-Up Payments

If your product has a monthly charge, a one-time charge, or both, you receive a payment from the customer immediately after the customer signs up for your product. The payment consists of the one-time charge and the prorated monthly charge (based on when the customer signs up during the month). We subtract the \$0.30 DevPay fee and deposit the remainder in your Amazon Payments account. For more information about the fees charged, see [DevPay Fees \(p. 20\)](#). For information about what happens if the customer's payment at sign-up is smaller than the \$0.30 DevPay fee we subtract, see [When You Have a Small Monthly Charge \(p. 38\)](#).

How do you know when you've received a customer's sign-up payment? It shows up on your DevPay transaction history, and it's listed in the DevPay reports. For more information, see [Your DevPay Activity Page \(p. 17\)](#) and [Reports \(p. 69\)](#).

Example Sign-Up Payment

Let's say that Joe signs up to use your product (called MyAMI) in the middle of the month (April 16). The pricing scheme for MyAMI includes usage charges, a one-time charge at sign-up, and a monthly charge. This example shows the charges when Joe signs up.

Note

AWS charges for Amazon EC2 based on the costs described in [Allowed Usage-Based Charges \(p. 30\)](#). However, for simplicity, this example reflects only some of those possible costs and assumes Joe only uses small instances of the AMI.

The following table shows the pricing for MyAMI.

Price Dimension		MyAMI Price
One-Time Sign-Up Charge		\$10.00
Monthly Charge		\$8.00
Usage Charges	per small instance-hour used	\$0.25
	per GB data transferred in	\$0.30
	per GB data transferred out	\$0.25

What are Joe's charges when he signs up to use MyAMI on April 16? He pays the sign-up charge of \$10.00, plus the prorated monthly charge, which is $0.5 * \$8.00 = \4.00 . Therefore his total bill when he signs up is \$14.00. When we receive the payment, we subtract the \$0.30 transaction fee and deposit the remainder (\$13.70) into your Amazon Payments account.

When You Have a Small Monthly Charge

If your product's monthly charge is too low, and a customer signs up on or near the last day of the month, the prorated amount the customer pays might not cover the \$0.30 transaction fee we subtract from the payment. We therefore charge you the difference.

For example, if your monthly fee is \$5.00, the prorated amount a customer owes when signing up on the last day of the month is around \$0.17 (depending on the number of days in that month). When we collect the \$0.17, we subtract \$0.30 from it, meaning that instead of receiving \$0.17, you owe \$0.13. One way to cover this deficit is to include a one-time fee that's large enough to ensure the customer's payment is always larger than the \$0.30 fee. Another way is to charge a monthly fee large enough to avoid the situation entirely.

If you don't include a one-time sign-up charge or a large enough monthly fee, and you owe small amounts on customer sign-ups, we don't automatically charge you the small amount when each customer signs up. Instead, we occasionally aggregate the small amounts from multiple customers and charge you the aggregated amount. The aggregated charge is listed in your DevPay transaction history as "Charge (Revenue collected minus \$0.30 fee per customer)." For an example of a DevPay transaction history page, see [Appendix: Example DevPay Activity Pages \(p. 239\)](#).

Monthly Payments

This section describes the process we use to pay you each month, and to collect the DevPay fees and the costs of AWS services that you owe.

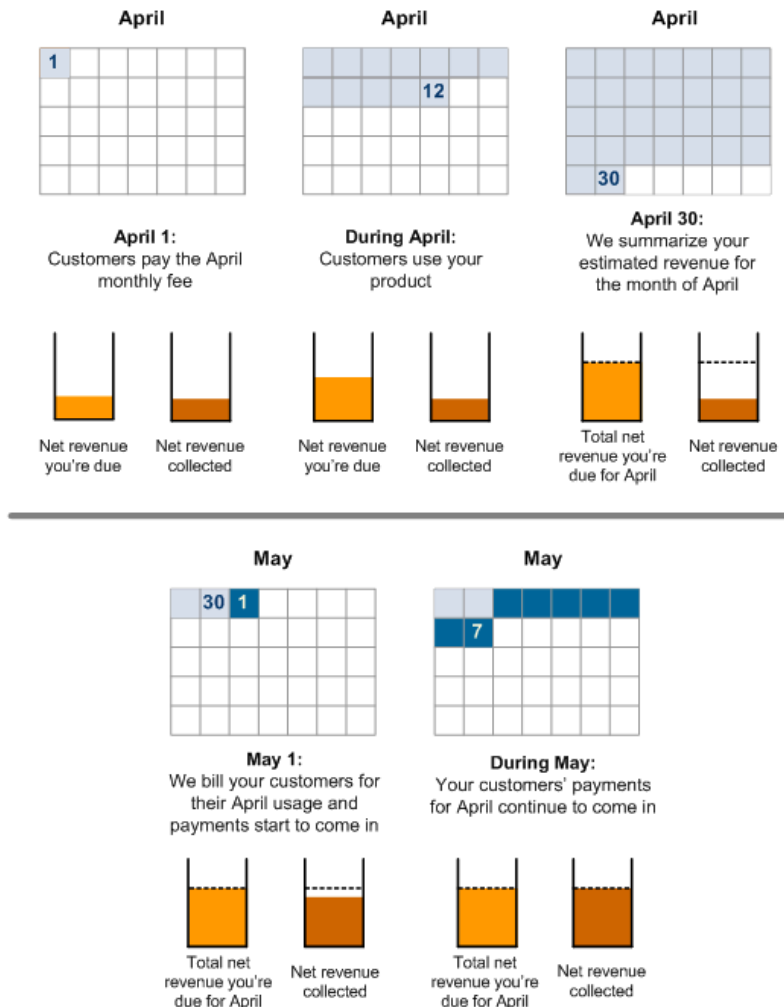
Overall Timing

After your customers sign up for your product, we bill them monthly on the first day of the month. On that day, we bill them for the previous month's usage and the current month's fee (assuming the product has both usage-based and monthly charges). For example, on May 1, we bill them for their usage in April and for the monthly fee to cover May.

If you think of this from the perspective of the *revenue for a particular month* (e.g., April), then you can see that you receive your April revenue at two main times (April 1 and May 1). At any point during April, you can view your *DevPay Activity Page* and see how much revenue we've actually collected in April monthly fees to date, and how much April revenue you *expect* to receive after we bill your customers for their April usage on May 1 (for more information about using the page, see [Your DevPay Activity Page \(p. 17\)](#)).

At the end of April, we summarize the data for that month and display it in the April revenue statement (for an example, see [Appendix: Example DevPay Activity Pages \(p. 239\)](#)). During May and thereafter, you can view the April revenue statement and see how much of the April usage-based revenue we've actually been able to collect from your customers.

The following figure illustrates the concepts of your *collected* revenue and *expected* revenue.



The Monthly Billing Process

As discussed in the preceding section, we bill your customers monthly. We also charge you monthly for the DevPay fees and for the cost of the AWS services your customers used during the month. This section describes how the process works. We'll continue to use the month of April as an example and focus on the May 1 billing cycle. We'll assume that your customers all paid their April monthly fees on April 1 (during the April 1 billing cycle).

The overall process for the monthly billing cycle is described in the following figure and table.



Process for Monthly Billing

1	We bill your customers. On May 1, we bill your customers for their usage in April and the monthly fee to cover May.
2	Your customers' payments come in during the first few days of May.
3	We subtract the \$0.30 DevPay fee per product from each payment and deposit the remaining amount in your Amazon Payments account. The customers' payments cover revenues for both April and May: usage-based revenue for April, and monthly fee revenue for May. However, the \$0.30 transaction fee collected on May 1 is reported on your April DevPay revenue statement.
4	We charge you a few days later. We calculate your <i>value-add</i> for each customer in April. We then charge you for: <ul style="list-style-type: none"> • The 3% DevPay fee on any value-add amount greater than zero for each customer • The costs of the AWS services each customer used during April <p>We attempt to charge your Amazon Payments account, and if the balance is insufficient, we charge the remainder to your credit card.</p> <p>Note We aggregate all the 3% fee amounts and AWS costs for your customers and charge you once, so you don't have a separate individual charge for each customer.</p>

You can withdraw funds from your Amazon Payments account at any time, but we recommend you wait until after we charge you in step 4 to be sure you have sufficient funds.

Let's continue with the earlier example about Joe who signs up for your product on April 16 (for that example, see [Sign-Up Payments \(p. 37\)](#)). What happens with the May 1 billing cycle? We'll follow the four steps in the monthly billing process described in the preceding table.

For step 1, we bill Joe for his usage from sign-up on April 16 through April 30. We also bill him for the monthly charge that covers all of May. The following figure shows his usage amounts, the corresponding AWS costs for Amazon EC2, and the details of the bill he pays on May 1. We're currently only interested in revenue related to April, so we'll ignore the monthly charge that covers May.

**Amazon DevPay Developer Guide
When and How You Get Paid**

Note

The Amazon EC2 prices shown in the following example were valid when this document was originally written. The prices might have changed since then. For the current Amazon EC2 prices, go to [the Amazon EC2 product page](#).

Joe's use during April (April 16 - April 30)

Small instance-hours used	25 Hours
GB data transferred in	10 GB
GB data transferred out	5 GB

Joe's bill on May 1

Your price for small instance-hours used: 25 hours * \$0.25/hour	\$6.25	Revenue related to April usage: \$10.50
Your price for GB data transferred in: 10 GB * \$0.30/GB	\$3.00	
Your price for GB data transferred out: 5 GB * \$0.25/GB	\$1.25	
Monthly charge for May 1 month * \$8.00/month	+ \$8.00	Revenue related to May usage
Total Joe pays on May 1	\$18.50	

Joe's Amazon EC2 costs for April

Amazon EC2 cost for small instance-hours used: 25 hours * \$0.10/hour	\$2.50
Amazon EC2 cost for GB data transferred in: 10 GB * \$0.10/GB	\$1.00
Amazon EC2 cost for GB data transferred out: 5 GB * \$0.17/GB	+ \$0.85
Total	\$4.35

For step 2, we collect Joe's May 1 payment of \$18.50. For step 3, we subtract the \$0.30 transaction fee and deposit the remainder (\$18.20) in your account.

Total Joe pays on May 1	\$18.50
Transaction fee	-\$ 0.30
Amount deposited in your account on May 1	\$18.20

Step 4 occurs around May 2. We calculate the total revenue Joe generated in April (\$24.50) and your value-add for April (\$20.15). We charge you 3% of your value-add (3% * \$20.15 = \$0.60) plus \$4.35 for the AWS costs for Amazon EC2. The total amount you pay is \$4.95.

**Amazon DevPay Developer Guide
When and How You Get Paid**

Joe's April revenue		Your April value-add for Joe		
April revenue from the April 16 sign-up payment	\$14.00	Joe's April revenue	\$24.50	
April revenue from the May 1 monthly payment	\$10.50	Joe's Amazon EC2 costs	- \$4.35	
Total	\$24.50	Your value-add	\$20.15	→ DevPay Fee on value-add: 3% * \$20.15 = \$0.60

3% DevPay fee on your value-add for April	\$ 0.60
Amazon EC2 costs for April	\$ 4.35
Amount we charge you around May 2	\$ 4.95

The following equation shows your net revenue for Joe for April.

April revenue from Joe's April 16 sign-up payment	\$14.00
Transaction fee for Joe's April 16 sign-up payment	- \$ 0.30
April revenue from Joe's May 1 monthly payment	+ \$10.50
Transaction fee for Joe's May 1 monthly payment	- \$ 0.30
3% DevPay fee on your value-add for April	- \$ 0.60
Amazon EC2 costs for April	- \$ 4.35
Your April net revenue	\$18.95

Related Topics

- [Your DevPay Activity Page \(p. 17\)](#)
- [Customer Refunds and Chargebacks \(p. 57\)](#)

If a Customer Doesn't Pay the Monthly Bill

The preceding description of the monthly billing process (see [The Monthly Billing Process \(p. 40\)](#)) assumes that the customers pay their bills the first time we bill them on the first day of the month (on May 1 in the preceding example).

What if the customer's payment fails? The details depend on whether your value-add is greater than or equal to zero, or less than zero. The following table summarizes what happens, and the subsequent sections give examples.

In either case, we first ask the customer in an e-mail to update the payment method (e.g., the credit card). Then we try to collect from the customer three more times (on May 7, May 14, and May 21). In the table and examples that follow, we assume that the first retry attempt (on May 7) succeeds.

Date	Value-Add Greater Than or Equal to Zero for That Customer	Value-Add Less than Zero for That Customer
May 1	We bill the customer and the payment fails	We bill the customer and the payment fails

Amazon DevPay Developer Guide
When and How You Get Paid

Date	Value-Add Greater Than or Equal to Zero for That Customer	Value-Add Less than Zero for That Customer
Around May 2	We charge you: <ul style="list-style-type: none"> • AWS costs up to the amount of any April revenue collected to date (e.g., the April monthly fee collected on April 1) • 3% of any value-add collected up to that point 	We charge you: <ul style="list-style-type: none"> • AWS costs up to the amount of any April revenue collected to date (e.g., the April monthly fee collected on April 1) • AWS costs not covered by the customer's bill (the amount of your (negative) value-add)
May 7	We: <ul style="list-style-type: none"> • Retry the payment and it succeeds • Subtract \$0.30 and deposit the remainder in your DevPay account 	We: <ul style="list-style-type: none"> • Retry the payment and it succeeds • Subtract \$0.30 and deposit the remainder in your DevPay account
Around May 8	We charge you: <ul style="list-style-type: none"> • Any remaining AWS costs we've not collected from you • 3% of any value-add collected on May 7 	We charge you: <ul style="list-style-type: none"> • Any remaining AWS costs we've not collected from you

If none of the retry attempts actually succeed, then effectively the actions in the final two rows of the table don't occur.

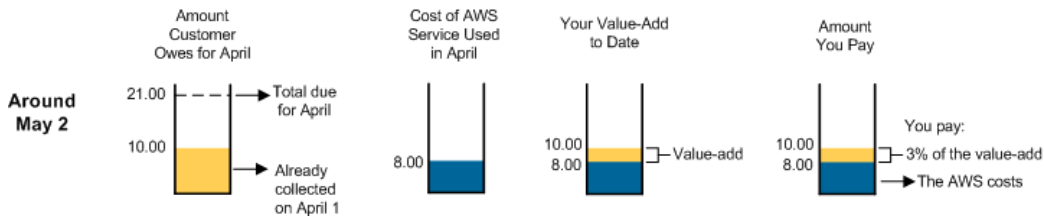
The following examples give specific details of how you get paid and pay AWS what you owe when your value-add is greater than or equal to zero, or less than zero.

Example When Your Value-Add Is Greater Than or Equal to Zero (Version 1)

If the customer doesn't pay after the first billing attempt, then we don't have any money to deposit into your account on May 1. However, if your product has a monthly fee, then we probably already collected the April monthly fee on April 1 and deposited it into your account (minus the \$0.30 transaction fee). Around May 2, we calculate how much April revenue we've collected for you to date and your April value-add to date. We then charge you for the amount of any AWS costs you owe, *up to the amount of April revenue we've already collected to date*. We also charge you 3% of any value-add you have at that point.

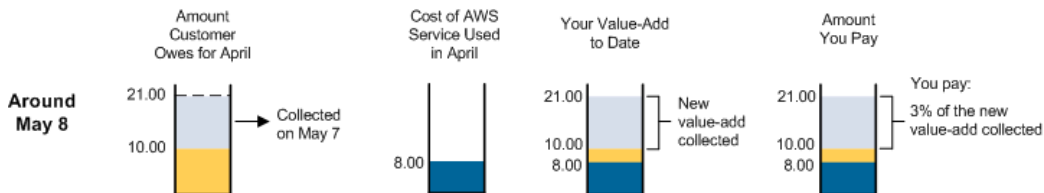
For example, let's say the customer owes \$21.00 total for the month of April (as shown in the following figure). The customer's payment for \$11.00 on May 1 fails, but we've already collected \$10.00 of the \$21.00 on April 1 (the April monthly fee). Let's also say that your total AWS costs for that customer for April are \$8.00.

On May 2, your value-add to date is \$10.00 - \$8.00 = \$2.00. At that time, we charge you the full \$8.00 in AWS costs (because they're entirely covered by the \$10.00 April monthly fee the customer already paid). We also charge you 3% of your value-add to date, which is $3\% * \$2.00 = \0.06 .



We try again to collect the \$11.00 from the customer on May 7, and the payment succeeds. We immediately subtract the \$0.30 and deposit the remainder in your account. Around May 8, we again calculate how much April revenue we've collected for you to date and your April value-add to date. We then charge you for the amount of any AWS costs you still owe, *up to the amount of April revenue we've just collected for you*. We also charge you 3% of any new value-add you have at that point.

In this example, because you've already covered the AWS costs for April, the customer's payment is all value-add (as shown in the following figure). Therefore, at this time we only charge you 3% of your new value-add. Your new value-add is $\$21.00 - \$10.00 = \$11.00$, so we collect $3\% * \$11.00 = \0.33 .

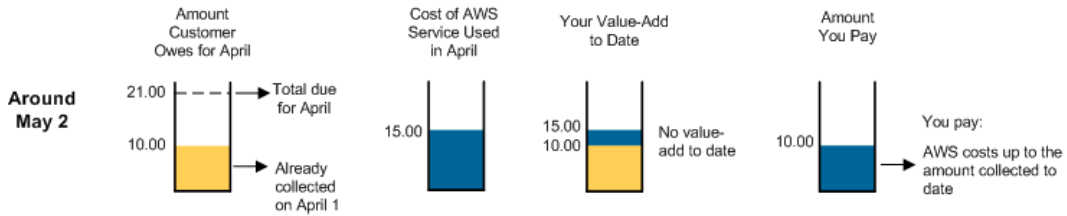


Example When Your Value-Add Is Greater Than or Equal to Zero (Version 2)

This example is the same as the preceding one, except that your total AWS costs for April are *higher* than the \$10.00 April monthly fee—they are \$15.00 instead of \$8.00.

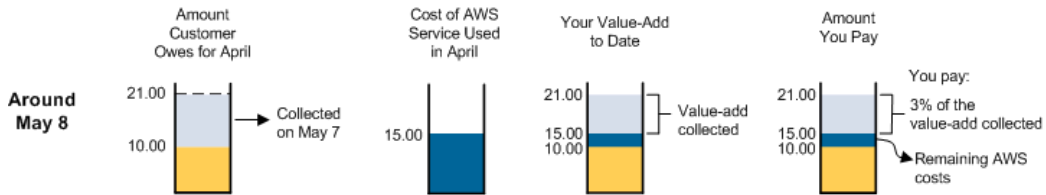
Around May 2, we calculate your value-add to date. It's currently \$0.00 (because all the money we've collected to date will be used to cover AWS costs), so we don't charge you a 3% fee at that time. We do, however, charge you \$10.00 in AWS costs (the amount covered by the \$10.00 April monthly fee the customer already paid). You still owe \$5.00 in AWS costs, but we won't collect that until after one of the retry attempts succeeds and your customer pays the remaining April revenue you're owed.

The following figure illustrates the example.



We try again to collect the \$11.00 from the customer on May 7, and the payment succeeds. We immediately subtract the \$0.30 and deposit the remainder in your account. Around May 8, we again calculate how much April revenue we've collected for you to date, your April value-add to date, and any remaining AWS costs you owe. We then charge you for the amount of any AWS costs you still owe, *up to the amount of April revenue we've just collected for you*. We also charge you 3% of any new value-add you have at that point.

Around May 8, you finally have a value-add that is greater than zero (\$21.00 - \$15.00 = \$6.00, as shown in the following figure). In this case you can also think of it as the difference between the payment you just received and the remaining AWS costs you still owe (\$11.00 - \$5.00 = \$6.00). At this time, we collect the remaining \$5.00 in AWS costs you still owe, plus 3% of your value-add (3% * \$6.00 = \$0.18).



If instead the customer's \$11.00 payment had failed at each retry attempt, you wouldn't get the remaining \$6.00 of your value-add, and we wouldn't get the remaining \$5.00 in AWS costs. We also wouldn't collect the \$0.30 transaction fee or the 3% DevPay fee on that \$6.00. Finally, we would cancel the customer's subscription to your product.

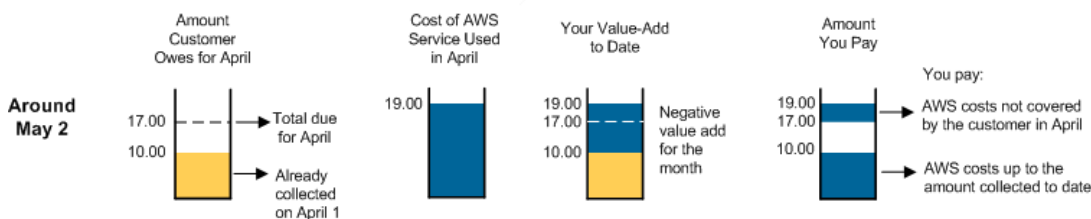
Example When Your Value-Add Is Less Than Zero

This example differs from the two preceding ones in that the amount the customer owes for April is *less* than the total AWS costs for April. The revenue is \$17.00, whereas the AWS costs are \$19.00. Your value-add is $\$17.00 - \$19.00 = -\$2.00$.

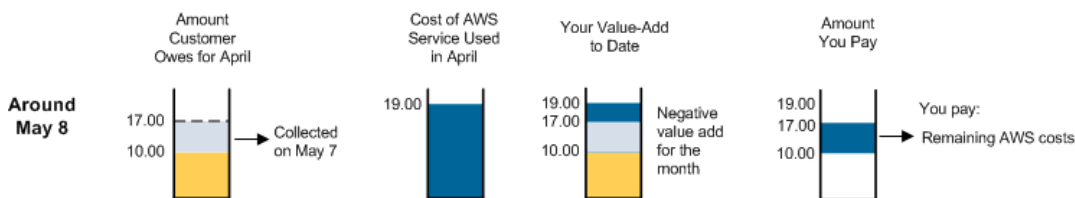
Here, the customer owes \$17.00 total for the month of April, and we've already collected \$10.00 of it on April 1 (the April monthly fee). The customer's payment for \$7.00 on May 1 fails. Around May 2, we calculate how much April revenue we've collected for you to date and charge you two amounts:

- AWS costs up to the amount of revenue already collected (the amount of the April monthly fee)
- AWS costs not covered by the total revenue for April (the absolute value of your negative value-add)

So we collect \$10.00 in AWS costs (which is covered by the customer's April monthly fee payment of \$10.00), and \$2.00 for the AWS costs not covered by the total April revenue. The following figure illustrates the example.



We try again to collect the \$7.00 from the customer on May 7, and the payment succeeds. We immediately subtract the \$0.30 and deposit the remainder in your account. Around May 8, we charge you for the amount of any AWS costs you still owe, *up to the amount of April revenue we've just collected for you*. Because your value-add is negative, we effectively charge you the amount of the customer's payment (as show in the following figure).



E-mails You Receive

We send e-mails to you for the following reasons:

- When the monthly revenue and cost statement is available
- When the monthly customer reports are available (for more information, see [Reports \(p. 69\)](#))
- When you withdraw money from your Amazon Payments account
- If you attempt to withdraw money from your Amazon Payments account and the withdrawal fails
- When we charge you
- If we have problems charging your credit card

Note

We send the e-mails to the e-mail address for the AWS developer account you used to register the DevPay product.

Customer Billing

Topics

- [About Customer Accounts \(p. 49\)](#)
- [When Customers Are Billed \(p. 50\)](#)
- [Where Customers Get Information About Their Bills \(p. 51\)](#)
- [Customer Cancellation \(p. 53\)](#)
- [When Customers Resubscribe \(p. 54\)](#)
- [Where Customers Manage the Payment Method \(p. 55\)](#)
- [When Customers Don't Pay \(p. 56\)](#)
- [E-mail Contact with Your Customers \(p. 56\)](#)

This section discusses how we bill your customers and other situations involving your customers' payments.

Important

When customers buy your product and are later billed for using it, they deal entirely with *Amazon Payments*, and not *Amazon DevPay*. Your customers receive no direct exposure to Amazon DevPay. Therefore, you don't need to point your customers to the Amazon DevPay documentation or information for any reason. For information about how to handle customers' questions, see [Customer Support \(p. 59\)](#).

About Customer Accounts

Each customer who purchases your product must provide the e-mail address and password for an Amazon.com account during the purchase. The customer can't use that same Amazon.com account to purchase your product again, unless the customer cancels the account's subscription to the product first.

Note

Each Amazon.com account is identified by an e-mail address and a specific password. It's possible to have multiple Amazon.com accounts that use the same e-mail address, but different passwords. AWS developer accounts are just Amazon.com accounts that have been enabled for use with Amazon Web Services.

When Customers Are Billed

If your product has a one-time charge, a monthly charge, or both, your customers are immediately billed when they sign up for your product. The charge includes the one-time charge and the first monthly charge. The monthly charge is prorated based on when the customer signs up during the month. For an example, see [Sign-Up Payments \(p. 37\)](#).

Customers are also billed on the first day of each month for:

- The previous month's usage of all Amazon DevPay products they use (for the products that have usage-based charges)
- The monthly charge for the current month (for the products that have a monthly charge)

For an example, see [Monthly Payments \(p. 38\)](#).

They receive a monthly Amazon Payments billing e-mail showing the total amount they have been charged for all the DevPay products (for more information, see the e-mail template at [When the Customer Is Billed Each Month \(p. 209\)](#)). They can see more details about their usage on their Application Billing page (<http://www.amazon.com/dp-applications>). The Amazon Payments bill doesn't show the name of any underlying AWS service that your DevPay product uses. Only your product's name is displayed.

Note

For customers who are also AWS developers (who use AWS products such as Amazon EC2 or Amazon S3), this Amazon Payments bill is separate from their AWS developer bill. If the customer uses Amazon EC2 elastic IP addresses, Amazon Elastic Block Store (Amazon EBS), Auto Scaling, or Amazon CloudWatch with your paid AML, any charges for those are part of the customer's AWS developer bill. For more information, see [Costs You're Not Responsible For \(p. 27\)](#).

Where Customers Get Information About Their Bills

At any time, customers can view their **Application Billing** page (<http://www.amazon.com/dp-applications>). This page shows usage and billing information for all the DevPay products they use. You should provide a link to this page on your web site. Following is an example of the page.

amazonpayments Application Billing Payment Method Application Activation

Hello, Em Vingt. (If you're not Em Vingt, [click here.](#))

Application Billing

[View Previous Statement](#)

Billing Statement as of March 20, 2009†

Billing Cycle for this Report: March 1 - March 31, 2009
This page shows application activity and usage charges through approximately 03/20/2009 17:59 GMT.

Application Name	Rate	Usage	Total(\$)
Sky Storage Sold by: Skysonsa, Inc. View/Cancel Application			
	Recurring monthly charge for April	Recurring monthly charge	5.00
	\$0.150 per GB-Month of storage used	18.343 GB-Month	2.75
	\$0.100 per GB of data transfer in	0.146 GB	0.01
	\$0.180 per GB of data transfer out	0.242 GB	0.04
			7.80
Cactuss Sold by: Hambotext View/Cancel Application			
	Recurring monthly charge for April	Recurring monthly charge	1.50
	\$0.20 per GB-Month of storage used (First 20 GB-Month)	20.000 GB-Month	4.00
	\$0.15 per GB of data transfer out (Greater than 20 GB-Month)	39.440 GB-Month	5.92
	\$0.120 per GB of data transfer in	11.780 GB	1.41
	\$0.190 per GB of data transfer out	0.385 GB	0.07
	\$0.020 per 1,000 PUT or LIST requests	493592 requests	9.87
	\$0.020 per 10,000 GET and all other requests	487746 requests	0.98
			23.75
Charges Due on April 1, 2009†			31.55

† All charges for this billing cycle will be charged to your credit card on your next billing date, April 1, 2009. The current billing cycle starts March 1, 2009 and ends March 31, 2009 00:00 GMT. Not included in the charges displayed here are any additional usage charges you will accrue in this billing cycle.

View a Previous Statement: February 2009

> [Printer Friendly Version](#)

The page shows the current month-to-date usage and related costs of all the DevPay products the customer owns. In this example, the customer owns two different applications. At the end of the month, the customer is billed and the usage and billing information for the month is archived. The customer can view the previous month's statement by clicking the link at the top of the page. The customer can view any month's statement by using the pull-down menu at the bottom of the page. The customer can also get a printer friendly version of the page.

Below the name of each product on the page is a **View/Cancel Application** link. The resulting page displays the product information, the contact information for the company, and the pricing details.

The screenshot shows the Amazon Payments interface for an application named "Sky Storage". At the top, there are three tabs: "Application Billing" (selected), "Payment Method", and "Application Activation". Below the tabs, a blue bar contains the text "Hello, Em Vingt. (If you're not Em Vingt, [click here.](#))". The main heading is "Application Billing > View/Cancel Application". The application name "Sky Storage" is displayed in a light green box. Under "Application Information", the description states: "Sky Storage is an application that lets you store files and backup data securely. Sky Storage makes it easy to store any data securely online, and access it from any computer just like a local hard drive. You can back up your important files like photos, home movies, music, business records, and much more. Sharing files between computers are also easy. You can move large files between work and home or share it with your friends and family." The seller is "Skysonsa, Inc.", the purchase date is "September 18, 2007", and the technical inquiries email is "support@skystorage.com". The "Pricing" section contains a table with the following data:

Price (\$)	Pricing Dimension	Description
0.150	per GB -Month of storage used	This is approximately 200 MP3's stored for a month.
0.100	per GB of data transfer in	This is approximately 200 MP3's uploaded.
0.180	per GB of data transfer out	This is approximately 200 MP3's downloaded.
1.00	one-time charge	This charge only occurs at sign up.
5.00	recurring monthly charge	This charge only occurs at sign up and will be prorated.

At the bottom of the application details, there is a link: "> [Cancel the use of Sky Storage](#)".

If your customers have any billing questions, they should contact us at <application-payments@amazon.com>.

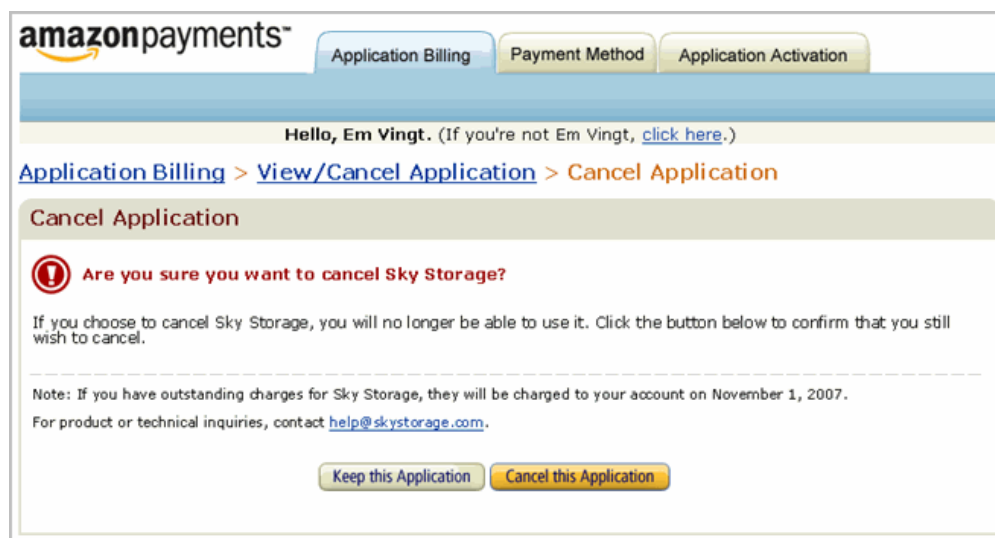
Customer Cancellation

Topics

- [When a Customer Cancels](#) (p. 53)
- [If You Want to Cancel a Customer](#) (p. 54)

When a Customer Cancels

Your customers can cancel their use of your product at any time. To cancel, they go to their **Application Billing** page, click **View/Cancel Application**, and then click **Cancel the use of [product]**. Following is an example of the resulting page.



When customers cancel, access to your product is discontinued immediately. They can no longer access any data they have stored with AWS using your product (for example, data stored in Amazon S3 with your product). For more information about the data, see [Customer Access Stored Data](#) (p. 11).

If your product has a monthly charge, the customer immediately receives a refund for any unused portion of the monthly charge. One-time charges are never refunded in any situation. For information about how we charge you to cover a customer refund, see [Customer Refunds and Chargebacks](#) (p. 57).

DevPay doesn't notify you by e-mail when a customer cancels. For information about determining when customers cancel, see [Reports](#) (p. 69) and [Verifying the Customer's Subscription Status](#) (p. 59).

At the end of the month, we bill the customer for any use of your product up to the point of cancellation (if your product has usage-based charges).

Important

Because customers can't use a different product to access data they store with an AWS service using your product, you should warn your customers that:

- They should retrieve their data before they cancel use of your product
- If they don't pay their bills for your product, we cancel their access to your product and their data becomes unavailable

If You Want to Cancel a Customer

If you want to cancel a customer, e-mail us at <DevPay@amazon.com>. Note that there isn't a programmatic way for you to cancel a customer.

When Customers Resubscribe

If your product has a one-time sign-up charge, and a customer cancels use of your product but then later decides to resubscribe, we assess the one-time sign-up charge again.

If a customer cancels and resubscribes *in the same month*, the customer's Application Billing page shows one contiguous period of subscription, and not two separate periods. Also, we don't restart the customer's usage measurements after the resubscription. Instead, we continue the usage measurements where they left off when the customer canceled. This is important if your product has tiered pricing. For more information, see [Tiered Usage-Based Pricing \(p. 31\)](#).

Where Customers Manage the Payment Method

If customers have issues related to the payment method they use to pay for DevPay products (such as needing to update credit card information), they should follow your link to the **Application Billing** page and then click the **Payment Method** tab. Following is an example page.

The screenshot shows the Amazon Payments interface. At the top, there are three tabs: 'Application Billing', 'Payment Method' (which is selected), and 'Application Activation'. Below the tabs, a greeting says 'Hello, Em Vingt. (If you're not Em Vingt, [click here.](#))'. The main heading is 'Payment Method', followed by a sub-heading 'Edit Payment Method'. Under 'Current Payment Method', the details for an Amazon.com Visa card are listed: Type: Amazon.com Visa, Credit Card Number: *****11111, Expiration Date: 01/2010, Cardholder's Name: Em Vingt, and Billing Address: Em Vingt, 123 Any Street, Any City, State 12345, United States, Phone: 1234567890. There is an 'Edit this Card' button. Below this, a note states: 'If you select a new or different payment method, it will be used for all your applications going forward.' The next section is 'Select a Different Payment Method', which lists two credit cards from the user's account: MasterCard (number *****99535, expiration 10/2007) and Amazon.com Visa (number *****11111, expiration 01/2010). The Amazon.com Visa card is selected. Below this is the 'Or Enter a New Payment Method' section, which has a form with fields for Payment Type (a dropdown menu showing 'Visa'), Credit Card Number (a text input), Expiration Date (two dropdown menus for month and year, showing '01' and '2006'), and Cardholder's Name (a text input). A 'Continue' button is at the bottom.

A single payment method is used for all DevPay products the customer owns. The customer can change the payment method, add a new payment method, or change information related to the payment method, such as the card expiration date or the billing address.

When Customers Don't Pay

What happens if a customer doesn't pay? If we're unable to charge your customer's credit card, we e-mail the customer and ask for a valid payment method. We retry the payment up to three times, once each week. If the customer still hasn't paid after the third retry (21 days after the initial billing attempt), we cancel the customer's access to your product. Your customer can no longer access any data stored with AWS using your product.

The amount you pay to AWS decreases if a customer doesn't pay a bill. For more information, see [If a Customer Doesn't Pay the Monthly Bill \(p. 42\)](#).

DevPay doesn't notify you by e-mail if a customer cancels or doesn't pay. For information about determining when customers cancel or don't pay, see [Reports \(p. 69\)](#) and [Verifying the Customer's Subscription Status \(p. 59\)](#).

E-mail Contact with Your Customers

Amazon Payments sends e-mails to your customers regarding the purchase and billing of your product. For information about when those e-mails are sent and what they look like, see [Appendix: E-mails Sent to Your Customers \(p. 206\)](#).

Customer Refunds and Chargebacks

Topics

- [Refunds \(p. 57\)](#)
- [Chargebacks \(p. 58\)](#)

Refunds

If your product has a monthly charge, customers automatically receive a refund if:

- They cancel use of the product (for more information, see [Customer Cancellation \(p. 53\)](#)). We refund the unused prorated portion of the monthly charge.
- You decrease the product's monthly charge sometime during the month (for more information, see [Changing Pricing \(p. 64\)](#)). We refund the net prorated amount at the time of the price change.

In both of these cases, the refund occurs in the same month as the original monthly charge. For example, if the customer pays the July monthly charge and then cancels in the middle of July, we give the refund immediately in July (not later in August). Or, if the customer pays the July monthly charge, and then in the middle of July you decrease your product's monthly charge, we give the refund immediately in July. This means we're able to take that refund into account when we calculate your July value-add for the customer in early August. In other words, you don't pay the 3% DevPay fee on money that you've refunded to the customer.

When we give a refund, we pay the amount on your behalf and then charge you the amount. We first attempt to collect the money from your Amazon Payments account. If the funds in the account are insufficient, we bill your credit card for the remaining amount. We do not refund you the \$0.30 transaction fee.

Important

We recommend that you leave money in your Amazon Payments account to cover refunds.

Your DevPay Activity page includes a line item with the sum of the refunds we charged you for during the month. When we charge you, your DevPay transaction history includes a "Customer Refunds" line item. The charge can be for a single refund or multiple (we might aggregate any refunds into a single charge). For an example, see the example pages for July in [Appendix: Example DevPay Activity Pages \(p. 239\)](#)

Note

If you contact us and ask us to refund a customer for a charge that occurred in a previous month, we credit you any 3% DevPay fee you paid earlier related to that charge. However, we do not refund the \$0.30 transaction fee.

This type of refund is not included in the refunds line item on your DevPay Activity page. It does, however, appear in your DevPay transaction history for the month the refund occurred.

Chargebacks

Customers can dispute charges that appear on their credit card bills. When a customer calls the credit card company to dispute a charge, it's called a *chargeback*.

When a customer initiates a chargeback, you can contest it. If you don't, or if the customer wins the chargeback, we return the money to the customer on your behalf and then charge you the amount we returned to the customer.

For chargebacks classified as "Unauthorized Payments," you can contact us at <devpay@amazon.com> and we will refund you the cost of AWS services and the associated 3% Amazon DevPay fee that you paid earlier for the charge in dispute. For chargebacks other than "Unauthorized Payments," if you believe the chargeback was because of performance related to AWS services, you can appeal to get a refund of the AWS costs based on the published service-level agreements (SLAs). For more information, go to the [Amazon Web Services Customer Agreement](#) and [Amazon Payments User Agreement](#).

When charging you for a chargeback, we first attempt to collect the money from your DevPay account. If the funds in the account are insufficient, we bill your credit card for the remaining amount. Chargebacks do not appear on your DevPay Activity page.

Important

We recommend that you leave money in your Amazon Payments account to cover chargebacks.

Customer Support

If you need to contact us about your DevPay product, e-mail us at <DevPay@amazon.com>.

If your customers have technical questions about your product, they should contact you. Make sure to display your contact information on your site. Also, the customer contact information that you provide during product registration is available from the Application Billing page (for information about the page, see [Where Customers Get Information About Their Bills \(p. 51\)](#)).

If your customers have any billing questions, they should contact us at <application-payments@amazon.com>.

If your customers have issues related to the payment method they use to pay for DevPay products (such as needing to update credit card information), they should follow the link you provide to the **Application Billing** page and then click the **Payment Method** tab. For an example of this page, see [Where Customers Manage the Payment Method \(p. 55\)](#).

Important

Verifying the Customer's Subscription Status

When a customer contacts your company for support, your customer service representative might want to verify whether the customer's subscription to your product is active. AWS offers a way for you to programmatically determine if the customer has canceled use of your product. How it works depends on the type of product:

- For Amazon EC2 AMI products, see [Verification of a Customer's Subscription Status \(p. 99\)](#)
- For Amazon Simple Storage Service desktop products, see [Verification of the Customer's Subscription Status \(p. 120\)](#)
- For Amazon Simple Storage Service web products, see [Verification of the Customer's Subscription Status \(p. 135\)](#)

Registering Your Product

The *Amazon DevPay Getting Started Guide* discusses the steps you follow to use DevPay. One of these steps is registering your product with DevPay. During registration, you provide product information such as pricing, and you receive information you need to sell your product. The information that follows explains the steps to registering a product.

Note

AWS must approve your product after you register it. The approval process typically occurs within one business day. During that time you can begin integrating your product with DevPay. The process of product registration and approval *does not* make your product visible to the world. Customers can't sign up for it *until* you advertise the purchase URL (which is where customers sign up). AWS doesn't automatically advertise your product or purchase URL anywhere. For more information about advertising, see [Advertising Your Product \(p. 84\)](#).

You provide the following information during registration:

- Company name
- Product name
- Product description (as you want your customers to see it)
- Redirect URL (the page you want customers to see after they have purchased the product)
- Any terms and conditions you want displayed (optional)
- Contact e-mail address and telephone number (to be used by AWS and not displayed to customers)
- Contact e-mail or URL (to be displayed to customers)
- Pricing for use of the product

The information you display at the redirect URL might vary depending on the type of product. For an Amazon EC2 AMI, you might display information about the AMI. For an Amazon Simple Storage Service desktop product, you might display information about downloading and installing the product. For an Amazon S3 web product, you might display a link taking customers to the site for your product.

Because registration is an important process, the following procedure includes **Example** links for many of the steps. Each link takes you to an example web page for that step in the registration process. In the example pages, we register a fictional product called ABC AMI with DevPay.

Tip

To see the whole set of example pages from start to finish instead of jumping to each one individually in the following procedure, see [Appendix: Product Registration Flow \(p. 217\)](#).

To register your product

1. Decide which AWS developer account you want to use to register your product.
We recommend you create a new AWS developer account and use it for all DevPay products you create. Use a business e-mail address as the login for this account (an address not associated with a specific person).
2. Go to <http://aws.amazon.com/devpay> and click the sign-up button.
Make sure to create the new AWS developer account with the business e-mail address you determined in the previous step.
3. If you are prompted, provide the information required to create an Amazon Payments Business Account.
You're only prompted to do this if you don't already have this type of Amazon Payments account. For more information, see [Your Amazon Payments Account \(p. 15\)](#).
[Example \(p. 218\)](#)
4. Provide the DevPay product information:

- a. Enter the basic product information (your company's name, the product's name, etc.) and click **Continue**.
[Example \(p. 219\)](#)
 - b. Select which AWS service the product uses (Amazon EC2 or Amazon S3).
If you select Amazon EC2, also select the check boxes for the regions and instance types you want the product to cover (for more information, see [Your Product's Configuration and Price \(p. 88\)](#)).
[Example \(p. 220\)](#)
 - c. Click **Continue**.
5. Provide the price for your product:
We recommend you read the sections of this guide about product pricing and fees so you understand the implications of the price you choose (for more information, see [How Do You Set Your Product's Price? \(p. 19\)](#)). You can change your product's price later (for more information, see [Changing Pricing \(p. 64\)](#)).
- a. Read the first page (which gives an overview of DevPay pricing) and click **Continue**.
[Example \(p. 222\)](#)
 - b. Enter any usage-based charges you want to include and click **Continue**.
For the usage-based charges, you can set tiered prices (for more information, see [Tiered Usage-Based Pricing \(p. 31\)](#)).
[Example \(p. 223\)](#)
 - c. Enter any one-time sign-up charges and recurring monthly charges you want to include and click **Continue**.
[Example \(p. 227\)](#)
 - d. Review your overall product pricing, make any changes you want, and click **Continue**.
[Example \(p. 228\)](#)
 - e. Enter any price descriptions you want to include (to give customers additional details about each of the charges) and click **Continue**.
[Example \(p. 229\)](#)
6. Confirm the information you've provided:
- a. On the **Confirmation** page, review the information you've provided.
[Example \(p. 230\)](#)
 - b. Preview what we will display to your customers by clicking **Preview Order Page** at the bottom of the page.
[Example \(p. 232\)](#)
 - c. Click **Return to Previous** to go back to the **Confirmation** page.
 - d. Make any changes you want, and when you're satisfied with the information, on the **Confirmation** page click **Register Product**.
A **Thank You** page is displayed.
[Example \(p. 236\)](#)

The product approval process typically occurs within one business day. During the approval process, you can begin integrating your product with DevPay.

7. On the **Thank You** page, click the link that refers to visiting your product details page.

The product details page for your product is displayed. This page contains all the information about your product. For an example of the page, see [Product Details Page \(p. 237\)](#). At the top are the *product code*, the *product token*, and the *purchase URL*, items you need when integrating with DevPay. You can return to this page at any time by going to your DevPay Activity page (at <http://aws.amazon.com/devpayactivity>) and clicking **View Product Details**.

8. If we created an Amazon Payments Business Account for you during product registration and the **Thank You** page alerts you, verify your e-mail address with Amazon Payments:
 - a. Check for a message that Amazon Payments has sent to the e-mail address you used as the login when you registered the DevPay product.
 - b. Click the link in the e-mail message.
You've verified your e-mail address.

Note

You can't use Amazon DevPay or Amazon Payments until you verify your e-mail address with Amazon Payments.

Important

If we created an Amazon Payments Business Account for you during product registration, you might need to associate a bank account with your Amazon Payments account and verify the bank account. You must do this if you expect to make more than \$10,000 total each month from all your DevPay customers, or if you want to withdraw money from your Amazon Payments account. For instructions, see [Adding and Verifying Your Bank Account \(p. 15\)](#).

Making Changes to Your Product

This topic describes how to make changes to your product's information and how to cancel your product's use of Amazon DevPay.

Changing Product Information

You can change all the basic product information at any time from your DevPay Activity page.

To change product information

1. Go to your **DevPay Activity** Page at <http://aws.amazon.com/devpayactivity>.
2. When prompted, log in with the AWS developer login you used to register the product with DevPay.
3. In the table displaying the activity for the product, click **View Product Details**.
4. On the resulting page, click **Edit**.

Canceling Your Product's Use of DevPay

To cancel your product's use of DevPay, contact <DevPay@amazon.com>.

Related Topics

- [Changing Pricing \(p. 64\)](#)

Changing Pricing

Topics

- [About Price Changes \(p. 64\)](#)
- [Important Dates Related to Price Changes \(p. 65\)](#)
- [Passive vs. Active Authorization \(p. 65\)](#)
- [Changing Your Product's Pricing \(p. 66\)](#)
- [Canceling a Pending Price Change \(p. 66\)](#)
- [Your Customers' Experience of a Price Change \(p. 67\)](#)

This section describes how you change your Amazon DevPay product's price.

Note

If the product is for Amazon EC2 AMIs, all of the information in this section also applies to changing the product's *configuration* (e.g., which Amazon EC2 regions, environments, or instance types the DevPay product covers). For more information about DevPay product configuration, see [Your Product's Configuration and Price \(p. 88\)](#) and [Changing a Product's Configuration \(p. 92\)](#).

About Price Changes

You can change the pricing for a product that already uses Amazon DevPay. When you change pricing for your product, the change affects all customers—both existing customers and new ones who sign up for your product. You cannot change the price just for new customers signing up while leaving existing customers on the original price.

The timing of a price change affects all customers at the same time; it is not on a per-customer basis relative to the customer's sign-up date. In other words, you can't set up a price change to automatically occur six months after the customer signs up.

When you set up a price change, you provide the new pricing information and dates related to the change, as described in the following sections. You can have only one pending price change. For example, you can't schedule a price change for June 1 and at the same time schedule a price change for September 1. You must wait for the June 1 price change to occur before you can schedule another price change for the product. Or, you can cancel the June 1 price change and schedule a new one for September 1.

Note

Price changes occur on Coordinated Universal Time (Greenwich Mean Time).

General Process for a Price Change

1	You schedule the price change. You don't need to get prior approval from AWS for a price change. You should keep in mind all the pricing guidelines when determining your new price (for more information, see How Do You Set Your Product's Price? (p. 19)).
2	Amazon Payments sends a <i>notification e-mail</i> to your existing customers making them aware of the upcoming price change.
3	The price change takes effect, and Amazon Payments sends a <i>confirmation e-mail</i> to your existing customers confirming that the price change has occurred.

The timing of the steps is discussed in the following section.

Important Dates Related to Price Changes

When you request a price change for your product, you must specify two dates:

- **E-mail Notification**—When you want Amazon Payments to send the notification e-mail (to tell your existing customers about the upcoming price change)
- **Effective Date**—When you want the price change to go into effect

You can schedule a price change up to 180 days before the effective date.

For information about how a price change and the preceding dates affect the pricing information displayed to new customers signing up for your product, see [New Customers Who Sign Up \(p. 67\)](#).

Passive vs. Active Authorization

You must specify whether the price change requires *passive authorization* or *active authorization* from your existing customers. We recommend you use passive authorization whenever possible because it is a better experience for your customers.

Passive Authorization

Existing customers receive the notification e-mail from Amazon Payments regarding the upcoming price change and don't need to take any action to accept the price change. For the e-mail template, see [When You Schedule a Price Change with Passive Authorization \(p. 213\)](#). They are switched automatically to the new price on the effective date. The notification e-mail includes a link to the Application Billing page where they can cancel if they don't want to accept the price change. For information about customer cancellation, see [Customer Cancellation \(p. 53\)](#).

You have the opportunity to modify a portion of the e-mail's contents when you schedule the price change. We recommend you include information about the consequences of canceling. For example, if your product stores images for your customers, let your customers know their access to those images will be discontinued when they cancel. For the template we use for the e-mail, see [When You Schedule a Price Change with Passive Authorization \(p. 213\)](#).

Timing

You must specify a notification e-mail date that is at least 14 calendar days before the price change effective date.

Active Authorization

Existing customers receive the notification e-mail from Amazon Payments regarding the upcoming price change, and they must actively accept the price change by going to the URL provided in the e-mail. For the e-mail template, see [When You Schedule a Price Change with Active Authorization \(p. 213\)](#).

Customers who instead choose to immediately cancel their service have their access to the product canceled immediately. They receive a refund for any unused portion of the monthly charge. They also receive an e-mail when their access is canceled. For the e-mail template, see [If the Customer Cancels Use of Your Product \(p. 208\)](#).

Customers who don't accept or cancel by the time the price change takes effect have their access to the product canceled when the price change takes effect. They receive a refund for any unused portion of the monthly charge. They also receive an e-mail when their access is canceled. For the e-mail template, see [If the Customer Doesn't Accept the Price Change by the Deadline, Causing Cancellation \(p. 215\)](#). If they want access to the product again, they must sign up for it again. If your product has a one-time charge, we assess that charge again when they resubscribe.

As previously mentioned, we send an e-mail to your existing customers requesting they take action to accept the price change or cancel. You have the opportunity to modify a portion of the e-mail's contents when you schedule the price change. We recommend you include information about the consequences of canceling or taking no action. For example, if your product stores images for your customers, let your customers know their access to those images will be discontinued if they cancel or if they haven't taken any action when the price change takes effect.

Timing

You must specify a notification e-mail date that is at least one calendar day before the price change effective date.

Changing Your Product's Pricing

Use the following procedure to change the pricing for your product. At any point before the price change goes into effect, you can edit the information you submitted.

Important

If you edit the information after the e-mail notifications have already been sent, a new round of e-mail notifications must be sent that reflect your edited information. If you're using passive authorization, during editing you must specify new dates that meet the 14-day requirement.

To set up a price change

1. Go to your **DevPay Activity** page at <http://aws.amazon.com/devpayactivity>.
2. When prompted, log in with the AWS developer login you used to register the product with DevPay.
3. In the table displaying the activity for the product, click **Change Pricing**.
4. Follow the instructions presented to you to provide the new pricing information and specify dates for the price change and the corresponding notification e-mail.

Canceling a Pending Price Change

You can cancel a pending price change up to the day before the effective date. If the e-mail notification informing your customers of the upcoming price change has already been sent, Amazon Payments sends another e-mail notifying your customers of the cancellation. For the e-mail template, see [When You Cancel an Upcoming Price Change \(p. 214\)](#).

To cancel a pending price change

1. Go to your **DevPay Activity** page at <http://aws.amazon.com/devpayactivity>.
2. When prompted, log in with the AWS developer login you used to register the product with DevPay.
3. In the table displaying the activity for the product, click **View Product Details**.
4. In the area where the upcoming price change information is listed, click **Delete Pricing Plan**.

If you're canceling the price change before the e-mail notification alerting your customers has been sent, the price change is deleted and you see a confirmation page.

If you're canceling the price change after the e-mail notification alerting your customers has already been sent, you are first prompted to review the new e-mail that will go to your customers to alert them of the cancellation, and then you receive a confirmation page.

Your Customers' Experience of a Price Change

How existing customers are notified of and accept a price change is covered in the section [Passive vs. Active Authorization](#) (p. 65).

If your product has a one-time charge and you increase or decrease that charge, existing customers are not affected. The change to the one-time charge only applies to customers who sign up after the price change takes effect.

With either passive or active authorization, customers might decide to cancel use of your product because of the price change. Customers' access to your product is discontinued immediately, and they receive a refund for any unused portion of the monthly charge. For more information, see [Customer Cancellation](#) (p. 53).

Effect of a Mid-Month Price Change

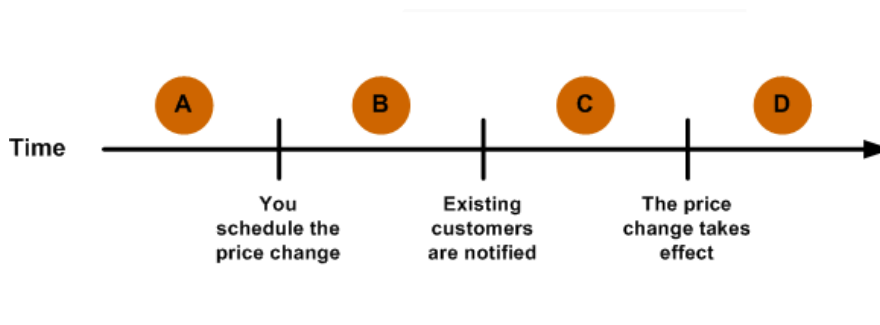
If the price change occurs in the middle of a month, the customer's Application Billing page (which shows usage and billing information) displays the old and the new pricing information separately. The customers' bills are affected as follows:

- **If the monthly charge increases**—Your customers are not billed for that increase until the next regular monthly billing cycle; however, they'll be charged a prorated amount at that time.
For example, if your product's pricing increases on April 15, the customer isn't billed for that increased monthly charge on April 15. Instead, on May 1, they're billed to cover the increased monthly charge for April 15-30 and also to cover the entire monthly charge for May.
- **If the monthly charge decreases**—Your customers are refunded the net prorated amount at the time of the price change.
The confirmation e-mail that indicates the price change has taken effect includes information about the refund.

If your product has tiered prices for any of the usage-based components, we always consider the customer's total usage for the month when determining how to charge the customer before and after the price change. For more information, see [Price Changes with Tiered Pricing](#) (p. 33).

New Customers Who Sign Up

The experience new customers have of a price change depends on when they sign up. The following figure and table describe new customers' experience at different points during the process of a price change.



Time Point	When New Customers Sign Up	Customer Experience
A	Before you schedule a price change	During sign-up, customers see the original pricing.

Amazon DevPay Developer Guide
Changing Pricing

Time Point	When New Customers Sign Up	Customer Experience
B	After you schedule a price change, but before the e-mail notification goes to existing customers	During sign-up, customers see the original pricing. They receive the e-mail notification like everybody else.
C	After the e-mail notification goes to existing customers, but before the price change effective date	During sign-up, customers see the original pricing, and the new pricing and when it takes effect. If your price change requires active authorization, the customer's purchase of the product implies authorization of the price change.
D	After the price change effective date	During sign-up, customers only see the new pricing.

Reports

Topics

- [Subscription Report \(p. 69\)](#)
- [Revenue Report \(p. 71\)](#)
- [Usage Report \(p. 76\)](#)

Amazon DevPay provides you with three reports to help you understand your revenue, your costs, and your customers' use of your product.

Subscription Report

DevPay provides a subscription report for your DevPay products. The report is always available at any time in your DevPay Activity page and always displays information for the current month. You don't have to generate it. You can modify it to include one DevPay product, or all of your products. The following table lists the information included in the report.

Item	Description
Total Subscriptions To Date That Haven't Been Canceled	The total number of subscriptions to date that haven't been canceled.
New Subscriptions This Month That Haven't Been Canceled	The total number of customers who subscribed this month and have not canceled. When a customer subscribes in the current month, the value increases by one; if that customer then cancels in the same month, the value decreases by one, and the number of cancellations increases by one (see the next item in this table).
Canceled Subscriptions This Month	The total number of canceled subscriptions in the current month. This includes cancellations of subscriptions created in the current month and previous months. If a customer subscribes and cancels multiple times in the current month, the value increments each time the customer cancels.

To view the subscription report

1. Go to your **DevPay Activity** page at <http://www.amazon.com/devpayactivity>.
2. Click **View Customer Reports** in the table for any of your products.
The **DevPay Customer Reports** page is displayed, and the report is at the top of the page. You can view the report for a single product or all of your DevPay products.

Note

In the **Revenue** section of your *DevPay Activity* page, we display the number of transactions that we will charge the \$0.30 transaction fee for (see the image below). Do not confuse that number with the number of customers you have for the product (which is shown in the subscription report). The number of transactions includes the sign-up payments new customers make, plus the payments *all* customers make at the beginning of the next month. For more information, see the examples covered in [Appendix: Example DevPay Activity Pages \(p. 239\)](#).

DevPay Activity

Account Number: 12 [REDACTED] 99

View:

Summary (July 1 - July 31, 2009)

	Billed (\$)	Collected (\$)
Total Revenue	295.84	120.64
AWS Costs	-263.27	-0.00
DevPay Fee	-3.99	-0.60
Customer Refunds	-6.45	-6.45
Total Net Proceeds	22.13	113.59

ABC AMI Activity (July 1 - July 31, 2009)

[View Product Details](#) |
 [Change Pricing](#) |
 [View Customer Reports](#)

Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	120.64
Amazon EC2 small instance-hours used	\$0.20 × 81 hours	16.20
Amazon EC2 large instance-hours used	\$0.50 × 12 hours	6.00
Amazon EC2 x-large instance-hours used	\$0.90 × 170 hours	153.00
		295.84
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.10 × 81 hours	-8.10
Amazon EC2 large instance-hours used	\$0.40 × 12 hours	-4.80
Amazon EC2 x-large instance-hours used	\$0.80 × 170 hours	-136.00
Amazon EC2 data uploaded	\$0.10 × 479 GB	-47.90
Amazon EC2 data downloaded	\$0.17 × 391 GB	-66.47
DevPay Fee	(3% × \$42.87) + 9 transactions × \$0.30	-3.99
		-267.26

Related Topics

- [Revenue Report \(p. 71\)](#)
- [Usage Report \(p. 76\)](#)

Revenue Report

The purpose of the revenue report is to help you understand the revenue you've received from each individual customer, across all your DevPay products (if you have multiple ones). You can use this report to determine who your customers are, which customers haven't paid their bills, and the difference between what a customer paid and the corresponding AWS costs you paid.

The report is available in comma-separated value (CSV) format on or around the fifteenth day of each month. We don't generate a revenue report if you don't have any customers yet.

The report covers revenue from all your customers' activity during the previous month. Each row in the report lists the revenue and costs associated with an individual customer's use of a single DevPay product. The report includes subscribed customers even if they didn't use the product during the previous month.

Keep in mind that if your product has a monthly charge, we bill your customers for that on the first day of the month, along with the charges for their usage the previous month. This means that the revenue for the monthly charge appears in the next month's report. For example, the report you receive on January 15 includes revenue for your customers' usage in December. However, the monthly charges to cover January (which we bill customers for on January 1) appear in the February 15 report.

If a customer cancels and then resubscribes to your product in the same month, we present the revenue from each subscription period separately in that month's report. For example, if the customer cancels on January 10 and then resubscribes on January 20, the report you receive in February has one row for the revenue for January 1 to January 10, and a separate row for the revenue for January 20 to January 31. The value in the *Customer Since* field for the second subscription period will be January 20.

Important

The revenue report includes sensitive customer information such as the customer's name and e-mail address. Make sure to take the necessary precautions to keep the information safe. For more information, go to the *Limits on Your Use of Information* section in the [Amazon Payments User Agreement](#).

To view the revenue report

1. Go to your **DevPay Activity** page at <http://www.amazon.com/devpayactivity>.
2. Click **View Customer Reports** in the table for any of your products.
The **DevPay Customer Reports** page is displayed.
3. In the **Customer Revenue and Usage Reports** section of the page, select the time period for the report, click **Revenue**, and click **Download Report**.
You're prompted to save the CSV report to the location of your choice.

The following table describes each of the columns presented in the report in the order they appear. All values in the report (except column names) are enclosed in quotation marks.

Column Name	Description
Customer Email	The customer's e-mail address. Type: String with 64 characters maximum length Example: john@example.com
Customer Name	The customer's first and last name. Type: String with 150 characters maximum length Example: John Smith

Amazon DevPay Developer Guide
Reports

Column Name	Description
Postal Code	The postal code from the customer's billing address. For U.S. ZIP codes, only the five basic digits are included. Type: String with 32 characters maximum length Example: 98144
Country Code	The ISO country code for the country in the customer's billing address. For a list of the ISO country codes, go to the ISO web site . Type: String with 2 characters maximum length Example: US
Billing Period	The billing month the report covers. Type: Date with format MMM-YYYY Example: DEC-2007
Application	The name of the DevPay product (important if you have multiple DevPay products). Type: String with 80 characters maximum length Example: My Application
Customer Status	The status of the customer on the day the report is generated. Note that you can programmatically determine at any time whether a customer is active or canceled. For more information, see Verifying the Customer's Subscription Status (p. 59) . Type: String with 32 characters maximum length Valid values: Active Activation Pending Cancelled Activation Pending means that the customer's payment at sign-up has not yet cleared. Customers with this status are not actually signed up for the product yet; however, in this report we display the revenue from the sign-up charge that we are attempting to collect.
Customer Since	The date the customer signed up for the product (and if the product has up-front fees at sign-up, this is the date the customer's sign-up payment succeeded). If the customer's status is Activation Pending, this field is empty. For information about what this field displays if a customer cancels and then resubscribes in the same month, see the paragraphs preceding this table. Type: Date with format DD-MMM-YY Example: 12-DEC-07
Cancellation Date	The date the customer canceled use of the product. If the customer hasn't canceled, this field is empty. Type: Date with format DD-MMM-YY Example: 15-DEC-07

Amazon DevPay Developer Guide
Reports

Column Name	Description
Revenue Billed	<p>The amount we billed the customer for the product during the previous month (including applicable sign-up and usage charges), regardless of whether the customer has subsequently paid the bill. For the amount the customer has actually paid as of the report generation date, see the description of the <i>RevenueCollectedCurrBillPeriod</i> column that follows in this table.</p> <p>Type: Currency rounded to two decimal places Example: 12345.99</p>
AWS Costs	<p>The cost to you of the underlying AWS service your customer used. Because it is a debit, this value is always negative.</p> <p>Note that this is the total amount we <i>expect</i> to charge you. The amount we have <i>actually</i> charged you as of the report generation date is listed in the <i>AWSCostsChargedCurrBillPeriod</i> column. To understand why the amount we've actually charged you might be less than the amount we expect to charge you, see If a Customer Doesn't Pay the Monthly Bill (p. 42). Any additional AWS costs for the current billing period that we collect <i>after</i> the report generation date will appear in next month's report, in the <i>AWSCostsChargedPrevBillPeriod</i> column.</p> <p>Note that data transfer out is a special case. Because it has a tiered cost structure, the value used to determine its cost is the cost-per-unit across the tiers.</p> <p>For example, let's say that data transfer out costs \$0.10 for the first gigabyte used by all your customers, and \$0.20 for the remaining gigabytes used by all your customers. If you have three customers, and they each use 0.5 GB, then the total amount of data transfer out is 1.5 GB. Therefore, the cost-per-unit is: $[(0.5 \text{ GB} * \\$0.10/\text{GB}) + (0.5 \text{ GB} * \\$0.10/\text{GB}) + (0.5 \text{ GB} * \\$0.20/\text{GB})] / 1.5 \text{ GB} = \\0.13 per GB.</p> <p>Type: Currency rounded to two decimal places Example: -12345.99</p>
DevPay Fee	<p>The DevPay fees you owe for the billing period. Because it is a debit, this value is always negative.</p> <p>Note that this is the total amount we <i>expect</i> to charge you. The amount we have <i>actually</i> charged you as of the report generation date is listed in the <i>DevPayFeeChargedCurrBillPeriod</i> column. To understand why the amount we've actually charged you might be less than the amount we expect to charge you, see If a Customer Doesn't Pay the Monthly Bill (p. 42). Any additional DevPay fees for the current billing period that we collect <i>after</i> the report generation date will appear in next month's report, in the <i>DevPayFeeChargedPrevBillPeriod</i> column.</p> <p>Type: Currency rounded to two decimal places Example: -12345.99</p>
Refunds Issued	<p>Refunds we've issued to your customer because of a cancellation or a price change to the product. Because it is a debit, this value is always negative.</p> <p>Type: Currency rounded to two decimal places Example: -12345.99</p>

Amazon DevPay Developer Guide
Reports

Column Name	Description
RevenueCollectedCurrBillPeriod	<p>Revenue collected for the billing period listed in the report, as of the report generation date. This includes sign-up charges (if the customer signed up during the billing period), monthly charges, and usage charges.</p> <p>For example, the January 15 report shows sign-up charges we collected in December for customers who signed up in December, monthly charges we collected in December to cover December use, and usage charges we collected in January (up until January 15) for customers who used the product in December. It's possible we'll collect a customer's payment for December usage after January 15 (for example on January 21). That revenue will appear in the February 15 report in the <i>RevenueCollectedPrevBillPeriod</i> column (see the next entry in this table).</p> <p>Type: Currency rounded to two decimal places Example: 12345.99</p> <p>Note The value reported is the value <i>before</i> AWS subtracts the corresponding DevPay fees.</p>
RevenueCollectedPrevBillPeriod	<p>Revenue collected since the last revenue report for usage during the months prior to the current billing period (see the preceding description for <i>RevenueCollectedCurrBillPeriod</i>).</p> <p>The value reported is the value <i>before</i> AWS subtracts the corresponding DevPay fees.</p> <p>Type: Currency rounded to two decimal places Example: 12345.99</p>
AWSCostsChargedCurrBillPeriod	<p>AWS costs we've actually collected for the billing period listed in the report, as of the report generation date. Compare this to the AWS costs we <i>expect</i> to collect (see the description for the preceding <i>AWS Costs</i> column). Because it is a debit, this value is always negative.</p> <p>Type: Currency rounded to two decimal places Example: -12345.99</p>
AWSCostsChargedPrevBillPeriod	<p>AWS costs collected since the last revenue report for usage during the months prior to the current billing period (see the preceding description for <i>AWSCostsChargedCurrBillPeriod</i>). Because it is a debit, this value is always negative.</p> <p>Type: Currency rounded to two decimal places Example: -12345.99</p>
DevPayFeeChargedCurrBillPeriod	<p>DevPay fees we've actually collected for the billing period listed in the report, as of the report generation date. Compare this to the DevPay fees we <i>expect</i> to collect (see the description for the preceding <i>DevPay Fee</i> column). Because it is a debit, this value is always negative.</p> <p>Type: Currency rounded to two decimal places Example: -12345.99</p>

Amazon DevPay Developer Guide Reports

Column Name	Description
DevPayFeeChargedPrevBillPeriod	DevPay fees collected since the last revenue report for usage during the months prior to the current billing period (see the preceding description for <i>DevPayFeeChargedCurrBillPeriod</i>). Because it is a debit, this value is always negative. Type: Currency rounded to two decimal places Example: -12345.99

Example Revenue Report

```
Customer Email, Customer Name, Postal Code, Country Code, Billing Period, Application, Customer Status, Customer Since, Cancellation Date, Revenue Billed, AWS Costs, DevPay Fee, Refunds Issued, RevenueCollectedCurrBillPeriod, RevenueCollectedPrevBillPeriod, AWSCostsChargedCurrBillPeriod, AWSCostsChargedPrevBillPeriod, DevPayFeeChargedCurrBillPeriod, DevPayFeeChargedPrevBillPeriod
"joesmith@example.com", "Joe Smith", "92009", "US", "FEB-2008", "My Product A", "Cancelled", "27-FEB-08", "04-MAR-08", ".31", "-.14", "-.31", "0", ".31", "0", "-.14", "0", "-.31", "0"
"janedoe@example.com", "Jane Doe", "M4E3W7", "CA", "FEB-2008", "My Product B", "Active", "21-FEB-08", "", "3.54", "-.03", "-.71", "0", "3.54", "0", "-.03", "0", "-.71", "0"
"markcooper@example.com", "Mark Cooper", "50014", "US", "FEB-2008", "My Product C", "Active", "24-FEB-08", "", "1.1", "0", "-.62", "0", "1.1", "0", "0", "0", "-.62", "0"
"samanthajones@example.com", "Samantha Jones", "NW1 9XN", "GB", "FEB-2008", "My Product D", "Active", "27-DEC-07", "", "62.28", "-40.91", "-.94", "0", "20.00", "0", "-20.00", "0", "0", "0"
```

Related Topics

- [Subscription Report \(p. 69\)](#)
- [Usage Report \(p. 76\)](#)

Usage Report

The purpose of the usage report is to help you understand the specific usage details that resulted in the revenue collected for each individual customer the previous month. You can use this report to understand the usage patterns of your product and the associated AWS costs.

Like the revenue report, this report is available in comma-separated value (CSV) format on or around the fifteenth day of each month. We don't generate a usage report if none of your customers used your product or generated any revenue for you during the previous month.

Each row of the report represents a single revenue item related to a customer's activity with a single DevPay product. For example, a row might represent the revenue from a sign-up charge, a monthly charge, or a specific AWS service usage dimension (such as data transfer in or data transfer out). Only dimensions related to a customer's use during the month are included in the report. For example, if your product uses Amazon EC2, but the customer didn't transfer any data in or out during the previous month, the report has no rows for data transfer in or out for that customer.

If a customer cancels and then resubscribes to your product in the same month, we present the revenue from each subscription period separately in that month's report. For example, if the customer cancels on January 10 and then resubscribes on January 20, the report you receive in February has one set of rows for the revenue for January 1 to January 10, and a separate set for the revenue for January 20 to January 31. The value in the *Customer Since* field for the second subscription period will be January 20.

To view the usage report

1. Go to your **DevPay Activity** page at <http://www.amazon.com/devpayactivity>.
2. Click **View Customer Reports** in the table for any of your products.
The **DevPay Customer Reports** page is displayed.
3. In the **Customer Revenue and Usage Reports** section of the page, select the time period for the report, click **Usage**, and click **Download Report**.
You're prompted to save the CSV report to the location of your choice.

The following table describes each of the columns presented in the report in the order they appear. All values in the report (except column names) are enclosed in quotation marks.

Column Name	Description
Customer Email	The customer's e-mail address. Type: String with 64 characters maximum length Example: john@example.com
Billing Period	The billing month the report covers. Type: Date with format MMM-YYYY Example: DEC-2007
Application	The name of the DevPay product (important if you have multiple DevPay products). Type: String with 80 characters maximum length Example: My Application

Amazon DevPay Developer Guide
Reports

Column Name	Description
Customer Since	The date the customer signed up for the product (and if the product has up-front fees at sign-up, this is the date the customer's sign-up payment succeeded). If the customer's status is <code>Activation Pending</code> , this field is empty. Type: Date with format DD-MMM-YY Example: 12-DEC-07
Amazon Web Service	The underlying AWS service. The field has a value only when <i>Type</i> is <code>Usage</code> ; otherwise it is empty. Type: String with 32 characters maximum Example: AmazonEC2
Type	The type of charge. Type: String with 32 characters maximum Valid values: <ul style="list-style-type: none"> • DevPayFee—The DevPay fees related to this customer's use of your product • OneTimeFee—The one-time sign-up charge • Refund—A refund issued because of a cancellation or change to your product's price • Subscription—The monthly charge • Usage—A charge for a usage dimension for the underlying AWS service. See the following description for <i>Usage Dimension</i>.
Usage Dimension	If the <i>Type</i> column's value is <code>Usage</code> , then this field lists the specific usage dimension for the revenue reported in this row. For example, <code>DataTransfer-In-Bytes</code> . The values shown in this field are the same values displayed when you generate an AWS Usage Report for your AWS developer account (creating an AWS Usage Report is one of the options you have when you log in to your AWS developer account). Type: String with 256 characters maximum length Example: <code>DataTransfer-Out-Bytes</code>
Usage Amount	The amount the customer used of the specified dimension. Type: Number with 24 characters maximum length Example: 0.032988
Usage Unit	The unit of measure for the specified dimension. Type: String with 30 characters maximum length Example: GB
Revenue Billed	The revenue we billed the customer for the specified dimension, regardless of whether the customer has actually paid the bill. If the <i>Type</i> column's value is <code>Refund</code> , then the value in this field is negative. Type: Currency rounded to two decimal places Example: 12345.99

Amazon DevPay Developer Guide
Reports

Column Name	Description
Costs	<p>If the <i>Type</i> is <code>DevPayFee</code>, this field lists the DevPay fee related to this customer's use of your product. If the <i>Type</i> is <code>Usage</code>, this field lists the cost to you of the underlying AWS service the customer used.</p> <p>Note that data transfer out is a special case. Because it has a tiered cost structure, the value used to determine its cost is the cost-per-unit across the tiers.</p> <p>For example, let's say that data transfer out costs \$0.10 for the first gigabyte used by all your customers, and \$0.20 for the remaining gigabytes used by all your customers. If you have three customers, and they each use 0.5 GB, then the total amount of data transfer out is 1.5 GB. Therefore, the cost-per-unit is: $[(0.5 \text{ GB} * \\$0.10/\text{GB}) + (0.5 \text{ GB} * \\$0.10/\text{GB}) + (0.5 \text{ GB} * \\$0.20/\text{GB})] / 1.5 \text{ GB} = \\$0.2 / 1.5 \text{ GB} = \\0.13 per GB.</p> <p>Because it is a debit, this value is always negative.</p> <p>Type: Currency rounded to two decimal places</p> <p>Example: -12345.99</p>

Example Usage Report

```
Customer Email, Customer Name, Billing Period, Application, Customer Status, Customer
Since, Cancellation Date, Revenue Billed, AWS Costs, DevPay Fee, Refunds Issued, Rev
enueCollectedCurrBillPeriod, RevenueCollectedPrevBillPeriod
"joesmith@example.com", "FEB-2008", "My Product A", "27-FEB-08", "AmazonS3", "Us
age", "DataTransfer-In-Bytes", ".0049798339605331", "GB", ".01", "0"
"joesmith@example.com", "FEB-2008", "My Product A", "27-FEB-08", "AmazonS3", "Us
age", "DataTransfer-Out-Bytes", ".0037920186296105", "GB", ".01", "0"
"joesmith@example.com", "FEB-2008", "My Product A", "27-FEB-08", "AmazonS3", "Us
age", "Requests-Tier1", "12310", "Requests", ".25", "-.12"
"joesmith@example.com", "FEB-2008", "My Product A", "27-FEB-08", "AmazonS3", "Us
age", "Requests-Tier2", "12737", "Requests", ".03", "-.01"
"joesmith@example.com", "FEB-2008", "My Product A", "27-FEB-08", "AmazonS3", "Us
age", "TimedStorage-ByteHrs", ".00030796040363353", "GB-Mo", ".01", "0"
"joesmith@example.com", "FEB-2008", "My Product A", "27-FEB-08", "", "DevPay
Fee", "", "", "", "0", "-.31"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "AmazonS3", "Us
age", "DataTransfer-In-Bytes", ".00011754874140024", "GB", ".01", "0"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "AmazonS3", "Us
age", "DataTransfer-Out-Bytes", ".00035172142088413", "GB", ".01", "0"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "AmazonS3", "Us
age", "Requests-Tier1", "893", "Requests", ".02", "-.01"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "AmazonS3", "Us
age", "Requests-Tier2", "22178", "Requests", ".04", "-.02"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "AmazonS3", "Us
age", "TimedStorage-ByteHrs", ".000010993909733049", "GB-Mo", ".01", "0"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "", "Subscrip
tion", "", "", "", "3.45", "0"
"janedoe@example.com", "FEB-2008", "My Product B", "21-FEB-08", "", "DevPay
Fee", "", "", "", "0", "-.7"
"markcooper@example.com", "FEB-2008", "My Product C", "24-FEB-08", "", "OneTime
Fee", "", "", "", "1", "0"
"markcooper@example.com", "FEB-2008", "My Product C", "24-FEB-08", "", "Subscrip
tion", "", "", "", ".1", "0"
"markcooper@example.com", "FEB-2008", "My Product C", "24-FEB-08", "", "DevPay
Fee", "", "", "", "0", "-.6"
"samanthajones@example.com", "FEB-2008", "My Product D", "27-DEC-
07", "AmazonEC2", "Usage", "BoxUsage", "409", "Hrs", "57.26", "-40.9"
"samanthajones@example.com", "FEB-2008", "My Product D", "27-DEC-
07", "AmazonEC2", "Usage", "DataTransfer-In-Bytes", ".017763266339898", "GB", ".01", "0"
"samanthajones@example.com", "FEB-2008", "My Product D", "27-DEC-
07", "AmazonEC2", "Usage", "DataTransfer-Out-Bytes", ".03164594899863", "GB", ".01", "-
.01"
"samanthajones@example.com", "FEB-2008", "My Product D", "27-DEC-07", "", "Subscrip
tion", "", "", "", "5", "0"
"samanthajones@example.com", "FEB-2008", "My Product D", "27-DEC-07", "", "DevPay
Fee", "", "", "", "0", "-2.17"
```

Related Topics

- [Subscription Report \(p. 69\)](#)
- [Revenue Report \(p. 71\)](#)

Recommendations for Product Development

When developing your product and planning how your system will work, we suggest you follow the recommendations described in the following table.

Recommendation	Description
Special AWS developer account for your products	When registering your first product with DevPay, create a single AWS developer account to use with all your DevPay products. All the products will then appear on the DevPay Activity page associated with that AWS developer account. Use a business e-mail address not associated with a specific person as the login for the account.
Amazon Payments branding	On your site, mention that Amazon Payments handles the billing for your product. Include the Amazon Payments trademark or logo to help orient your customers. The trademark, logo, and guidelines for using them with Amazon DevPay are available in the co-marketing area of the Developer Connection web site .
Link to Application Billing page	Display an obvious link to the Application Billing page (http://www.amazon.com/dp-applications). Explain that this is the location where customers get usage and billing information about your product, cancel use of your product, and manage their payment methods.
Your customer service contact information	Display your own customer service information prominently so that customers know how to contact you with questions about your product.
Amazon Payments contact information	Display the <code><application-payments@amazon.com></code> e-mail address. This is where customers should go with billing questions.
Area for product news on your site	Include an area on your site where you post the latest news or information about your product, and make customers aware of the area.
Activation key	<p>When you read the sections in this guide about integrating your product with DevPay, you'll see that you probably need to obtain an <i>activation key</i> from DevPay for each customer. Whether you need to do this depends on the type of product you have and whether you want to verify if a customer is still subscribed to your product. The details of these situations are discussed later in the guide.</p> <p>AWS makes the activation key available to you in several ways discussed elsewhere in the guide. One of the ways involves customer interaction and requires the customer to get the key and give it to you. Another way doesn't involve the customer; instead, you programmatically obtain the key from a redirect URL. We recommend you use the programmatic method but also have a backup plan to obtain the key from the customer, should the programmatic method fail.</p> <p>Any time you ask the customer to get the activation key for you (for reasons described in subsequent sections in this guide), display the <i>activate URL</i> (http://www.amazon.com/dp-activate), which goes to a page where customers can obtain an activation key for your product (for an example, see The Application Activation Page (p. 203)). Customers must log in with their Amazon.com login IDs.</p>

Amazon DevPay Developer Guide
Recommendations for Product Development

Recommendation	Description
Implication of cancellation	Make customers aware of what happens when they cancel use of your product. For example, if your product stores images for them, let the customers know they can't access those images once they cancel. Caution them to retrieve their data before canceling (for more information about customer cancellation and access to data, see Customer Cancellation (p. 53) and Customer Access Stored Data (p. 11)).
Amazon EC2 sign-up	If your product is an Amazon EC2 AMI product, make your customers aware that they must be signed up for Amazon EC2 to purchase your product. Customers who are not already signed up for Amazon EC2 will be prompted to sign up when they purchase your product.
Product security	Design your product and system with security in mind. For information about some possible product risks you should be aware of, see Appendix: Risks to DevPay Products (p. 261) . If your product is an Amazon EC2 AMI, don't enable root access to the AMI unless it's necessary. For more information, go to the section about protecting shared AMIs in the <i>Amazon Elastic Compute Cloud Developer Guide</i> .

Testing and Going Live

This section describes the process we recommend you follow to test your DevPay product and to take it into production.

Important

DevPay has no testing sandbox. You can only test your product in a live environment. Remember that you will incur real charges during testing and your credit card will be charged accordingly.

Overall Process for Testing and Going Live with Your Product

1	Understand how product pricing and DevPay fees work. For more information, see How Do You Set Your Product's Price? (p. 19) . The section links you to several other sections that describe possible implications of your pricing scheme.
2	Create an AWS developer account to use when creating your product. We recommend you have a single AWS developer account for all your DevPay products. Don't use the personal AWS account of one of the employees in your company.
3	Determine your pricing and the other product information you'll provide when registering your product. For more information, see Registering Your Product (p. 60) .
4	Register your product using the AWS developer account you created and get the product token, product code, and purchase URL for your product. After AWS approves your product, the purchase URL becomes live and available for anyone to use. Don't expose the purchase URL to the public yet.
5	Do any coding required to integrate your product with DevPay. For an Amazon EC2 paid AMI, see Using DevPay with Your Amazon EC2 AMI (p. 85) . For an application using Amazon S3, see Using DevPay with Your Amazon S3 Product (p. 104) .
6	Use the purchase URL yourself to sign up for your product with a test customer account. You must provide an active credit card when you sign up. If you use the product with the test customer account, the credit card is charged accordingly.
7	If your system is designed to programmatically obtain the activation key, confirm your system gets it correctly when a test customer signs up for your product.
8	Use your product with the test customer account you've created.
9	Watch your DevPay Activity page and the Application Billing page for the test customer. These show the usage information and corresponding revenue you expect to receive for the usage of your product by your test customer. For an example of how charges appear on your DevPay Activity page and DevPay transaction history, see Appendix: Example DevPay Activity Pages (p. 239) .
10	Look at the revenue your product generates and verify that the product's pricing meets your requirements. If necessary, make changes to the pricing. For more information, see Changing Pricing (p. 64) .
11	If you're using the License Service API to validate the status of a customer's subscription, use the API to confirm the status of your test customer. Confirm the process fits correctly into your customer support workflow. For more information, see Verifying the Customer's Subscription Status (p. 59) .
12	Update the product information if you find you need to make any changes. For more information, see Making Changes to Your Product (p. 63) .

13	After you're satisfied with the pricing and product information and you're ready to go into production, make the purchase URL visible to the public. For more information, see Advertising Your Product (p. 84) .
----	---

Advertising Your Product

When you've completed testing of your product and want to let customers buy it, you can advertise your product through AWS. AWS doesn't automatically advertise your product or purchase URL anywhere; it's up to you to do it.

To advertise an Amazon S3 application

- Post information about the application in the [Solutions Catalog](#) on the AWS Developer Connection site.

To advertise an Amazon EC2 paid AMI

- Post information about the AMI in the [Solutions Catalog](#) on the AWS Developer Connection site and on the [Amazon Machine Images \(AMIs\)](#) page on the AWS Resource Center.

For information about co-marketing your product with AWS, go to the [co-marketing area of the Developer Connection site](#).

Using DevPay with Your Amazon EC2 AMI

Topics

- [Overview of DevPay with Amazon EC2 \(p. 86\)](#)
- [Your Product's Configuration and Price \(p. 88\)](#)
- [The Product Code and AMI Rebundling \(p. 89\)](#)
- [If You Have Multiple AMIs to Sell \(p. 90\)](#)
- [Changing a Product's Configuration \(p. 92\)](#)
- [Basic Paid AMI Tasks \(p. 94\)](#)
- [Frequently Asked Questions \(p. 97\)](#)
- [Web Site Development \(p. 98\)](#)
- [Verification of a Customer's Subscription Status \(p. 99\)](#)
- [Supported AMIs \(p. 102\)](#)
- [Additional Details \(p. 103\)](#)

This section describes how to use Amazon DevPay with Amazon EC2.

Overview of DevPay with Amazon EC2

Topics

- [Paid AMIs \(p. 86\)](#)
- [Current Limitations \(p. 86\)](#)
- [Summary of How Paid AMIs Work \(p. 86\)](#)

Paid AMIs

The most popular way to use Amazon EC2 and Amazon DevPay together is by selling an AMI you've created. An AMI that you sell through DevPay is called a *paid AMI*.

With a paid AMI, your customers:

- Must be signed up to use Amazon EC2 themselves

Important

The process they go through to purchase your AMI product prompts them to sign up for Amazon EC2 if they aren't already signed up. However, to reduce any possible confusion, we encourage you to inform your customers on your site that they must be signed up for Amazon EC2 to purchase your product.

- Buy your paid AMI and then launch instances of it
- Always use *their own* AWS credentials when launching instances; you don't launch instances of your paid AMI for them with your credentials
- Pay the price you set for the paid AMI, and not the normal Amazon EC2 rates

You can also use Amazon EC2 and Amazon DevPay together with a *supported AMI*. For more information about supported AMIs, see [Supported AMIs \(p. 102\)](#).

Current Limitations

With the current implementation of Amazon DevPay:

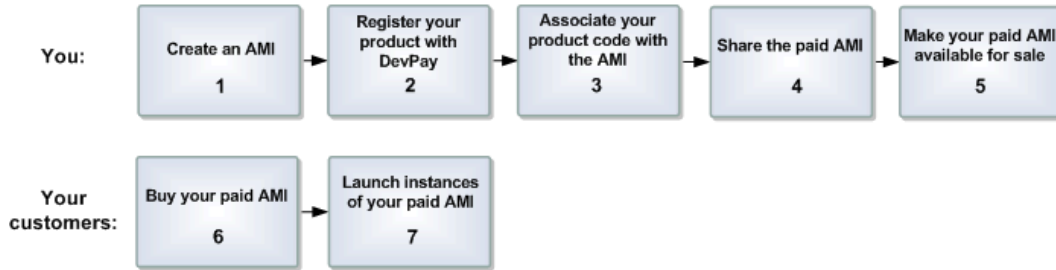
- Your paid or supported AMIs must be backed by Amazon S3. Paid or supported AMIs backed by Amazon Elastic Block Store are currently not supported. Therefore, your paid AMIs cannot run Windows 2008 Server or SQL Server 2008 at this time.
- You can't use Elastic Load Balancing (either by itself or in conjunction with Auto Scaling) with instances of paid or supported AMIs.
- The discounts from Amazon EC2 Reserved Instances don't apply to paid or supported AMIs. That is, if you purchase Reserved Instances, you don't get the lower usage price associated with them when your customers launch your paid or supported AMIs. Also, if your customers purchase Reserved Instances, and they use your paid or supported AMIs, they continue to pay the price you specified for the use of your paid or supported AMIs.
- Your customers can't make Spot Instance requests for your paid or supported AMIs; if they do, Amazon EC2 returns an error.

For more information about any of the preceding Amazon EC2 features, go to the [Amazon EC2 product page](#).

Summary of How Paid AMIs Work

The following figure and table summarize the basic flow for creating and using paid AMIs.

Amazon DevPay Developer Guide
Overview of DevPay with Amazon EC2



Process for Creating and Using Paid AMIs

1	You create an AMI. The AMI must be Amazon S3-backed and not Amazon EBS-backed. For more information, go to the Amazon Elastic Compute Cloud Developer Guide .
2	You register your DevPay product with DevPay. As part of this process, you provide a product description, product pricing, etc. This registration process creates an 8-character <i>product code</i> for the product, a <i>product token</i> for the product, and a <i>purchase URL</i> where customers can sign up to use the product. For more information, see Registering Your Product (p. 60) .
3	You associate the product code with the AMI. This makes the AMI a paid AMI. For instructions, see Associating a Product Code with an AMI (p. 94) .
4	You share the AMI with select customers or the public. For instructions, see Sharing Your Paid AMI with Select Users or the Public (p. 95) . Note that even though you've shared the AMI, because it's a paid AMI (it has a product code associated with it), no one can use the AMI until they sign up for it (see the following steps).
5	You make your paid AMI available for sale. To do this, you make the aforementioned purchase URL available. You can advertise your paid AMI in the Solutions Catalog on the AWS Developer Connection site and on the Amazon Machine Images (AMIs) page on the AWS Resource Center.
6	Customers click the purchase URL and sign up for your product. For an example of what the customer sees during the purchase process, see Appendix: The Customer Purchase Experience (p. 199) . If they're not already signed up for Amazon EC2, they'll be prompted to sign up. They purchase your product with their Amazon.com accounts. They must also have the credentials needed to launch Amazon EC2 instances. At this point, they have the AMI ID of your paid AMI.
7	Customers launch instances of your paid AMI. Because you associated the AMI with the product code, you are charged for the Amazon EC2 costs (instance-hours and bandwidth) the customers incur when using instances of that AMI. However, the customers are also charged at the rate you specified during product registration.

Note

You can associate your DevPay product code with more than one AMI. However, a single AMI can be associated with only one product code. If you plan to sell multiple AMIs, you could sell them all under a single product code, or different product codes (by registering multiple DevPay products). For information about why you might choose a single product code or multiple product codes, see [If You Have Multiple AMIs to Sell \(p. 90\)](#).

Each customer's usage for the paid AMI is displayed on their Application Billing page. For more information, see [Where Customers Get Information About Their Bills \(p. 51\)](#).

At any time, you can confirm the customer is still currently subscribed to your product. For more information, see [Verification of a Customer's Subscription Status \(p. 99\)](#).

Note

In the preceding process, you associate your product code with your own AMI and sell the AMI as a DevPay product. There's another scenario for using DevPay with Amazon EC2 in which you sell software or a service to EC2 users and let them associate your product code with their own AMIs. For more information, see [Supported AMIs \(p. 102\)](#).

Your Product's Configuration and Price

During product registration (see [Registering Your Product \(p. 60\)](#)), you set up the product and its price. Specifically, you designate which regions, environments, and instance types the product covers. We call this *configuring* your DevPay product (don't confuse this with configuring your AMI, which is different and covered in the Amazon EC2 documentation).

You configure the product by simply selecting check boxes for the regions, environments, and instance types that you want the product to support (see the following image, which shows the interface you use to register and configure the product). In this image, the product is configured for Linux/UNIX AMIs only, in all available instance types (small, etc.). If you wanted to also sell Windows AMIs under this product (for example), you would select the corresponding check boxes and provide prices for the specific instance types you selected. For more information about changing a product's configuration, see [Changing a Product's Configuration \(p. 92\)](#).

Usage Charges

Specify **Your Price** for each of the selected instance types. Your customers can launch ABC AMI on the instance types you have checked below. If you choose not to charge for a particular pricing dimension, set the price to 0.00 and your customers will not see that pricing dimension.

		US-East (Northern Virginia) Region	EU (Ireland) Region	US-West (Northern California) Region		
Instance	Details	Your Price(\$) (per hour [†])	Amazon EC2 Cost(\$)	=	Markup Before DevPay Fees [†] (\$)	Tiered Pricing (what's this?)
<input checked="" type="checkbox"/> Amazon EC2 running Linux/UNIX						
Box Usage	<input checked="" type="checkbox"/> Small	0.140	0.085		0.055	Add
	<input checked="" type="checkbox"/> Large	0.440	0.340		0.100	Add
	<input checked="" type="checkbox"/> Extra Large	0.840	0.680		0.160	Add
	<input type="checkbox"/> Double Extra Large	0.000	1.200		N/A	Add
	<input type="checkbox"/> Quadruple Extra Large	0.000	2.400		N/A	Add
	<input checked="" type="checkbox"/> High-CPU Medium	0.240	0.170		0.070	Add
	<input checked="" type="checkbox"/> High-CPU Extra Large	0.840	0.680		0.160	Add
	<input type="checkbox"/> Amazon EC2 running Windows					
<input type="checkbox"/> Amazon EC2 running Windows and SQL Server						
Data Transfer						
Data Transfer Type	Details	Your Price(\$)	Amazon EC2 Cost(\$)	=	Markup Before DevPay Fees [†] (\$)	Tiered Pricing (what's this?)
Data Transfer	GB of data transfer in	0.140	0.100		0.040	Add
	GB of data transfer out	0.200	0.170		0.030	Add
	GB of regional data transfer in/out	0.020	0.010		0.010	Add

Associating your product code with an AMI turns it into a paid AMI that you can sell through DevPay. Letting other EC2 users associate *your* product code with *their* AMIs turns those AMIs into *supported AMIs* (for more information, see [Supported AMIs \(p. 102\)](#)). Only product owners can associate their product code with AMIs that don't match the product's configuration. For example, if your product is configured only for Linux/UNIX AMIs, you can still associate the product code with Windows AMIs, but other EC2 users can't.

Note

You might associate your product code with an AMI that doesn't match the product's configuration in expectation that you will then update the product's configuration to cover that AMI.

AWS also prevents all EC2 users (including you) from launching instance types that don't match the product's configuration. For example, if you configure the product only for small instance types, no one can launch other instance sizes of the AMI. If they try, Amazon EC2 returns an error.

When you set your product's price, you can include a one-time sign-up charge, a monthly charge, data transfer charges, and hourly instance charges for the specific instance types the product is configured for. For information about setting your product's price, see [How Do You Set Your Product's Price? \(p. 19\)](#). Remember that you're responsible for paying the Amazon EC2 costs (bandwidth and instance-hours) your customers incur when using an instance of your paid AMI, just as if you were using the AMI yourself.

Note

You are not responsible for any costs related to your customers' use of elastic IP addresses, Amazon Elastic Block Store, Auto Scaling, or Amazon CloudWatch with your paid AMIs. For more information, see [Costs You're Not Responsible For \(p. 27\)](#).

Caution

If you configure your product for a particular instance type, your customers can launch that instance type regardless of the price you set. You can set the price to any value, including \$0.00. Setting the price to \$0.00 does not prevent customers from launching that instance type; however, it prevents that instance type from appearing in your product's price. Customers could still launch that instance type and incur EC2 costs that you would be responsible for. But, because you set the price to \$0.00, the customers would pay no hourly instance charges. Keep this in mind when you set the price for your product.

The Product Code and AMI Rebundling

Associating a product code with an AMI turns it into a paid AMI that EC2 users must sign up for to use. Can you ensure that the product code stays with the AMI if someone rebundles the AMI? The answer varies for Linux/UNIX AMIs and Windows AMIs. These are described in the following sections.

Linux/UNIX AMIs

If you give the customer root access to your paid Linux/UNIX AMI, the customer can rebundle it. If your customer uses AWS tools to rebundle the AMI, the rebundled AMI inherits the product code. When launching instances of the rebundled AMI, the customer is still billed for usage based on your price. However, if the customer doesn't use the AWS tools when rebundling, the rebundled AMI won't inherit the product code, and the customer will pay normal Amazon EC2 rates and not your price.

When a customer contacts you for support for a paid AMI, you can confirm your product code is associated with the AMI and the customer's instance is currently running the AMI. For more information, see [Confirming an Instance Is Running an AMI Associated with a Product Code \(p. 95\)](#).

If you have software installed on the AMI, the software can retrieve the instance metadata to determine if the product code is associated with the instance. For more information, see [Getting the Product Code from Within an Instance \(p. 95\)](#).

Keep in mind that the preceding methods for confirming the association of the product code with the instance are not foolproof because a customer with root access to the instance could return false information indicating the product code is associated with the instance.

Windows AMIs

When you associate a product code with a Windows AMI, the association is permanent. Therefore, we recommend you keep a separate, base copy of the AMI that has no product code associated with it.

Anyone who purchases a Windows AMI can rebundle it. The product code is automatically transferred to the rebundled AMI. When EC2 users launch instances of the rebundled AMI, they pay the rates you set when you registered your DevPay product. In turn, you're charged for the EC2 costs they incur.

If You Have Multiple AMIs to Sell

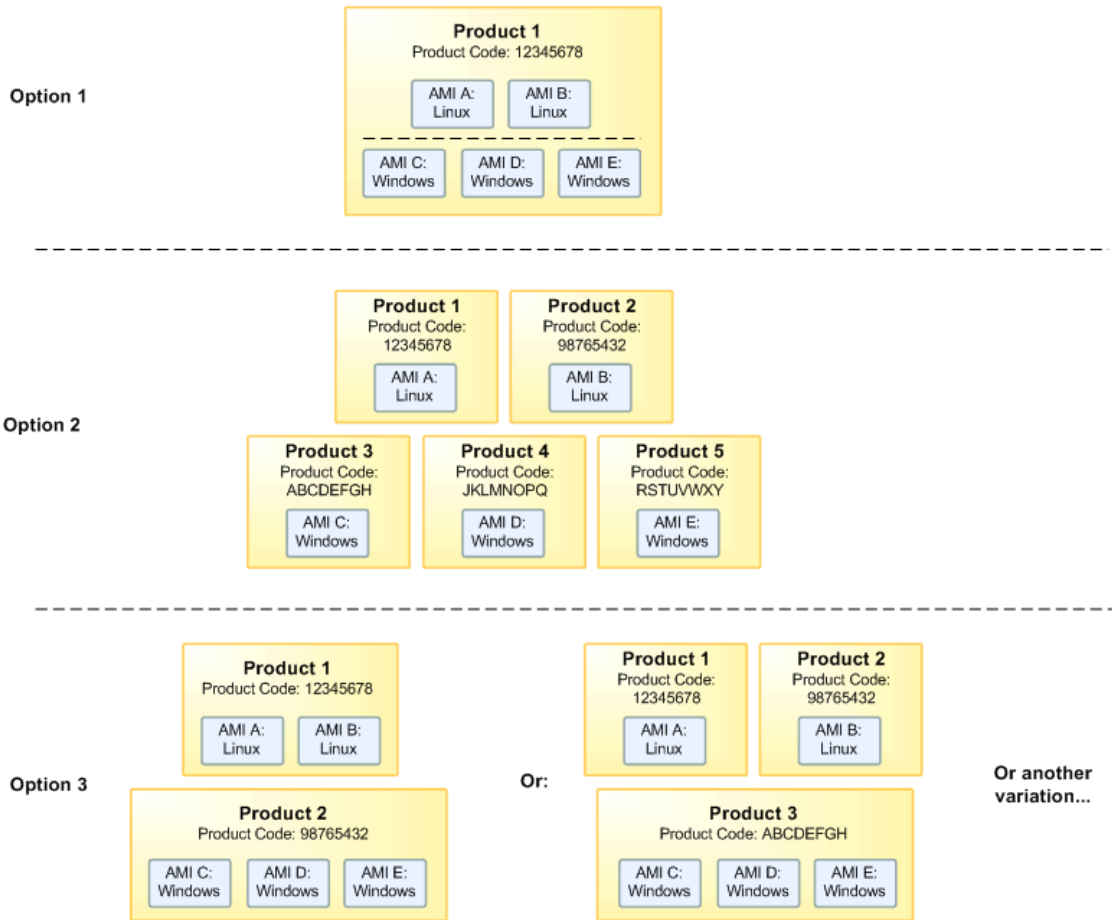
If you have more than one AMI you want to sell, this section can help you decide whether to create one or multiple DevPay products to cover your AMIs.

To recap the basic process for creating a paid AMI: you register a DevPay product, receive a product code, and associate the product code with your AMI (for more information, see [Summary of How Paid AMIs Work \(p. 86\)](#)). Your customers must sign up for your product before they can launch instances of your AMI.

You can associate the product code with multiple AMIs, but each AMI can have only one product code. After an AMI is associated with your product code, it sells for the price you set when you registered the DevPay product. For information about what happens to the product code when you rebundle an AMI, see [The Product Code and AMI Rebundling \(p. 89\)](#).

What if you have more than one AMI to sell? For example, let's say you have two Linux/UNIX AMIs and three Windows AMIs to sell. You have three options (also illustrated in the following figure):

- **Option 1**—Register a single DevPay product that covers them all
- **Option 2**—Register a separate DevPay product for each
- **Option 3**—Do something in between
For example, you could register one DevPay product for the Linux/UNIX AMIs and another for the Windows AMIs.



What's the best path to follow? The answer depends on your needs. The following sections describe factors to consider as you decide how many DevPay products to create for your multiple AMIs.

Product Pricing

If you charge a monthly fee, that single fee covers the customer's access to *all* the different AMIs covered by the product. For example, if the product covers five different types of AMIs (two that use Linux/UNIX and three that use Windows), the customer can use all five types for the one monthly fee (plus any usage-based charges). The customer doesn't pay the monthly fee for *each* type of AMI. This also applies if your product covers multiple AMIs all of the same type. For example, you could assign a single product code to 10 Linux/UNIX AMIs, and a customer could use all 10 for the single monthly fee (plus the usage-based charges). The customer doesn't pay the monthly fee for *each* of the 10 AMIs.

Amazon EC2 charges different prices depending on the region, environment, instance type, etc. (for more information, go to the [Amazon EC2 detail page](#).) Likewise, you can specify which regions, environments, instance types, etc., your product covers, and charge your customers different prices based on those variables. If you want to vary a specific instance type's hourly charge for one AMI versus another, then you need more than one DevPay product. For example, if you want to charge \$0.25 per small instance-hour for one Linux/Unix AMI, but \$0.35 per small instance-hour for another, then you need a separate DevPay product for each AMI.

Customer Experience

Before customers can launch instances of a paid AMI, they must sign up for the DevPay product that covers that AMI. If you decide to have a separate DevPay product for each AMI you sell, your customers must go through the purchase process for *each* AMI they use. You need to decide if this is the customer experience you want. To see an example of the customer purchase experience, see [Appendix: The Customer Purchase Experience](#) (p. 199).

Costs You Pay

You get the benefit of the tiered pricing structure that AWS uses for data transfer out (for more information, see [The Cost of Tiered Usage Types](#) (p. 24)). AWS applies the tiered pricing to the usage across *all customers of a single DevPay product*. Therefore, the more AMIs a single DevPay product covers, the easier it is for that product to reach the usage levels where you pay a lower cost for data transfer out.

Per-AMI Usage Data

DevPay provides you usage data *per product*, but not *per AMI* (for more information, see [Usage Report](#) (p. 76)). If it's important for you to have usage data *per AMI*, you can register a different DevPay product for each AMI. However, we recommend you consider the other customer-impacting factors discussed in this section before choosing this path.

Multiple Versions of a Product

You can have multiple versions of a DevPay product. For example, you could have a regular version and a more expensive *support* version that includes customer support that you provide. In this case you would register a separate DevPay product for each version.

Important

You can't automatically upgrade or migrate a customer between DevPay products. Instead, the customer must cancel the subscription to the first (we refund any unused portion of the monthly fee), and then go through the purchase process for the second. No monthly fees, account information, or usage information can be transferred between the two products.

Changing a Product's Configuration

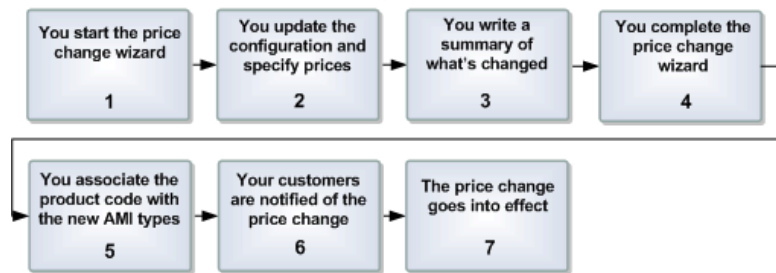
You might want to change an existing DevPay product's configuration. *Changing the configuration* simply means you change which regions, environments, or instance types the DevPay product covers (for more information about configuration, see [Your Product's Configuration and Price](#) (p. 88)). Essentially when you change a product's configuration, you go through the process of changing the product's price (for more information, see [Changing Pricing](#) (p. 64)).

For example, let's say you have a product that covers one or more Linux/UNIX AMIs, and you also want to sell some Windows AMIs under the same product code. You can change the product's configuration to cover your new Windows AMIs, and as part of that process, you specify the price you want to charge for the Windows AMIs. We recommend you understand the implications of selling multiple AMIs under a single or multiple product codes. For more information, see [If You Have Multiple AMIs to Sell](#) (p. 90).

Important

We recommend you warn your customers at least 14 days in advance of any change to your product, especially if you plan to remove items from the configuration. When you change your product's configuration, you simply walk through the price change wizard, which includes a step for specifying the product's configuration (see [Changing Pricing](#) (p. 64)). As part of the price change wizard, you can specify a message to include in the price change notification e-mail your customers get. In that message you can summarize what's changed about the product.

The following diagram and table describe how to change your product's configuration.



Process to Change Your Product's Configuration

1	<p>You start the price change wizard.</p> <p>For more information, see Changing Pricing (p. 64). You can do either an active or a passive price change.</p> <p>To <i>add</i> new instance types to the configuration, we recommend you do a passive price change with at least a 14-day notice.</p> <p>To <i>remove</i> instance types from the configuration, we recommend you do an active price change with at least a 14-day notice.</p>
2	<p>In the price change interface, you add or remove the desired items from the configuration. Specifically, you select or clear the check boxes for the items you want to add or remove from the configuration (to see what this looks like, see Your Product's Configuration and Price (p. 88)). If you add new items, you also specify their prices.</p>
3	<p>You write a brief summary of how the product has changed.</p> <p>When you set up the price change, you have the option to include your own message as part of the notification e-mail Amazon Payments sends to your customers. We recommend you include a brief summary of the change.</p> <p>For example, "As of [effective date of price change], your \$20 monthly fee will also cover several Windows AMIs that we now offer, in addition to the existing Linux/UNIX AMIs. If you launch an instance of one of these new Windows AMIs, you will pay the usage prices shown here. Thank you for using our product!".</p>
4	<p>You complete the price change wizard.</p>
5	<p>After preparing any new AMIs you want to sell, you associate the product code with them. For more information, see Associating a Product Code with an AMI (p. 94) and The Product Code and AMI Rebundling (p. 89).</p>
6	<p>Your customers receive the e-mail about the price change.</p> <p>Up until the price change takes effect, new customers who sign up see both the old and new pricing, and the date when the new pricing goes into effect.</p>
7	<p>The price change goes into effect on the date you specified.</p> <p>Existing and new customers can begin to launch instances of any new AMIs that the product covers.</p>

Important

If your product configuration change *discontinues* any instance types, you're not responsible for any charges your customers accrue for the discontinued instance types after the price change takes effect.

Basic Paid AMI Tasks

Topics

- [Associating a Product Code with an AMI \(p. 94\)](#)
- [Sharing Your Paid AMI with Select Users or the Public \(p. 95\)](#)
- [Confirming an Instance Is Running an AMI Associated with a Product Code \(p. 95\)](#)
- [Getting the Product Code from Within an Instance \(p. 95\)](#)

This section describes several tasks you might perform when managing your paid AMIs.

Tip

This section presents the instructions using the Amazon EC2 command line tools. Amazon EC2 also has programmatic APIs (Query and SOAP) and a GUI-based console (at <https://console.aws.amazon.com/>). For more information about using the console, go to the [Amazon Elastic Compute Cloud Console and Command Line User Guide](#). For more information about using the APIs, go to the [Amazon Elastic Compute Cloud Developer Guide](#) and the [Amazon Elastic Compute Cloud API Reference](#).

Associating a Product Code with an AMI

Each AMI can have a single product code associated with it. You must be the owner of an AMI to associate a product code with it. You can associate a single product code with more than one AMI. For more information, see [If You Have Multiple AMIs to Sell \(p. 90\)](#).

Important

You can associate a product code only with Amazon S3-backed AMIs (those using an instance store root device), and not Amazon EBS-backed AMIs (those using an Amazon EBS root device).

To associate a product code with an AMI

- Use the `ec2-modify-image-attribute` command:

```
PROMPT> ec2-modify-image-attribute ami-5bae4b32 --product-code 774F4FF8  
productCodes      ami-5bae4b32      productCode      774F4FF8
```

To verify the product code is associated with the AMI

- Use the `ec2-describe-image-attribute` command:

```
PROMPT> ec2-describe-image-attribute ami-5bae4b32 --product-code  
productCodes      ami-5bae4b32      productCode      774F4FF8
```

You can't change or remove the `productCodes` attribute after you've set it. If you want to use the same image without the product code or associate a different product code with the image, you must reregister the image to obtain a new AMI ID. You can then use that AMI without a product code or associate the new product code with the AMI ID. For more information, see [The Product Code and AMI Rebundling \(p. 89\)](#).

Sharing Your Paid AMI with Select Users or the Public

If you're creating a paid AMI, after you associate the product code with the AMI, you need to share the AMI with select customers or the public.

To share an AMI

- Use the `ec2-modify-image-attribute` command.

The following example shares the AMI with the public.

```
PROMPT> ec2-modify-image-attribute ami-5bae4b32 --launch-permission -a all  
  
launchPermission      ami-5bae4b32      ADD      group      all
```

Even though you've shared the AMI, no one can use it until they sign up for your product by going to the purchase URL. Once customers sign up, any instances of the paid AMI they launch will be billed at the rate you specified during product registration.

Confirming an Instance Is Running an AMI Associated with a Product Code

If you have created a product for others to use with their AMIs (the supported AMI scenario), you might want to confirm that a particular AMI is associated with your product code and a particular instance is currently running that AMI.

Note

You must be the owner of the product code to successfully call `ec2-confirm-product-instance` with that product code. Because your customers don't own the product code, they should describe their instances to confirm their instances are running with your product code.

To confirm an instance is running an AMI associated with your product code

- Use the `ec2-confirm-product-instance` command:

```
PROMPT> ec2-confirm-product-instance 774F4FF8 -i i-10a64379  
  
774F4FF8      i-10a64379      true      111122223333
```

If the AMI is associated with the product code, `true` is returned along with the AMI owner's account ID. Otherwise, `false` is returned.

Getting the Product Code from Within an Instance

A running Amazon EC2 instance can determine if a DevPay product code is attached to itself. The instance retrieves the product code like it retrieves its other metadata. For information about the retrieval of metadata, go to the ["Instance Metadata" section](#) of the *Amazon Elastic Compute Cloud Developer Guide*.

The instance retrieves the product code by querying a web server with this REST-like API call:

```
GET http://169.254.169.254/2007-03-01/meta-data/product-codes
```

Following is an example response.

```
774F4FF8
```

Frequently Asked Questions

This section includes several questions commonly asked by Amazon EC2 developers about paid AMIs.

- Q: Can I sell an AMI that uses an Amazon EBS root device through Amazon DevPay?
A: No, the AMI must have an instance store root device (must be Amazon S3-backed).
- Q: How can Amazon EC2 users determine if a particular AMI is a paid AMI?
A: By describing the image (`ec2describe-images`). An AMI is a paid AMI if a product code is returned. Example: run `ec2describe-images --filters Name=product-codes,Values=amazon`, and the AMI `ami-bd9d78d4` will be returned with a product code (`A79EC0DB`).
- Q: How can Amazon EC2 users determine what paid AMIs are available?
A: By describing images (`ec2describe-images`) with the `-a` flag. This shows all AMIs the user has access to. The AMIs with product codes listed are paid AMIs. Example: run `ec2describe-images -a`, and the result contains an AMI with ID `ami-bd9d78d4`. This is our Demo Paid AMI with product code `A79EC0DB`.
- Q: Is there anything that prevents a paid AMI from being rebundled? How can this be restricted?
A: For Linux/UNIX AMIs: Paid AMIs are comparable to shared AMIs with regards to rebundling and trying to restrict rebundling. If you choose to allow a user running the AMI to see all of its contents (for example, by giving root access to the AMI), then the user could easily rebundle these into their own AMI. For more information, go to [Protecting a Shared AMI \(Linux and UNIX\)](#) in the *Amazon Elastic Compute Cloud Developer Guide*.

For Windows AMIs: Users can rebundle Windows paid AMIs; you can't restrict this.

- Q: Will the product code be inherited by the rebundled AMI?
A: For Linux/UNIX AMIs: If your customer uses AWS tools to rebundle the AMI, the product code associated with the AMI is inherited by the rebundled AMI. When launching the rebundled AMI, the customer is still billed for usage based on your price. Note that the inheritance of the product code during rebundling is a convenience feature and not a guarantee that the product code will always be attached to rebundled AMIs. The customer's workflow could bundle the AMI outside of Amazon EC2, or the customer could use modified versions of the AWS tools, preventing the product code from being inherited. You can include software on the AMI that checks the instance's metadata to determine if the product code is associated with the instance. For more information, see [Getting the Product Code from Within an Instance \(p. 95\)](#).

For Windows AMIs: The product code associated with the AMI is always inherited by the rebundled AMI. Therefore, AWS bills any customers who use instances of the rebundled AMI based on the original product code.

- Q: Will the kernel/RAM disk be inherited by the rebundled AMI?
A: If the AMI is rebundled on an instance that is running with a certain kernel/RAM disk, then the kernel/RAM disk will be inherited by the rebundled AMI. Note that this is a convenience feature, and not a guarantee that the kernel/RAM disk will always be attached to rebundled AMIs.
- Q: Why can't I query a particular AMI's attributes to see if the AMI is paid?
A: Only the owner of an AMI can query the AMI attributes. However, anyone can tell if an AMI is paid by describing images (`ec2describe-images`). An AMI is paid if a product code is returned. Example: run `ec2describe-images --filters Name=product-codes,Values=amazon`, and the AMI with ID `ami-bd9d78d4` will be returned with a product code (`A79EC0DB`).
- Q: Who can use the `confirm-product-instance` command?
A: Only the owner of the AMI can use this command. Owners use this command with supported AMIs to determine if a supported instance with a given product code attached is up and running.
- Q: I created my paid AMIs with one AWS developer account, but I want to sell them using a different AWS developer account. Can I transfer them?
A: No, you can't automatically transfer AMIs from one account to another. You would have to upload them again using the second AWS developer account and then register them with DevPay using that account. Alternately, you could leave the AMIs with the original account (the AMI owner account) and register them with DevPay using another AWS developer account (the product owner account).

You could then use the AMI owner account to associate the product code with the AMIs. However, keep in mind that only the product owner (and not the AMI owner in this case) can use the `ec2-confirm-product-instance` command, which confirms that an instance is running an AMI associated with the product owner's product code. For more information about this command, go to the [Amazon Elastic Compute Cloud Developer Guide](#).

Q: Can my customers use Amazon EBS for persistent data storage with my paid AMI?

A: Your customers can use Amazon EBS volumes for persistent data storage with instances of your paid AMI, just like they can with any Amazon EC2 instances. Amazon EBS volumes are associated with the customer's account and not with an individual instance. Therefore you are not able to charge your customers for their use of Amazon EBS. AWS charges your customers directly for their use of Amazon EBS. Your customers use their credentials in the calls to create an Amazon EBS volume or to attach it to an instance (as they do when they launch an instance of your paid AMI).

Q: Do the discounts from Reserved Instances apply to my paid or supported AMIs?

A: No. The discounts from Amazon EC2 Reserved Instances don't apply to paid or supported AMIs. That is, if you purchase Reserved Instances, you don't get the lower usage price associated with them when your customers launch your paid or supported AMIs. Also, if your customers purchase Reserved Instances, and they use your paid or supported AMIs, they continue to pay the price you specified for the use of your paid or supported AMIs.

Q: Can customers use Spot Instances with my paid or supported AMIs?

A: No. If your customers make Spot Instance requests for your paid or supported AMI, Amazon EC2 returns an error.

Web Site Development

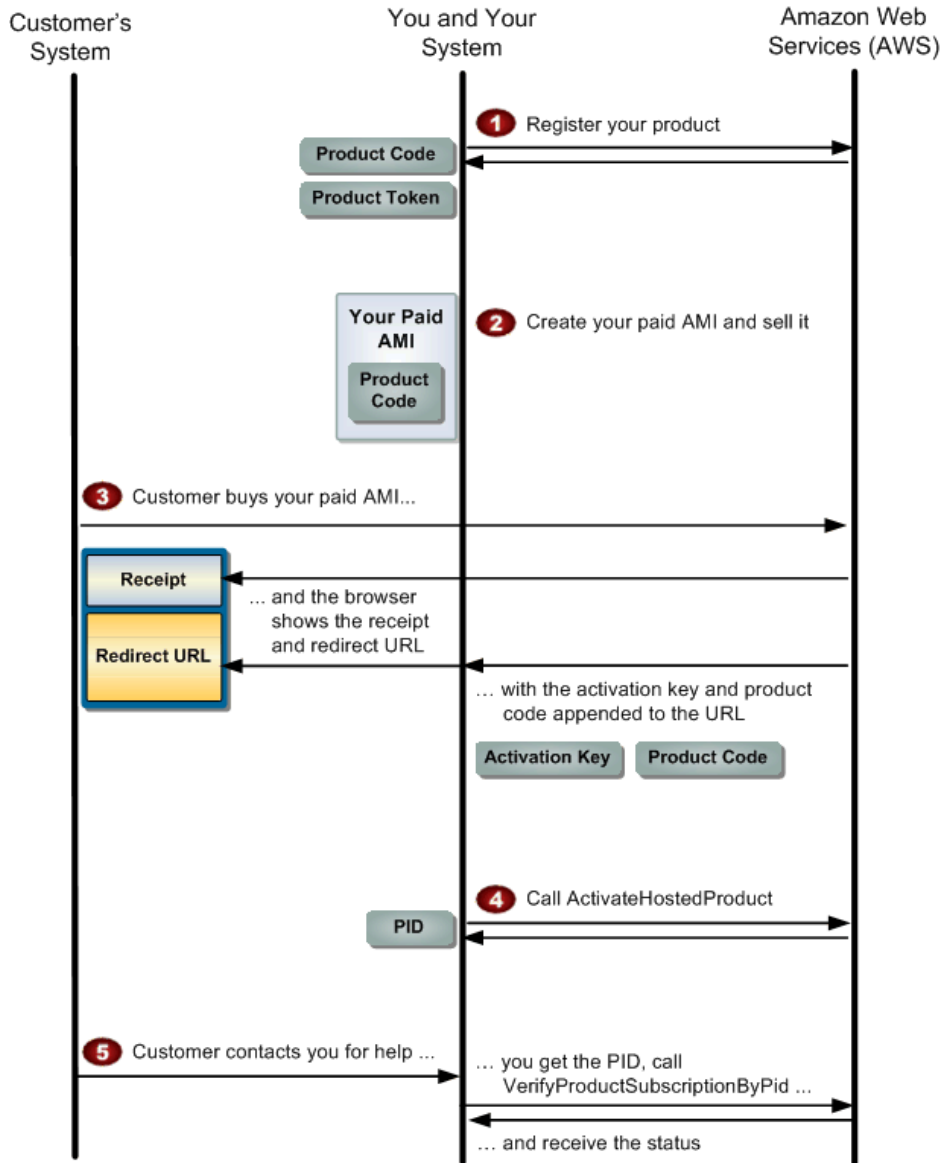
It's your responsibility to provide a web site and sign-up process that includes information about your AMI product, such as its purpose and benefits. In addition, your web site should:

- Provide a sign-up link for your product (the purchase URL you receive when registering your product). For more information about registering your product, see [Registering Your Product \(p. 60\)](#).
- Follow the recommendations for product development. For more information, see [Recommendations for Product Development \(p. 80\)](#).

Verification of a Customer's Subscription Status

If you provide customer support for your paid or supported AMI products, you can confirm that any customer who calls for support is an active subscriber to the AMI product in question. To do this, you must integrate with the License Service API (for information, see [License Service API Reference \(p. 149\)](#)).

The process for verifying the customer's status is described in the following diagram and corresponding table.



Process for Confirming Customer Status

1	<p>You register your AMI product with DevPay. As part of the registration, you receive a <i>product code</i> and a <i>product token</i>. You store those two values together. For more information about product registration, see Registering Your Product (p. 60).</p>
----------	--

Amazon DevPay Developer Guide
Verification of a Customer's Subscription Status

2	<p>You implement your paid or supported AMI.</p> <p>For information about paid AMIs, see Summary of How Paid AMIs Work (p. 86). For information for supported AMIs, see Supported AMIs (p. 102).</p>
3	<p>Later, a customer signs up for your AMI product. At the end of the sign-up process, the customer's browser window splits into two frames. The top frame contains a confirmation for the purchase. The bottom frame contains the <i>redirect URL</i> that you provided when you registered your AMI product with DevPay, with two values appended as query parameters: an <i>activation key</i> and the product code for the AMI product.</p> <p>For example, if your redirect URL is <code>http://www.example.com/productActivation.html</code>, the customer would be redirected to <code>http://www.example.com/productActivation.html?ActivationKey=<activation key value>&ProductCode=<product code value></code>.</p> <p>The activation key is also displayed in the top frame of the window. For information about why it's displayed there, see Exception Handling for the Redirect URL (p. 101).</p>
4	<p>You provide an application that retrieves the two values appended to the redirect URL. The application looks up the product token that is associated with the product code. Then the application makes a signed request to the License Service action ActivateHostedProduct (p. 177) with the product token and the activation key. The response includes a <i>persistent identifier (PID)</i> for the customer. Your application stores the PID.</p>
5	<p>Later, when a customer contacts you for support on one of your AMI products, your customer service representative looks up the PID for the customer. The application then makes a signed request to the License Service action VerifyProductSubscriptionByPid (p. 190) with the PID and the product code for the AMI product in question. The response confirms whether the customer is currently subscribed to the AMI product.</p>

Note

The PID represents the relationship between you (the AMI product owner) and the customer. Even if a customer buys 10 different AMI products from you, you have only one PID for that customer.

Exception Handling for the Redirect URL

Although the failure rate for the redirect is expected to be low, your application must be prepared to handle any failure cases. It's plausible the customer could close the browser window and prevent your application from retrieving the activation key and product code from the redirect. Your application must handle the situation in which one or both of the parameters are not available through the redirect. See the suggested actions in the following table.

Activation Key Retrieved	Product Code Retrieved	Suggested Action
No	Yes	<p>Your web page should display the name of the AMI product and prompt the customer for the activation key, which was displayed in the top frame of the browser window.</p> <p>The customer can then either provide the activation key from the top frame or obtain a new activation key by clicking the <i>activate URL</i> that you display. In either case, you should provide a way for the customer to paste the activation key into a form on your page.</p>
Yes	No	<p>Your web page should display a list of your AMI products and ask the customer to select the AMI product just purchased (the one the activation key is associated with).</p> <p>The customer can then select the product, enabling your application to retrieve the product token to use in a request for <code>ActivateHostedProduct</code>. If the customer chooses the wrong product, the request fails.</p>
No	No	<p>One reason this situation occurs is if the customer closes the browser window.</p> <p>Your web page should display a list of your AMI products and ask the customer to select the AMI product (the one the activation key is associated with). The page should also prompt the customer for the activation key, which was displayed in the top frame of the browser window.</p> <p>The customer can then either provide the activation key from the top frame or obtain a new activation key by clicking the activate URL that you display. In either case, you should provide a way for the customer to paste the activation key into a form on your page.</p>

Supported AMIs

Another way to use Amazon EC2 and Amazon DevPay together is with a *supported AMI*. With a supported AMI, you charge for software or a service you provide that customers use with their own AMIs. The customer's AMI must be Amazon S3-backed (use an instance store root device) and not Amazon EBS-backed (use an Amazon EBS root device).

The main difference between a *paid AMI* and a *supported AMI* is how the AMI is associated with a product code:

- **Paid AMI**—You associate your own product code with your own AMI
- **Supported AMI**—Other EC2 users associate your product code with their own AMIs

The following figure and table summarize the flow to create and use supported AMIs.



Process to Create and Use Supported AMIs

1	<p>You register your product with DevPay.</p> <p>As part of this process, you provide a product description, configure the product, and specify its price. This registration process creates an 8-character <i>product code</i> for the product, a <i>product token</i> for the product, and a <i>purchase URL</i> where customers can sign up to use the product. For more information, see Registering Your Product (p. 60) and Your Product's Configuration and Price (p. 88).</p> <p>Important</p> <p>DevPay works only with Amazon S3-backed AMIs. In your product description, make it clear your product is for Amazon S3-backed AMIs (those using an instance store root device), and not Amazon EBS-backed AMIs (those using an Amazon EBS root device).</p>
2	<p>You make your product available for sale.</p> <p>To do this, you make the aforementioned purchase URL available. You can advertise the product in the Solutions Catalog on the AWS Developer Connection site.</p>
3	<p>Customers click the purchase URL and sign up for your product.</p> <p>For an example of what the customer sees during the purchase process, see Appendix: The Customer Purchase Experience (p. 199). If they're not already signed up for Amazon EC2, they'll be prompted to sign up. They purchase your product with their Amazon.com accounts. They must also have an AWS developer account and the credentials needed to launch Amazon EC2 instances.</p> <p>At this point, they have the product code.</p>
4	<p>Customers associate the product code with their own AMIs.</p> <p>For more information, see Associating a Product Code with an AMI (p. 94).</p>

5	<p>Customers launch instances of their own AMIs.</p> <p>Because the customers signed up for your product and then associated your product code with their AMIs, you are charged for the Amazon EC2 costs (instance-hours and bandwidth) they incur when using those AMIs. However, the customers are also charged at the rate you specified during product registration.</p>
---	--

Note

Amazon EC2 prevents your customers (but not you as the product code owner) from associating your product code with AMI types the product isn't configured for. For example, if the product is configured only for Linux/UNIX AMIs, your customers can't associate the product code with Windows AMIs. Also, Amazon EC2 prevents your customers from launching specific instance types your product isn't configured for.

Caution

If you configure your product for a particular instance type, your customers can launch that instance type regardless of the price you set. You can set the price to any value, including \$0.00. Setting the price to \$0.00 does not prevent customers from launching that instance type; however, it prevents that instance type from appearing in your product's price. Customers could still launch that instance type and incur EC2 costs that you would be responsible for. But, because you set the price to \$0.00, the customers would pay no hourly instance charges. Keep this in mind when you set the price for your product.

Each customer's usage for the AMI is displayed on their Application Billing page. For more information, see [Where Customers Get Information About Their Bills \(p. 51\)](#).

At any time, you can confirm the customer is still currently subscribed to your product. For more information, see [Verification of a Customer's Subscription Status \(p. 99\)](#).

If your product is software that customers install on their AMIs, you can have the software retrieve the instance metadata to check if the customer has actually associated the product code with the AMI. For more information, see the section about instance metadata in the *Amazon Elastic Compute Cloud Developer Guide*. Keep in mind that for Linux/UNIX AMIs, this method is not foolproof because a customer with root access could return false information indicating the product code is associated with the instance.

Additional Details

This guide covers the basic Amazon EC2 commands for setting up paid and supported AMIs.

For complete details about the commands and API calls you can use to manage your paid and supported AMIs, go to the [Amazon Elastic Compute Cloud Developer Guide](#).

Using DevPay with Your Amazon S3 Product

Topics

- [Overview of DevPay with Amazon S3 \(p. 105\)](#)
- [Development with Amazon S3 \(p. 107\)](#)
- [Quick Reference for Amazon S3 Products \(p. 108\)](#)
- [Setting Up Desktop Products \(p. 112\)](#)
- [Setting Up Web Products \(p. 123\)](#)
- [Using Amazon S3 Copy with DevPay \(p. 137\)](#)
- [Using Amazon S3 Logs with DevPay \(p. 138\)](#)
- [Using Amazon S3 Requester Pays with DevPay \(p. 142\)](#)
- [Using Both a Desktop and Web Product \(p. 148\)](#)

This section describes what is required to make your Amazon Simple Storage Service product work with Amazon DevPay.

Overview of DevPay with Amazon S3

With Amazon DevPay and Amazon S3:

- You create a product that uses Amazon S3 and sell it through DevPay
- Your Amazon S3 product can be a desktop application or a web application
- To use your product, your customers *don't have to be signed up to use Amazon S3*
- Your customers don't need to obtain AWS credentials (like a Secret Access Key and Access Key ID); you obtain the credentials that DevPay requires and use them on your customers' behalf
- Your customers pay the price you set for your product, and not the normal Amazon S3 rates; they won't be aware that the product uses Amazon S3 (unless you tell them)

Requirements for Amazon S3

To use Amazon DevPay with your Amazon S3 product:

- Your product must use the REST interface when calling Amazon S3 (web products may also use pre-signed URLs). SOAP is not supported.
- Your product must create a separate bucket in Amazon S3 for each customer who buys and uses the product. This is true even if your DevPay product is a hosted product. Each DevPay product can create up to 100 buckets per customer. For example, a customer who uses three different DevPay products can have up to 300 DevPay buckets, plus any other buckets created outside of DevPay (i.e., those created with a personal AWS account).
Once your product has created a bucket and put objects in it, only your product can access that bucket and the objects in it. For more information about restrictions on data access, see [Customer Access Stored Data \(p. 11\)](#).
- You must be able to do business in the United States and have a U.S. bank account.
For additional requirements see:
 - [Amazon Payments User Agreement](#)
 - [Amazon Web Services Customer Agreement](#)

Summary of How Amazon S3 Products Work

The following figure and table summarize the basic process for implementing Amazon DevPay products that use the Amazon Simple Storage Service. The specific details are covered in subsequent sections and differ for desktop products and web products. For more information about desktop products, see [Setting Up Desktop Products \(p. 112\)](#). For more information about web products, see [Setting Up Web Products \(p. 123\)](#).



Process for Using an Amazon DevPay Product with Amazon S3

1	You create your product.
2	You register your DevPay product with DevPay (for more information, see Registering Your Product (p. 60)).
3	You integrate the product token and product code you received from registration into your product as required for desktop or web products.
4	You make the product available for sale by making the <i>purchase URL</i> available to the public. You can also advertise the product through AWS (for more information, see Advertising Your Product (p. 84)).
5	A customer buys the product and gets an activation key.
6	The customer uses the product for the first time.
7	The product gets the activation key (either through an automatic redirect or from the customer).
8	The product calls the License Service to activate itself using the activation key.
9	The product receives special DevPay credentials specific to the customer and stores them safely.
10	The product calls Amazon S3 on behalf of the customer and uses the customer's special DevPay credentials in the call.

Two Versions of a Product

You can have two versions of a DevPay product. For example, you could have a regular version and a more expensive "support" version that includes customer support that you provide. In this case you would register a separate DevPay product for each version.

Important

You can't upgrade or migrate a customer between two DevPay products. Instead, the customer must cancel the subscription to the first, and then go through the purchase process for the second. No monthly fees, account information, customer data, or usage information can be transferred between the two products.

Development with Amazon S3

This section describes development work required for your web site and for your desktop or web product.

Web Site Development

It's your responsibility to provide a web site and sign-up process that includes information about your product, such as its purpose and benefits. In addition, your web site should handle these items:

- Provide a sign-up link for your product (the purchase URL you receive when registering your product). For more information about registering your product, see [Registering Your Product \(p. 60\)](#).
- Follow the recommendations for product development. For more information, see [Recommendations for Product Development \(p. 80\)](#).

Product Development

The development work required varies for desktop products and web products.

For a quick reference summarizing the important identifiers and authentication used by desktop product and web products, see [Quick Reference for Amazon S3 Products \(p. 108\)](#).

For all the details for desktop products, see [Setting Up Desktop Products \(p. 112\)](#).

For all the details for web products, see [Setting Up Web Products \(p. 123\)](#).

Quick Reference for Amazon S3 Products

Topics

- [Desktop Products \(p. 108\)](#)
- [Web Products \(p. 109\)](#)

The purpose of this section is to bring summary information about identifiers and authentication for desktop and web products together in one convenient location. The details are given elsewhere in this guide as indicated.

Desktop Products

Your desktop product uses identifiers for itself and for customers. For information about when the desktop product uses each of the identifiers, see [Authentication and Parameter Requirements \(p. 108\)](#).

Product Identifiers

Identifier	When Obtained
product token	When you register your product (for more information, see Registering Your Product (p. 60)).

Your Identifiers

Your desktop product uses none of your own identifiers. Your Access Key ID and Secret Access Key are not used at all by desktop products. For information about your Access Key ID and Secret Access Key, see [Your AWS Access Credentials \(p. 154\)](#).

Customer Identifiers

Identifier	When Obtained
Access Key ID and Secret Access Key	When your desktop product calls ActivateDesktopProduct (p. 173) . Note The Access Key ID and Secret Access Key work only with AWS service calls associated with Amazon DevPay. They can't be used for regular AWS service calls.
activation key	When the customer buys your product or goes to the <i>activate URL</i> . The desktop product then gets the activation key from the customer.
user token	When your desktop product calls <code>ActivateDesktopProduct</code> . The user token is <i>not</i> the customer's Access Key ID.

Authentication and Parameter Requirements

Desktop products make calls to both the License Service and Amazon S3. The authentication and parameter requirements are summarized in the following table.

Note

The License Service actions typically require other parameters in addition to those listed in the following table. For a complete list of parameters that an action uses, see the action description in the API reference.

Service or Action	Authentication and Parameter Requirements
ActivateDesktopProduct	REST-Query and SOAP calls do not need to be signed. The requests must be made over HTTPS. The request must include the product token and the activation key. The response includes the customer's user token, Secret Access Key, and Access Key ID.
VerifyProductSubscriptionByTokens (p. 194)	REST-Query calls must include the <i>customer's</i> Access Key ID and be signed using the <i>customer's</i> Secret Access Key. The requests must be made over HTTPS. SOAP requests are not allowed for this action. The request must include the product token and the customer's user token. The response contains a Boolean value.
Amazon S3	REST calls must include the <i>customer's</i> Access Key ID and be signed using the <i>customer's</i> Secret Access Key. They must not use your own Access Key ID and Secret Access Key. The REST request must include the product token and the customer's user token. Important Amazon S3 requests that use DevPay must be REST requests or pre-signed URLs; SOAP requests are not supported for DevPay.

Web Products

Your web product uses identifiers for itself, for you, and for customers. For information about when your web product uses each of the identifiers, see [Authentication and Parameter Requirements \(p. 110\)](#).

Product Identifiers

Identifier	When Obtained
product code	When you register your product (for more information, see Registering Your Product (p. 60)).
product token	When you register your product.

Your Identifiers

Identifier	When Obtained
Access Key ID and Secret Access Key	When you created the AWS developer account that you used to register your product (for more information, see Your AWS Access Credentials (p. 154)).

Customer Identifiers

Identifier	When Obtained
activation key	When the customer buys your product or goes to the activate URL. Your web product then gets the activation key from the customer (alternately, your system can get it directly from AWS; for more information, see The Activation Key (p. 125)).
persistent identifier (PID)	When your web product calls <code>ActivateHostedProduct</code> (p. 177).
user token	When your web product calls <code>ActivateHostedProduct</code> .

Unlike desktop products, web products do not use a Secret Access Key and Access Key ID *for the customer*; only *your* Secret Access Key and Access Key ID are used.

Authentication and Parameter Requirements

Web products make calls to both the License Service and Amazon S3. The authentication and parameter requirements are summarized in the following table.

Note

The License Service actions typically require other parameters in addition to those listed in the following table. For a complete list of parameters that an action uses, see the topic for the action.

Service or Action	Authentication and Parameter Requirements
<code>ActivateHostedProduct</code>	<p>REST-Query calls must include your Access Key ID and be signed using your Secret Access Key. The requests must be made over HTTPS.</p> <p>SOAP requests must use WS-Security and your X.509 certificate. The requests must be made over HTTPS.</p> <p>The request must include the product token and the activation key. The response includes the customer's user token and persistent identifier (PID).</p>
GetActiveSubscriptionsByPid (p. 182)	<p>REST-Query calls must include your Access Key ID and be signed using your Secret Access Key. The requests must be made over HTTPS.</p> <p>SOAP requests must use WS-Security and your X.509 certificate. The requests must be made over HTTPS.</p> <p>The request must include the customer's persistent identifier (PID). The response contains a list of product codes.</p>
VerifyProductSubscriptionByPid (p. 190)	<p>REST-Query calls must include your Access Key ID and be signed using your Secret Access Key. The requests must be made over HTTPS.</p> <p>SOAP requests must use WS-Security and your X.509 certificate. The requests must be made over HTTPS.</p> <p>The request must include the product code and the customer's persistent identifier (PID). The response contains a Boolean value.</p>

Service or Action	Authentication and Parameter Requirements
Amazon S3	<p>REST calls and pre-signed URLs must include your Access Key ID and be signed using your Secret Access Key (like regular Amazon S3 REST calls or pre-signed URLs that don't use DevPay).</p> <p>The REST request must contain the customer's user token. The product token is required only if your web product is using a user token created on or before May 15, 2008. Calls to <code>ActivateHostedProduct</code> made after that date return a new type of user token that eliminates the need to include the product token in the Amazon S3 request. Pre-signed URLs your web product creates must include this new type of user token and should <i>not</i> include the product token.</p> <p>Important Amazon S3 requests that use DevPay must be REST requests or pre-signed URLs; SOAP requests are not supported for DevPay.</p>

Setting Up Desktop Products

Topics

- [Prerequisites \(p. 112\)](#)
- [Overall Authentication Process \(p. 112\)](#)
- [Desktop Product Activation \(p. 114\)](#)
- [Making Amazon S3 REST Calls with Desktop Products \(p. 117\)](#)
- [Query String Authentication with Desktop Products \(p. 118\)](#)
- [Amazon S3 POST with Desktop Products \(p. 118\)](#)
- [Desktop Product Exceptions \(p. 119\)](#)
- [Verification of the Customer's Subscription Status \(p. 120\)](#)

This section describes how desktop products work with Amazon DevPay.

The *Amazon DevPay Getting Started Guide* discusses sample code that demonstrates how to update a basic Amazon Simple Storage Service library to work with DevPay.

Prerequisites

To integrate your desktop product with Amazon DevPay, you must have the *purchase URL* and the *product token* for your product, which you receive during product registration. Your product needs to use them as described in the subsequent sections. For more information about product registration, see [Registering Your Product \(p. 60\)](#).

Overall Authentication Process

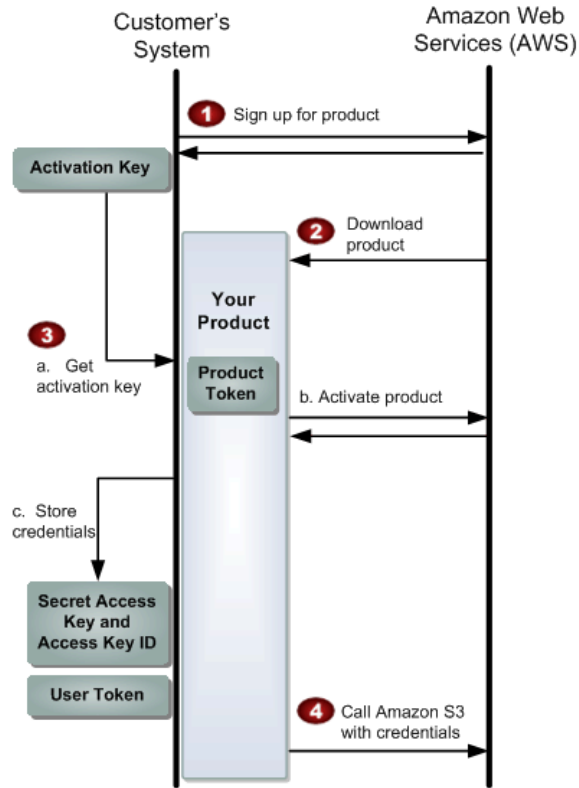
Much of the development required to make your product work with Amazon DevPay involves customer authentication. Your product must integrate with the License Service, which provides your product with the necessary authentication credentials for the customer. Your product then uses those credentials and the product token when making a request for Amazon S3 on behalf of that customer. This enables Amazon to bill the customer for the usage of your product and the Amazon S3 requests the product makes.

Important

Your DevPay product must create a separate bucket in Amazon S3 for each customer who buys and uses the product. Each DevPay product can create up to 100 buckets per customer. For example, a customer who uses three different DevPay products can have up to 300 DevPay buckets, plus any other buckets created outside of DevPay (i.e., those created with a personal AWS account).

Once your product has created a bucket and put objects in it, only your product can access that bucket and the objects in it. For more information about restrictions on data access, see [Customer Access Stored Data \(p. 11\)](#).

The process for customer authentication is described in the following diagram and corresponding steps.



Overall Authentication Process for Desktop Products

1	<p>The customer signs up for the product by clicking the <i>purchase URL</i> you received during product registration, and as part of the process, the customer:</p> <ul style="list-style-type: none"> • Logs in with an Amazon.com login (or creates a new login; note that this is a regular Amazon.com login and <i>not</i> an AWS developer account) • Signs up to use your product • Obtains an <i>activation key</i> required by your product
2	<p>The customer downloads and installs your product.</p>
3	<p>As part of the installation, or immediately after, the product goes through an activation process:</p> <ol style="list-style-type: none"> Your product obtains the activation key from the customer. For more information, see The Activation Key (p. 114). Your product sends a request to the License Service to activate itself and obtain a Secret Access Key, Access Key ID, and <i>user token</i> for the customer. Your request includes the product token for your product and the customer's activation key. For more information, see The Request for Activation (p. 115). Your product appropriately stores the credentials it has received so the customer can use the product seamlessly in the future. For more information, see Credential Storage (p. 115).

4 Later, when the customer uses the product, the product makes an Amazon S3 REST request on behalf of the customer. In the process, the product retrieves and uses the Secret Access Key, Access Key ID, user token, and the product token for the product. For more information, see [Making Amazon S3 REST Calls with Desktop Products \(p. 117\)](#).

Note
Amazon S3 requests that use DevPay must be REST requests. SOAP requests are not supported for DevPay. Alternately, your desktop product could use Amazon S3 POSTs or pre-signed URLs to access Amazon S3, but for security reasons, we don't recommend you use those with a DevPay desktop product. For more information about those methods, see [Query String Authentication with Desktop Products \(p. 118\)](#) and [Amazon S3 POST with Desktop Products \(p. 118\)](#).

The next sections give additional details about the process.

Desktop Product Activation

Topics

- [The Activation Key \(p. 114\)](#)
- [The Request for Activation \(p. 115\)](#)
- [Credential Storage \(p. 115\)](#)
- [Activation and Subscription Timing \(p. 116\)](#)

Your desktop product must go through a process of *activation* before each customer can use it. This process is part of the overall process desktop products follow to work with Amazon DevPay (for more information, see [Overall Authentication Process \(p. 112\)](#)).

Activation means the product contacts AWS with a key identifying the customer, and AWS replies with credentials the product must use when making Amazon Simple Storage Service requests for that customer. The credentials are valid only for your specific product and for the specific customer. The following sections describe how activation works.

The Activation Key

To purchase your product, the customer goes through a purchase process, which starts when the customer clicks the *purchase URL* (for an example of what the customer sees during the purchase process, see [Appendix: The Customer Purchase Experience \(p. 199\)](#)). At the end of this process, the customer's browser splits into two frames. The top frame contains a confirmation for the purchase. The bottom frame contains the *redirect URL* you provided during product registration.

Also displayed in the top frame of the browser is an *activation key* that contains information identifying the customer and the product. The key looks similar to this: `ADMAY7DVLJTWJ76MMBMOEXAMPLE`.

Your desktop product needs the activation key in order to get credentials the product needs for that customer. The following table describes the typical flow for getting the key.

Process for Getting the Activation Key

1	The redirect URL displays a download link where the customer can download your product.
2	The customer downloads and installs the product.

3	During the installation, the product prompts the customer for the activation key, indicating that it was displayed in the top frame of the browser window. The product also indicates that if the activation key is no longer available (if the customer closed the browser, for example), the customer can get a new key at the <i>activate URL</i> .
4	The customer pastes the key from the browser into the form your product provides. Or if the key isn't available that way, the customer clicks the link to the activate URL, logs in with an Amazon login, gets a new activation key, and pastes it into the form.

Activation keys expire one hour after creation for security reasons.

Important

To successfully activate your product, the activation key your desktop product provides to AWS must be associated with the product token. In other words, do not provide an activation key that a customer obtained when signing up for some other product that uses Amazon DevPay besides yours. Your product should not store activation keys.

The Request for Activation

Once the product has the activation key, it activates itself by requesting the License Service action [ActivateDesktopProduct](#) (p. 173). The request must include the product token and the customer's activation key. The response includes the Secret Access Key, Access Key ID, and user token for the customer. The Secret Access Key and Access Key ID work only with AWS service calls associated with DevPay. They can't be used for regular AWS service calls.

No harm occurs if your product activates itself more than once. Each time the product activates itself, the License Service returns a new Secret Access Key, Access Key ID, and user token for the customer. There might be times when the product needs to reactivate itself (for more information, see [Desktop Product Exceptions](#) (p. 119)). You should design your product so that it can reactivate itself without requiring the customer to reinstall the product. In this reactivation case, the product should overwrite the old credentials with the new credentials received during reactivation.

You can also let a customer install your product on multiple desktops (for example, on a work computer and on a home computer). Each instance of your desktop product that the customer installs needs to activate separately and receive its own set of credentials to use when making Amazon S3 calls for the customer. Activating a second or third computer for a customer doesn't invalidate the credentials from the first computer's activation. The customer can use the same activation key for each computer, or different ones (the customer can get a new activation key at any time by going to the activate URL).

The requests to activate your desktop product do not require any special authentication (but they must be made over HTTPS). If you're creating both a desktop version and web version of your product, be aware that desktop products don't have to authenticate their requests for the License Service, but web products do (for more information about web product activation, see [The Request for Activation](#) (p. 126)).

Credential Storage

Your product should encrypt and store the Secret Access Key, Access Key ID, and user token on the customer's file system.

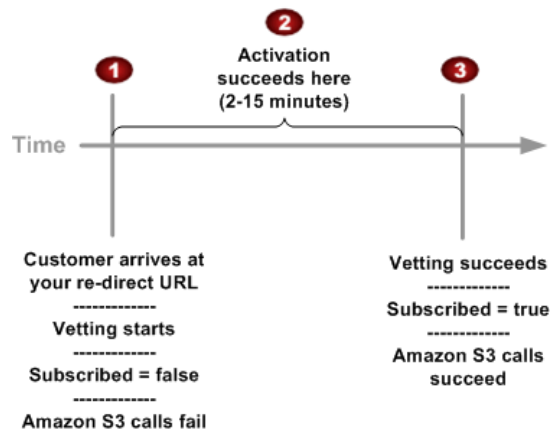
Important

The Secret Access Key, Access Key ID, and user token that your product receives work only for a specific customer and a specific product. If your customer purchases more than one Amazon DevPay desktop product, there will be a separate set of credentials for each product. For example, if customer John Smith purchases DevPay Product A and DevPay Product B, there will be a set of credentials for John Smith associated with Product A, and another set associated with Product B. It doesn't matter if he bought both products from you or if one was sold by another vendor.

You must design your products such that each product separates its credentials on John Smith's system from any other products' credentials.

Activation and Subscription Timing

When customers sign up for your product, they must provide a credit card. However, they're not officially subscribed until we confirm the credit card is valid (a process known as *vetting*). The following diagram and discussion describe the timing of when the customer is officially subscribed.



1	When customers sign up for the product, they're redirected to your URL and receive the activation key. At that point, we start the process of vetting the card. The customer isn't yet officially subscribed to the product.
2	The credit card vetting process usually takes 2 minutes, but can take up to 15 minutes. During this time, your product can activate the customer and get the customer's credentials. However, until the vetting succeeds, any calls your product makes to Amazon S3 on behalf of the customer return an error saying the customer isn't signed up for Amazon S3 (the error is <code>NotSignedUp</code>).
3	When the vetting succeeds, the customer is then officially subscribed to your product. Within a few seconds, Amazon S3 begins to accept your product's requests without returning the <code>NotSignedUp</code> error.

You should design your product to activate the customer and get the customer's credentials immediately after the customer is redirected to your URL. We recommend immediate activation because activation keys have a limited lifetime (one hour).

Once your product has activated the customer, it should wait until the customer is officially subscribed before sending any requests to Amazon S3. The product can determine the subscription status by polling `VerifyProductSubscriptionByTokens` at a regular interval (e.g., 30 seconds). Until the customer is officially subscribed, the action returns `false` for the subscription status. For more information about the action, see [VerifyProductSubscriptionByTokens](#) (p. 194).

During the credit card vetting period, if customers go to their Application Billing page (at <http://www.amazon.com/dp-applications>), they see a message that says "Authorizing your account to access this application."

If the vetting fails, the customer receives an e-mail (see [If the Validation of the Customer's Credit Card Fails](#) (p. 207)). The customer needs to update the payment method with a valid card. Once the payment

method is updated, we then vet the new information and switch the customer's subscription status to `true` (assuming the vet succeeds). For information about how customers update the payment method, see [Where Customers Manage the Payment Method \(p. 55\)](#).

Making Amazon S3 REST Calls with Desktop Products

Your product will make calls to the Amazon Simple Storage Service on behalf of a customer. These calls are part of the overall process desktop products follow to work with Amazon DevPay (for more information, see [Overall Authentication Process \(p. 112\)](#)).

Your product's Amazon S3 requests must be REST requests. Amazon S3 SOAP requests are not supported with DevPay. This section describes how to make REST requests.

Alternately, your desktop product could use Amazon S3 POSTs or pre-signed URLs to access Amazon S3, but for security reasons, we don't recommend you use those with a DevPay desktop product. For more information about those methods, see [Query String Authentication with Desktop Products \(p. 118\)](#) and [Amazon S3 POST with Desktop Products \(p. 118\)](#).

Making a REST request to Amazon S3 with DevPay is similar to making a REST request for Amazon S3 without DevPay. These are the differences:

- Your product must include the product token and the user token in the Amazon S3 REST request
- Your product must sign the request using *the customer's* Secret Access Key and provide the customer's Access Key ID in the request instead of your own

Caution

Do not distribute your *own* Secret Access Key with your product.

Adding the Tokens to Amazon S3 REST Requests

Each request for Amazon S3 that your desktop product makes on behalf of a customer must be a REST request that includes the product token for your product and the user token for the customer. You include the tokens in the REST request by adding two `x-amz-security-token` headers: one to hold the product token, and one to hold the user token.

The following example shows a basic Amazon S3 PUT request that a product registered with DevPay could make on behalf of a customer.

```
PUT/ bucketname/objectname HTTP/1.0
Content-Length: 0
Host: s3.amazonaws.com
Date: Wed, 12 Dec 2007 03:40:41 GMT
Authorization: AWS 0GS7553JW74RRM612K02EXAMPLE:frJIUN8DYpKDtOL
Cwo//y1lqDzgEXAMPLE=
x-amz-security-token: {UserToken}AAAHVXN1clRrbgfOpSykBAXO7g/zG...[long encoded token]...
x-amz-security-token: {ProductToken}MIIBzTCCATagAwIBAgIGARBlqe...[long encoded token]...
```

Alternately, you can add a single `x-amz-security-token` header with the product token and user token separated by a comma.

Amazon S3 returns two errors related to the user token: `ExpiredToken` and `InvalidToken`. For more information about the errors Amazon S3 returns, see the [Amazon Simple Storage Service Developer Guide](#).

Signing the Amazon S3 REST Request

DevPay requests for Amazon S3 are not anonymous, so they require authentication. In general, you sign REST requests for Amazon S3 with DevPay the same way you sign REST requests without DevPay.

The main difference between REST requests with and without DevPay is whose credentials you use for the signature. For requests without DevPay, you use your own credentials. For requests with DevPay, you use the customer's credentials. This means you include the customer's Access Key ID instead of your own in the `Authorization` header, and you use the customer's Secret Access Key to create the signature.

The Amazon S3 documentation instructs you to include all headers that start with `x-amz` in the string to sign. This means that you must include the two `x-amz-security-token` headers when creating the signature. For more information, see the [Amazon Simple Storage Service Developer Guide](#).

Query String Authentication with Desktop Products

Amazon Simple Storage Service users can give direct third-party access to their private Amazon S3 data, without proxying the request. They do this by constructing a "pre-signed" request and encoding it as a URL that can be used in a browser. This feature is commonly referred to as *query string authentication* or *pre-signed URLs*. For complete information about the Amazon S3 query string authentication feature, see the [REST authentication section](#) in the *Amazon Simple Storage Service Developer Guide*.

Caution

Query string authentication is designed for use by web products, not desktop products. Although it's technically possible that your desktop product could use the feature, we strongly discourage it because it's a less secure way for your desktop product to access a customer's bucket. Any request your desktop product makes using query string authentication exposes the product token in clear text, enabling anyone to get the product token and use it maliciously to your detriment. Using a regular call without query string authentication doesn't expose the product token in clear text, making it the more secure option. Web products can make calls using query string authentication without including the product token if they're using a user token created after May 15, 2008. Desktop products cannot.

If you have a desktop product but still want to use query string authentication, we recommend you enable a server to run a web (hosted) interface to your product, set up the server to handle query string authentication, and have your customers use that web interface. Thereafter, to use your product, the customers could use their desktop client or the web interface (their subscription covers both). In this case, you need to activate the customer's use of the product twice: once with [ActivateDesktopProduct \(p. 173\)](#) (for use with the desktop client) and once with [ActivateHostedProduct \(p. 177\)](#) (for use with the web interface). For more information about creating a DevPay web product, see [Setting Up Web Products \(p. 123\)](#). For information about using query string authentication with a web product, see [Query String Authentication with Web Products \(p. 130\)](#).

Amazon S3 POST with Desktop Products

Amazon Simple Storage Service users can upload content directly to Amazon S3 using a web browser and a POST request. For complete information about the Amazon S3 POST feature, see the [section on browser-based uploads using POST](#) in the *Amazon Simple Storage Service Developer Guide*.

Caution

The Amazon S3 POST feature is designed for use by web products, not desktop products. Although it's technically possible that your desktop product could use the feature, we strongly discourage it because it's a less secure way for your desktop product to put data in a customer's bucket. An Amazon S3 POST by your desktop product exposes the product token in clear text, enabling anyone to get the product token and use it maliciously to your detriment. Using a regular PUT call instead doesn't expose the product token in clear text, making it the more secure option.

Web products can do a POST without including the product token if they're using a user token created after May 15, 2008. Desktop products cannot.

If you have a desktop product but still want to use the Amazon S3 POST feature, we recommend you enable a server to run a web (hosted) interface to your product, set up the server to handle Amazon S3 POSTs, and have your customers use that web interface. Thereafter, to use your product, the customers could use their desktop client or the web interface (their subscription covers both). In this case, you need to activate the customer's use of the product twice: once with [ActivateDesktopProduct \(p. 173\)](#) (for use with the desktop client) and once with [ActivateHostedProduct \(p. 177\)](#) (for use with the web interface). For more information about creating a DevPay web product, see [Setting Up Web Products \(p. 123\)](#). For information about using an Amazon S3 POST with a web product, see [Amazon S3 POST with Web Products \(p. 132\)](#).

Desktop Product Exceptions

The following table lists several types of exceptions that your product must handle.

Exception	Description	How to Handle
Errors related to the activation key	When the product tries to activate itself, the activation key the customer provides could be expired, invalid, or incorrect. Activation keys expire one hour after creation.	For information about handling these situations, see Errors (p. 169) .
Invalid or expired user token	If the user token is invalid or expired, the Amazon Simple Storage Service returns an error.	Reactivate the product (see the following procedure).
Missing user token, Secret Access Key, or Access Key ID	When the product tries to retrieve the customer's credentials, any of them could be missing.	Reactivate the product (see the following procedure).
Authentication Error	When AWS authenticates an Amazon S3 request, an authentication error could occur (e.g., if the wrong key is used or the credentials are corrupted).	Reactivate the product (see the following procedure).

If at any time the product consistently receives an error, the product should attempt to reactivate itself.

Reactivating the Product

Your product must be prepared to reactivate itself when certain exceptions listed in the preceding table occur.

To reactivate the product

1. Prompt the customer to go to the *activate URL* to get a new activation key (for an example of this page, see [The Application Activation Page \(p. 203\)](#)).
2. Prompt the customer for the new activation key.
3. Request [ActivateDesktopProduct \(p. 173\)](#) with the new activation key and other required parameters.
4. Delete the old user token, Secret Access Key, and Access Key ID for that customer.

5. Store the new user token, Secret Access Key, and Access Key ID appropriately and use them for future Amazon S3 requests for that customer.

Verification of the Customer's Subscription Status

When a customer contacts your company for support, the customer service representative might want to verify whether the customer is still subscribed to your desktop product (customers can cancel use of your product at any time). This information is available from AWS programmatically. This section describes how your system gets the information.

Process for Verifying the Customer's Subscription Status

1	When your customer completes the process to buy your product, you get the activation key for that customer. You can get the activation key in different ways. For more information, see Obtaining the Activation Key (p. 120) .
2	You then request ActivateHostedProduct (p. 177) to obtain a <i>persistent identifier (PID)</i> for the customer. You store the PID. Note You might be thinking that for desktop products, you're supposed to call <code>ActivateDesktopProduct</code> and not <code>ActivateHostedProduct</code> . This is true if your goal is to get the credentials your desktop product needs so it can make AWS requests on behalf of the customer. If your goal is to get a customer PID (which is required to verify the customer's status), then you need to call <code>ActivateHostedProduct</code> . Your desktop product installed on the customer's system must always call <code>ActivateDesktopProduct</code> to get the credentials it needs. Separately, you can optionally call <code>ActivateHostedProduct</code> to get a PID.
3	Later, when your customer support representative needs to verify the status of the customer's subscription, the representative looks up the PID. Your system makes a request to VerifyProductSubscriptionByPid (p. 190) with the PID and the <i>product code</i> for the product in question. The action's response indicates whether the customer is still subscribed or has canceled use of your product.

Note

The PID represents the relationship between you (the product owner) and the customer. Even if a customer buys 10 different products from you, you have only one PID for that customer.

Obtaining the Activation Key

The following table describes the ways your application can get the activation key.

Method	Description
From the <i>redirect URL</i>	<p>You can obtain the activation key without interacting with the customer. At the end of the sign-up process, the customer's browser window splits into two frames. The top frame contains a confirmation for the purchase. The bottom frame contains the redirect URL that you provided when you registered your desktop product with DevPay, with two values automatically appended as query parameters: the <i>activation key</i> you need and the product code for the desktop product the customer just purchased.</p> <p>For example, if your redirect URL is <code>http://www.example.com/productActivation.html</code>, the customer would be redirected to <code>http://www.example.com/productActivation.html?ActivationKey=<activation key value>&ProductCode=<product code value></code>.</p> <p>You provide an application that retrieves the two values appended to the redirect URL. The application looks up the product token associated with the product code. Then the application makes a signed request to the License Service action <code>ActivateHostedProduct</code> to get the customer's PID.</p> <p>If you decide to use this method to retrieve the activation key, you should design your application to handle the situation when the redirect URL fails. For more information, see Exception Handling for the Redirect URL (p. 121).</p>
From the customer	<p>You can obtain the activation key by prompting the customer to provide it. The customer can get it from the top frame of the browser; the key is automatically displayed there. The customer can also generate an activation key at any time by going to the <i>activate URL</i>. You should provide customers that URL as an alternate method for obtaining the activation key for you. They'll be asked to log in with an Amazon.com login.</p>

Note

After your customers purchase your desktop product, they download and install it. During the installation, the installed product prompts the customer for an activation key (for more information, see [Desktop Product Activation \(p. 114\)](#)). It doesn't matter if the customer provides a different activation key than the one your other application obtained and used in the `ActivateHostedProduct` request (see step 2 in the preceding process). The activation key is just a timestamped, short-lived value that represents the relationship between the customer and your product. At any time, you can ask the customer to go to <http://amazon.com/dp-activate>, log in with an Amazon.com login, and obtain another activation key.

Exception Handling for the Redirect URL

Although the failure rate for the redirect is expected to be low, your application must be prepared to handle any failure cases. It's plausible the customer could close the browser window and prevent your application from retrieving the activation key and product code from the redirect. Your application must handle the situation in which one or both of the parameters are not available through the redirect. See the suggested actions in the following table.

Amazon DevPay Developer Guide
Setting Up Desktop Products

Activation Key Retrieved	Product Code Retrieved	Suggested Action
No	Yes	<p>Your web page should display the name of the desktop product and prompt the customer for the activation key, which was displayed in the top frame of the browser window.</p> <p>The customer can then either provide the activation key from the top frame or obtain a new activation key by clicking the activate URL.</p> <p>In either case, you should provide a way for the customer to paste the activation key into a form on your page.</p>
Yes	No	<p>Your web page should display a list of your DevPay products and ask the customer to select the product just purchased (the one the activation key is associated with).</p> <p>The customer can then select the product, enabling your application to retrieve the product token to use in a request for <code>ActivateHostedProduct</code>. If the customer chooses the wrong product, the request fails.</p>
No	No	<p>One reason this situation occurs is if the customer closes the browser window.</p> <p>Your web page should display a list of your DevPay products and ask the customer to select the product just purchased (the one the activation key is associated with). The page should also prompt the customer for the activation key, which was displayed in the top frame of the browser window.</p> <p>The customer can then either provide the activation key from the top frame or obtain a new activation key by clicking the activate URL that you display.</p> <p>In either case, you should provide a way for the customer to paste the activation key into a form on your page.</p>

Setting Up Web Products

Topics

- [Prerequisites \(p. 123\)](#)
- [Overall Authentication Process \(p. 123\)](#)
- [Web Product Activation \(p. 125\)](#)
- [Making Amazon S3 REST Calls with Web Products \(p. 128\)](#)
- [Query String Authentication with Web Products \(p. 130\)](#)
- [Amazon S3 POST with Web Products \(p. 132\)](#)
- [Web Product Exceptions \(p. 135\)](#)
- [Verification of the Customer's Subscription Status \(p. 135\)](#)

This section describes how web products work with Amazon DevPay.

Prerequisites

To integrate your web product with Amazon DevPay, you must have the following items:

- The *purchase URL*, *product code*, and *product token* for your product, which you receive during product registration. Make sure to associate the product code with the product token in your system, so you can use the product code to look up the product token. For information about product registration, see [Registering Your Product \(p. 60\)](#).
- Your AWS Secret Access Key and accompanying Access Key ID. You'll be using these credentials to authenticate requests for the License Service and Amazon Simple Storage Service. You received the AWS Secret Access Key and Access Key ID when you registered with Amazon Web Services as a developer. For more information, see [Your AWS Access Credentials \(p. 154\)](#).

Overall Authentication Process

Much of the work required to make your product work with Amazon DevPay involves customer authentication. Your product must integrate with the License Service, which provides your product with a user token for the specific customer. Your product then includes that user token and the product token when making a request for the Amazon Simple Storage Service on behalf of that customer. This enables Amazon to bill the customer for the usage of your product and the Amazon S3 requests the product makes.

Important

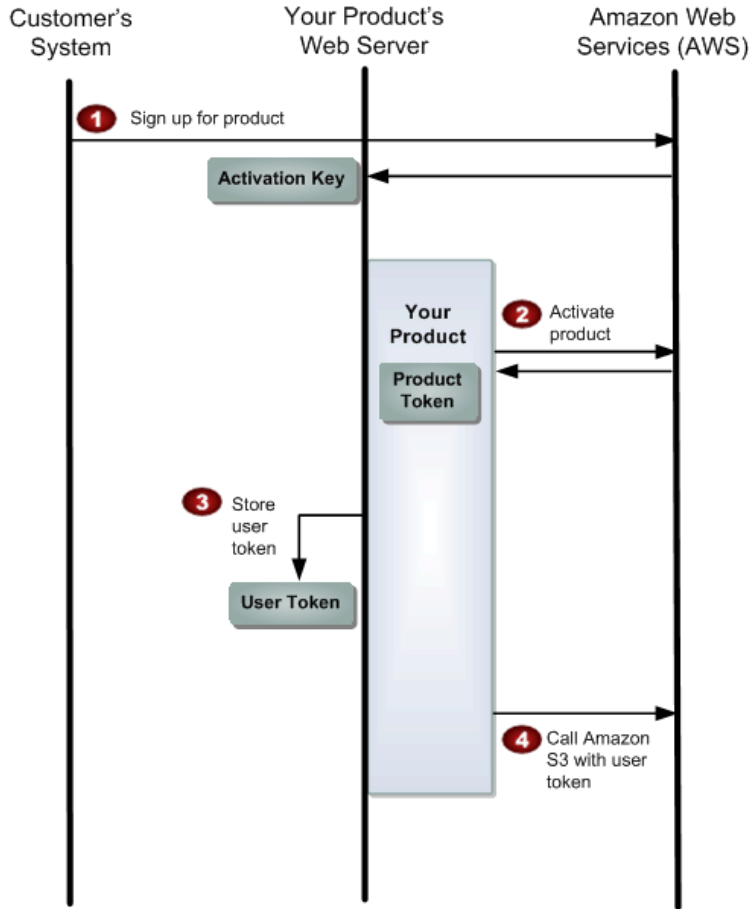
Your DevPay product must create a separate bucket in Amazon S3 for each customer who buys and uses the product. Each DevPay product can create up to 100 buckets per customer. For example, a customer who uses three different DevPay products can have up to 300 DevPay buckets, plus any other buckets created outside of DevPay (i.e., those created with a personal AWS account).

Once your product has created a bucket and put objects in it, only your product can access that bucket and the objects in it. For more information about restrictions on data access, see [Customer Access Stored Data \(p. 11\)](#).

Important

It's your responsibility to design your web product so it can recognize each customer who returns to your site and retrieve the user token associated with that customer.

The process for customer authentication is described in the following diagram and corresponding steps.



Overall Process of Authentication for Web Products

1	The customer signs up for the product by clicking the <i>purchase URL</i> you received during product registration. When the customer completes the purchase, AWS generates an activation key for that customer and makes it available to your server. For more information, see The Activation Key (p. 125) .
2	Your product sends an authenticated request to the License Service to activate itself and obtain a <i>user token</i> for the customer. The request includes the product token for your product and the activation key. For more information, see The Request for Activation (p. 126) .
3	Your product appropriately stores the user token it has received. For more information, see Storage of the User Token (p. 127) . Your product should associate the user token with the customer who is logged in to your web product.

4 Later, when the customer uses the product, the product makes an Amazon S3 REST request on behalf of the customer. In the process, the product retrieves and includes the customer's user token and the product token in the request. For more information, see [Making Amazon S3 REST Calls with Web Products \(p. 128\)](#).

Note
Amazon S3 requests that use DevPay must be REST requests or pre-signed URLs; SOAP requests are not supported for DevPay. The product token is optional in REST requests if you have the new version of the user token that `ActivateHostedProduct` began returning after May 15, 2008. Pre-signed URLs must include this new version of the user token and should not include the product token.

The next sections give additional details about the process.

Web Product Activation

Topics

- [The Activation Key \(p. 125\)](#)
- [The Request for Activation \(p. 126\)](#)
- [Storage of the User Token \(p. 127\)](#)
- [Exception Handling for the Redirect URL \(p. 127\)](#)
- [Activation and Subscription Timing \(p. 127\)](#)

Your web product must go through a process of *activation* before each customer can use it. This process is part of the overall process web products follow to work with Amazon DevPay (for more information, see [Overall Authentication Process \(p. 123\)](#)).

Activation means the product contacts AWS with a key identifying the customer, and AWS replies with a user token the product must use when making Amazon Simple Storage Service requests for that customer. The user token is valid only for your specific product and for the specific customer. The following sections describe how activation works.

The Activation Key

To purchase your product, the customer goes through a purchase process, which starts when the customer clicks the *purchase URL* (for an example of what the customer sees during the purchase process, see [Appendix: The Customer Purchase Experience \(p. 199\)](#)). At the end of this process, the customer's browser splits into two frames. The top frame contains a confirmation for the purchase. The bottom frame contains the *redirect URL* you provided during product registration.

When the purchase is complete, AWS generates an *activation key* that contains information identifying the customer and the product. The key looks similar to this: `ADMAY7DVLJTWJ76MMBEXAMPLE`.

Your web product needs the activation key in order to get the user token for the customer. The following table describes the different ways AWS makes the key available; you can use any of the ways to get the activation key.

Method	Description
Displayed in the top frame of the customer's browser	AWS automatically displays the key in the top frame of the browser. The first time the customer uses your web product, it prompts the customer to copy and paste the key from the browser window into a form the product provides.

Method	Description
Appended to the redirect URL	<p>AWS automatically appends the activation key to the redirect URL as a query parameter, along with the <i>product code</i>.</p> <p>For example, if your redirect URL is <code>http://www.example.com/productActivation.html</code>, the customer would be redirected to <code>http://www.example.com/productActivation.html?ActivationKey=<activation key value>&ProductCode=<product code value></code>.</p> <p>You have an application that retrieves the activation key and product code. This method is invisible to the customer and provides a friendlier experience. However, if you choose this method to obtain the activation key, you must design your system to handle some possible exceptions. For more information, see Exception Handling for the Redirect URL (p. 127).</p>
Provided at the <i>activate URL</i>	<p>AWS provides a URL (http://www.amazon.com/dp-activate) where the customer can go to obtain a new activation key at any time (for an example of this page, see The Application Activation Page (p. 203)). You should display the activate URL any time you prompt the customer for the activation key. If you choose to use one of the other methods in this table, implement this one as an alternate in case the other method fails.</p>

Activation keys expire one hour after creation for security reasons.

Important

To successfully activate your product, the activation key you provide during activation must be associated with the product token. In other words, do not provide an activation key that a customer obtained when signing up for some other product that uses DevPay besides yours. Your product should not store activation keys.

The Request for Activation

Once the application has the activation key and product code, it looks up the product token associated with the product code. The application then makes a signed request to the License Service action [ActivateHostedProduct \(p. 177\)](#). The request must include the product token for the customer and the customer's activation key. The response includes the user token for the customer.

Note

The product calls `ActivateHostedProduct` to activate itself when the customer initially signs in to use the product. The product might require reactivation at other times. For more information, see [Web Product Exceptions \(p. 135\)](#).

The requests for `ActivateHostedProduct` must be authenticated as follows:

- **REST-Query Requests**—With an HTTPS request and an HMAC-SHA1 signature created with your Secret Access Key
 For more information and instructions, see [Authentication of REST-Query Requests \(p. 155\)](#).
- **SOAP Requests**—With an HTTPS request and a digital signature using the standard X.509 WS-Security profile
 Your product must sign the SOAP message (the body and time stamp) with your private key. The request must include your X.509 certificate in the SOAP header. For more information about calling the License Service using SOAP, see [Authentication of SOAP Requests \(p. 162\)](#).

Storage of the User Token

The user token must be available and ready to use each time the product makes an Amazon Simple Storage Service request on behalf of the customer. We recommend your product encrypt the user token and store it securely. If the user token is ever missing, the product must get a new one. For more information, see [Web Product Exceptions](#) (p. 135).

Important

It's your responsibility to design your web product so it can recognize each customer who returns to your site and retrieve the user token associated with that customer.

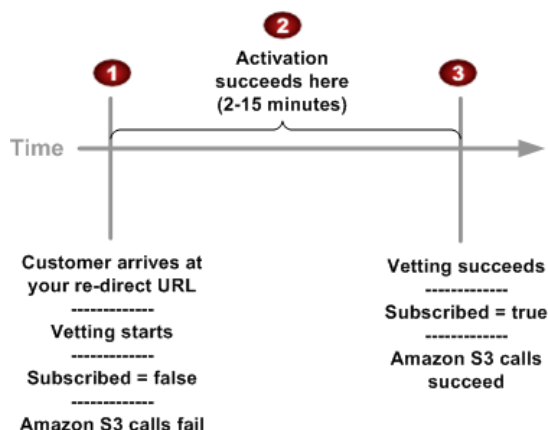
Exception Handling for the Redirect URL

Although the failure rate for the redirect is expected to be low, your application must be prepared to handle any failure cases. It's plausible the customer could close the browser window and prevent your application from retrieving the activation key and product code from the redirect. Your application must handle the situation in which one or both of the parameters are not available through the redirect. See the suggested actions in the following table.

Activation Key Retrieved	Product Code Retrieved	Suggested Action
No	Yes	Your web page displays the name of the web product and prompts the customer for the activation key, which was displayed in the top frame of the browser window. The customer either provides the activation key from the top frame or obtains a new activation key by clicking the activate URL that you display. In either case, you need to provide a way for the customer to paste the activation key into a form on your page.
Yes	No	Your web page should display a list of your DevPay products and ask the customer to select the product just purchased (the one the activation key is associated with). The customer can then select the product, enabling your application to retrieve the product token to use in a request for <code>ActivateHostedProduct</code> . If the customer chooses the wrong product, the request fails.
No	No	Your web page should display a list of your DevPay products and ask the customer to select the product just purchased (the one the activation key is associated with). The page should also prompt the customer for the activation key, which was displayed in the top frame of the browser window. The customer can then either provide the activation key from the top frame or obtain a new activation key by clicking the activate URL that you display. In either case, you should provide a way for the customer to paste the activation key into a form on your page.

Activation and Subscription Timing

When customers sign up for your product, they must provide a credit card. However, they're not officially subscribed until we confirm the credit card is valid (a process known as *vetting*). The following diagram and discussion describe the timing of when the customer is officially subscribed.



1	When customers sign up for the product, they're redirected to your URL and receive the activation key. At that point, we start the process of vetting the card. The customer isn't yet officially subscribed to the product.
2	The credit card vetting process usually takes 2 minutes, but can take up to 15 minutes. During this time, your product can activate the customer and get the customer's credentials. However, until the vetting succeeds, any calls your product makes to Amazon S3 on behalf of the customer return an error saying the customer isn't signed up for Amazon S3 (the error is <code>NotSignedUp</code>).
3	When the vetting succeeds, the customer is then officially subscribed to your product. Within a few seconds, Amazon S3 begins to accept your product's requests without returning the <code>NotSignedUp</code> error.

You should design your product to activate the customer and get the customer's credentials immediately after the customer is redirected to your URL. We recommend immediate activation because activation keys have a limited lifetime (one hour).

Once your product has activated the customer, it should wait until the customer is officially subscribed before sending any requests to Amazon S3. The product can determine the subscription status by polling `VerifyProductSubscriptionByPid` at a regular interval (e.g., 30 seconds). Until the customer is officially subscribed, the action returns `false` for the subscription status. For more information about the action, see [VerifyProductSubscriptionByPid](#) (p. 190).

During the credit card vetting period, if customers go to their Application Billing page (at <http://www.amazon.com/dp-applications>), they see a message that says "Authorizing your account to access this application."

If the vetting fails, the customer receives an e-mail (see [If the Validation of the Customer's Credit Card Fails](#) (p. 207)). The customer needs to update the payment method with a valid card. Once the payment method is updated, we then vet the new information and switch the customer's subscription status to `true` (assuming the vet succeeds). For information about how customers update the payment method, see [Where Customers Manage the Payment Method](#) (p. 55).

Making Amazon S3 REST Calls with Web Products

Your product will make calls to the Amazon Simple Storage Service on behalf of a customer. These calls are part of the overall process web products follow to work with Amazon DevPay (for more information, see [Overall Authentication Process](#) (p. 123)).

Your product's Amazon S3 requests must be REST requests, Amazon S3 POSTs, or *pre-signed URLs*. Amazon S3 SOAP requests are not supported with Amazon DevPay. This section describes how to make REST requests. For information about using pre-signed URLs with your web product, see [Query String Authentication with Web Products \(p. 130\)](#). For information about using Amazon S3 POSTs with your web product, see [Query String Authentication with Web Products \(p. 130\)](#).

Making a REST request to Amazon S3 with DevPay is similar to making a REST request for Amazon S3 without DevPay. The only difference is that your product must include the product token and the user token in the Amazon S3 request. This has implications for forming the request and for signing the request.

Important

For web products only, the product token is optional in the REST request if the web product is using the new version of the user token that [ActivateHostedProduct \(p. 177\)](#) began returning after May 15, 2008. To ensure that a web product user token is the new version, use the [RefreshUserToken \(p. 185\)](#) action.

Adding the Tokens to Amazon S3 REST Requests

Each Amazon S3 REST request that your web product makes on behalf of a customer must include the customer's user token. You include the user token in the REST request by adding an `x-amz-security-token` header. If you need to pass the product token, you can use the same header and include both the user token and the product token separated by a comma. Alternately, you can include a second `x-amz-security-token` header with the product token.

Important

If your web product has a user token created on or before May 15, 2008, each Amazon S3 REST request from your product must include the product token. Calls to `ActivateHostedProduct` made after that date return a new type of user token that eliminates the need to include the product token in the Amazon S3 request.

The following example shows a basic Amazon S3 PUT request that a product registered with DevPay could make on behalf of a customer.

```
PUT/ bucketname/objectname HTTP/1.0
Content-Length: 0
Host: s3.amazonaws.com
Date: Sat, 17 May 2008 03:40:41 GMT
Authorization: AWS OGS7553JW74RRM612K02EXAMPLE:frJIUN8DYpKDtOL
Cwo//y11qDzgEXAMPLE=
x-amz-security-token: {UserToken}AAAHVXN1clRrbgfOpSykBAXO7g/zG...[long encoded
token]...
x-amz-security-token: {ProductToken}MIIBzTCCATagAwIBAgIGARBlqe...[long encoded
token]...
```

Amazon S3 returns two errors related to the user token: `ExpiredToken` and `InvalidToken`. For more information about the errors Amazon S3 returns, see the [Amazon Simple Storage Service Developer Guide](#).

Signing the Amazon S3 REST Request

DevPay requests for Amazon S3 are not anonymous, so they require authentication. You sign the request essentially the same way you sign a request without DevPay. You still use your own Secret Access Key and Access Key ID.

The Amazon S3 documentation instructs you to include all headers that start with `x-amz` in the string to sign. This means that you must include any `x-amz-security-token` headers when creating the signature. For more information, see the [Amazon Simple Storage Service Developer Guide](#).

Query String Authentication with Web Products

Topics

- [Before You Begin](#) (p. 130)
- [Scope of Use](#) (p. 130)
- [Access to Data with Pre-Signed URLs](#) (p. 130)
- [Who Pays for Data Access Using Query String Authentication?](#) (p. 130)
- [Using a Pre-Signed URL](#) (p. 131)

Amazon Simple Storage Service users can give direct third-party access to their private Amazon S3 data, without proxying the request. They do this by constructing a *pre-signed* request and encoding it as a URL that can be used in a browser. This feature is commonly referred to as *query string authentication* or *pre-signed URLs*. This section describes the special work your Amazon DevPay web product must do to use this feature.

Before You Begin

We assume that you are familiar with using Amazon S3 query string authentication. If you're not, go to the [REST authentication section](#) of the *Amazon Simple Storage Service Developer Guide* and read about it, and then come back to this section.

Scope of Use

Query string authentication is designed for use with DevPay web products and not DevPay desktop products. If you currently have a desktop product and want to use query string authentication, we recommend you enable a server to run a web (hosted) interface to your product, set up the server to handle query string authentication, and have your customers use the web interface. Thereafter, to use your product, the customers can either use the desktop client or the web interface (their subscription covers both). In this case, you need to activate the customer's use of the product twice: once with [ActivateDesktopProduct](#) (p. 173) (for use with the desktop client) and once with [ActivateHostedProduct](#) (p. 177) (for use with the web interface).

The primary use case for query string authentication is for GETs in a web browser, although you can also use it programmatically with GETs, PUTs, or DELETEs.

Access to Data with Pre-Signed URLs

Your web product can make the pre-signed URL available to one person, multiple people, or the entire public. The person clicking the URL does *not* need to be subscribed to your DevPay product or use the URL through your DevPay web product. Anyone can click a pre-signed URL to download data from the customer's Amazon S3 bucket. This is an exception to the Amazon DevPay data sharing policy (for more information about data sharing, see [Customer Access Stored Data](#) (p. 11)).

Who Pays for Data Access Using Query String Authentication?

Pre-signed URLs are just another way to download data from the customer's bucket. It might be the customer, or someone else, who uses the pre-signed URL to download data. However, the customer pays the price you specify for downloading objects with your DevPay product (the exception to this is if the bucket is a requester pays bucket; for more information, see [Using Amazon S3 Requester Pays with DevPay](#) (p. 142)). Likewise, you pay AWS for the corresponding Amazon S3 costs that you would for downloading objects for the customer. If your web product makes the pre-signed URL available to the general public (for example), millions of people could click the URL and download the object, causing you and your customer to incur the corresponding costs, which could be significant. Keep this in mind when deciding who your web product gives the pre-signed URL to and how long the URL will be valid.

Downloads that originate with a pre-signed URL are included in the download usage statistics and dollar amount the customer sees on the Application Billing page.

Using a Pre-Signed URL

The following table describes the basic process to use a pre-signed URL with your DevPay web product.

Process for Using a Pre-Signed URL

1	Make sure your web product has the required version of the user token. For more information, see The Required Version of the User Token (p. 131) .
2	Create the URL according to the instructions in the <i>Amazon Simple Storage Service Developer Guide</i> , with the specific changes required for DevPay products. For more information, see Changes to the URL (p. 131) .

The Required Version of the User Token

To make query string authentication work with your web product, you must use a particular version of the web product user token, which [ActivateHostedProduct \(p. 177\)](#) began returning after May 15, 2008. This version of the user token lets you omit the product token from the request, which is desirable because it's a security risk to send the product token in clear text in the URL. Use the [RefreshUserToken \(p. 185\)](#) action to ensure your web product has the required version of the user token.

Your web product can still use the original version of the user token in requests that don't use query string authentication. In that case, the web product must continue to include the product token in the request. The newer user token is compatible with older requests so you don't have to reconstruct them. That is, if you have refreshed the user token, you can still send it together with the product token even though the product token is no longer required.

Changes to the URL

When creating a pre-signed URL for a DevPay bucket, you must add the query parameter `x-amz-security-token` to pass the user token and include the user token when creating the signature. The example that follows shows the `x-amz-security-token` parameter in the request.

The instructions for creating a signature for query string authentication tell you to include any headers that start with `x-amz`. When creating the string to sign, you treat the `x-amz-security-token` query parameter as if it were a header, so that the signature includes the user token. Your web product also must use your Access Key ID and Secret Access Key to create the signature, as it does for all Amazon S3 requests it makes on behalf of the customer.

Important

Remember to URL encode the user token when including it as a query parameter in the URL, but *not* to URL encode it when including it in the string to sign.

Example Request

The following example shows an example pre-signed URL that you would use in a web browser. This URL gets the `/photos/puppy.jpg` object from the customer's bucket. The presence of the user token enables AWS to bill the customer for the GET request. Line breaks have been added and the user token has been shortened to make the URL easier to read.

```
http://johnsmith.s3.amazonaws.com/photos/puppy.jpg
?x-amz-security-token={UserToken}BA4PXRK1...Kczg==
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Signature=vjbyPxybdZaNmGa%2By%3DEXAMPLE
&Expires=1141889120
```

The following example shows the same request in the format that you would use with a programmatic GET.

```
GET /photos/puppy.jpg?x-amz-security-token={UserToken}BA4PXRK1...Kczg==&
AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&
Signature=vjbyPxybdZaNmGa%2By%3DEXAMPLE&
Expires=1141889120 HTTP/1.1

Host: johnsmith.s3.amazonaws.com
```

Amazon S3 POST with Web Products

Topics

- [Before You Begin](#) (p. 132)
- [Scope of Use](#) (p. 132)
- [Access to Buckets with a POST](#) (p. 133)
- [Who Pays for Data Upload Using POST?](#) (p. 133)
- [Changes to the POST for DevPay](#) (p. 133)

Amazon Simple Storage Service users can upload content directly to Amazon S3 using a web browser and a POST request. This section describes the special work your Amazon DevPay web product must do to use this feature.

Before You Begin

We assume that you're familiar with using Amazon S3 POST. If you're not, go to the *Amazon Simple Storage Service Developer Guide* and read the [section on browser-based uploads using POST](#), and then come back to this section.

Scope of Use

The Amazon S3 POST feature is designed for use by DevPay web products and not DevPay desktop products. If you currently have a desktop product and want to use the POST feature, we recommend you enable a server to run a web (hosted) interface to your product, set up the server to handle Amazon S3 POSTs, and have your customers use that web interface. Thereafter, to use your product, the customers could use their desktop client or the web interface (their subscription covers both). In this case, you need to activate the customer's use of the product twice: once with [ActivateDesktopProduct](#) (p. 173) (for use with the desktop client) and once with [ActivateHostedProduct](#) (p. 177) (for use with the web interface).

Access to Buckets with a POST

Your web product can make the POST form available to one person, multiple people, or the entire public. The person using the form does *not* need to subscribe to your DevPay product. This is an exception to the Amazon DevPay data sharing policy (for more information about data sharing, see [Customer Access Stored Data \(p. 11\)](#)).

Who Pays for Data Upload Using POST?

Amazon S3 POST is just another way to upload data to the customer's bucket. It might be the customer, or someone else who uses the POST form to upload data. However, the customer pays the price you specify for uploading objects with your DevPay product. Likewise, you pay AWS for the corresponding Amazon S3 costs that you would for uploading objects for the customer.

Uploads that originate with an Amazon S3 POST are included in the upload usage statistics and dollar amount the customer sees on the Application Billing page.

Changes to the POST for DevPay

With a POST, your web product provides the customer with a web page containing an HTML form. When the customer clicks the button on the page to upload the object, the page POSTs the HTML form to Amazon S3. The form includes a policy that your web product constructs. This policy dictates any restrictions on the content the customer wants to upload. Amazon S3 validates the content against the policy to make sure it meets your conditions.

When creating the policy and the HTML form, your web product must include two additional items required for DevPay products:

- The `x-amz-security-token` field in the HTML form
- The `x-amz-security-token` field in the policy itself

The value for both of the fields must be the user token.

Important

Before the update to Amazon DevPay on May 15, 2008, you were required to provide *both* the user token *and* the product token (separated by a comma) as the value for the `x-amz-security-token` field. You can still do that. However, your web product is no longer required to provide the product token if it's using the user token version that [ActivateHostedProduct \(p. 177\)](#) began returning as of May 15, 2008. Because it's a security risk to send the product token in clear text in the request, we recommend your web product use this newer version of the web product user token and omit the product token when performing POSTs. Use the [RefreshUserToken \(p. 185\)](#) action to ensure you have the latest version of the web product user token.

Example Policy

The following example policy shows the required `x-amz-security-token` field with an example user token (the token has been shortened for readability).

```
{
  "expiration": "2008-05-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "johnsmith"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "http://johnsmith.s3.amazonaws.com/successful_upload.html"},
    {"x-amz-security-token": "{UserToken}BA4PXRK1...Kczg=="},
    ["starts-with", "$Content-Type", "image/"],
    {"x-amz-meta-uuid": "1436512EXAMPLE"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Example HTML Form

The following example HTML form shows the required `x-amz-security-token` field with an example user token (the token has been shortened for readability).

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
    ...
    <form action="http://johnsmith.s3.amazonaws.com/" method="post" enctype="multipart/form-data">
      Key to upload: <input type="input" name="key" value="user/eric/" /><br />
      <input type="hidden" name="acl" value="public-read" />
      <input type="hidden" name="success_action_redirect" value="http://johnsmith.s3.amazonaws.com/successful_upload.html" />
      Content-Type: <input type="input" name="Content-Type" value="image/jpeg" /><br />
      <input type="hidden" name="x-amz-meta-uuid" value="1436512EXAMPLE" />
      Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
      <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
      <input type="hidden" name="Policy" value="<policy goes here>" />
      <input type="hidden" name="Signature" value="<signature goes here>" />
      <input type="hidden" name="x-amz-security-token" value="{UserToken}BA4PXRK1...Kczg==" />
      File: <input type="file" name="file" /> <br />
      <!-- The elements after this will be ignored -->
      <input type="submit" name="submit" value="Upload to Amazon S3" />
    </form>
    ...
  </html>
```

Web Product Exceptions

The following table lists several types of exceptions that your product must handle.

Exception	Description	How to Handle
Errors related to the activation key	When the product tries to activate itself, the activation key the customer provides could be expired, invalid, or incorrect. Activation keys expire one hour after creation.	For information about handling these situations, see Errors (p. 169) .
Invalid or expired user token	If the user token is invalid or expired, the Amazon Simple Storage Service returns an error.	Reactivate the product (see the following procedure).
Missing user token	When the product tries to retrieve the user token, it could be missing.	Reactivate the product (see the following procedure)

If at any time the product consistently receives an error, the product should attempt to reactivate itself.

Reactivating the Product

Your product must be prepared to reactivate itself when certain exceptions listed in the preceding table occur.

To reactivate the product

1. Prompt the customer to go to the *activate URL* to get a new activation key (for an example of this page, see [The Application Activation Page \(p. 203\)](#)).
2. Prompt the customer for the new activation key.
3. Request `ActivateHostedProduct` ([p. 177](#)) with the new activation key and other required parameters.
4. Delete the old user token for that customer.
5. Store the new user token appropriately and use it for future Amazon S3 requests for that customer.

Verification of the Customer's Subscription Status

When a customer contacts your company for support, the customer service representative might want to verify whether the customer is still subscribed to your web product (customers can cancel use of your product at any time). This information is available from AWS programmatically. This section describes how your system gets the information.

Process for Verifying the Customer's Subscription Status

1	The first time the customer goes to your web product in a browser and logs in to use it, your product goes through an activation process (for more information, see Web Product Activation (p. 125)). As part of this process, the product obtains an activation key for the customer and requests <code>ActivateHostedProduct</code> (p. 177).
2	The response from <code>ActivateHostedProduct</code> includes a <i>persistent identifier (PID)</i> for the customer. Your product stores the PID.

3	Later, when your customer support representative needs to verify the status of the customer's subscription, the representative looks up the PID. Your system makes a request to VerifyProductSubscriptionByPid (p. 190) with the PID and the <i>product code</i> for the product in question. The action's response indicates whether the customer is currently subscribed or has canceled use of your product.
---	---

Note

The PID represents the relationship between you (the product owner) and the customer. Even if a customer buys 10 different products from you, you have only one PID for that customer.

Using Amazon S3 Copy with DevPay

The Amazon Simple Storage Service lets you copy an object. For more information about the Amazon S3 copy feature, go to the [section about copying objects](#) in the *Amazon Simple Storage Service Developer Guide*.

To understand how DevPay products can use the copy feature, think of the copy operation as two steps: the DevPay product reads the object from the source bucket and then writes the object to the destination bucket. Amazon S3 requires your product to use the same Secret Access Key and Access Key ID for both steps. Therefore, the DevPay product must have access to the required credentials for the owner of the source bucket and the owner of the destination bucket.

AWS also has rules about data access with DevPay products (for more information, see [Customer Access Stored Data \(p. 11\)](#)). These rules affect how your DevPay product can use the copy feature.

Taking into account the preceding requirements, your product can:

- Copy an object to the same bucket
- Copy an object to another bucket that your product has created and is owned by the same customer
- Copy an object to another bucket that your product has created and is owned by a different customer
This works only if your product created both the source and destination buckets (i.e., both customers used your product to create their buckets) and the required Amazon S3 access control is enabled. For example, Bob can copy an object from Alice's bucket to his own bucket if he and Alice are both using your DevPay product and Alice has given him permission to read her object using the Amazon S3 access control lists. Likewise, Alice can copy her object to Bob's bucket if Bob has given Alice write permission for his bucket.

By contrast, your product cannot:

- Copy an object to a bucket created by another DevPay product, even if you own both products and the same customer owns both buckets
- Copy an object between a bucket created by a DevPay product and a bucket not created by a DevPay product

The customer who sends the PUT request for the copy is charged your product's price for the copy request. The customer who owns the bucket where the copy is stored is charged your product's price for the copy storage. AWS doesn't charge for bandwidth related to copying, so you are charged only for the normal Amazon S3 PUT request and storage charges whenever your customers copy objects.

Using Amazon S3 Logs with DevPay

Topics

- [Before You Begin](#) (p. 138)
- [Overview](#) (p. 138)
- [Who Pays for Log File Storage and Download?](#) (p. 138)
- [Access to Log Files](#) (p. 139)
- [Setting Up and Using Logs](#) (p. 139)
- [Sharing Log Files](#) (p. 139)

This section covers how Amazon DevPay products can produce reports for customers based on the Amazon Simple Storage Service server access logs.

Before You Begin

We assume you understand how Amazon S3 server access logs work and how Amazon S3 access control lists (ACLs) work. If you aren't already familiar with Amazon S3 logs or ACLs, we recommend you read about them in the [Amazon Simple Storage Service Developer Guide](#) first and then come back to this section.

Overview

You might want your Amazon S3 DevPay product to give your customers reports related to their use of Amazon S3. Your product can use the information in the Amazon S3 server access logs to produce those reports. Each record in a bucket's server access log contains details about an individual Amazon S3 request on that bucket, such as the request type, the resource the request worked with, and the date and time the request was processed.

When you enable logging for a bucket (the *source bucket*), you specify where you want the logs to be delivered (the *target bucket*), and Amazon S3 aggregates available log records into log files and delivers them to the target bucket (typically every few hours). The target bucket can be the same as the source bucket or a different bucket. However, the same Amazon S3 user must own both the source and target buckets.

Note

Because Amazon S3 requires the same user to own both the source and target buckets, you can't create a single bucket in Amazon S3 to hold all the log files for all your DevPay customers.

Who Pays for Log File Storage and Download?

Log files are essentially just objects in a customer's bucket, so the customer pays the same price you charge for storing objects and downloading objects with your DevPay product. Likewise, you pay AWS for the corresponding Amazon S3 costs that you would for storing objects for the customer. AWS does not charge to enable logging on a bucket, and there is no data transfer charge for log file delivery to the customer's target bucket.

Because log files are just like other objects in the bucket, any log file storage or access that your DevPay product performs is included in the usage statistics and dollar amount the customer sees on the Application Billing page. You should design your product to access the log files efficiently and delete the files as soon as they are no longer needed so that you minimize costs to the customer.

Access to Log Files

Because log files are essentially just like the other objects in the customer's bucket, your product accesses them on behalf of the customer the same way it accesses any other object in the bucket (using the customer credentials your product obtained from a call to the License Service). Objects in the customer's bucket (including log files) can be accessed only through your DevPay product (for more information, see [Customer Access Stored Data \(p. 11\)](#)).

Your DevPay product can use the Amazon S3 access control functionality to share any object in the customer's bucket with other users of your DevPay product. Likewise, the product can give other users of your product access to the log files. The important thing to remember is that all requests to access content in either the source or target bucket must come from your DevPay product. For example, your customer or another Amazon S3 user who has permission to access data in either bucket can't get access by using the Amazon S3 Firefox plugin. They must use your product.

Setting Up and Using Logs

This section lists the tasks your product must perform to enable and use logs, and to share logs with other Amazon S3 users. It is assumed that the target bucket already exists. For specific instructions on how your product uses the Amazon S3 API to execute these tasks, see the *Amazon Simple Storage Service Developer Guide*.

Basic Process for Enabling Logging and Using Logs

1	Your product sets up the target bucket. This consists of setting the bucket's ACL to grant <code>WRITE</code> and <code>READ_ACP</code> permissions to the Amazon S3 log delivery group. This lets Amazon S3 deliver the logs to the target bucket.
2	Your product enables logging on the source bucket. This consists of modifying the source bucket's <code>BucketLoggingStatus</code> resource.
3	Amazon S3 produces log files and puts them in the target bucket.
4	Your product gets the log files, produces the report for the customer, and deletes the log files.

Sharing Log Files

The following table describes what your product must do to share logs with another user of your product. For specific instructions on how your product uses the Amazon S3 API to execute these tasks, see the *Amazon Simple Storage Service Developer Guide*.

You might want another user of your product to have the ability to read and delete the log files in the target bucket. That user also needs to be able to list the objects in the bucket to know which log files are available to read and delete.

To give the other user these abilities, your DevPay product grants the following Amazon S3 permissions:

- `READ` on the target bucket (which enables listing all the objects in the bucket)
- `WRITE` on the target bucket (which enables deleting the log files and any other objects in the bucket)
- `READ` on the `BucketLoggingResource` for the source bucket (which enables the user to read only the logs—any other objects in the target bucket can't be read)

The following XML snippet shows an example of the `Grant` elements your product adds to the target bucket's ACL to allow the log delivery group to write the log files.

To help you understand how sharing works for log files, this example also shows grants related to sharing log files with another user of your product with the AWS account login `user@example.com`. The two grants are the `READ` and `WRITE` permissions that let the user list the contents of the target bucket and delete objects from the bucket. The XML snippet that follows this one shows the additional grant required to let the user actually read the log files.

For complete information about using Amazon S3 access control policies, go to the *Amazon Simple Storage Service Developer Guide*.

Example XML for Setting Up the Target Bucket

```
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="Group">
    <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
  </Grantee>
  <Permission>WRITE</Permission>
</Grant>
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="Group">
    <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
  </Grantee>
  <Permission>READ_ACP</Permission>
</Grant>

<!-- The following grants let the user list and delete the objects in the
bucket -->
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="AmazonCustomerByEmail">
    <EmailAddress>user@example.com</EmailAddress>
  </Grantee>
  <Permission>READ</Permission>
</Grant>
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="AmazonCustomerByEmail">
    <EmailAddress>user@example.com</EmailAddress>
  </Grantee>
  <Permission>WRITE</Permission>
</Grant>
```

The following XML snippet shows an example of the `BucketLoggingStatus` element your product provides to enable logging on the source bucket. This example:

- Uses the `TargetBucket` element to set the target bucket to `CustomerA_LogBucket`
- Uses the `TargetPrefix` element to add the literal string `CustomerA-access_log-` as a prefix to the name of all the log files

This example also continues with the preceding example of showing how to share log files with the `user@example.com` user. The following XML includes a `READ` grant that lets the user actually read the log files.

For complete information about using the `BucketLoggingStatus` element, go to the *Amazon Simple Storage Service Developer Guide*.

Example XML for Enabling Logging on the Source Bucket

```
<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>CustomerA_LogBucket</TargetBucket>
    <TargetPrefix>CustomerA-access_log-/</TargetPrefix>

    <!-- The following grant lets the user read only the log files in the
target bucket -->
    <TargetGrants>
      <Grant>
        <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:type="AmazonCustomerByEmail">
          <EmailAddress>user@example.com</EmailAddress>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Using Amazon S3 Requester Pays with DevPay

Topics

- [Overview of Amazon S3 Requester Pays \(p. 142\)](#)
- [Using Requester Pays with DevPay to Sell Data \(p. 142\)](#)

This section describes how the Amazon Simple Storage Service Requester Pays feature works with Amazon DevPay.

Overview of Amazon S3 Requester Pays

Typically, the owner of an Amazon S3 bucket pays all the costs associated with that bucket (for storage, data transfer, and requests). However, Amazon S3 lets a bucket owner designate a bucket as a *Requester Pays* bucket. This means that each requester who accesses objects in that particular bucket pays for their own data transfer and request costs. The bucket owner still pays the storage costs.

This scenario is useful for Amazon S3 users who want to let the public access objects in their buckets, but don't want to pay the data transfer and request costs. Essentially the bucket owner designates the bucket as a Requester Pays bucket, and then requesters add the `x-amz-request-payer=requester` header to the GET requests to indicate that they agree to pay for the data transfer and requests. For signed URLs, you include `requestPayer=requester` in the query string. For more information about Requester Pays buckets, go to [Requester Pays Buckets](#) in the *Amazon Simple Storage Service Developer Guide*.

Using Requester Pays with DevPay to Sell Data

The Requester Pays feature (used alone) lets you give other Amazon S3 users access to your data, but you can't make a profit; you can only avoid paying data transfer and request costs.

DevPay (used alone) lets you give access to your data to anyone who is signed up for your product (regardless if they're Amazon S3 users). But because the DevPay bucket isn't a *Requester Pays* bucket, you (as the owner of the bucket) still pay for data transfer and requests (at the DevPay product's price).

You need to use DevPay *together* with a Requester Pays bucket if you want to charge people a premium to download your data (the overall process is described in [Selling Your Data \(p. 146\)](#)). Note that because you're using DevPay, your customers don't have to be Amazon S3 users.

When you register your DevPay product, you specify the price you want customers to pay to download the data. The price could include a monthly fee, a mark-up on the data transfer out costs, and a mark-up on the GET request costs. Because the bucket is a *Requester Pays* DevPay bucket, the customers pay for the data transfer and requests according to your DevPay product's price when they download your data.

Important

No matter how you price your product, keep in mind that you're still responsible for *all* AWS costs your customers incur, including AWS costs for data transfer and requests (even though the bucket is a Requester Pays bucket). In other words, your customers pay you your premium to download the data, but you still pay the basic AWS costs for that download.

Note

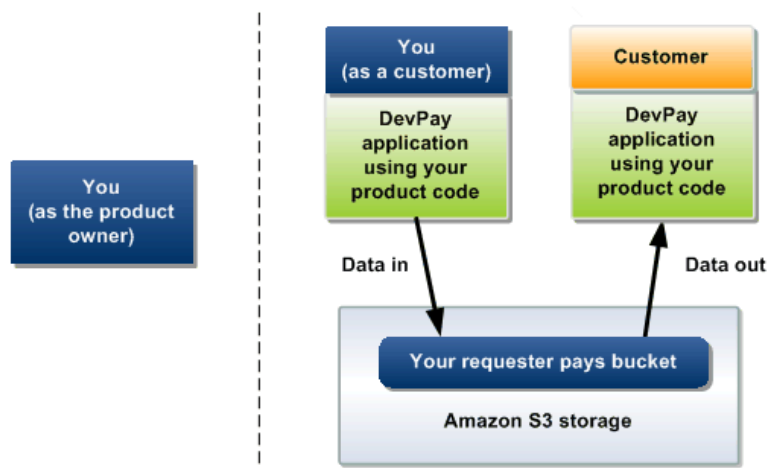
When you use DevPay with your Requester Pays bucket, your customers download your data to a location *outside* Amazon S3. They don't copy the data from your bucket to theirs.

If you want to use DevPay with a Requester Pays bucket to sell your data, there are some important data access and pricing considerations you need to understand.

Data Access

If you want to use DevPay to charge people to download your data, then *everyone* who accesses the bucket must be using the same DevPay application to access it (both you and your customers; for more information, see [Customer Access Stored Data \(p. 11\)](#)). By *same DevPay application*, we mean tools that use the same DevPay *product code*. You and your customers don't all have to be using the same specific tool; for example, you could create a tool for your own use to upload data, and a separate tool for your customers to download data. However, both of these tools must use the same DevPay product code.

This restriction on data access means you must sign up for your own DevPay product and become a customer. So (as illustrated in the following figure), you play two roles: You're the DevPay product owner, and you're a customer who owns the Requester Pays bucket. The fact that you play two different roles has implications for how you price your product.



Product Pricing

To help you understand the pricing considerations, we'll first summarize the roles and responsibilities of the different players.

You (as the DevPay product owner):

- Set the price your customers pay to use your product
- Pay AWS the Amazon S3 costs your customers incur
- Pay a 3% DevPay fee on the value-add for each customer and a \$0.30 fixed transaction fee per customer bill collected
- Are billed monthly for the preceding items (you pay with your Amazon Payments account, with your credit card as a backup if needed)

You (as a customer and the bucket owner):

- Pay the product's price to get the data into the bucket and store it
- Are billed (as a customer) monthly for the data transfer, storage, monthly fee, etc. (you pay with your credit card)

Your customers (who buy the data):

- Pay the product's price to get the data

- Are billed monthly for the costs of getting the data

The important point here is that *you are a customer of your own product*, and you're treated just like any other customer of your product. You (as the customer) pay to use your product according to the price you set (as the product owner). This also means that you (as the product owner) pay AWS costs and DevPay fees based on your use (as a customer), just as you do for any other customer. There is an important difference, however, between you (as a customer) and the other customers of your product. You (as a customer) are only interested in uploading and storing data in your Requester Pays bucket. Your customers are only interested in downloading the data. This difference has implications for how you design your DevPay application and set your product's price.

One way to handle this difference is to design your DevPay product specifically for selling data from a Requester Pays bucket. You could price your product so you (as a customer) don't pay to upload and store the data. The following table shows an example of how you might price your product (for brevity, it includes pricing only for Amazon S3 in the United States). The "Description" column shows the descriptions you could include to help customers understand your product's price (for more information, see [Your Description Page \(p. 229\)](#)).

Dimension	Price	Description
Monthly fee	\$10.00	You can access the data as often as you like for a single low monthly fee, plus data transfer costs
Storage	\$0.00	[Your customers won't see this pricing dimension in the product's price]
Data transfer in	\$0.00	[Your customers won't see this pricing dimension in the product's price]
Data transfer out	\$0.18 per GB	You pay this rate per GB to transfer the data out
PUT or LIST requests	\$0.00	[Your customers won't see this pricing dimension in the product's price]
GET and all other requests	\$0.00	[Your customers won't see this pricing dimension in the product's price]

If the product has a monthly fee, you (as the customer) would be required to pay the monthly fee, just like any other customer. In this case, we assume the monthly fee covers the costs of the GET requests (which we don't charge for).

You could set the prices for storage, data transfer in, and PUT or LIST requests at \$0.00. This means you (as a customer) pay nothing to transfer the data into the bucket and store it there.

You could mark up the price for data transfer out. Note that you could also mark up the price for GET requests (but in this example, we've chosen not to charge for them). So effectively, the price your customers see in this example includes only a \$10.00 monthly fee, and a data transfer out price of \$0.18 per GB.

Note

If a request to get the data in your bucket comes from an Amazon EC2 instance, the requester doesn't pay for data transfer (because data transfer between Amazon S3 and Amazon EC2 is free).

If you were to set your product's price as listed in the preceding table, what would the charges be if Bob were to access 500 GB of data in your Amazon S3-US bucket once in the month of April? Let's assume you upload the data at the beginning of April, and 1000 requests are required to either upload or download the data. For a list of the costs to use Amazon S3, go to <http://aws.amazon.com/s3>.

Amazon DevPay Developer Guide
Using Amazon S3 Requester Pays with DevPay

The following table shows your usage in April as a customer and the corresponding price you pay.

Price Dimension	Price You Pay as Customer
Monthly fee	\$10.00
PUT requests	\$0.00
Data transfer in (for 500 GB)	\$0.00
Storage (for 500 GB for entire month)	\$0.00
Total	\$10.00

The next table shows the corresponding AWS costs you pay as the product owner.

Price Dimension	AWS Costs You Pay as Product Owner
Monthly fee	N/A
PUT requests	\$0.01
Data transfer in (for 500 GB)	\$50.00
Storage (for 500 GB for entire month)	\$75.00
Total	\$125.01

You (as the customer) pay a bill of \$10.00 for April. AWS collects the \$0.30 transaction fee and gives you (the product owner) the remaining \$9.70. There's no value-add in this case, so you (as the product owner) pay no 3% DevPay fee for you (as a customer). You (as the product owner) pay the \$125.01 in AWS costs that you (as a customer) incurred in April to upload and store the data.

Now, let's look at Bob's usage. The following table shows his usage in April and the corresponding price he pays.

Price Dimension	Price Bob Pays
Monthly fee	\$10.00
GET requests	\$0.00
Data transfer out (for 500 GB)	\$90.00
Total	\$100.00

The next table shows the corresponding AWS costs you pay as the product owner.

Price Dimension	AWS Costs You Pay as Product Owner
Monthly fee	N/A
GET requests	\$0.001

Price Dimension	AWS Costs You Pay as Product Owner
Data transfer out (for 500 GB)	\$85.00
Total	\$85.01

Note

AWS rounds up any line item that is less than a penny to one penny. Therefore, the \$0.001 charge for GET requests is rounded up to \$0.01.

Bob pays a bill of \$100.00 for April. AWS collects the \$0.30 transaction fee and gives you (the product owner) the remaining \$99.70. Your value-add for Bob in April is \$100.00 - \$85.01 = \$14.99, so you pay 3% * \$14.99 = \$0.45 in DevPay fees. You also pay the \$85.01 in AWS costs that Bob incurred in April to download the data. Your net profit for Bob in April is \$14.24. You can see that if you get just 10 customers in April, your profits more than cover the costs to upload and store the data in your bucket during the month.

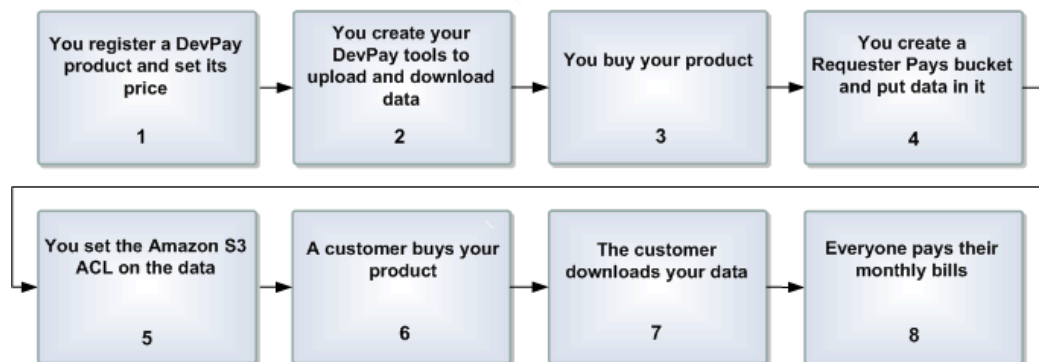
Note

If you hadn't designated your DevPay bucket as a Requester Pays bucket, then you as the bucket owner (and not Bob) would be billed the \$90.00 for the data transfer out. This is why it's important that you use DevPay with a *Requester Pays* bucket, and not just an ordinary bucket.

How would the numbers for this scenario look in future months? If you continue to sell the data that's already in the bucket, then you would incur no new data upload costs and pay only for storage, effectively increasing your profits.

Selling Your Data

The following diagram and table describe the flow for using DevPay with a Requester Pays bucket to sell your data. The process assumes you use a DevPay upload tool and a separate DevPay download tool.



Process for Using DevPay and Requester Pays to Sell Data

1	<p>You register a product with Amazon DevPay and set its price. In return you get the product code. For more information, see Registering Your Product (p. 60).</p>
---	---

2	<p>You create DevPay tools to upload and download data.</p> <p>The upload tool must use your product code and (at a minimum) be able to create a Requester Pays bucket, upload data to it, and set the Amazon S3 ACL on the data.</p> <p>The download tool must use your product code and be able to include the <code>x-amz-request-payer</code> header in the GET requests for your data.</p> <p>For more information, go to Requester Pays Buckets in the <i>Amazon Simple Storage Service Developer Guide</i>.</p>
3	<p>You buy your product, so you can upload your data.</p>
4	<p>You create a Requester Pays bucket and put data in it.</p> <p>As a customer, you use the DevPay upload tool to create the bucket and put your data in it.</p>
5	<p>You set the Amazon S3 ACL on the data. Make sure you set the read permission on the <i>objects</i> in the bucket, and not the bucket itself. We recommend you don't give read permission on the bucket.</p> <p>You use your upload tool to grant read permission to authenticated users. For more information, go to Access Control Lists in the <i>Amazon Simple Storage Service Developer Guide</i>.</p>
6	<p>A customer buys your product.</p>
7	<p>The customer downloads your data.</p> <p>The customer uses your download tool, which includes the <code>x-amz-request-payer</code> header in the GET requests.</p>
8	<p>All the involved parties pay their monthly bills.</p> <p>You (the customer and bucket owner) pay yourself (the product owner) the product's price to upload and manage the data.</p> <p>The customers pay you (the product owner) the product's price to download the data.</p> <p>You (the product owner) pay <i>all</i> the AWS costs that all your customers incurred, plus the applicable DevPay fees. Remember that the AWS costs include the costs you incur as a customer, <i>and</i> the data transfer and request costs your other customers incur.</p>

Related Topics

- [DevPay Fees \(p. 20\)](#)
- [Price Components \(p. 29\)](#)
- [When and How You Get Paid \(p. 37\)](#)

Using Both a Desktop and Web Product

You can create a desktop version of your product, a web version of your product, or both. The web version can be a regular web application with a full user interface, or it can be just a web server that performs the functions you need.

Why would you have both a desktop and web version? Some Amazon S3 features are designed to work with DevPay web products and not desktop products, such as the POST feature and query string authentication. If you have a desktop product but still want to use those features, we recommend you also create a web application or web server to handle them.

Keep the following in mind if you have both a desktop product and web product:

- You use the same product code and product token with both the desktop product and web product
- When customers purchase your product, their subscription covers use of both versions
- You must activate each customer twice: once for the desktop version, and once for the web version (the web version can use an automatic redirect that requires no customer interaction)
- The desktop and web products receive different credentials to make Amazon S3 calls for the customer

Related Topics

- [Amazon S3 POST with Web Products \(p. 132\)](#)
- [Query String Authentication with Web Products \(p. 130\)](#)
- [Desktop Product Activation \(p. 114\)](#)
- [Web Product Activation \(p. 125\)](#)

License Service API Reference

Topics

- [WSDL Location \(p. 149\)](#)
- [Requests \(p. 150\)](#)
- [Request Authentication \(p. 153\)](#)
- [Responses \(p. 167\)](#)
- [Errors \(p. 169\)](#)
- [Actions \(p. 172\)](#)

This section describes the License Service (version 2008-04-28).

WSDL Location

The WSDL for a given version of the License Service can be found at a URL that corresponds to the API version. For example, the current WSDL for the 2008-04-28 version of the License Service can be found here:

<https://ls.amazonaws.com/doc/2008-04-28/AmazonLS.wsdl>

The WSDL includes the description of the request and response messages.

The previous and only other version of the License Service is 2007-06-05. For information about what changed with the 2008-04-28 release, see [the Amazon DevPay release notes](#).

Requests

Topics

- [REST-Query Requests \(p. 150\)](#)
- [SOAP Requests \(p. 151\)](#)

This section describes how to create REST-Query and SOAP requests.

REST-Query Requests

The License Service supports REST-Query requests for calling service actions. REST-Query is also commonly known as Query or HTTP Query. REST-Query requests are simple HTTPS requests, using the GET or POST method with query parameters in the URL. REST-Query requests must contain an *Action* parameter to indicate the action to be performed. The response is an XML document that conforms to the License Service WSDL. You might use REST-Query requests when a SOAP toolkit is not available for your platform, or when REST-Query requests are easier to make than a heavier SOAP equivalent.

Important

The REST-Query requests your product creates to call the License Service are not the same as the REST requests the product creates to call the Amazon Simple Storage Service. Although they are both HTTP requests, the Amazon S3 REST requests use additional HTTP methods (such as DELETE) and HTTP headers. Make sure your product follows the instructions here for forming REST-Query requests when calling the License Service.

Important

All requests to the License Service must be made using HTTPS. HTTP requests are not accepted.

Structure of a REST-Query Request

REST-Query requests are URLs. The following is an example REST-Query request to [ActivateHostedProduct \(p. 177\)](#).

```
https://ls.amazonaws.com/?Action=ActivateHostedProduct&ActivationKey=ADMAY7DVLJTW  
WHJ76MMBMQEXAMPLE&ProductToken={ProductToken}AAAHVXNlclRrbgfOpSykBA...[long  
encoded token]...&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Version=2008-04-28  
&Timestamp=2008-05-19T12:00:00Z&Signature=1B/P67vCvGlDMBQ1dofZxg8E8SUEXAMPLE=
```

Because the preceding format is hard to read, all REST-Query examples in this guide are presented in the following format.

```
https://ls.amazonaws.com/  
?Action=ActivateHostedProduct  
&ActivationKey=ADMAY7DVLJTW  
WHJ76MMBMQEXAMPLE  
&ProductToken={ProductToken}AAAHVXNlclRrbgfOpSykBA...[long encoded token]...  
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE  
&Version=2008-04-28  
&Timestamp=2008-05-19T12:00:00Z  
&Signature=1B/P67vCvGlDMBQ1dofZxg8E8SUEXAMPLE=  
&SignatureVersion=1
```

The first line represents the *endpoint* of the request. This is the resource the request acts on. This will always be `https://ls.amazonaws.com/` for the License Service requests.

After the endpoint is a question mark (?), which separates the endpoint from the parameters. Each parameter is separated by an ampersand (&).

The *Action* parameter indicates the action to perform on the requested endpoint. The actions correspond to the License Service API actions.

Each action uses additional parameters, and they are listed in the topic for each action.

Important

Because REST-Query requests are URLs, you must URL encode the parameter values.

Related Topics

- [Authentication of REST-Query Requests \(p. 155\)](#)
- [Responses \(p. 167\)](#)

SOAP Requests

The License Service supports the SOAP message protocol for calling service actions over an HTTPS connection. The easiest way to use the SOAP interface with your product is to use a SOAP toolkit appropriate for your programming platform. SOAP toolkits are available for most popular programming languages and platforms.

Important

All requests to the License Service must be made using HTTPS. HTTP requests are not accepted.

The service's Web Services Description Language (WSDL) file describes the actions along with the format and data types of the actions' requests and responses. For the location of the WSDL, see [WSDL Location \(p. 149\)](#). Your SOAP toolkit interprets the WSDL file to provide your product access to the actions. For most toolkits, your product calls a service action using routines and classes provided or generated by the toolkit.

The topics for each of the License Service actions describe the request parameters for each action and their values. You might find it useful to refer to the WSDL file directly to see how the parameters appear in the XML of the request generated by your toolkit, and to understand how your toolkit makes the actions available to your product code.

Structure of a SOAP Request

A SOAP request is an XML data structure that your SOAP toolkit generates and sends to the service. As described by the service WSDL, the root element of this structure is named after the License Service action. You include the parameters for the request inside the root element, according to the WSDL.

The following example shows the XML for a SOAP message that calls the [ActivateHostedProduct \(p. 177\)](#) action. Although you probably won't manually build the SOAP message for a service request, it is useful to see what your SOAP toolkit tries to produce when provided with the appropriate values. Many SOAP toolkits require you to build a request data structure similar to the XML to make a request.

```
<?xml version="1.0" encoding="UTF-8" ?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ActivateHostedProduct xmlns="http://ls.amazonaws.com/doc/2008-04-28/">
      <ActivationKey>ADMAY7DVLJTWJ76MMBQEXAMPLE</ActivationKey>
      <ProductToken>{ProductToken}AAAHVXNlclRrbgOpSykBA... [long encoded
```

```
token]...</ProductToken>
  </ActivateHostedProduct>
</soapenv:Body>
<timestamp>
  <Created>2007-06-09T10:57:35Z</Created>
  <Expires>2007-06-09T11:02:35Z</Expires>
</Timestamp>
</soapenv:Envelope>
```

Related Topics

- [Authentication of SOAP Requests \(p. 162\)](#)
- [Responses \(p. 167\)](#)

Request Authentication

Topics

- [About Authentication](#) (p. 153)
- [Authentication of REST-Query Requests](#) (p. 155)
- [Authentication of SOAP Requests](#) (p. 162)

The subsequent sections discuss authentication in general and the specific authentication methods required when making REST-Query and SOAP requests for the License Service.

About Authentication

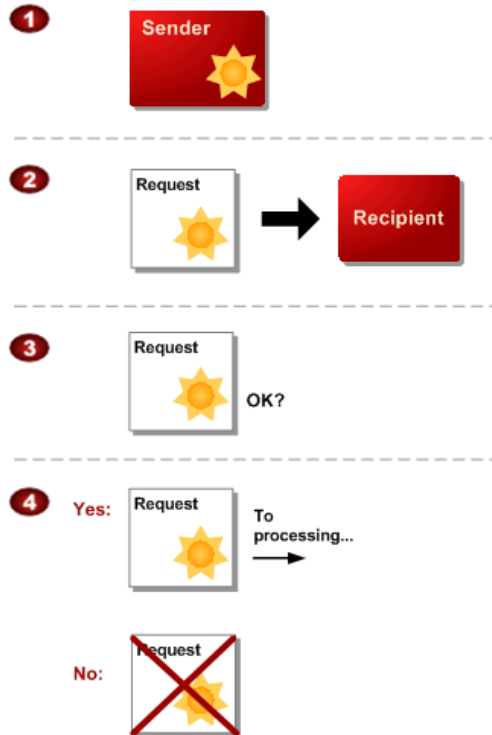
Topics

- [What Is Authentication?](#) (p. 153)
- [Your AWS Account](#) (p. 154)
- [Your AWS Access Credentials](#) (p. 154)

The following sections give basic information about authentication and the identifiers AWS uses.

What Is Authentication?

Authentication is a process for identifying and verifying who is sending a request. The following diagram shows a simplified version of an authentication process.



General Process of Authentication

1	The sender obtains the necessary credential.
2	The sender sends a request with the credential to the recipient.
3	The recipient uses the credential to verify the sender truly sent the request.
4	If yes, the recipient processes the request. If no, the recipient rejects the request and responds accordingly.

During authentication, AWS verifies both the identity of the sender and whether the sender is registered to use services offered by AWS. If either test fails, the request is not processed further.

For further discussion of authentication, go to the techencyclopedia.com entry for [authentication](#). For definitions of common industry terms related to authentication, go to the [RSA Laboratories Glossary](#).

Your AWS Account

To access any web services offered by AWS, you must first create an AWS account at <http://aws.amazon.com>. An AWS account is simply an Amazon.com account that is enabled to use AWS products; you can use an existing Amazon.com account login and password when creating the AWS account.

Alternately, you could create a new AWS-enabled Amazon.com account by using a new login and password. The e-mail address you provide as the account login must be valid. You'll be asked to provide a credit card or other payment method to cover the charges for any AWS products you use.

From your AWS account you can view your AWS account activity, view usage reports, and manage your AWS account access identifiers.

Your AWS Access Credentials

When you create an AWS account, AWS assigns you a pair of related identifiers:

- Access Key ID (a 20-character, alphanumeric sequence)
For example: AKIAIOSFODNN7EXAMPLE
- Secret Access Key (a 40-character sequence)
For example: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

These are your *AWS access key identifiers*.

Caution

Your Secret Access Key is a secret and only you and AWS should know it. It is important to keep it confidential to protect your account. Store it securely in a safe place. Never include it in your requests to AWS, and never e-mail it to anyone. Do not share it outside your organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon will ever ask you for your Secret Access Key.

The Access Key ID is associated with your AWS account. You include it in authenticated REST-Query requests to identify yourself as the sender of the request.

The Access Key ID is not a secret, and anyone could use your Access Key ID in requests to AWS. To provide proof that you truly are the sender of the request, you must also include a digital signature. For authenticated REST-Query requests, you calculate the signature using your Secret Access Key. AWS

uses the Access Key ID in the request to look up your Secret Access Key and then calculates a digital signature with the key. If the signature AWS calculates matches the signature you sent, the request is considered authentic. Otherwise, the request fails authentication and is not processed.

Viewing Your AWS Access Credentials

Your Access Key ID and Secret Access Key are displayed to you when you create your AWS account. They are not e-mailed to you. If you need to see them again, you can view them at any time from your AWS account.

To get your AWS access credentials

1. Go to the Amazon Web Services web site at <http://aws.amazon.com>.
2. Point to **Your Account** and click **Security Credentials**.
3. Log in to your AWS account.
Your Access Key ID is displayed in the **Access Credentials** section of the resulting page (following is an example).
4. To display your Secret Access Key, click **Show**.

Access Credentials

There are three types of access credentials used to authenticate your requests to AWS services: (a) access keys, (b) X.509 certificates, and (c) key pairs. Each access credential type is explained below.

[Access Keys](#) [X.509 Certificates](#) [Key Pairs](#)

Use access keys to make secure REST or Query protocol requests to any AWS service API. We create one for you when your account is created — see your access key below.

Your Access Keys

Created	Access Key ID	Secret Access Key	Status
November 13, 2009	AKIAING.....SQJ	Show	Active (Make Inactive)

[Create a new Access Key](#)

For your protection, you should never share your secret access keys with anyone. In addition, industry best practice recommends frequent key rotation.

[Learn more about Access Keys](#)

Authentication of REST-Query Requests

Topics

- [Which Requests Need to Be Authenticated?](#) (p. 156)
- [Required Information for Authentication](#) (p. 156)
- [General Authentication Process](#) (p. 156)
- [Calculating an HMAC-SHA1 Request Signature](#) (p. 158)
- [About the Time Stamp](#) (p. 160)
- [Java Sample Code for Base64 Encoding](#) (p. 160)
- [Java Sample Code for Calculating HMAC-SHA1 Signatures](#) (p. 161)

Which Requests Need to Be Authenticated?

Authentication requirements for the License Service requests vary for desktop products and web products:

- **Desktop Products**—HTTPS always required; HMAC-SHA1 signature required depending on the License Service action you're calling
- **Web Products**—HMAC-SHA1 signature and HTTPS required

Required Information for Authentication

REST-Query requests for the License Service that require authentication must include the following items:

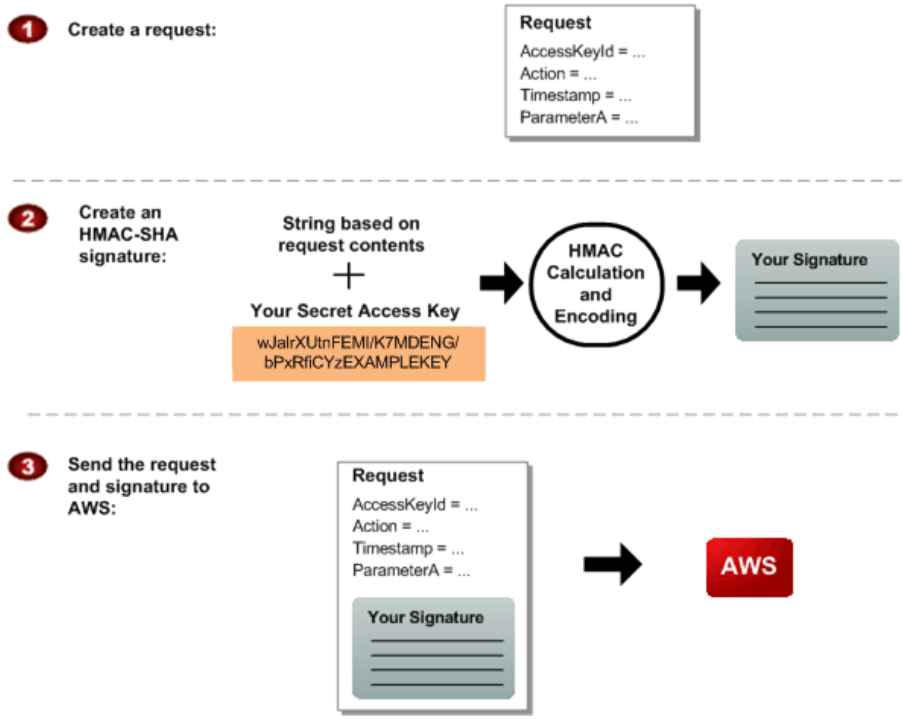
- **AWS Access Key ID**—Your Access Key ID, which AWS uses to look up your Secret Access Key. For more information, see [Your AWS Access Credentials \(p. 154\)](#).
- **Date**—A request time stamp or a request expiration time. For more information, see [About the Time Stamp \(p. 160\)](#).
- **Signature**—A valid request signature. You calculate a request signature using your Secret Access Key, which is a shared secret known only to you and AWS.
- **Signature Version**—Which signature version is being used. This is AWS-specific information that tells AWS the algorithm you used to form the string that is the basis of the signature. AWS calls the algorithm we're using *signature version 1* (for more information about the algorithm, see [Calculating an HMAC-SHA1 Request Signature \(p. 158\)](#)).

General Authentication Process

Following is the series of tasks required to authenticate requests to AWS. It is assumed you have already created an AWS account and received an Access Key ID and Secret Access Key. For more information about these items, see [Your AWS Account \(p. 154\)](#) and [Your AWS Access Credentials \(p. 154\)](#).

You perform the first three tasks.

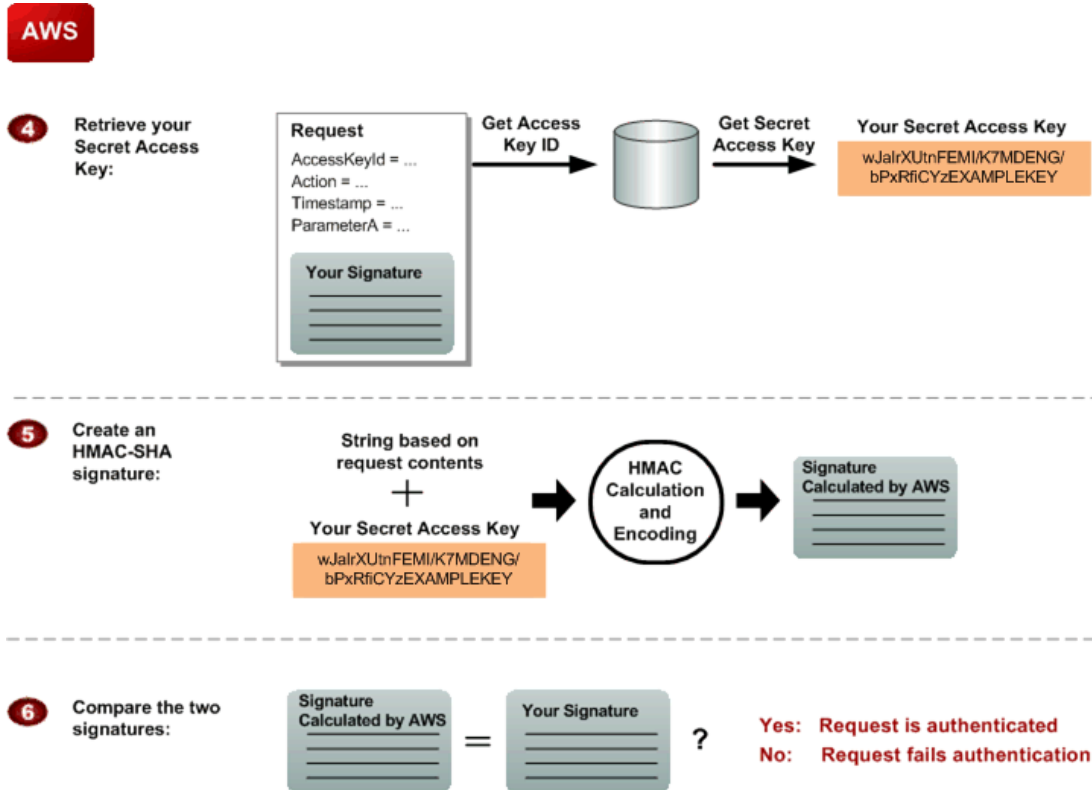
You



Process for Authentication: Tasks You Perform

1	You construct a request to AWS.
2	You calculate a keyed-hash message authentication code (HMAC-SHA1) signature using your Secret Access Key (for information about HMAC, go to http://www.faqs.org/rfcs/rfc2104.html)
3	You include the signature and your Access Key ID in the request, and then send the request to AWS.

AWS performs the next three tasks.



Process for Authentication: Tasks AWS Performs

4	AWS uses the Access Key ID to look up your Secret Access Key.
5	AWS generates a signature from the request data and the Secret Access Key using the same algorithm you used to calculate the signature you sent in the request.
6	If the signature generated by AWS matches the one you sent in the request, the request is considered authentic. If the comparison fails, the request is discarded, and AWS returns an error response.

Calculating an HMAC-SHA1 Request Signature

The request must include an HMAC-SHA1 signature. The signature is used as the value for the *Signature* parameter in the request URL being constructed. The string you use to compute the HMAC signature is constructed using the method described in the following procedure.

Important

Do not URL encode the concatenated string before computing the signature. URL encode the computed signature and other query parameters as specified in RFC1738, section 2.2. In addition, make sure to encode the + character although it is not required by RFC1738. This is required because the + character is interpreted as a blank space by Sun Java classes that perform URL decoding.

To calculate an HMAC-SHA1 signature

1. Create the string you'll use to generate the signature:

- a. Sort all request parameters alphabetically, ignoring case.

Include *SignatureVersion* in the list but not *Signature*. Do not list as "empty" any optional parameters that are not included in the request. In other words, if no value for *ParameterA* is specified in the request, do not include a *ParameterA* entry in this sorted list.

- b. Form a string by concatenating each request parameter's name with its value.

The format of the string is:

```
Param-name1Param-value1Param-name2Param-value2...Param-nameNParam-valueN
```

The parameter names are case sensitive. Do not include any separators in this string, such as question marks (?), ampersands (&), or equals signs (=). Do not URL encode the parameter values.

This is illustrated with the following request and the corresponding string to be signed (the request is borrowed from the Amazon Simple Queue Service documentation).

```
?Action=CreateQueue
&QueueName=queue2
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&SignatureVersion=1
&Expires=2007-01-12T12:00:00Z
&Version=2006-04-01
```

Following is the string to be signed for the preceding request.

```
ActionCreateQueueAWSAccessKeyIdAKIAIOSFODNN7EXAMPLEExpires2007-01-
12T12:00:00ZQueueNamequeue2SignatureVersion1Version2006-04-01
```

2. Calculate an RFC 2104-compliant HMAC-SHA1 signature, using your Secret Access Key as the key and the string you just created.

For more information, go to <http://www.faqs.org/rfcs/rfc2104.html>.

3. Convert the resulting value to base64.

For the preceding string, assuming the Secret Access Key is *fake-secret-key*, this is the base64 encoded value:

```
wlv84EOcHQk800Yq6QHgX4AdJfk=
```

4. URL encode the resulting value as specified in RFC 1738, section 2.2.

This is required because base64 encoding can result in characters that are not legal in a URL, such as plus signs (+), slashes (/), and equals signs (=).

For the preceding signature, this is the URL encoded value:

```
wlv84EOcHQk800Yq6QHgX4AdJfk%3D
```

5. Pass this final value in the *Signature* request parameter.

The following Java code snippet constructs the string.

```
/*
 * Assumes parameters are in a java.util.Map named paramMap
 * where the key is the parameter name.
 */
Set<String> sortedKeys = new TreeSet<String>(String.CASE_INSENSITIVE_ORDER);
sortedKeys.addAll(paramMap.keySet());

// Don't include Signature in the string to sign.
sortedKeys.remove("Signature");

StringBuilder stringBuilder = new StringBuilder();
for(String key : sortedKeys) {
    stringBuilder.append(key);
    stringBuilder.append(paramMap.get(key));
}

System.out.println("String to sign : " + stringBuilder.toString());
```

About the Time Stamp

The time stamp (or expiration time) you use in the request must be a `dateTime` object (for more information, go to <http://www.w3.org/TR/xmlschema-2/#dateTime>). Although it is not required, we recommend you provide the time stamp in the Coordinated Universal Time (Greenwich Mean Time) time zone. For example: 2007-01-31T23:59:59Z.

If you specify a time stamp (instead of an expiration time), the request automatically expires 15 minutes after the time stamp (in other words, AWS does not process a request if the request time stamp is more than 15 minutes earlier than the current time on AWS servers). Make sure your server's time is set correctly.

Important

If you are using .NET you must not send overly specific time stamps, due to different interpretations of how extra time precision should be dropped. To avoid overly specific time stamps, manually construct `dateTime` objects with no more than millisecond precision.

Java Sample Code for Base64 Encoding

HMAC-SHA1 request signatures must be base64 encoded. The following sample code shows how to perform base64 encoding.

```
package amazon.webservices.common;
/**
 * This class defines common routines for encoding * data in AWS requests.
 */
public class Encoding {
    /**
     * Performs base64-encoding of input bytes.
     *
     * @param rawData * Array of bytes to be encoded.
     * @return * The base64 encoded string representation of rawData.
     */
    public static String EncodeBase64(byte[] rawData) {
        return Base64.encodeBytes(rawData);
    }
}
```

```
}  
}
```

Java Sample Code for Calculating HMAC-SHA1 Signatures

The following Java code sample shows how to calculate an HMAC request signature.

```
package amazon.webservices.common;  
  
import java.security.SignatureException;  
import javax.crypto.Mac;  
import javax.crypto.spec.SecretKeySpec;  
  
/**  
 * This class defines common routines for generating  
 * authentication signatures for AWS requests.  
 */  
public class Signature {  
    private static final String HMAC_SHA1_ALGORITHM = "HmacSHA1";  
  
    /**  
     * Computes RFC 2104-compliant HMAC signature.  
     * * @param data  
     * The data to be signed.  
     * @param key  
     * The signing key.  
     * @return  
     * The Base64-encoded RFC 2104-compliant HMAC signature.  
     * @throws  
     * java.security.SignatureException when signature generation fails  
     */  
    public static String calculateRFC2104HMAC(String data, String key)  
        throws java.security.SignatureException  
    {  
        String result;  
        try {  
  
            // get an hmac_shal key from the raw key bytes  
            SecretKeySpec signingKey = new SecretKeySpec(key.getBytes(), HMAC_SHA1_ALGORITHM);  
  
            // get an hmac_shal Mac instance and initialize with the signing key  
            Mac mac = Mac.getInstance(HMAC_SHA1_ALGORITHM);  
            mac.init(signingKey);  
  
            // compute the hmac on input data bytes  
            byte[] rawHmac = mac.doFinal(data.getBytes());  
  
            // base64-encode the hmac  
            result = Encoding.EncodeBase64(rawHmac);  
  
        } catch (Exception e) {  
            throw new SignatureException("Failed to generate HMAC : " + e.getMessage());  
        }  
        return result;  
    }  
}
```

```
}  
}
```

Related Topics

- [REST-Query Requests](#) (p. 150)
- [Responses](#) (p. 167)

Authentication of SOAP Requests

Topics

- [Which Requests Need to Be Authenticated?](#) (p. 162)
- [About WS-Security](#) (p. 162)
- [What Needs to Be Signed](#) (p. 162)
- [Message Expiration](#) (p. 162)
- [X.509 Certificates](#) (p. 163)
- [Example Request to Use When Troubleshooting](#) (p. 164)

Which Requests Need to Be Authenticated?

Authentication requirements for the License Service requests vary for desktop products and web products:

- **Desktop Products**—HTTPS required
- **Web Products**—WS-Security and HTTPS required

About WS-Security

WS-Security, which is officially called Web Services Security: SOAP Message Security, is an open standard published by OASIS that defines mechanisms for signing and encrypting SOAP messages. The License Service supports version 1.0 of the WS-Security specification. For more information and a link to the WS-Security 1.0 specification, go to [the OASIS-Open web site for WS-Security](#).

Tip

The easiest way to comply with the WS-Security requirements is to use a SOAP toolkit that supports WS-Security 1.0 and X.509 certificates.

What Needs to Be Signed

You must sign the `Timestamp` element, and if you're using WS-Addressing, we recommend you also sign the `Action` header element. Alternately, you can instead sign `Timestamp`, `Body`, the `Action` header element, and the `To` header element. For information about WS-Addressing, go to <http://www.w3.org/Submission/ws-addressing/>.

Message Expiration

AWS requires request messages to expire so they can't be used in malicious replay attacks. The best practice for specifying the expiration of SOAP/WS-Security requests is to include a `Timestamp` element with an `Expires` child element. In this case, the message expires at the time established in the `Expires` element.

If no `Timestamp` element is present in the request, the request is rejected as invalid. If you include a `Timestamp` element with a `Created` child element but no `Expires` child element, the message expires 15 minutes after the value of the `Created` element.

X.509 Certificates

An X.509 certificate is a security device designed to carry a public key and bind that key to an identity. X.509 certificates are used in public key cryptography. For more information about public key cryptography and X.509 certificates, go to the techencyclopedia.com entries for "digital signature" and "public key cryptography".

To use SOAP to access the License Service, you must have a private key and a related X.509 certificate, and you must associate that X.509 certificate with your AWS developer account. You use the private key instead of your AWS Secret Access Key to sign SOAP requests (for information about your Secret Access Key, see [Your AWS Access Credentials \(p. 154\)](#)). For information about obtaining an X.509 certificate from AWS or using an X.509 certificate you obtained elsewhere, see the following sections.

Note

AWS does not implement a full public key infrastructure. The certificate information is used only to authenticate requests to AWS. AWS uses X.509 certificates only as carriers for public keys and does not trust or use in any way any identity binding that might be included in an X.509 certificate.

The WS-Security 1.0 specification requires you to sign the SOAP message with your private key and include the X.509 certificate in the SOAP message header. Specifically, you must represent the X.509 certificate as a `BinarySecurityToken` as described in the WS-Security X.509 token profile (also available if you go to [the OASIS-Open web site](#)).

Using Your Own X.509 Certificate

If you have an X.509 certificate you want to use, you can upload the certificate to AWS (without the private key value). This associates the certificate with your AWS account.

AWS accepts any syntactically and cryptographically valid X.509 certificate. Certificates can be self-signed or signed by any key. The certificate must be in Privacy Enhanced Mail (PEM) format and include a base64 encoded Distinguished Encoding Rules (DER) certificate body.

Important

When you upload the certificate, AWS checks the certificate's contents to confirm that the certificate has not expired. AWS doesn't check certificate revocation lists (CRLs) to determine if the certificate has been revoked, nor does AWS validate the certificate with a certificate authority (CA) or any trusted third parties.

To upload your own X.509 certificate

1. Go to the Amazon Web Services web site at <http://aws.amazon.com>.
2. Point to **Your Account** and click **Security Credentials**.
3. Log in to your AWS account.
The **Security Credentials** page is displayed.
4. In the **Access Credentials** section of the page, click the **X.509 Certificates** tab.
5. Click **Upload Your Own Certificate**.
6. Follow the instructions presented to upload your certificate.

Using an X.509 Certificate Generated by AWS

If you don't already have an X.509 certificate, or if you want a new certificate to use with AWS, you can have AWS generate one and automatically associate it with your AWS account. Certificates generated by AWS are signed by an AWS internal certificate authority.

To have AWS create an X.509 certificate for you

1. Go to the Amazon Web Services web site at <http://aws.amazon.com>.
2. Point to **Your Account** and click **Security Credentials**.
3. Log in to your AWS account.
The **Security Credentials** page is displayed.
4. In the **Access Credentials** section of the page, click the **X.509 Certificates** tab.
5. Click **Create a New Certificate**.
Your X.509 certificate and corresponding private key are generated.
6. From the dialog box, download your private key file and X.509 certificate file.

Example Request to Use When Troubleshooting

The following example shows the initial portion of a SOAP request that uses WS-Security with an X.509 certificate. If you're using a SOAP toolkit that supports WS-Security and X.509 certificates, the toolkit constructs the request automatically for you, so you don't have to create a request like this yourself. The example is included here as a reference to use if you're troubleshooting authentication issues with your SOAP requests. Several requirements are listed following the example; the numbers highlight where in the example the requirements are satisfied.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header>
  <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-secext-1.0.xsd">

    <wsse:BinarySecurityToken
      xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecur
ity-utility-1.0.xsd"
      1 EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary"
      2 ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
x509-token-profile-1.0#X509v3"
      wsu:Id="CertId-1064304">
      3 [Your base64 encoded X.509 certificate...]
    </wsse:BinarySecurityToken>

    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
      <ds:SignedInfo>

        <ds:CanonicalizationMethod 4 http://www.w3.org/2001/10/xml-exc-
c14n#"></ds:CanonicalizationMethod>

        <ds:SignatureMethod 5 Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></ds:SignatureMethod>
```

```
<ds:Reference URI="#id-17984263">
  <ds:Transforms>
    <ds:Transform 6 Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></ds:Transform>
  </ds:Transforms>
  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmld
sig#sha1"></ds:DigestMethod>
  <ds:DigestValue>0pjZ1+TvgPf6uG7o+Yp3l2YdGZ4=</ds:DigestValue>
</ds:Reference>

<ds:Reference URI="#id-15778003">
  <ds:Transforms>
    <ds:Transform 6 Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></ds:Transform>
  </ds:Transforms>
  <ds:DigestMethod 7 Algorithm="http://www.w3.org/2000/09/xmld
sig#sha1"></ds:DigestMethod>
  <ds:DigestValue>HhRbxBBmc200348f8nLNZyo4AOM=</ds:DigestValue>
</ds:Reference>

</ds:SignedInfo>

<ds:SignatureValue>bmVx24Qom4kd9QQtclxWIlgLk4QsQBPaKESi79x479xgb09PEStXMi
HZuBAi9luuKdNTcfQ8UE/d
jjHKZKEQRCOLLVy0Dn5ZL1RlMHsv+OzJzzvIJFTq3LQKNrzJzsNe</ds:SignatureValue>

<ds:KeyInfo Id="KeyId-17007273">
  8 <wsse:SecurityTokenReference
    xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd" wsu:Id="STRId-22438818">
    <wsse:Reference URI="#CertId-1064304"
      ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
x509-token-profile-1.0#X509v3">
    </wsse:Reference>
  </wsse:SecurityTokenReference>
</ds:KeyInfo>

</ds:Signature>

<wsu:Timestamp
  xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecur
ity-utility-1.0.xsd" 9 wsu:Id="id-17984263">
  <wsu:Created>2006-06-09T10:57:35Z</wsu:Created>
  <wsu:Expires>2006-06-09T11:02:35Z</wsu:Expires>
</wsu:Timestamp>

</wsse:Security>
</SOAP-ENV:Header>
```

Requirements for BinarySecurityToken and Signatures

- 1 The `EncodingType` attribute for the `BinarySecurityToken` element must be `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary`.
- 2 The `ValueType` attribute for the `BinarySecurityToken` element must be `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3` or `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509PKIPathv1`.
- 3 The `BinarySecurityToken` element must contain the base64 encoding of the leaf X.509 certificate if the `ValueType` is `#X509v3`, or it must contain the base64 encoding of the full X.509 certificate chain if the `ValueType` is `#X509PKIPathv1`.
- 4 The `Algorithm` attribute of the `CanonicalizationMethod` element must be `http://www.w3.org/2001/10/xml-exc-c14n#`.
- 5 The `Algorithm` attribute of the `SignatureMethod` element must be `http://www.w3.org/2000/09/xmldsig#rsa-sha1`.
- 6 The `Algorithm` attribute of the `Transform` element for each `Reference` element must be either `http://www.w3.org/2001/10/xml-exc-c14n#` or `http://www.w3.org/TR/2001/REC-xml-c14n-20010315`.
- 7 The `Algorithm` attribute of the `DigestMethod` element for each `Reference` element must be `http://www.w3.org/2000/09/xmldsig#sha1`.
- 8 The `KeyInfo` element must contain a `SecurityTokenReference` element. The `SecurityTokenReference` element must contain a `Reference` element with a `URI` attribute. The `URI` attribute must use a local particle reference to identify the `BinarySecurityToken` element that contains the X.509 certificate (for example: the `URI` attribute equals `#CertId-1064304` in the preceding example request).
- 9 You must include a `wsu:Id` attribute in any message elements that you sign. You can sign any SOAP header and the entire SOAP `Body`. Do not sign any other elements (such as children of the `Body` element). AWS ignores those elements for the purposes of signature validation, even if you include a `wsu:ID` attribute in them. If you sign elements that shouldn't be signed, the signature validation will fail.

Related Topics

- [SOAP Requests \(p. 151\)](#)
- [Responses \(p. 167\)](#)

Responses

In response to an action request, the License Service returns an XML data structure that contains the results of the request. This data conforms to the License Service WSDL. For the location of the WSDL, see [WSDL Location \(p. 149\)](#).

For SOAP requests, this data structure is the SOAP message body of the response. SOAP toolkits typically convert the response data into structures for use with your programming language or allow you to specify your own data bindings.

For REST-Query requests, this data structure is simply the body of the HTTP response. You can use a data binding method for REST-Query responses or use an XML parser directly to process the information.

Aside from using a message envelope for SOAP, the schema for the results is the same for REST-Query and SOAP responses. The WSDL defines the response messages, and REST-Query users can access the WSDL file directly.

Structure of a Successful Response

If the request succeeded, the main response element is named after the action, but with "Response" appended. For example, `ActivateHostedProductResponse` is the response element returned for a successful `ActivateHostedProduct` request. This element contains the following child elements:

- An optional element containing action-specific results; for example, the `ActivateHostedProductResponse` element includes an element called `ActivateHostedProductResult`
- `ResponseMetadata`, which contains the `RequestId` child element

The License Service WSDL includes a description of what the XML response message looks like for each License Service action.

The following is an example of a successful response.

```
<ActivateHostedProductResponse
  xmlns="http://ls.amazonaws.com/doc/2008-04-28/">
  <ActivateHostedProductResult>
    <UserToken>
      {UserToken}AAAHVXNlclRrbgOpSykBAX07g/zG...[long encoded token]...
    </UserToken>
    <PersistentIdentifier>
      PMNGLKRRYHLOXDQKEMKEXAMPLE
    </PersistentIdentifier>
  </ActivateHostedProductResult>
  <ResponseMetadata>
    <RequestId>cb919c0a-9bce-4afe-9b48-9bdf2412bb67</RequestId>
  </ResponseMetadata>
</ActivateHostedProductResponse>
```

Structure of an Error Response

If a request is unsuccessful, the main response element is `ErrorResponse`, regardless of the action's name. This element contains an `Error` element and a `RequestId` element. Each `Error` includes:

- A `Type` element that identifies whether the error was a receiver or sender error

- A `Code` element that identifies the type of error that occurred
- A `Message` element that describes the error condition in a human-readable form
- A `Detail` element that might give additional details about the error or might be empty

The following is an example of an error response.

```
<ErrorResponse xmlns="http://ls.amazonaws.com/doc/2008-04-28/">
  <Error>
    <Type>
      Sender
    </Type>
    <Code>
      ExpiredActivationKey
    </Code>
    <Message>
      The activation key has expired
    </Message>
    <Detail/>
  </Error>
  <RequestId>
    c75dbc1c-af20-409a-95de-650bba351890
  </RequestId>
</ErrorResponse>
```

Related Topics

- [REST-Query Requests \(p. 150\)](#)
- [SOAP Requests \(p. 151\)](#)

Errors

The following table lists the errors returned by all the License Service API actions. Errors specific to a particular action are listed in the topic for that action.

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
AccessDenied	Access to the resource is denied.
CannotValidateCredentials	The provided security credentials are not valid.
ConflictingQueryParameter	The query parameter <i><parameter></i> is invalid. Its structure conflicts with that of another parameter.
ElementNotSigned	The element <i><element></i> is not signed.
InternalServerError	We encountered an internal error. Please try again.
InvalidAccessKeyId	AWS was not able to validate the provided access credentials.
InvalidAction	The action <i><action></i> is not valid for this web service.
InvalidAddress	The address <i><address></i> is not valid for this web service.
InvalidBatchRequest	Invalid batch request. Reason: <i><reason></i> .
InvalidClientTokenId	The security token included in the request is invalid and the caller cannot be identified.
InvalidHttpAuthHeader	The HTTP authorization header is bad, use format: <i><format></i> .
InvalidHttpRequest	Invalid HTTP request. Reason: <i><reason></i> .
InvalidParameterCombination	The parameter <i><parameter></i> cannot be used with the parameter <i><parameter></i> .
InvalidParameterValue	Value <i><value></i> for parameter <i><parameter></i> is invalid. Reason: <i><reason></i> .
InvalidQueryParameter	The query parameter <i><parameter></i> is invalid. Please see service documentation for correct syntax.
InvalidRequest	The service cannot handle the request. Request is invalid.
InvalidResponseGroups	The following response groups are invalid: <i><groups></i> .
InvalidSecurity	The provided security credentials are not valid. Reason: <i><reason></i> .

Amazon DevPay Developer Guide
Errors

Name	Description
InvalidSecurityToken	The security token used in the request is invalid. Reason: <i><reason></i> .
InvalidService	The Web Service <i><service></i> does not exist.
InvalidURI	Could not parse the specified URI: <i><URI></i> .
InvalidWSAddressingProperty	WS-Addressing parameter <i><parameter></i> has a wrong value: <i><value></i> .
MalformedSOAPSignature	Invalid SOAP Signature. Reason: <i><reason></i> .
MalformedVersion	Version not well formed: <i><version></i> . Must be in YYYY-MM-DD format.
MissingAction	No action was supplied with this request.
MissingClientTokenId	Request must contain AWSAccessKeyId or X.509 certificate.
MissingCredentials	AWS was not able to authenticate the request: access credentials are missing.
MissingDateHeader	Authorized request must have a "date" or "x-amz-date" header.
MissingParameter	The request must contain the parameter <i><parameter></i> .
MissingSOAPRequestInfo	Unexpected: missing SOAPRequestInfo from the request.
MissingWSAddressingProperty	WS-Addressing is missing a required parameter: <i><parameter></i> .
NoAttachmentContent	Attachment content is not available.
NoMIMEBoundary	No MIME boundary found for attachment part.
NoSuchVersion	The requested version (<i><version></i>) is not valid.
RequestExpired	Request has expired.
RequestThrottled	Request is throttled.
RequiresSSL	SSL connection required for backward compatible SOAP authentication.
SOAP11IncorrectDateFormat	Timestamp must be in XSD date format.
SOAP11MissingAction	The <i><Action></i> request element is missing in SOAP 1.1 request.
SoapBodyMissing	Could not find the SOAP body in the request.
SoapEnvelopeMissing	Could not find the SOAP envelope in the request.
SoapEnvelopeParseError	Could not parse the SOAP envelope.
SoapEnvelopeTooDeep	The SOAP envelope exceeded the maximum allowed depth.

Amazon DevPay Developer Guide
Errors

Name	Description
SoapEnvelopeTooLong	The SOAP envelope exceeded the maximum allowed length.
UnknownEnvelopeNamespace	Envelope Namespace must be for either SOAP 1.1: http://schemas.xmlsoap.org/soap/envelope , or SOAP 1.2: http://www.w3.org/2003/05/soap-envelope .
UnsupportedEncodingException	Encoding (most likely US-ASCII) not supported - internal service error.
UnsupportedHttpVerb	The requested HTTP verb is not supported: <verb>.
URITooLong	The URI exceeded the maximum limit of <length>.
WSSecurityCorruptSignedInfo	Signed info is corrupt.
WSSecurityCreatedDateIncorrectFormat	Timestamp for created date must be in ISO8601 format.
WSSecurityEncodingTypeError	BinarySecurityToken must have EncodingType of <type>.
WSSecurityExpiresDateIncorrectFormat	Timestamp for expires date must be in ISO8601 format.
WSSecurityIncorrectValuetype	BinarySecurityToken has bad ValueType.
WSSecurityMissingValuetype	BinarySecurityToken must have attribute ValueType.
WSSecurityMultipleCredentialError	Request must not contain more than one BinarySecurityToken with valueType <type>.
WSSecurityMultipleUsernameError	Request cannot contain more than one UsernameToken.
WSSecuritySignatureError	Error while processing signature element.
WSSecuritySignatureMissing	SignatureValue is missing or empty.
WSSecuritySignedInfoError	Error while processing signed info.
WSSecuritySignedInfoMissing	Request has no SignedInfo.
WSSecurityTimestampExpired	Request has expired.
WSSecurityTimestampExpiresMissing	Timestamp must have Expires element.
WSSecurityTimestampMissing	Security Header Element is missing the timestamp element.
WSSecurityUsernameContainsPswd	UsernameToken must not contain Password.
WSSecurityUsernameMissing	Request must contain Username element in UsernameToken.
WSSecurityX509CertCredentialError	Request cannot contain both Credential and an X.509 certificate.

Name	Description
WSSecurityMultipleX509Error	Request must not contain more than one BinarySecurityToken with valueType <i><type></i> or <i><type></i> .
WSSecurityX509SignatureError	Failed to check signature with X.509 certificate.
X509ParseError	Could not parse X.509 certificate.

Actions

Topics

- [ActivateDesktopProduct](#) (p. 173)
- [ActivateHostedProduct](#) (p. 177)
- [GetActiveSubscriptionsByPid](#) (p. 182)
- [RefreshUserToken](#) (p. 185)
- [VerifyProductSubscriptionByPid](#) (p. 190)
- [VerifyProductSubscriptionByTokens](#) (p. 194)

This section describes the License Service actions that you can perform.

ActivateDesktopProduct

Description

The `ActivateDesktopProduct` action is meant for desktop products or products installed in any type of distributed location (e.g., desktops, handheld devices, etc.). We recommend that web products use `ActivateHostedProduct` (p. 177).

Calling `ActivateDesktopProduct` provides your desktop product with the following credentials for a specific customer:

- User token
- Secret Access Key
- Access Key ID

Note

The Secret Access Key and Access Key ID that `ActivateDesktopProduct` provides are not identical to the credentials available from the AWS web site. They work only with AWS service calls associated with Amazon DevPay. They can't be used for regular AWS service calls.

For information about how your product should store the credentials, see [Credential Storage](#) (p. 115).

The credentials must be available and ready to use each time your product makes an Amazon Simple Storage Service request on behalf of that customer. The items work together as a unit. If one part is missing, your product must get a whole new set. For more information, see [Desktop Product Exceptions](#) (p. 119).

Your product calls `ActivateDesktopProduct` to activate itself when the customer initially installs it. The product might require reactivation at other times. For more information, see [Desktop Product Exceptions](#) (p. 119).

Activation succeeds regardless of whether the credit card the customer presents at sign-up time is valid. Even though activation succeeds, your product shouldn't send requests to Amazon S3 until the customer is actually subscribed (which depends on the credit card vetting or sign-up charge succeeding). For more information about how to confirm when a customer is actually subscribed, see [Activation and Subscription Timing](#) (p. 116).

Request Authentication

Requests for `ActivateDesktopProduct` must be made over HTTPS, but they do not require any other special authentication.

Request Parameters

Name	Description	Required
<i>Action</i>	The License Service action to perform. Set this to <code>ActivateDesktopProduct</code> . Type: String Default: None Condition: Required for REST-Query requests (for more information, see REST-Query Requests (p. 150)). For SOAP requests, the action is automatically indicated in the SOAP request's root element (for more information, see SOAP Requests (p. 151)).	Conditional
<i>ActivationKey</i>	A product activation key, which the product obtains from the customer. For more information, see Desktop Product Activation (p. 114) . Type: String with a maximum 30 characters; the length of this key could increase in the future Default: None	Yes
<i>ProductToken</i>	The product token for your product, which is obtained during product registration. For more information, see Registering Your Product (p. 60) . The product token is prefixed with the literal string <code>{ProductToken}</code> . Type: String Default: None	Yes
<i>TokenExpiration</i>	The amount of elapsed time (in seconds) before the user token returned by the service should expire. The user token never expires unless you use this optional parameter. Once the user token expires, the product must reactive itself, or the customer won't be able to use it. For more information, see Reactivating the Product (p. 119) . Type: <code>xsd:duration</code> (in seconds) Default: No expiration time	No
<i>Version</i>	The version of the API to use. For example: <code>2008-04-28</code> Type: String Default: <code>2007-06-05</code> Condition: Highly recommended for REST-Query requests. Not applicable to SOAP requests (the API version is indicated in the namespace).	Conditional

Response Elements

The response to this action contains the following elements in addition to the elements returned in all successful responses (for more information, see [Structure of a Successful Response \(p. 167\)](#)).

Name	Description
<code>AWSAccessKeyId</code>	The customer's AWS Access Key ID. Type: String Ancestor: <code>ActivateDesktopProductResult</code>
<code>SecretAccessKey</code>	The customer's Secret Access Key, which you use to sign requests for Amazon S3. Type: String Ancestor: <code>ActivateDesktopProductResult</code>
<code>UserToken</code>	The user token associated with the returned Secret Access Key and Access Key ID. The user token is prefixed with the literal string <code>{UserToken}</code> . The user token length varies, and we recommend you allot 1024 characters to store the value in your database. Note that the user token length is currently much shorter than this, but it could increase or decrease without notice. Each time you request <code>ActivateDesktopProduct</code> for a particular customer, the user token you receive is different because the creation time is part of the information included in the token. Type: String Ancestor: <code>ActivateDesktopProductResult</code>

Errors

The errors listed in the following table are returned in addition to the errors returned by all actions. For more information, see [Errors \(p. 169\)](#).

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
<code>ExpiredActivationKey</code>	The activation key is expired. Activation keys expire one hour after creation. Action to take: <ol style="list-style-type: none"> 1. Prompt the customer to go to the <i>activate URL</i> and generate a new activation key (the customer is asked to log in with an Amazon.com login). 2. Prompt the customer for the new activation key. 3. Re-request the action to activate the product with the new activation key and the other required parameters.
<code>IncorrectActivationKey</code>	The activation key does not correspond to the product token. Each activation key is associated with a particular customer and a particular product. This error occurs if you provide an activation key for a DevPay product other than the one represented by the product token. Action to take: Follow the instructions listed for <code>ExpiredActivationKey</code> .

Name	Description
InvalidActivationKey	The activation key is invalid or malformed. Action to take: Same action as for ExpiredActivationKey
InvalidProductToken	The product token is invalid for the signer. Action to take: Make sure you have copied the product token correctly. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).

Examples

The following example REST-Query request activates a desktop product.

Sample Request

```
https://ls.amazonaws.com/  
?Action=ActivateDesktopProduct  
&ActivationKey=ADMAY7DVLJTWHJ76MMBQEXAMPLE  
&ProductToken={ProductToken}YLFRL3e4lOHTgGSQa8BOKA....[long encoded token]...  
&Version=2008-04-28
```

Sample Response

```
<ActivateDesktopProductResponse>  
  <ActivateDesktopProductResult>  
    <UserToken>  
      {UserToken}AAAHVXNlclRrbgfOpSykBAX07g/zG....[long encoded token]...  
    </UserToken>  
    <AWSAccessKeyId>  
      AKIAIOSFODNN7EXAMPLE  
    </AWSAccessKeyId>  
    <SecretAccessKey>  
      wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY  
    </SecretAccessKey>  
  </ActivateDesktopProductResult>  
  <ResponseMetadata>  
    <RequestId>  
      cb919c0a-9bce-4afe-9b48-9bdf2412bb67  
    </RequestId>  
  </ResponseMetadata>  
</ActivateDesktopProductResponse>
```

Related Actions

- [RefreshUserToken](#) (p. 185)
- [VerifyProductSubscriptionByTokens](#) (p. 194)

ActivateHostedProduct

Description

The `ActivateHostedProduct` action takes in an *activation key* for the customer and the *product token* for your product and responds with a *persistent identifier (PID)* and *user token* for a customer. Whether your product uses one or both of these depends on the type of product and its needs:

- Amazon Simple Storage Service web products must use the user token (for more information, see [Overall Authentication Process \(p. 123\)](#)). They can optionally also use the PID (for more information, see [Verification of the Customer's Subscription Status \(p. 135\)](#)).
- Amazon S3 desktop products can optionally use the PID (for more information, see [Verification of the Customer's Subscription Status \(p. 120\)](#)).
- Amazon EC2 AMI products can optionally use the PID (for more information, see [Verification of a Customer's Subscription Status \(p. 99\)](#)).

Note

If you want to use query string authentication with your web product (for more information, see [Query String Authentication with Web Products \(p. 130\)](#)), you must have a user token returned by a call made to `ActivateHostedProduct` after May 15, 2008. You can update any user token so that it works with query string authentication by calling [RefreshUserToken \(p. 185\)](#).

Activation succeeds regardless of whether the credit card the customer presents at sign-up time is valid. Even though activation succeeds, your product shouldn't send requests to Amazon S3 until the customer is actually subscribed (which depends on the credit card vetting or sign-up charge succeeding). For more information about how to confirm when a customer is actually subscribed, see [Activation and Subscription Timing \(p. 127\)](#).

Request Authentication

Requests for `ActivateHostedProduct` must be made over HTTPS and authenticated by AWS. For more information, see [Authentication of REST-Query Requests \(p. 155\)](#) (for REST-Query) or [Authentication of SOAP Requests \(p. 162\)](#) (for SOAP).

Request Parameters

Name	Description	Required
<i>Action</i>	The License Service action to perform. Set this to <code>ActivateHostedProduct</code> . Type: String Default: None Condition: Required for REST-Query requests (for more information, see REST-Query Requests (p. 150)). For SOAP requests, the action is automatically indicated in the SOAP request's root element (for more information, see SOAP Requests (p. 151)).	Conditional
<i>ActivationKey</i>	The activation key. For more information, see The Activation Key (p. 125) . Type: String with a maximum 30 characters; the length of this key could change in the future Default: None	Yes

Amazon DevPay Developer Guide
ActivateHostedProduct

Name	Description	Required
<i>AWSAccessKeyId</i>	Your Access Key ID. For more information, see Your AWS Access Credentials (p. 154) . Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>Expires</i>	The date and time at which the signature included in the request expires. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162) .	Conditional
<i>ProductToken</i>	The product token for your product, which is obtained during product registration. For more information about registration, see Registering Your Product (p. 60) . The product token is prefixed with the literal string {ProductToken}. Type: String Default: None	Yes
<i>Signature</i>	A request signature. For more information, see Calculating an HMAC-SHA1 Request Signature (p. 158) . Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>SignatureVersion</i>	A legacy parameter AWS requires. Set this value to 1. Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>Timestamp</i>	The date and time the request is signed. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: Authenticated REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162) .	Conditional

Name	Description	Required
<i>TokenExpiration</i>	The amount of elapsed time (in seconds) before the user token returned by the service should expire. The user token never expires unless you use this optional parameter. Once the user token expires, the product must reactive itself, or the customer won't be able to use it. For more information, see Reactivating the Product (p. 135) . Type: xsd:duration (in seconds) Default: No expiration time	No
<i>Version</i>	The version of the API to use. For example: 2008-04-28 Type: String Default: 2007-06-05 Condition: Highly recommended for REST-Query requests. Not applicable to SOAP requests (the API version is indicated in the namespace).	Conditional

Response Elements

The response to this action contains the following elements in addition to the elements returned in all successful responses. For more information, see [Structure of a Successful Response \(p. 167\)](#).

Name	Description
<i>PersistentIdentifier</i>	A static identifier of 53 to 64 characters for the customer, specifically used by developers who own either Amazon EC2 AMI products (for more information, see Verification of a Customer's Subscription Status (p. 99)) or Amazon S3 desktop products (for more information, see Verification of the Customer's Subscription Status (p. 120)). Type: String Ancestor: <code>ActivateHostedProductResult</code>
<i>UserToken</i>	The user token, which is prefixed with the literal string <code>{UserToken}</code> . This is used by Amazon S3 web products (for more information, see Setting Up Web Products (p. 123)). The user token length varies, and we recommend you allot 1024 characters to store the value in your database. Note that the user token length is currently much shorter than this, but it could increase or decrease without notice. Each time you request <code>ActivateHostedProduct</code> for a particular customer, the user token you receive is different because the creation time is part of the information included in the token. Type: String Ancestor: <code>ActivateHostedProductResult</code>

Errors

The errors listed in the following table are returned in addition to the errors returned by all actions. For more information, see [Errors \(p. 169\)](#).

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
ExpiredActivationKey	The activation key is expired. Activation keys expire one hour after creation. Action to take: <ol style="list-style-type: none">1. Prompt the customer to go to the <i>activate URL</i> and generate a new activation key (the customer is asked to log in with an Amazon.com login).2. Prompt the customer for the new activation key.3. Repeat the request to activate the product with the new activation key and the other required parameters.
IncorrectActivationKey	The activation key does not correspond to the product token. Each activation key is associated with a particular customer and a particular product. This error occurs if you provide an activation key for a DevPay product other than the one represented by the product token. Action to take: Follow the instructions listed for <code>ExpiredActivationKey</code> .
InvalidActivationKey	The activation key is invalid or malformed. Action to take: Same action as for <code>ExpiredActivationKey</code> .
InvalidProductToken	The product token is invalid for the signer. Action to take: Make sure you have copied the product token correctly. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).

Examples

The following example REST-Query request gets the user token and PID for a customer.

Sample Request

```
https://ls.amazonaws.com/  
?Action=ActivateHostedProduct  
&ActivationKey=ADMAY7DVLJTWHJ76MMBQEXAMPLE  
&ProductToken={ProductToken}YLFRL3e41OHTgGSQa8BOKA...[long encoded token]...  
&Version=2008-04-28  
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE  
&Timestamp=2008-05-19T12:00:00Z  
&Signature=Dqlp3Sd61jTUA9Uf6SGtEEexUQEEXAMPLE=  
&SignatureVersion=1
```

Sample Response

```
<ActivateHostedProductResponse>
  <ActivateHostedProductResult>
    <UserToken>
      {UserToken}AAAHVXNlclRrbgfOpSykBAXO7g/zG....[long encoded token]...
    </UserToken>
    <PersistentIdentifier>
      PMNGLKRRYHLOXDQKEMKLEEXAMPLE
    </PersistentIdentifier>
  </ActivateHostedProductResult>
  <ResponseMetadata>
    <RequestId>
      cb919c0a-9bce-4afe-9b48-9bdf2412bb67
    </RequestId>
  </ResponseMetadata>
</ActivateHostedProductResponse>
```

Related Actions

- [GetActiveSubscriptionsByPid](#) (p. 182)
- [RefreshUserToken](#) (p. 185)
- [VerifyProductSubscriptionByPid](#) (p. 190)

GetActiveSubscriptionsByPid

Description

The `GetActiveSubscriptionsByPid` action takes a customer's *persistent identifier (PID)* and returns a *product code* list for the products the customer is actively subscribed to. Only the product codes for products you own are returned. To use `GetActiveSubscriptionsByPid`, you must have previously requested `ActivateHostedProduct` (p. 177) to get a PID for the customer. For more information about how to use `GetActiveSubscriptionsByPid`, see [Verification of a Customer's Subscription Status](#) (p. 99).

Request Authentication

Requests for `GetActiveSubscriptionsByPid` must be made over HTTPS and authenticated by AWS. For more information, see [Authentication of REST-Query Requests](#) (p. 155) (for REST-Query) or [Authentication of SOAP Requests](#) (p. 162) (for SOAP).

Request Parameters

Name	Description	Required
<i>Action</i>	The License Service action to perform. Set this to <code>GetActiveSubscriptionsByPid</code> . Type: String Default: None Condition: Required for REST-Query requests (for more information, see REST-Query Requests (p. 150)). For SOAP requests, the action is automatically indicated in the SOAP request's root element (for more information, see SOAP Requests (p. 151)).	Conditional
<i>AWSAccessKeyId</i>	Your Access Key ID. For more information, see Your AWS Access Credentials (p. 154). Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>Expires</i>	The date and time at which the signature included in the request expires. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162).	Conditional
<i>PersistentIdentifier</i>	The customer's <i>PID</i> . Type: String Default: None	Yes

Name	Description	Required
<i>Signature</i>	A request signature. For more information, see Calculating an HMAC-SHA1 Request Signature (p. 158) . Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>SignatureVersion</i>	A legacy parameter AWS requires. Set this value to 1. Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>Timestamp</i>	The date and time the request is signed. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: Authenticated REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162) .	Conditional
<i>Version</i>	The version of the API to use. For example: 2008-04-28 Type: String Default: 2007-06-05 Condition: Highly recommended for REST-Query requests. Not applicable to SOAP requests (the API version is indicated in the namespace).	Conditional

Response Elements

The response to this action contains the following element in addition to the elements returned in all successful responses. For more information, see [Structure of a Successful Response \(p. 167\)](#).

Name	Description
<i>ProductCode</i>	The product code for the product the customer is subscribed to. The response can contain multiple <ProductCode> elements. Type: String Ancestor: GetActiveSubscriptionsByPidResult

Special Errors

The errors listed in the following table are returned in addition to the errors returned by all actions. For more information, see [Errors \(p. 169\)](#).

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
InvalidPersistentIdentifier	The persistent identifier (PID) is invalid or malformed.
MissingPersistentIdentifier	The request is missing the required parameter <i>PersistentIdentifier</i> .

Examples

The following example REST-Query request gets a product code list for the products the customer is subscribed to.

Sample Request

```
https://ls.amazonaws.com/  
?Action=GetActiveSubscriptionsByPid  
&PersistentIdentifier=PMNGLKRRYHLOXDQKEMKLEXAMPLE  
&Version=2008-04-28  
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE  
&Timestamp=2008-05-19T12:00:00Z  
&Signature=Dqlp3Sd6ljTUA9Uf6SGtEExwUQEEXAMPLE=  
&SignatureVersion=1
```

Sample Response

```
<GetActiveSubscriptionsByPidResponse>  
  <GetActiveSubscriptionsByPidResult>  
    <ProductCode>  
      6883959E  
    </ProductCode>  
    <ProductCode>  
      774F4FF8  
    </ProductCode>  
  </GetActiveSubscriptionsByPidResult>  
  <ResponseMetadata>  
    <RequestId>  
      cb919c0a-9bce-4afe-9b48-9bdf2412bb67  
    </RequestId>  
  </ResponseMetadata>  
</GetActiveSubscriptionsByPidResponse>
```

Related Actions

- [ActivateHostedProduct](#) (p. 177)
- [VerifyProductSubscriptionByPid](#) (p. 190)

RefreshUserToken

Description

Occasionally AWS might change the contents of the user token to support new functionality. The `RefreshUserToken` action accepts a user token and returns an up-to-date version of that token. You can then use the updated token to take advantage of the new functionality. Even if the user token you send is already the latest version, the action succeeds and returns the latest version of the user token.

For example, after May 15, 2008, the [ActivateHostedProduct \(p. 177\)](#) action began returning a new version of the user token that allows you to use query string authentication with your web product without having to pass the product token in clear text in the URL (for more information, see [Query String Authentication with Web Products \(p. 130\)](#)). You can exchange your existing web product user tokens for the new version by using `RefreshUserToken`.

Important

Desktop products must use REST-Query to access this action; they cannot use SOAP.

Request Authentication

Requests for `RefreshUserToken` must be made over HTTPS and authenticated by AWS. For more information, see [Authentication of REST-Query Requests \(p. 155\)](#) (for REST-Query) or [Authentication of SOAP Requests \(p. 162\)](#) (for SOAP, which is only allowed for web products).

Request Parameters

Name	Description	Required
<i>Action</i>	<p>The License Service action to perform. Set this to <code>RefreshUserToken</code>.</p> <p>Type: String</p> <p>Default: None</p> <p>Condition: Required for REST-Query requests (for more information, see REST-Query Requests (p. 150)). For SOAP requests, the action is automatically indicated in the SOAP request's root element (for more information, see SOAP Requests (p. 151)).</p>	Conditional
<i>AdditionalTokens</i>	<p>A list of tokens. Currently, the only token you might need to provide in this list is the product token, which you obtain during product registration. For more information about registration, see Registering Your Product (p. 60). The product token is prefixed with the literal string <code>{ProductToken}</code>.</p> <p>Type: String</p> <p>Default: None</p> <p>Condition: You must provide the product token if the product is a desktop product, or if it's a web product and the user token was originally created from a call to <code>ActivateHostedProduct</code> made on or before May 15, 2008.</p>	Conditional

Amazon DevPay Developer Guide
RefreshUserToken

Name	Description	Required
<i>AWSAccessKeyId</i>	<p>The Access Key ID associated with the Secret Access Key used to sign the request. For desktop products, this is the customer's Access Key ID. For web products, this is your Access Key ID. For more information about Access Key IDs, see Your AWS Access Credentials (p. 154).</p> <p>Type: String</p> <p>Default: None</p> <p>Condition: Required for REST-Query requests. Not applicable to SOAP requests.</p>	Conditional
<i>Expires</i>	<p>The date and time at which the signature included in the request expires.</p> <p>Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard.</p> <p>Default: None</p> <p>Condition: REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162).</p>	Conditional
<i>Signature</i>	<p>A request signature. Desktop products must sign the request with the customer's Secret Access Key. Web products must sign the request with your Secret Access Key. For more information about creating the signature, see Calculating an HMAC-SHA1 Request Signature (p. 158).</p> <p>Type: String</p> <p>Default: None</p> <p>Condition: Required for REST-Query requests. Not applicable to SOAP requests.</p>	Conditional
<i>SignatureVersion</i>	<p>A legacy parameter AWS requires. Set this value to 1.</p> <p>Type: String</p> <p>Default: None</p> <p>Condition: Required for REST-Query requests. Not applicable to SOAP requests.</p>	Conditional

Name	Description	Required
<i>Timestamp</i>	<p>The date and time the request is signed.</p> <p>Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard.</p> <p>Default: None</p> <p>Condition: Authenticated REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162).</p>	Conditional
<i>UserToken</i>	<p>The unexpired user token you want to refresh.</p> <p>Type: String</p> <p>Default: None</p>	Yes
<i>Version</i>	<p>The version of the API to use. You must set this to 2008-04-28 (or newer) to use <code>RefreshUserToken</code>.</p> <p>Type: String</p> <p>Default: 2007-06-05</p> <p>Condition: Highly recommended for REST-Query requests. Not applicable to SOAP requests (the API version is indicated in the namespace).</p>	Conditional

Response Elements

The response to this action contains the following element in addition to the elements returned in all successful responses. For more information, see [Structure of a Successful Response \(p. 167\)](#).

Name	Description
UserToken	<p>The refreshed user token, which is prefixed with the literal string <code>{UserToken}</code>.</p> <p>Type: String</p> <p>Ancestor: RefreshUserTokenResult</p>

Errors

The errors listed in the following table are returned in addition to the errors returned by all actions. For more information, see [Errors \(p. 169\)](#).

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
InvalidClientId	<p>Authentication failure. Note that an invalid user token or product token might also trigger this error.</p> <p>Action to take: Make sure you have copied the product token and user token correctly and that you use the correct credentials when you sign the request. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).</p>
InvalidProductToken	<p>The product token is invalid for the signer.</p> <p>Action to take: Make sure you have copied the product token correctly and that you use the correct credentials when you sign the request. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).</p>
InvalidParameterValue	<p>One or more of the parameters required in the request is missing or invalid.</p>

Examples

The following example REST-Query request refreshes a user token.

Sample Request

```
https://ls.amazonaws.com/
?Action=RefreshUserToken
&AdditionalTokens={ProductToken}YLFRL3e4lOHTgGSQa8BOKA...[long encoded token]...
&UserToken={UserToken}AAAHVXN1clRrbgfOpSykBAX07g/zG...[long encoded token]...
&Version=2008-04-28
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Timestamp=2008-05-19T12:00:00Z
&Signature=Dqlp3Sd6lJTUA9Uf6SGtEEExwUQEEXAMPLE=
&SignatureVersion=1
```

Sample Response

```
<RefreshUserTokenResponse>
  <RefreshUserTokenResult>
    <UserToken>
      {UserToken}DRTNXLBkql7XfpfOwrEPuD4Rl/Qm...[long encoded token]...
    </UserToken>
  </RefreshUserTokenResult>
  <ResponseMetadata>
    <RequestId>
      cb919c0a-9bce-4afe-9b48-9bdf2412bb67
    </RequestId>
  </ResponseMetadata>
</RefreshUserTokenResponse>
```

Related Actions

- [ActivateDesktopProduct](#) (p. 173)
- [ActivateHostedProduct](#) (p. 177)

VerifyProductSubscriptionByPid

Description

The `VerifyProductSubscriptionByPid` action lets you verify whether a customer who purchased your product is actively subscribed to the product (e.g., and hasn't canceled). To use this action, you must have previously requested `ActivateHostedProduct` (p. 177) to get a *persistent identifier (PID)* for the customer.

Request Authentication

Requests for `VerifyProductSubscriptionByPid` must be made over HTTPS and authenticated by AWS. For more information, see [Authentication of REST-Query Requests](#) (p. 155) (for REST-Query) or [Authentication of SOAP Requests](#) (p. 162) (for SOAP).

Request Parameters

Name	Description	Required
<i>Action</i>	The License Service action to perform. Set this to <code>VerifyProductSubscriptionByPid</code> . Type: String Default: None Condition: Required for REST-Query requests (for more information, see REST-Query Requests (p. 150)). For SOAP requests, the action is automatically indicated in the SOAP request's root element (for more information, see SOAP Requests (p. 151)).	Conditional
<i>AWSAccessKeyId</i>	Your Access Key ID. For more information, see Your AWS Access Credentials (p. 154). Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>Expires</i>	The date and time at which the signature included in the request expires. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162).	Conditional
<i>PersistentIdentifier</i>	The customer's <i>PID</i> . Type: String Default: None	Yes

Name	Description	Required
<i>ProductCode</i>	The <i>product code</i> for your product, which you obtained during product registration. For more information, see Registering Your Product (p. 60) . Type: String Default: None	Yes
<i>Signature</i>	A request signature. For more information, see Calculating an HMAC-SHA1 Request Signature (p. 158) . Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>SignatureVersion</i>	A legacy parameter AWS requires. Set this value to 1. Type: String Default: None Condition: Required for REST-Query requests. Not applicable to SOAP requests.	Conditional
<i>Timestamp</i>	The date and time the request is signed. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: Authenticated REST-Query requests must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both. For information about handling message expiration for SOAP requests, see Message Expiration (p. 162) .	Conditional
<i>Version</i>	The version of the API to use. For example: 2008-04-28 Type: String Default: 2007-06-05 Condition: Highly recommended for REST-Query requests. Not applicable to SOAP requests (the API version is indicated in the namespace).	Conditional

Response Elements

The response to this action contains the following elements in addition to the elements returned in all successful responses. For more information, see [Structure of a Successful Response \(p. 167\)](#).

Name	Description
Subscribed	Whether the customer is currently subscribed to your product. Type: Boolean Valid Values: true false Ancestor: VerifyProductSubscriptionByPidResult

Errors

The errors listed in the following table are returned in addition to the errors returned by all actions. For more information, see [Errors \(p. 169\)](#).

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
InvalidPersistentIdentifier	The persistent identifier (PID) is invalid or malformed.
InvalidProductToken	The product token is invalid for the signer. Action to take: Make sure you have copied the product token correctly. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).
MissingPersistentIdentifier	The request is missing the required parameter <i>PersistentIdentifier</i> .
MissingProductToken	The request is missing the required parameter <i>ProductToken</i> .

Examples

The following example REST-Query request verifies if a customer is still subscribed to the product.

Sample Request

```
https://ls.amazonaws.com/  
?Action=VerifyProductSubscriptionByPid  
&PersistentIdentifier=PMNGLKRRYHLOXDQKEMKLEXAMPLE  
&ProductCode=774F4FF8  
&Version=2008-04-28  
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE  
&Timestamp=2008-05-19T12:00:00Z  
&Signature=Dqlp3Sd61jTUA9Uf6SGtEEExwUQEEXAMPLE=  
&SignatureVersion=1
```

Sample Response

```
<VerifyProductSubscriptionByPidResponse>  
  <VerifyProductSubscriptionByPidResult>  
    <Subscribed>  
      true  
    </Subscribed>  
  </VerifyProductSubscriptionByPidResult>  
  <ResponseMetadata>  
    <RequestId>  
      cb919c0a-9bce-4afe-9b48-9bdf2412bb67  
    </RequestId>
```

```
</ResponseMetadata>  
</VerifyProductSubscriptionByPidResponse>
```

Related Actions

- [ActivateHostedProduct](#) (p. 177)
- [GetActiveSubscriptionsByPid](#) (p. 182)

VerifyProductSubscriptionByTokens

Description

The `VerifyProductSubscriptionByTokens` action lets desktop products verify whether the customer who purchased and installed the product is actively subscribed to it (that is, the customer hasn't canceled the subscription). For example, a desktop product might call `VerifyProductSubscriptionByTokens` each time the customer opens the product. If the subscription is no longer active, the desktop product can disable the customer's access.

Important

You must use REST-Query to access this action; using SOAP is not allowed for this action.

Request Authentication

Requests for `VerifyProductSubscriptionByTokens` must be made over HTTPS and authenticated by AWS. For more information, see [Authentication of REST-Query Requests \(p. 155\)](#).

Important

The desktop product must sign the request using the *customer's* Secret Access Key and Access Key ID (which were obtained from a previous request the desktop product made to [ActivateDesktopProduct \(p. 173\)](#)).

Request Parameters

Name	Description	Required
<i>Action</i>	The License Service action to perform. Set this to <code>VerifyProductSubscriptionByTokens</code> . Type: String Default: None	Yes
<i>AWSAccessKeyId</i>	The Access Key ID of the customer, which the desktop product obtained from a request to <code>ActivateDesktopProduct</code> . Type: String Default: None	Yes
<i>Expires</i>	The date and time at which the signature included in the request expires. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: The request must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both.	Conditional

Name	Description	Required
<i>ProductToken</i>	The product token for your product, which you obtained during product registration. For more information, see Registering Your Product (p. 60) . The product token is prefixed with the literal string {ProductToken}. Type: String Default: None	Yes
<i>Signature</i>	A request signature. For more information, see Calculating an HMAC-SHA1 Request Signature (p. 158) . You must use the customer's Secret Access Key and Access Key ID instead of your own when creating the signature. Type: String Default: None	Yes
<i>SignatureVersion</i>	A legacy parameter AWS requires. Set this value to 1. Type: String Default: None	Yes
<i>Timestamp</i>	The date and time the request is signed. Type: String in the format YYYY-MM-DDThh:mm:ssZ, as specified in the ISO 8601 standard. Default: None Condition: The request must include either the <i>Timestamp</i> parameter or the <i>Expires</i> parameter, but not both.	Conditional
<i>UserToken</i>	The user token for the customer, which the desktop product received from a request to <code>ActivateDesktopProduct</code> . The user token is prefixed with the literal string {UserToken}. Type: String Default: None	Yes
<i>Version</i>	The version of the API to use. For example: 2008-04-28 Type: String Default: 2007-06-05 Condition: Highly recommended for REST-Query requests.	Conditional

Response Elements

The response to this action contains the following elements in addition to the elements returned in all successful responses. For more information, see [Structure of a Successful Response \(p. 167\)](#).

Name	Description
Subscribed	Whether the customer is currently subscribed to your product. Type: Boolean Valid Values: true false Ancestor: VerifyProductSubscriptionByTokensResult

Errors

The errors listed in the following table are returned in addition to the errors returned by all actions. For more information, see [Errors \(p. 169\)](#).

Important

We might throttle requests to the License Service as necessary. When we throttle, we return a 503 (service unavailable) HTTP status code. Your system should be prepared to retry any request that receives a 503 code.

Name	Description
InvalidClientTokenId	Authentication failure. Note that an invalid user token or product token might also trigger this error. Action to take: Make sure you have copied the product token and user token correctly and that you use the correct credentials when you sign the request. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).
InvalidProductToken	The product token is invalid for the signer. Action to take: Make sure you have copied the product token correctly and that you use the correct credentials when you sign the request. You can view the product token for your product at any time by clicking View Product Details on your DevPay Activity page (at http://aws.amazon.com/devpayactivity).
InvalidParameterValue	One or more of the parameters required in the request is missing or invalid.

Examples

The following example REST-Query request verifies if a customer is still subscribed to the product.

Sample Request

```
https://ls.amazonaws.com/  
?Action=VerifyProductSubscriptionByTokens  
&ProductToken={ProductToken}AAAHVXNlclRrbgfOpSykBA...[long encoded token]...  
&UserToken={UserToken}AAAHVXNlclRrbgfOpSykBAXO7g/zG...[long encoded token]...  
&Version=2008-04-28  
&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE  
&Timestamp=2008-05-19T12:00:00Z
```

```
&Signature=Dqlp3Sd61jTUA9Uf6SGtEEExwUQEEXAMPLE=  
&SignatureVersion=1
```

Sample Response

```
<VerifyProductSubscriptionByTokensResponse>  
  <VerifyProductSubscriptionByTokensResult>  
    <Subscribed>  
      true  
    </Subscribed>  
  </VerifyProductSubscriptionByTokensResult>  
  <ResponseMetadata>  
    <RequestId>  
      cb919c0a-9bce-4afe-9b48-9bdf2412bb67  
    </RequestId>  
  </ResponseMetadata>  
</VerifyProductSubscriptionByTokensResponse>
```

Related Actions

- [ActivateDesktopProduct](#) (p. 173)

Appendices

Topics

- [Appendix: The Customer Purchase Experience \(p. 199\)](#)
- [Appendix: E-mails Sent to Your Customers \(p. 206\)](#)
- [Appendix: Product Registration Flow \(p. 217\)](#)
- [Appendix: Example DevPay Activity Pages \(p. 239\)](#)
- [Appendix: Risks to DevPay Products \(p. 261\)](#)

Appendix: The Customer Purchase Experience

Topics

- [The Product Purchase Flow \(p. 200\)](#)
- [The Application Activation Page \(p. 203\)](#)
- [The Application Billing Page \(p. 204\)](#)

This appendix describes what your customers see when purchasing DevPay products.

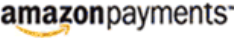
Important

When customers buy your product and are later billed for using it, they deal entirely with *Amazon Payments*, and not *Amazon DevPay*. Your customers receive no direct exposure to Amazon DevPay. Therefore, you don't need to point your customers to the Amazon DevPay documentation or information for any reason.

The Product Purchase Flow

To purchase your product, customers first click the *purchase URL* that you present to them (you receive the URL when you register your product; for more information, see [Registering Your Product \(p. 60\)](#)). They then go through a series of web pages that are branded with Amazon Payments. The following example pages mention several fictional companies and products. For example, a company called Skysonsa, Inc. is selling a product called Sky Storage.

Your customers are first prompted to log in with an Amazon.com login. The page indicates that you've teamed up with Amazon Payments to bill for the product.



Skysonsa, Inc. has teamed with Amazon Payments to make billing quick, easy, and secure.

Please sign in to buy Sky Storage.

Amazon.com Sign In

You may sign in using your existing Amazon.com account or you can create a new account by selecting "I am a new customer."

Enter your e-mail address:

I am a new customer.
(You'll create a password later)

**I am a returning customer,
and my password is:**

[Forgot your password? Click here](#)

[Has your e-mail address changed since your last order?](#)


About Amazon.com Sign In

Amazon.com Sign In allows you to log in to applications that use Amazon technology using your Amazon.com account. To protect your information, you should only enter your Amazon.com e-mail address and password into a web site if the address of the site starts with <https://www.amazon.com> (see your browser address bar to check this). You can find out about how your personal information is protected in our [Privacy Notice](#).

Amazon Payments, Inc. is a wholly owned subsidiary of Amazon.com that provides a safe means to process transactions online. Amazon Payments does not share your financial information (such as credit card information) with any seller using Amazon Payments without your consent. All information associated with your use of Amazon Payments will be handled in accordance with the terms of the [Amazon Payments Privacy Notice](#).

[Conditions of Use](#) | [Privacy Notice](#)
© 2007, Amazon.com, Inc. or its Affiliates.

Your customers are next presented with the product details you provided when you registered the product with DevPay. If you have a monthly charge, a one-time charge, or both, the initial fee the customers are charged at sign-up is displayed in the top right corner of this page. The page also displays information about the credit card on file that will be used if the customer decides to purchase the product. At the bottom, the page includes links to the [Billing Services Agreement](#) and the terms and conditions you specified for your product. If customers want to purchase your product, they click **Place your order**.



Review the information below, then click "Place your order."
▶ Place your order

Total due today

One-time charge:	\$1.00
Prorated recurring monthly charge:	\$1.25
Total:	\$2.25

Payment Method: Change

Visa: ***-11111
Exp: 10/2010

Billing Address: Change

Em Vingt
123 Any Street
Any City, State 12345
Phone:

Application Information

Sky Storage

Sky Storage is an application that lets you store files and backup data securely. Sky Storage makes it easy to store any data securely online, and access it from any computer just like a local hard drive. You can back up your important files like photos, home movies, music, business records, and much more. Sharing files between computers is also easy.

Sky Storage Pricing

One-Time and Monthly Charges

Pricing Dimension	Your Price (\$)	Description
One-Time Charge	1.00	This charge only occurs at sign-up.
Recurring Monthly Charge	5.00	

Usage Charges - US Standard Region

Usage Type	Your Price (\$)	Details	Description
Data Transfer	GB of data transfer in	0.100 per GB	This is approximately 200 MP3's uploaded.
	GB of data transfer out	0.180 per GB	This is approximately 200 MP3's downloaded.
Storage	GB-Month of storage used	0.150 per GB-Mo	This is approximately 200 MP3's stored a month.

Note: Sky Storage is sold by Skysonsa, Inc. Amazon Payments will charge your credit card for any one time and recurring charges outlined in the pricing above. Until you cancel your subscription to Sky Storage, any recurring charges will be charged on the 1st of every month and will include any applicable usage charges.

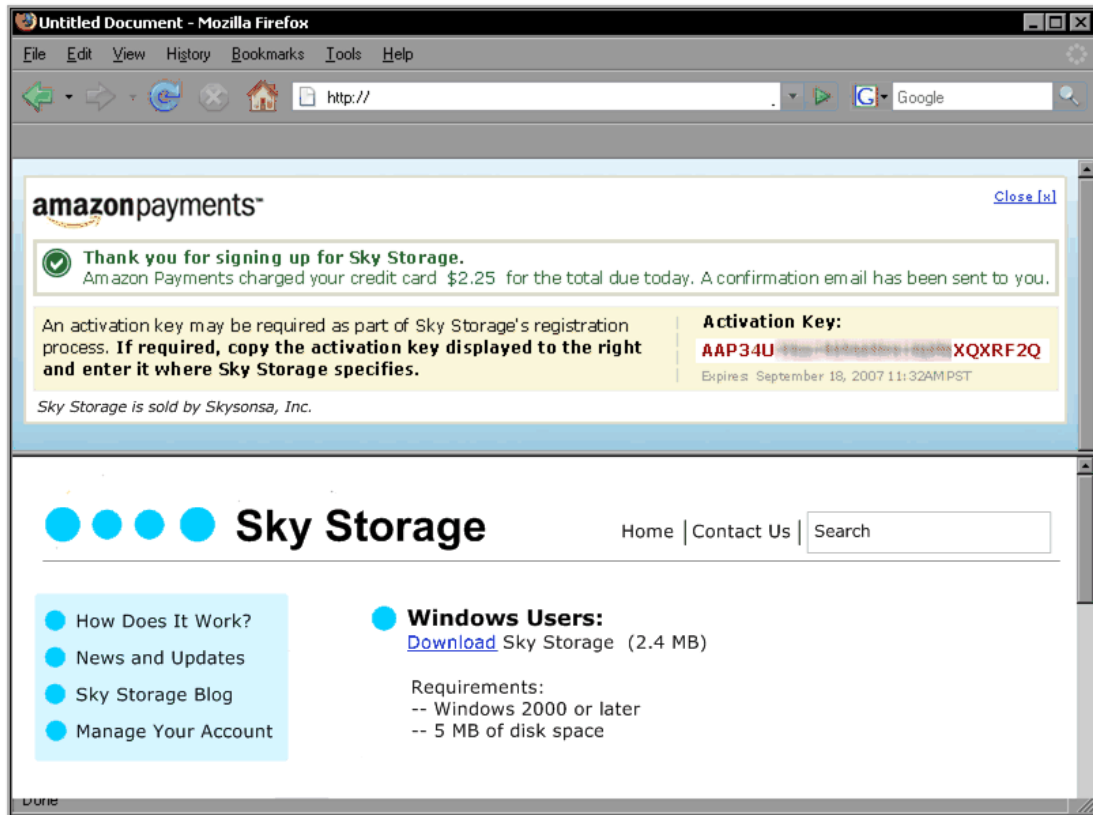
Terms and Conditions

By clicking the 'Place your Order' button, you indicate that you have read and agree to the [Billing Services Agreement](#).

Review the information above, then click "Place your order."
▶ Place your order

Amazon DevPay Developer Guide The Product Purchase Flow

After they click **Place your order**, the purchase is processed and the browser window splits into two frames. The top frame contains the *activation key*. The bottom frame contains the *redirect URL* you provided when you registered the product with DevPay.



At this point, the customer has purchased the product and can begin using it.

The Application Activation Page

Depending on the type of product you have and how you design your system, you might need to prompt your customers to provide you the *activation key*. Customers might not have the activation key, so you should point them to the *activate URL* (<http://www.amazon.com/dp-activate>) to get one. The activate URL goes to the **Application Activation** page. Following is an example.

amazonpayments™ Application Billing Payment Method Application Activation

Hello, Em Vingt. (If you're not Em Vingt, [click here.](#))

Application Activation

Generate an Application Activation Key

An activation key may be required as part of an application's registration process. If your application requires it, follow the steps below.

- Step 1:** Click on the **Generate Key** link to generate an activation key for an application.
- Step 2:** Copy the activation key.
- Step 3:** Click on **Go to Application** to access the application.
- Step 4:** Paste the activation key if and where the application specifies.

Application Name	Seller Name	Activation Key (What's this?)	Application Link
Sky Storage View/Cancel Application	Sky Storage Company	AAP34UY QXRF2Q Expires: September 18, 2007 11:32AM PST Generate a new key	Go to Application
SwiftSource View/Cancel Application	Wildfire Computing Systems	▼ Generate Key	Go to Application
Titanic Drive View/Cancel Application	Varuna Systems, Inc.	▼ Generate Key	Go to Application

The top part of the page has instructions for generating an activation key. The bottom part shows a list of the DevPay products the customer has purchased. The key is associated with a specific DevPay product, so the customer must generate a key for the appropriate product. If you try using the License Service with an activation key for another product, you get an error.

What happens to the key after the customer generates it depends on your system. This page instructs the customer to click the **Go to Application** link on the right side of the page. That link goes to the redirect URL you provided when you registered the product with DevPay (for more information, see [Registering Your Product \(p. 60\)](#)). We automatically append the activation key and product code to the redirect URL as query parameters, so if your system is set up to retrieve those values programmatically, then the customer just needs to click that link. Alternately, your system might need the customer to paste the key somewhere. If the location is not the destination of the redirect URL, then you need to make it clear to the customer where to paste the activation key.

The Application Billing Page

After customers have used the product, they can view their usage and corresponding costs on the **Application Billing** page (at <http://www.amazon.com/dp-applications>). This page is available at any time and shows usage and billing information for all the DevPay products they use. You should provide a link to this page on your web site.

The following image shows an example **Application Billing** page for a customer who has purchased paid AMIs.

amazonpayments Application Billing Payment Method Application Activation

Hello, Joe Johnson. (If you're not Joe Johnson, [click here.](#))

Application Billing

[View Previous Statement](#)

Billing Statement as of March 20, 2009†

Billing Cycle for this Report: March 1 - March 31, 2009
This page shows application activity and usage charges through approximately 03/20/2009 17:59 GMT.


Application Name	Rate	Usage	Total(\$)
ABC AMI Sold by: AMIMakers, Inc. View/Cancel Application			
	Recurring monthly charge for April	Recurring monthly charge	19.99
Amazon EC2 running Linux/UNIX			
	\$0.140 per Small instance-hour (or partial hour) consumed	55 Hrs	7.70
	\$0.840 per Extra Large instance-hour (or partial hour) consumed	19 Hrs	15.96
Bandwidth			
	\$0.140 per GB of data transfer in	390.443 GB	54.66
	\$0.200 per GB of data transfer out	584.251 GB	116.85
	\$0.020 per GB of regional data transfer in/out	2.892 GB	0.06
			215.22
Ham Overtone Server Sold by: Hambotext View/Cancel Application			
	Recurring monthly charge for April	Recurring monthly charge	7.00
Amazon EC2 running Linux/UNIX			
	\$0.130 per Small instance-hour (or partial hour) consumed	9 Hrs	1.17
Bandwidth			
	\$0.110 per GB of data transfer in	31.997 GB	3.52
	\$0.190 per GB of data transfer out (First 10240 GB)	51.003 GB	9.69
	\$0.020 per GB of regional data transfer in/out	5.513 GB	0.11
			21.49
Charges Due on April 1, 2009†			236.71

† All charges for this billing cycle will be charged to your credit card on your next billing date, April 1, 2009. The current billing cycle starts March 1, 2009 and ends March 31, 2009 00:00 GMT. Not included in the charges displayed here are any additional usage charges you will accrue in this billing cycle.

View a Previous Statement: February 2009

[Printer Friendly Version](#)

The next image shows an example **Application Billing** page for a customer who has purchased products based on Amazon S3.



Application Billing
Payment Method
Application Activation

Hello, Em Vingt. (If you're not Em Vingt, [click here.](#))

Application Billing

[View Previous Statement](#)

Billing Statement as of March 20, 2009†

Billing Cycle for this Report: March 1 - March 31, 2009
This page shows application activity and usage charges through approximately 03/20/2009 17:59 GMT.

Application Name	Rate	Usage	Total(\$)
Sky Storage Sold by: Skysonsa, Inc. View/Cancel Application			
	Recurring monthly charge for April	Recurring monthly charge	5.00
	\$0.150 per GB-Month of storage used	18.343 GB-Month	2.75
	\$0.100 per GB of data transfer in	0.146 GB	0.01
	\$0.180 per GB of data transfer out	0.242 GB	0.04
			7.80
Cactuss Sold by: Hambotext View/Cancel Application			
	Recurring monthly charge for April	Recurring monthly charge	1.50
	\$0.20 per GB-Month of storage used (First 20 GB-Month)	20.000 GB-Month	4.00
	\$0.15 per GB of data transfer out (Greater than 20 GB-Month)	39.440 GB-Month	5.92
	\$0.120 per GB of data transfer in	11.780 GB	1.41
	\$0.190 per GB of data transfer out	0.385 GB	0.07
	\$0.020 per 1,000 PUT or LIST requests	493592 requests	9.87
	\$0.020 per 10,000 GET and all other requests	487746 requests	0.98
			23.75
Charges Due on April 1, 2009†			31.55

† All charges for this billing cycle will be charged to your credit card on your next billing date, April 1, 2009. The current billing cycle starts March 1, 2009 and ends March 31, 2009 00:00 GMT. Not included in the charges displayed here are any additional usage charges you will accrue in this billing cycle.

View a Previous Statement:

[Printer Friendly Version](#)

Appendix: E-mails Sent to Your Customers

Topics

- [E-mails Related to Sign-Up or Cancellation \(p. 207\)](#)
- [E-mails Related to Billing \(p. 209\)](#)
- [E-mails Related to Price Changes \(p. 213\)](#)

This appendix shows the templates we use when sending e-mails to your customers.

E-mails Related to Sign-Up or Cancellation

This section includes the following e-mails:

- [When the Customer Signs Up to Use Your Product](#) (p. 207)
- [If the Validation of the Customer's Credit Card Fails](#) (p. 207)
- [If the Customer Cancels Use of Your Product](#) (p. 208)

When the Customer Signs Up to Use Your Product

Subject: Your *[product name]* Purchase

Greetings from Amazon Payments,

Thank you for purchasing *[product name]*, which is sold and supported by *[company name]*. Amazon Payments is processing payments for your subscription to *[product name]*. If you have any questions about this application, including technical support, please contact the provider at *[contact e-mail or URL]*.

[company name] may ask you for an Activation Key. If you do not have an Activation Key, you can generate a new key at any time by visiting the following web page:

<http://www.amazon.com/dp-activate>

[If there is a sign-up charge, a monthly charge, or both, the following paragraph is included:]

On *[date of sign up]*, Amazon Payments has charged your credit card on file with us in the amount of $\$[total amount]$ for your purchase of *[product name]*. *Click here* to view this transaction.

You have authorized Amazon Payments to charge your credit card for any recurring fees associated with your use of *[product name]* until you cancel your subscription. You will receive an e-mail notification when we charge your credit card on the first day of each month.

To view your transaction, review account history, change your credit card, or cancel your subscription, please visit the following web page:

<http://www.amazon.com/dp-applications>

If you have any questions regarding this payment, please contact us at application-payments@amazon.com.

Sincerely,
Amazon Payments

If the Validation of the Customer's Credit Card Fails

Subject: Billing Alert From Amazon Payments

Greetings from Amazon Payments,

Amazon Payments is performing the billing for your purchased application(s). We were unable to verify the payment method used. You must provide a valid payment method to continue using your application(s).

Please update your payment method by visiting the following web page:
<http://www.amazon.com/dp-applications>

If you have any questions regarding your payment method, please contact us at application-payments@amazon.com.

Sincerely,
Amazon Payments

If the Customer Cancels Use of Your Product

Subject: Cancellation of *[product name]*

Greetings from Amazon Payments,

This e-mail confirms that you have canceled your use of *[product name]* sold by *[company name]*. You will no longer be able to use this application. You will be billed at the end of the month for any use of this application prior to cancellation; after that you will receive no more bills for *[product name]*.

[If there is a refund related to the cancellation, the following paragraph is included:]

You will be issued a refund in the amount of *[refund amount]* for this application. This refund amount is the prorated portion of the monthly fee based on the number of days remaining in the month.

Additional information regarding your payments for this application is available at the following web page:

<http://www.amazon.com/dp-applications>

If you have any questions regarding your cancellation of this application, please contact application-payments@amazon.com. For all other questions regarding an application, including technical support, please contact *[company name]* at *[contact e-mail or URL]*.

Sincerely,
Amazon Payments

E-mails Related to Billing

This section includes the following e-mails:

- [When the Customer Is Billed Each Month](#) (p. 209)
- [If the Customer's Credit Card Charge Fails During Billing](#) (p. 209)
- [When the Customer Changes the Payment Method](#) (p. 210)
- [When a Retry Attempt Succeeds](#) (p. 210)
- [When the First or Second Retry Attempt Fails](#) (p. 211)
- [When the Final Retry Attempt Fails](#) (p. 211)

When the Customer Is Billed Each Month

Subject: Your Bill from Amazon Payments for *[billing period]*

Greetings from Amazon Payments,

Amazon Payments is processing payments for your subscription to the application(s) listed below. On *[date of charge]* we charged the credit card you currently have on file with us in the amount of \$*[total amount]* for your use of application(s) during *[billing period]*. [Click here](#) to view information about this transaction.

Application Details:

Application: *Product name*

Sold by: *Company name*

Contact information: *Contact e-mail or URL*

Additional information regarding your bill, individual application charges, and your account history are available at the following web page:
<http://www.amazon.com/dp-applications>

You can also change your credit card or cancel your subscriptions to any of the applications at this web site.

If you have any questions regarding payment, please contact us at application-payments@amazon.com. For all other questions, including technical support, please contact the provider using the information provided above.

Sincerely,
Amazon Payments

If the Customer's Credit Card Charge Fails During Billing

Subject: Billing Alert from Amazon Payments

Greetings from Amazon Payments,

Amazon Payments is performing the billing for your purchased application(s).

We have been unable to charge your credit card for *[total amount]* for your use of *[product name(s)]* during *[billing period]*. Please update your payment method by visiting the following web page:
<http://www.amazon.com/dp-applications>

Application details:

Application: *[product name]*
Sold by: *[company name]*
Contact information: *[contact e-mail or URL]*

If you have any questions regarding this payment, please contact us at application-payments@amazon.com. For all other questions regarding an application, including technical support, please contact the provider using the information listed above.

Sincerely,
Amazon Payments

When the Customer Changes the Payment Method

Subject: Account Alert from Amazon Payments

Greetings from Amazon Payments,

This e-mail is to confirm that you recently updated the payment method on file with us. This payment method will be used for all of your application charges going forward. You can review your changes by visiting the following web page:
<http://www.amazon.com/dp-applications>

If you have any questions regarding your payment method, please contact us at application-payments@amazon.com.

Sincerely,
Amazon Payments

When a Retry Attempt Succeeds

If the customer's payment fails the first time, we retry three more times as necessary (7 days later, 14 days later, and 21 days later). If one of those retry attempts succeeds, we send the following e-mail to the customer.

Subject: Billing Alert from Amazon Payments

Greetings from Amazon Payments,

We have successfully retried and charged your credit card the amount of *[amount]* for *[billing period]*. Your *[billing period]* bill is now paid in full. To view information about the applications and details about this charge, go to the following web page:
<http://www.amazon.com/dp-applications>

If you have any questions regarding payment, please contact us at application-payments@amazon.com.

For all other questions, including technical support, please contact the provider. Information for the provider can be found at <http://www.amazon.com/dp-applications>.

Sincerely,
Amazon Payments

When the First or Second Retry Attempt Fails

If the customer's payment fails the first time, we retry 7 days later, and then 14 days later if necessary. When either of these retry attempts fails, we send the following e-mail to the customer.

Subject: Billing Alert from Amazon Payments

Greetings from Amazon Payments,

Amazon Payments is performing the billing for your purchased application(s). This is a reminder that we have been unable to charge your credit card for *[amount]* for *[billing period]*.

Please update your payment method by visiting the following web page:
<http://www.amazon.com/dp-applications>

Please note that access to your application(s) will be cancelled if you fail to pay this bill.

To view information about the applications and details about this charge, go to the following web page:
<http://www.amazon.com/dp-applications>

If you have any questions regarding this payment, please contact us at application-payments@amazon.com.

For all other questions regarding an application, including technical support, please contact the provider. The information about the provider can be found at <http://www.amazon.com/dp-applications>.

Sincerely,
Amazon Payments

When the Final Retry Attempt Fails

If the third (and final) retry attempt fails, we cancel the customer's access to all DevPay applications and send the following e-mail to the customer.

Subject: Billing Alert from Amazon Payments

Greetings from Amazon Payments,

Amazon Payments is performing the billing for your purchased application(s).

Amazon DevPay Developer Guide
E-mails Related to Billing

After several attempts, we have been unable to charge your credit card for *[amount]* for *[billing period]*. As a result, your access to the following application(s) below has been cancelled.

Application Details:

Application: *Application name*

Sold by: *Company name*

Contact information: *contact e-mail or URL*

To view information about the applications or details of this charge, go to the following web page:

<http://www.amazon.com/dp-applications>

If you have any questions regarding this payment, please contact us at application-payments@amazon.com.

Sincerely,
Amazon Payments

E-mails Related to Price Changes

This section includes the following e-mails:

- [When You Schedule a Price Change with Passive Authorization \(p. 213\)](#) (for information about passive price changes, see [Passive vs. Active Authorization \(p. 65\)](#))
- [When You Schedule a Price Change with Active Authorization \(p. 213\)](#) (for information about active price changes, see [Passive vs. Active Authorization \(p. 65\)](#))
- [When You Cancel an Upcoming Price Change \(p. 214\)](#)
- [When a New Price for Your Product Goes Into Effect \(p. 214\)](#)
- [If the Customer Doesn't Accept the Price Change by the Deadline, Causing Cancellation \(p. 215\)](#)

When You Schedule a Price Change with Passive Authorization

Subject: Price Change for *[product name]*

Greetings from Amazon Payments,

[company name] is changing the price of *[product name]* on *[price change effective date]*.

[Details of new price]

You will be charged the new price for this application starting *[price change effective date]*.

You can cancel your subscription to the application at any time by visiting the following web page:

[http://www.amazon.com/dp-applications?productCode=*product code*](http://www.amazon.com/dp-applications?productCode=<i>product code</i>)

If you have any questions regarding this price change, please contact *[company name]* directly at *[contact e-mail or URL]*.

[If you requested to add your own message to this e-mail, the following is included:]

Here is a message from *[company name]* regarding this price change:

[Your message]

Sincerely,
Amazon Payments

When You Schedule a Price Change with Active Authorization

Subject: Price Change for *[product name]*

Greetings from Amazon Payments,

[company name] is changing the price of *[product name]*

on *[price change effective date]*.

[Details of new price]

In order to continue using this application after *[price change effective date]*, you must agree to the new price by visiting the following web page:
[http://www.amazon.com/dp-applications?productCode=*product code*](http://www.amazon.com/dp-applications?productCode=<i>product code</i>)

Failure to accept this price change will result in the cancellation of your subscription to *[product name]*.

If you have any questions regarding this price change, please contact *[company name]* directly at *[contact e-mail or URL]*.

[If you requested to add your own message to this e-mail, the following is included:]

Here is a message from *[company name]* regarding this price change:

[Your message]

Sincerely,
Amazon Payments

When You Cancel an Upcoming Price Change

You can cancel a scheduled price change up to the day before the price change's effective date. If you cancel after the e-mail notification has already been sent to your customers, Amazon Payments sends another e-mail to alert them of the cancellation.

Subject: Price Change Canceled for *[product name]*

Greetings from Amazon Payments,

[company name] has decided not to change the price for *[product name]*. You will continue to pay the current price of this application.

You can view a summary of your current and past charges by visiting the following web page:
<http://www.amazon.com/dp-applications>

If you have any questions, please contact *[company name]* at *[contact e-mail or URL]*.

Sincerely,
Amazon Payments

When a New Price for Your Product Goes Into Effect

Subject: Pricing Change for *[product name]*

Greetings from Amazon Payments,

This e-mail confirms that as of *[price change effective date]*, the price of *[product name]* sold by *[company name]* is now:

[new pricing summary]

[If your product has a monthly charge that has increased with the price change, the following paragraph is included:]

You will see a one-time charge on your next bill to cover the increased monthly cost of *[product name]* from *[old monthly charge amount]* to *[new monthly charge amount]* for the remainder of this month.

[If your product has a monthly charge that has decreased with the price change, the following paragraph is included:]

You will be issued a refund for the decreased monthly cost of *[product name]* from *[old monthly charge amount]* to *[new monthly charge amount]* for the remainder of this month.

Additional information regarding your payments for this application is available at the following web page:
<http://www.amazon.com/dp-applications>

If you have any questions regarding this price change, please contact *[company name]* at *[contact e-mail or URL]*.

Sincerely,
Amazon Payments

If the Customer Doesn't Accept the Price Change by the Deadline, Causing Cancellation

Subject: Use of *[product name]* Canceled

Greetings from Amazon Payments,

Your access to *[product name]* sold by *[company name]* has been canceled because you chose not to accept the new price for the application, which went into effect on *[price change effective date]*. You will no longer be able to use this application. You will be billed at the end of the month for any use of this application through *[cancellation date]*; after that you will receive no more bills for *[product name]*.

[If a refund is applicable (see [Customer Cancellation \(p. 53\)](#)), the following paragraph is included:]

You will be issued a refund in the amount of *[refund amount]* for this application. This refund amount is the prorated portion of the monthly fee based on the number of days remaining in the month.

If you have questions, please contact *[company name]* at *[contact e-mail or URL]*.

Sincerely,
Amazon Payments

Appendix: Product Registration Flow

Topics

- [Amazon Payments Account Page \(p. 218\)](#)
- [Product Information Page \(p. 219\)](#)
- [Which Amazon Web Services Page \(p. 220\)](#)
- [Pricing Overview Page \(p. 222\)](#)
- [Usage Charges Page \(p. 223\)](#)
- [One-Time Charge and Monthly Charges Page \(p. 227\)](#)
- [Review Pricing Page \(p. 228\)](#)
- [Your Description Page \(p. 229\)](#)
- [Confirmation Page \(p. 230\)](#)
- [Preview Order Page \(p. 232\)](#)
- [Thank You Page \(p. 236\)](#)
- [Product Details Page \(p. 237\)](#)

This appendix shows the web pages you see when registering a DevPay product. Specifically, it shows two things:


- The full registration flow for a fictional DevPay product called ABC AMI, which is a paid AMI
- Specific pages from the registration of another fictional DevPay product called Sky Storage, which uses Amazon S3

The flows for registering a paid AMI product and an Amazon S3 product are essentially the same. However, the pages for setting the usage-based pricing for an Amazon S3 product (e.g., Sky Storage) are slightly different and so are presented here. The Sky Storage pages show how to set tiered pricing (available for both Amazon EC2 products and Amazon S3 products). For more information, see [Tiered Usage-Based Pricing \(p. 31\)](#).

Amazon Payments Account Page

When you register your product, you're first asked to log in with an AWS developer login (for more information, see [Registering Your Product \(p. 60\)](#)). If you decide to create a new AWS developer account for your DevPay products as we recommend, you first go through the steps to set up that account (providing a password, a payment method, a billing address, etc.). Those pages are not shown here. For more information about product development recommendations, see [Recommendations for Product Development \(p. 80\)](#).


After your AWS developer account is set up, we check our records to see if you have an Amazon Payments Business Account associated with that developer account. If you don't, then on this page, you provide the information we need so we can create that account.

**Register Your Product**

[AWS ACCOUNT](#) **AMAZON PAYMENTS ACCOUNT** [PRODUCT INFORMATION](#) [PRICING INFORMATION](#) [CONFIRMATION](#)

Welcome John Smith | [Not You?](#)

Amazon DevPay uses Amazon Payments to process payments from your customers. This step will open an Amazon Payments Business account for you.


Create an Amazon Payments Business Account

Business Name:

Business Address: **AMIMakers, Inc.**
1200 12th Ave. S
Seattle, WA
98144
(206) 555-2222

Customer Service Email Address: Note: This email address and phone number will be made available to your customers.

Customer service phone number: (optional)

Sales Website: (optional)

Type of Ownership:

Primary Business Category:

Date Established: (mm/dd/yyyy)

Security Questions

Select two questions below that we can use to verify your identity and provide an addition level of security.

Security Question 1:
Answer:


Security Question 2:
Answer:

By clicking continue you agree to the terms and conditions of the [Privacy Notice](#) and [User Agreement](#) (including your consent to receive legal notices about your account electronically).

[Return to the registration procedure. \(p. 60\)](#)

Product Information Page

On the next page, the product registration process begins. You enter the information about your product (the name, description, etc.).

**Register Your Product**

[AWS ACCOUNT](#) [AMAZON PAYMENTS ACCOUNT](#) **PRODUCT INFORMATION** [PRICING INFORMATION](#) [CONFIRMATION](#)

Welcome John Smith

Product Information

The information you provide below is exactly what your customers will see when they purchase your product.

Company Name:

Product Name:

Product Description:

Please enter plain text only

Product Redirect URL:

Please provide the web site address of the page you want your customers to be directed to after they have completed their purchase.

Terms and Conditions (Optional)

If you wish, Amazon DevPay can require all customers to agree to Terms and Conditions that apply to the use of your product before they purchase it. If you would like to present Terms and Conditions, please enter them below. Otherwise, just leave this field blank.

Please enter plain text only

Contact Information

Please provide your contact information for product related issues.

Contact email address: †

Contact telephone number: †

Email address or URL for customers with product or technical inquiries:
This will be displayed to customers

† This information will be used for Amazon to contact you regarding any product related issues and will not be displayed to your customers.

[Return to the registration procedure. \(p. 60\)](#)

Which Amazon Web Services Page

On this page, you select which AWS service your product uses.



amazon web services™ Register Your Product

AWS ACCOUNT AMAZON PAYMENTS ACCOUNT **PRODUCT INFORMATION** PRICING INFORMATION CONFIRMATION

Which Amazon Web Services does ABC AMI use?

Amazon Simple Storage Service (Amazon S3)

Amazon Elastic Compute Cloud (Amazon EC2)

Back Continue

If you select Amazon EC2, you must also specify which Amazon EC2 Regions and instance types your product covers.

Note

The following image shows check boxes for three Regions and several instance types. In the future, Amazon EC2 might support additional Regions or instance types, so the interface might look different than the image shown here.

amazon
webservices™ Register Your Product

AWS ACCOUNT AMAZON PAYMENTS ACCOUNT **PRODUCT INFORMATION** PRICING INFORMATION CONFIRMATION

Which Amazon Web Services does ABC AMI use?

Amazon Simple Storage Service (Amazon S3)

Amazon Elastic Compute Cloud (Amazon EC2)

Which Amazon EC2 regions and environments do you want your customers to be able to run ABC AMI on? Your customers will only be able to run ABC AMI on the environments and instance types that you select.

US-West (Northern California) Region

- Amazon EC2 running Linux/UNIX
- Amazon EC2 running Windows
- Amazon EC2 running Windows and SQL Server

US-East (Northern Virginia) Region

- Amazon EC2 running Linux/UNIX
 - Which Amazon EC2 instance types will ABC AMI run on?
 - Small**
 - Large**
 - Extra Large**
 - Double Extra Large**
 - Quadruple Extra Large**
 - High-CPU Medium**
 - High-CPU Extra Large**
- Amazon EC2 running Windows
- Amazon EC2 running Windows and SQL Server

EU (Ireland) Region

- Amazon EC2 running Linux/UNIX
- Amazon EC2 running Windows
- Amazon EC2 running Windows and SQL Server

Back Continue

[Return to the registration procedure. \(p. 60\)](#)

Pricing Overview Page

This page is the first in a series in which you specify your product's price. The first page gives an overview of DevPay product pricing.

The screenshot shows the Amazon DevPay registration process. At the top, the Amazon Web Services logo is followed by the text "Register Your Product". Below this is a progress bar with five steps: "AWS ACCOUNT", "AMAZON PAYMENTS ACCOUNT", "PRODUCT INFORMATION", "PRICING INFORMATION" (which is highlighted), and "CONFIRMATION". A user is logged in as "John Smith". The main content area has a navigation menu with five items: "PRICING OVERVIEW" (selected), "USAGE CHARGES", "ONE-TIME & MONTHLY CHARGES", "REVIEW PRICING", and "YOUR DESCRIPTION". The "Pricing Overview" section explains that users will define the price for their customers and includes a list of Amazon DevPay fees: a 3% value-add fee and a \$0.30 per product fee. It also includes an "Important Pricing Consideration" section.

amazon web services™ Register Your Product

AWS ACCOUNT AMAZON PAYMENTS ACCOUNT PRODUCT INFORMATION **PRICING INFORMATION** CONFIRMATION

Welcome John Smith | Not You?

PRICING OVERVIEW USAGE CHARGES ONE-TIME & MONTHLY CHARGES REVIEW PRICING YOUR DESCRIPTION

Pricing Overview

In the following pages you will define the price that your customers will pay for ABC AMI. You can include one-time, recurring monthly, and/or usage-based charges.

Each month customers pay you based on the price you set, and we collect from you the cost of the AWS services (e.g., Amazon EC2) the customers consume plus the DevPay fees described below.

Amazon DevPay Fees

- 3% of your value-add, which is the difference between the revenue collected from a customer and the cost of AWS services (e.g., Amazon EC2) that the customer used. We charge you this fee only if the revenue from the customer is greater than the cost.
- \$0.30 per product for each bill we collect from your customer.

Important Pricing Consideration

If you set your price so that your customer's bill (e.g., \$20.00) is less than the cost of AWS services that the customer used (e.g., \$22.00), then you are responsible for the difference (\$2.00).

Back Continue

[Return to the registration procedure. \(p. 60\)](#)

Usage Charges Page

Topics


- [Usage Charges for ABC AMI \(p. 224\)](#)
- [Usage Charges for Sky Storage \(p. 225\)](#)

On this page, you enter any usage charges you want to include as part of the product's price. When you enter a value for a particular pricing dimension, the corresponding AWS service cost and your markup (before DevPay fees are subtracted) are displayed in the table.

Note

The following image shows tabs for three Regions and rows for several instance types. Amazon EC2 might support additional Regions or instance types in the future, so the interface might look different than the image shown here.

Usage Charges for ABC AMI



Register Your Product

AWS ACCOUNT
AMAZON PAYMENTS ACCOUNT
PRODUCT INFORMATION
PRICING INFORMATION
CONFIRMATION

PRICING OVERVIEW
USAGE CHARGES
ONE-TIME & MONTHLY CHARGES
REVIEW PRICING
YOUR DESCRIPTION

Usage Charges

Specify **Your Price** for each of the selected instance types. Your customers can launch ABC AMI on the instance types you have checked below. If you choose not to charge for a particular pricing dimension, set the price to 0.00 and your customers will not see that pricing dimension.

		US-East (Northern Virginia) Region	EU (Ireland) Region	US-West (Northern California) Region			
Instance	Details	Your Price(\$) (per hour [†])	-	Amazon EC2 Cost(\$)	=	Markup Before DevPay Fees [†] (\$)	Tiered Pricing (what's this?)
<input checked="" type="checkbox"/> Amazon EC2 running Linux/UNIX							
Box Usage	<input checked="" type="checkbox"/> Small	0.140		0.085		0.055	Add
	<input checked="" type="checkbox"/> Large	0.440		0.340		0.100	Add
	<input checked="" type="checkbox"/> Extra Large	0.840		0.680		0.160	Add
	<input type="checkbox"/> Double Extra Large	0.000		1.200		N/A	Add
	<input type="checkbox"/> Quadruple Extra Large	0.000		2.400		N/A	Add
	<input checked="" type="checkbox"/> High-CPU Medium	0.240		0.170		0.070	Add
	<input checked="" type="checkbox"/> High-CPU Extra Large	0.840		0.680		0.160	Add
<input type="checkbox"/> Amazon EC2 running Windows							
<input type="checkbox"/> Amazon EC2 running Windows and SQL Server							
Data Transfer Type	Details	Your Price(\$)	-	Amazon EC2 Cost(\$)	=	Markup Before DevPay Fees [†] (\$)	Tiered Pricing (what's this?)
Data Transfer	GB of data transfer in	0.140		0.100		0.040	Add
	GB of data transfer out	0.200		0.170		0.030	Add
	GB of regional data transfer in/out	0.020		0.010		0.010	Add

US-East (Northern Virginia) Region
EU (Ireland) Region
US-West (Northern California) Region

Notes: You cannot charge your customers for their use of Amazon Elastic IP addresses, Amazon Elastic Block Store, Elastic Load Balancing, Auto Scaling, or Amazon CloudWatch with your Paid AMIs; AWS bills them directly for this. [Learn more.](#)

[†] Your Markup Before DevPay Fees: The difference between 'Your Price' and the 'Amazon EC2 Cost'.

Back
Continue

[Return to the registration procedure. \(p. 60\)](#)

Usage Charges for Sky Storage

You see a different set of pricing dimensions if your product is based on Amazon S3. Following is the relevant part of the page for Sky Storage.

Note

The following image shows tabs for three Regions. Amazon S3 might support additional Regions in the future, so the interface might include additional tabs.

		US Standard Region	US West (Northern California) Region	EU (Ireland) Region	Asia Pacific (Singapore) Region		
Usage Type	Details	Your Price(\$)	-	Amazon S3 Cost(\$)	=	Markup Before DevPay Fees†(\$)	Tiered Pricing (what's this?)
Storage	GB-Month of storage used	<input type="text" value="0.000"/>		0.150		-0.150	Add
	GB-Month of Reduced Redundancy Storage used	<input type="text" value="0.000"/>		0.100		-0.100	Add
Requests	1,000 PUT or LIST requests	<input type="text" value="0.000"/>		0.010		-0.010	Add
	10,000 GET and all other requests	<input type="text" value="0.000"/>		0.010		-0.010	Add
Data Transfer	GB of data transfer in	<input type="text" value="0.000"/>		0.000		0.000	Add
	GB of data transfer out	<input type="text" value="0.000"/>		0.150		-0.150	Add

† Your Markup Before DevPay Fees: The difference between 'Your Price' and the 'Amazon S3 Cost'.

You can set tiers for any of the usage-based price components (for information about tiered pricing, see [Tiered Usage-Based Pricing \(p. 31\)](#)).

To add tiered pricing

1. Click **Add** in the **Tiered Pricing** column
The image that follows shows an example of the resulting window that is displayed.
2. Click **Add Tier** for each tier you want to add.
You can add as many tiers as you want.
3. Enter the tier value and corresponding price for each tier.
4. When you're done, click **Save & Close**.

The following image shows Sky Storage's tiered pricing for GB-Month of storage used.

[Close](#)

Tiered Pricing: Storage

Tiered pricing allows you to specify a price for your product that changes based on usage amount. Click **Add Tier** for each tier you want to add. Enter the tier value and corresponding price for each tier. When you are done, click **Save & Close**.

[+ Add Tier](#)

Tiers	Your Price(\$)	- Amazon Web Service Price(\$)	= Markup Before DevPay Fees(\$)	
1. per GB-Month of storage used (First <input type="text" value="5"/> GB-Month)	<input type="text" value="0.40"/>	0.15	0.25	Remove Tier
2. per GB-Month of storage used (Next <input type="text" value="10"/> GB-Month) (greater than 5 GB-Month through 15 GB-Month)	<input type="text" value="0.30"/>	0.15	0.15	Remove Tier
3. per GB-Month of storage used (Over 15 GB-Month)	<input type="text" value="0.20"/>	0.15	0.05	

[Remove Tiering](#)


[Save & Close](#) [Next: Data Transfer In](#)

[Close](#)

[Return to the registration procedure. \(p. 60\)](#)

One-Time Charge and Monthly Charges Page

On this page, you enter any sign-up charges and monthly charges you want to include as part of the product's price.

**Register Your Product**

[AWS ACCOUNT](#) [AMAZON PAYMENTS ACCOUNT](#) [PRODUCT INFORMATION](#) **[PRICING INFORMATION](#)** [CONFIRMATION](#)

[PRICING OVERVIEW](#) [USAGE CHARGES](#) **[ONE-TIME & MONTHLY CHARGES](#)** [REVIEW PRICING](#) [YOUR DESCRIPTION](#)

One-Time and Monthly Charges

- You may set a one-time sign-up charge (e.g., \$10.00 licensing charge) and/or a monthly charge (e.g., \$5.00 subscription charge) for the use of ABC AMI.
- DevPay charges a \$0.30 fee per transaction. One-time and/or monthly charges can be used to cover that cost.
- The monthly charge will be prorated for the month when your customers sign up. In subsequent months, they will be charged this fee at the beginning of each month.
- If you choose not to set a one-time or monthly charge, set the **Your Price** column to 0.00 and your customers will not see that charge.

Pricing Dimension	Your Price(\$)	- Amazon EC2 Cost(\$)	= Markup Before DevPay Fees [†] (\$)
One-time Charge	5.00	N/A	5.00
Monthly Charge [*]	19.99	N/A	19.99

[†] **Your Markup Before DevPay Fees:** The difference between 'Your Price' and the 'Amazon EC2 Cost'.


^{*} **Monthly Charge:** The monthly charge will be prorated for the first month.

[Back](#) [Continue](#)

[Return to the registration procedure. \(p. 60\)](#)

Review Pricing Page

This page shows a summary of the pricing information you've provided in the previous pages. You can change any part of the pricing from this page.


Register Your Product

AWS ACCOUNT
AMAZON PAYMENTS ACCOUNT
PRODUCT INFORMATION
PRICING INFORMATION
CONFIRMATION

PRICING OVERVIEW
USAGE CHARGES
ONE-TIME & MONTHLY CHARGES
REVIEW PRICING
YOUR DESCRIPTION

Review Your Pricing

If you want to further edit your price, click the **Change** buttons below.

Usage Charges Change

		US-East (Northern Virginia) Region	EU (Ireland) Region	US-West (Northern California) Region
Instance	Details	Your Price(\$) (per hour†)	Amazon EC2 Cost(\$)	Markup Before DevPay Fees(\$)
Amazon EC2 running Linux/UNIX				
Box Usage	Small	0.140	0.085	0.055
	Large	0.440	0.340	0.100
	Extra Large	0.840	0.680	0.160
	Double Extra Large	N/A	0.000	N/A
	Quadruple Extra Large	N/A	0.000	N/A
	High-CPU Medium	0.240	0.170	0.070
	High-CPU Extra Large	0.840	0.680	0.160
Amazon EC2 running Windows				
Amazon EC2 running Windows and SQL Server				
Data Transfer Type	Details	Your Price(\$)	Amazon EC2 Cost(\$)	Markup Before DevPay Fees(\$)
Data Transfer	GB of data transfer in	0.140	0.100	0.040
	GB of data transfer out	0.200	0.170	0.030
	GB of regional data transfer in/out	0.020	0.010	0.010

US-East (Northern Virginia) Region
EU (Ireland) Region
US-West (Northern California) Region

One-Time and Monthly Charges Change

- At sign-up we bill the customer the one-time charge and the prorated monthly charge for the first month.
- Each month, we bill the customer the monthly charge and any usage charges.

Pricing Dimension	Your Price(\$)	Amazon EC2 Cost(\$)	Markup Before DevPay Fees† (\$)
One-time Charge	5.00	N/A	5.00
Monthly Charge‡	19.99	N/A	19.99

† Your Markup Before DevPay Fees: The difference between 'Your Price' and the 'Amazon EC2 Cost'.

‡ Monthly Charge: The monthly charge will be prorated for the first month.

Back
Continue

[Return to the registration procedure. \(p. 60\)](#)

Your Description Page

On this page you can optionally provide descriptions for any of the charges.

For ABC AMI, we enter a fictional description for one pricing dimension to show you how it appears to customers.


Register Your Product

AWS ACCOUNT
AMAZON PAYMENTS ACCOUNT
PRODUCT INFORMATION
PRICING INFORMATION
CONFIRMATION

PRICING OVERVIEW
USAGE CHARGES
ONE-TIME & MONTHLY CHARGES
REVIEW PRICING
YOUR DESCRIPTION

Your Description (optional)

This is what your customer will see when they sign up for ABC AMI. You may enter any additional information you want to show your customers to help them understand ABC AMI pricing. For example, you may describe 'GB-month of storage used' as, "This stores about 200 MP3s a month".

ABC AMI Pricing

US-East (Northern Virginia) Region

Instance		Your Price (\$)	Details	Description
Amazon EC2 running Linux/UNIX				
Box Usage	Small	0.140	per hour (or partial hour) consumed	ABC AMI's description of this pricing
	Large	0.440	per hour (or partial hour) consumed	Enter your optional description here
	Extra Large	0.840	per hour (or partial hour) consumed	Enter your optional description here
	High-CPU Medium	0.240	per hour (or partial hour) consumed	Enter your optional description here
	High-CPU Extra Large	0.840	per hour (or partial hour) consumed	Enter your optional description here
Data Transfer Type		Your Price (\$)	Details	Description
Data Transfer	GB of data transfer in	0.140	per GB	Enter your optional description here
	GB of data transfer out	0.200	per GB	Enter your optional description here
	GB of regional data transfer in/out	0.020	per GB	Enter your optional description here

Pricing Dimension	Your Price (\$)	Description
One-time Charge	5.00	Enter your optional description here
Recurring Monthly Charge	19.99	Enter your optional description here

Back Continue

[Return to the registration procedure. \(p. 60\)](#)

Confirmation Page

The **Confirmation** page displays a summary of the product information for you to review. You can make changes from this page. Clicking **Preview Order Page** at the bottom of the page shows the information as it will be displayed to your customers (for an example, see [Preview Order Page \(p. 232\)](#)).

The **Confirmation** page is long, so we've displayed it here in two parts.

The screenshot shows the Amazon DevPay Confirmation Page for ABC AMI. At the top left is the Amazon Web Services logo and the text "Register Your Product". Below this is a progress bar with five steps: AWS ACCOUNT, AMAZON PAYMENTS ACCOUNT, PRODUCT INFORMATION, PRICING INFORMATION, and CONFIRMATION. The CONFIRMATION step is currently active. A blue callout box contains the text: "Please review your product information and pricing for ABC AMI. Click 'Register Your Product' at the bottom of the page when you are done." The main content area is divided into three sections, each with a "Change" button:

- ABC AMI Product Information**:
 - Company Name: AMIMakers, Inc.
 - Product Name: ABC AMI
 - Product Description: Product description for ABC AMI
 - Product Redirect URL: http://abcami.amimakers.com
- ABC AMI Contact Information**:
 - Please note, this information will **not** be shown to customers.
 - Contact Email Address: internalcontact@amimakers.com
 - Contact Phone Number: 206 555 1111
 - Contact Information for Customers with Product or Technical Inquiries: customercontact@amimakers.com
- ABC AMI Terms and Conditions**:
 - Terms and Conditions for ABC AMI

ABC AMI Pricing Information [Change](#)

US-East (Northern Virginia) Region

Instance	Your Price (\$)	Details	Description
Amazon EC2 running Linux/UNIX			
Box Usage	Small	0.140	per hour (or partial hour) consumed
	Large	0.440	per hour (or partial hour) consumed
	Extra Large	0.840	per hour (or partial hour) consumed
	High-CPU Medium	0.240	per hour (or partial hour) consumed
	High-CPU Extra Large	0.840	per hour (or partial hour) consumed
ABC AMI's description of this pricing item.			
Data Transfer			
Data Transfer Type	Your Price (\$)	Details	Description
Data Transfer	GB of data transfer in	0.140	per GB
	GB of data transfer out	0.200	per GB
	GB of regional data transfer in/out	0.020	per GB

Pricing Dimension	Your Price (\$)	Description
One-time Charge	5.00	
Recurring Monthly Charge	19.99	

ABC AMI Terms and Conditions

By clicking the 'Register Product' button, you agree to the [Amazon Web Services Customer Agreement](#).

Preview the ABC AMI order page as it will be displayed to your customers. [Preview Order Page](#)

Click "Register Product" to enable Amazon DevPay to bill for your product. [Register Product](#)

[Return to the registration procedure. \(p. 60\)](#)


Preview Order Page

Topics

- [Preview Page for ABC AMI \(p. 233\)](#)
- [Preview Page for Sky Storage \(p. 235\)](#)

This page shows what your customers see when they purchase your product. The **Place your order** button displayed on this preview page is inactive, so you can't click the button and go through the purchase experience from here. However, if you want to see an example of the full purchase experience, see [Appendix: The Customer Purchase Experience \(p. 199\)](#).

Preview Page for ABC AMI



Review the information below, then click "Place your order." [▶ Place your order](#)

<p>Total due today</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">One-time charge:</td> <td style="text-align: right; padding: 2px;">\$5.00</td> </tr> <tr> <td style="padding: 2px;">Prorated recurring monthly charge:</td> <td style="text-align: right; padding: 2px;">\$19.99</td> </tr> <tr> <td style="padding: 2px;">Total:</td> <td style="text-align: right; padding: 2px;">\$24.99</td> </tr> </table>	One-time charge:	\$5.00	Prorated recurring monthly charge:	\$19.99	Total:	\$24.99	<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">Payment Method: Change</td> <td style="padding: 2px;">Billing Address: Change</td> </tr> <tr> <td style="padding: 2px;">Visa: ***-11111</td> <td style="padding: 2px;">Em Vingt</td> </tr> <tr> <td style="padding: 2px;">Exp: 10/2010</td> <td style="padding: 2px;">123 Any Street</td> </tr> <tr> <td></td> <td style="padding: 2px;">Any City, State 12345</td> </tr> <tr> <td></td> <td style="padding: 2px;">Phone:</td> </tr> </table>	Payment Method: Change	Billing Address: Change	Visa: ***-11111	Em Vingt	Exp: 10/2010	123 Any Street		Any City, State 12345		Phone:
One-time charge:	\$5.00																
Prorated recurring monthly charge:	\$19.99																
Total:	\$24.99																
Payment Method: Change	Billing Address: Change																
Visa: ***-11111	Em Vingt																
Exp: 10/2010	123 Any Street																
	Any City, State 12345																
	Phone:																

Application Information

ABC AMI

Product description for ABC AMI

ABC AMI Pricing†

One-Time and Monthly Charges

Pricing Dimension	Your Price (\$)	Description
One-Time Charge	5.00	
Recurring Monthly Charge	19.99	

Usage Charges - US-East (Northern Virginia) Region

Instance	Your Price (\$)	Details	Description
Amazon EC2 running Linux/UNIX			
Small	0.140	per hour (or partial hour) consumed	ABC AMI's description of this pricing item.
Large	0.440	per hour (or partial hour) consumed	
Extra Large	0.840	per hour (or partial hour) consumed	
High-CPU Medium	0.240	per hour (or partial hour) consumed	
High-CPU Extra Large	0.840	per hour (or partial hour) consumed	
Box Usage			
Data Transfer Type	Your Price (\$)	Details	Description
GB of data transfer in	0.140	per GB	Data Transfer
GB of data transfer out	0.200	per GB	
GB of regional data transfer in/out	0.020	per GB	

† Price includes the Amazon EC2 price. If you use Amazon Elastic IP addresses, Amazon Elastic Block Store, Auto Scaling, or Amazon CloudWatch, you will be charged separately. At this time, you cannot use Elastic Load Balancing with instances of ABC AMI.

* Amazon EC2 with Windows Authentication Services and Amazon EC2 with Windows Authentication Services and SQL are deprecated and will be removed from Amazon DevPay products in November 2009.

Note: ABC AMI is sold by AMIMakers, Inc.. Amazon Payments will charge your credit card for any one time and recurring charges outlined in the pricing above. Until you cancel your subscription to ABC AMI, any recurring charges will be charged on the 1st of every month and will include any applicable usage charges.


Terms and Conditions

By clicking the 'Place your Order' button, you indicate that you have read and agree to the [Billing Services Agreement](#) and [ABC AMI Terms of Use](#).

Review the information above, then click "Place your order." [▶ Place your order](#)

[Return to the registration procedure. \(p. 60\)](#)

Preview Page for Sky Storage



Review the information below, then click "Place your order."
▶ Place your order

Total due today		Payment Method: Change	Billing Address: Change
One-time charge:	\$5.00	Visa: ***-11111	Em Vingt
Prorated recurring monthly charge:	\$19.99	Exp: 10/2010	123 Any Street
Total:	\$24.99		Any City, State 12345
			Phone:

Application Information

Sky Storage

Product description for Sky Storage

Sky Storage Pricing

One-Time and Monthly Charges

Pricing Dimension	Your Price (\$)
One-Time Charge	5.00
Recurring Monthly Charge	19.99

Usage Charges - US Standard Region

Usage Type	Your Price (\$)	Details
Data Transfer	GB of data transfer in	0.150 per GB
	GB of data transfer out	0.200 per GB
	GB-Month of storage used	0.400 per GB-Mo (First 5 GB-Month)
Storage		0.300 per GB-Mo (Greater than 5 GB-Month through 15 GB-Month)
		0.200 per GB-Mo (Over 15 GB-Month)

Usage Charges - EU (Ireland) Region

Usage Type	Your Price (\$)	Details
Storage	GB-Month of storage used	0.400 per GB-Mo (First 5 GB-Month)
		0.300 per GB-Mo (Greater than 5 GB-Month through 15 GB-Month)
		0.200 per GB-Mo (Over 15 GB-Month)
Data Transfer	GB of data transfer in	0.150 per GB
	GB of data transfer out	0.200 per GB

Note: Sky Storage is sold by Skysonsa, Inc. Amazon Payments will charge your credit card for any one time and recurring charges outlined in the pricing above. Until you cancel your subscription to Sky Storage, any recurring charges will be charged on the 1st of every month and will include any applicable usage charges.

Terms and Conditions

By clicking the 'Place your Order' button, you indicate that you have read and agree to the [Billing Services Agreement](#) and [Sky Storage Terms of Use](#).

Review the information above, then click "Place your order."
▶ Place your order


[Return to the registration procedure. \(p. 60\)](#)

Thank You Page


This is the page that is displayed after you click **Register Product** on the preceding **Confirmation** page.


If we created an Amazon Payments account for you during product registration, the page instructs you to verify your e-mail address with Amazon Payments (for more information, see [Getting Your Account](#) (p. 15)).

The page also instructs you to click **visit your product detail page** to go to the page where your product identifiers are displayed (for an example of the Product Details page, see [Product Details Page](#) (p. 237)).

 **Register Your Product**

AWS ACCOUNT AMAZON PAYMENTS ACCOUNT PRODUCT INFORMATION PRICING INFORMATION **CONFIRMATION**

 **ABC AMI has been submitted for approval to Amazon DevPay.**

 **Wait, you're not quite done yet.**

- **Check your email and click the verification link.**
Amazon Payments has just sent a message to johnsmith5@example.com. Please verify your email address by clicking the link in that message. This is mandatory — your Amazon DevPay product will not be approved until this has been done. Once your product is approved, you will receive an email.
- **Start integrating with Amazon DevPay.**
Visit the Amazon DevPay Activity page at <http://aws.amazon.com/devpayactivity> and click **View Product Details** to get your product identifiers or review/change information about your product.

[Return to the registration procedure.](#) (p. 60)

Product Details Page

The **Product Details** page contains the information you provided about your product, as well as items you need in order to integrate the product with DevPay (the product code, etc.). You can view this page at any time by going to your DevPay Activity Page at <http://aws.amazon.com/devpayactivity> and clicking **View Product Details**.

Amazon DevPay Developer Guide Product Details Page

Product Registration Information

Your application requires the credentials listed below, as described in the Amazon DevPay Developer Guide. When you are ready to sell ABC AMI, you can direct your customers to the Purchase URL below.

Status: Pending Approval *(as of October 30, 2009)*

Product Code: [Product Code here]

Product Identification Token: Please take care to safeguard this token, which you should never share with others directly. To learn how to integrate it securely with your product, please refer to the Amazon DevPay Developer Guide.
[+ Show](#)

Purchase URL: [Purchase URL here]

Product Information [Change](#)

Company Name: AMIMakers, Inc.

Product Name: ABC AMI

Product Redirect URL: http://abcami.amimakers.com

Contact Email Address: internalcontact@amimakers.com

Contact Phone Number: 206 555 1111

E-mail address or URL for customers with product or technical inquiries: customercontact@amimakers.com
(This will be displayed to customers)

Product Description:
Product description for ABC AMI

Terms and Conditions:
Terms and Conditions for ABC AMI

ABC AMI Pricing Plan [Change](#)

One-Time and Monthly Charges

Pricing Dimension	Your Price (\$)	Description
One-Time Charge	5.00	
Recurring Monthly Charge	19.99	

Usage Charges - US-East (Northern Virginia) Region

Instance	Your Price (\$)	Details	Description
Amazon EC2 running Linux/UNIX			
Box Usage	Small	0.140	per hour (or partial hour) consumed
	Large	0.440	per hour (or partial hour) consumed
	Extra Large	0.840	per hour (or partial hour) consumed
	High-CPU Medium	0.240	per hour (or partial hour) consumed
	High-CPU Extra Large	0.840	per hour (or partial hour) consumed
Data Transfer Type			
Data Transfer	GB of data transfer in	0.140	per GB
	GB of data transfer out	0.200	per GB
	GB of regional data transfer in/out	0.020	per GB

[> Cancel ABC AMI](#)

[Return to Previous](#)

[Return to the registration procedure. \(p. 60\)](#)

Appendix: Example DevPay Activity Pages

This appendix walks through a simple example in which you create an imaginary DevPay product to see what the numbers look like on the DevPay Activity page and in the DevPay transaction history over a period of a few months. The goal is to familiarize you with the pages and help you understand the billing, collection, and revenue reporting process DevPay uses. The example covers multiple months because:

- The billing and collection cycle for a single month's revenue spans two months
- We want to show the DevPay Activity page during a month when you have both existing customers and new ones signing up

In this example, you create a paid AMI called ABC AMI. You begin selling ABC AMI in June 2009. The following table shows the price you charge and the corresponding AWS cost.

Note

The Amazon EC2 prices shown here were valid when this document was originally written. The prices might have changed since then. For the current Amazon EC2 prices, go to [the Amazon EC2 product page](#).

Pricing Dimension	Your Price	AWS Cost
Sign-up charge	\$0.00	N/A
Monthly charge	\$20.00	N/A
Small instance-hour (or part) used	\$0.20	\$0.10
Large instance-hour (or part) used	\$0.50	\$0.40
Extra-large instance-hour (or part) used	\$0.90	\$0.80
Data transfer in	\$0.00	\$0.10
Data transfer out	\$0.00	\$0.17

Note that you *don't* charge customers for data transfer in and out. It's an unusual pricing scheme, but we've chosen it to shorten and simplify the example. Also, to keep the example short, we're ignoring any other instance types and EC2 features (like high-CPU instances, regional data transfer, Elastic Block Store, etc.). For the full list of usage-based costs for Amazon EC2, see [Allowed Usage-Based Charges \(p. 30\)](#).

The following sections show the activity for your products over a period of three months:

- [June \(p. 240\)](#)
- [July \(p. 251\)](#)
- [August \(p. 259\)](#)

June

Topics

- [Your June DevPay Activity Page on June 14 \(p. 240\)](#)
- [Your June DevPay Activity Page at the End of June \(p. 245\)](#)
- [Your DevPay Transaction History in June \(p. 248\)](#)
- [Your June DevPay Activity Page in Early July \(p. 250\)](#)

On June 1 you introduce ABC AMI, and several days later people begin signing up for it. During June, five people sign up. They pay a prorated monthly charge to cover the remaining portion of June.

Your June DevPay Activity Page on June 14

We'll first focus on what your DevPay Activity Page looks like in the middle of June after your customers have been using ABC AMI for a while.

Usage

The following table shows your customers' usage details as of June 14. By this date, only three of the five customers who will sign up in June have signed up and started to use ABC AMI. Customer A signed up on June 3, Customer B on June 4, and Customer C on June 5. Notice that Customer C signs up to use your product, but doesn't use it during June.

	Customer			
Small instance-hours used	0	8	0	8
Large instance-hours used	3	0	0	3
X-large instance-hours used	5	0	0	5
GB data uploaded	64	40	0	104
GB data downloaded	35	10	0	45

Revenue

The next table shows your revenue as of June 14. This includes *actual* revenue and *expected* revenue. The actual revenue covers the prorated monthly fees your customers already paid at sign-up. The expected revenue covers their usage up to June 14 (they don't actually pay that money until we bill them on July 1). In this example, your usage-based revenue comes only from small, large, and extra-large instances used. Remember that to keep the example short, we're ignoring all other usage-based dimensions for Amazon EC2 (such as high-CPU instance types, Elastic Block Store, regional data transfer, etc).

	Customer			
June prorated monthly fee	\$18.67	\$18.00	\$17.33	\$54.00
Small instance-hours used	\$0.00	\$1.60	\$0.00	\$1.60
Large instance-hours used	\$1.50	\$0.00	\$0.00	\$1.50
X-large instance-hours used	\$4.50	\$0.00	\$0.00	\$4.50

	Customer			
Total	\$24.67	\$19.60	\$17.33	\$61.60

AWS Costs

The next table shows the AWS costs you owe based on your customers' usage amounts. These costs are based on the amount AWS charges for Amazon EC2.

Note

The Amazon EC2 prices shown here were valid when this document was originally written. The prices might have changed since then. For the current Amazon EC2 prices, go to [the Amazon EC2 product page](#).

	Customer			
Small instance-hours used	\$0.00	\$0.80	\$0.00	\$0.80
Large instance-hours used	\$1.20	\$0.00	\$0.00	\$1.20
X-large instance-hours used	\$4.00	\$0.00	\$0.00	\$4.00
GB data uploaded	\$6.40	\$4.00	\$0.00	\$10.40
GB data downloaded	\$5.95	\$1.70	\$0.00	\$7.65
Total	\$17.55	\$6.50	\$0.00	\$24.05

Value-Add

The next table shows your value-add for each customer, which is the difference between the revenue from the customer and the AWS costs the customer incurred (both of these values are shown in the preceding tables).

	Customer		
Revenue	\$24.67	\$19.60	\$17.33
AWS Costs	\$17.55	\$6.50	\$0.00
Value-Add	\$7.12	\$13.10	\$17.33
3% DevPay Fee on Value-Add	\$0.21	\$0.39	\$0.52

As of June 14, each of the individual value-add amounts that you expect to receive from your customers is greater than zero. The sum of them is $\$7.12 + \$13.10 + \$17.33 = \37.55 (we calculate the value-add amounts per individual customer, but we're mentioning this summed value because it explicitly appears on the DevPay Activity page).

We charge you a 3% DevPay fee on each value add amount that is greater than zero. This means that as of June 14, you owe $\$0.21 + \$0.39 + \$0.52 = \1.13 .

DevPay Activity Page

The preceding tables show the details of the usage, revenue, and value-add amounts for each customer up to June 14. We present a summary of this information across all your customers on your DevPay Activity page, which you can view at any time. Following is an image of your DevPay Activity page on June 14, which shows the month-to-date summary for ABC AMI.

Note

During the month, the DevPay Activity page reflects the current month's data up to and through the previous day. Therefore, on the first day of the month, the page has no data to display yet and so shows zeros.

The dates displayed on the DevPay Activity page and related pages are in Coordinated Universal Time (Greenwich Mean Time).

The page is divided into two main areas: the main summary and then a larger area containing a table for each DevPay product you have. The **Summary** section at the top summarizes the revenue, costs, and fees across *all* your DevPay products (in this example you have only one product). It shows both the revenue you *expect* to receive (assuming all your customers pay their bills), and the *actual* revenue that we have collected. You can come back to this page at any time to see how much of your expected revenue for the month we've collected from your customers. Note that during the month, there is typically a big difference between the *expected* values and the *collected* values, because we don't bill you or your customers for much of that money until after the end of the month.

Following the summary is a table for each DevPay product you have. Each table has three sections:

- The top part shows the product's name and three links on the right:
 - **View Product Details**—Displays the information you provided when you registered the product. You can edit the information there. The page also displays the *purchase URL*, *product code*, and *product token* for the product.
 - **Change Pricing**—Steps you through the process of changing the product's price (for more information, see [Changing Pricing \(p. 64\)](#)).
 - **View Customer Reports**—Takes you to a page where you can access the DevPay reports (for more information, see [Reports \(p. 69\)](#)).
- The middle part of the table shows the revenue the product has generated during the month. Currently it shows month-to-date values as of June 14.
- The bottom part shows the costs the product incurred during the month. The section includes both the AWS service costs (in this case, the costs of Amazon EC2) and the DevPay fees. Currently it shows month-to-date values as of June 14.

DevPay Activity

View:

Summary (snapshot as of June 14, 2009)

	1 Expected (\$)	Collected (\$)
Total Revenue	61.60	54.00
DevPay Fee	-2.93	-0.90
AWS Costs	-24.05	-0.00
Total Net Proceeds	34.62	53.10

ABC AMI Activity (snapshot as of June 14, 2009)

[View Product Details](#) | [Change Pricing](#) | [View Customer Reports](#)

Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	2 54.00
Amazon EC2 small instance-hours used	\$0.200 × 8 hours	1.60
Amazon EC2 large instance-hours used	\$0.500 × 3 hours	1.50
Amazon EC2 x-large instance-hours used	\$0.900 × 5 hours	4.50
		61.60
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.100 × 8 hours	-0.80
Amazon EC2 large instance-hours used	\$0.400 × 3 hours	-1.20
Amazon EC2 x-large instance-hours used	\$0.800 × 5 hours	-4.00
Amazon EC2 data uploaded	\$0.100 × 104 GB	-7.65
Amazon EC2 data downloaded	\$0.170 × 45 GB	-7.65
DevPay Fee	4 (3% × \$37.55) + (6 transactions × \$0.30)	-2.93
	Totals	-26.98

- 1** The summary includes two columns: **Expected** and **Collected**. The **Expected** column shows:
- Your total expected revenue for June to date (assuming your customers pay their bills when we bill them on July 1)
 - The DevPay fees you're expected to pay for June to date
 - The AWS costs you're expected to pay when we bill you after July 1 (the amount is based on your customers' usage of your product to date in June)
- For June to date, you expect to receive \$61.60 in revenue. You've already received \$54.00 of that in the form of prorated monthly fees, so the \$54.00 shows up in the **Collected** column. You're also expected to pay \$2.93 in DevPay fees for June to date. You've already paid \$0.90 of that in the form of the \$0.30 transaction fees we subtracted when your first three customers signed up and paid the prorated monthly charges. We'll bill you for any remaining fees you owe after July 1. Finally, you're also expected to pay \$24.05 in AWS costs for June to date. However, we won't bill you for that until after July 1, so 0.00 is displayed in the **Collected** column for now.
- 2** \$54.00 is the total of the prorated monthly charges your three customers paid immediately when they signed up. The description says "Monthly charges including prorated monthly amounts," but at this point you have only prorated monthly amounts. Next month you'll have both full monthly charges and prorated amounts because your existing customers will be charged full monthly fees, and then more people will sign up and be charged prorated monthly fees.
- 3** Your customers have used 8 small, 3 large, and 5 extra-large instance hours by June 14. The page shows the price they pay you for this usage and the corresponding costs you pay to AWS.

4

The DevPay fee includes 3% of the sum of the individual value-add amounts that are greater than zero. These were shown in a preceding table to be \$7.12 for Customer A, \$13.10 for Customer B, and \$17.33 for Customer C, and they total \$37.55.

The DevPay fee also includes the \$0.30 transaction fee that we collect each time we charge your customers. Therefore we include six instances of the \$0.30 fee because we charged each of your three customers when they signed up during June, and we will charge them on July 1 for their June usage.

Note

The July 1 bill will cover your revenue for both June (usage) and July (monthly fees); however, the \$0.30 transaction fee per customer collected on July 1 is reported on the June statement.

Notice that the number of transactions shown here (six) does not equal the number of customers you currently have (three). To get the number of customers you have for a product, use the subscription report (for more information, see [Subscription Report \(p. 69\)](#)).

Important

In the main **Revenue** section of the table, there could be a slight discrepancy between the math shown in the **Details** column (in the middle) and the value in the **Totals** column (on the right). This is due to rounding. We round sub-penny charges up to a whole penny, and we round charges over a penny up or down to the closest penny.

Your June DevPay Activity Page at the End of June

Now we'll look at your DevPay Activity page at the end of June. After June 14, your three existing customers continue to use ABC AMI, and two new customers sign up (one on June 15 and the other on June 20). The following tables (which are similar in format to the preceding ones) show your ABC AMI customers' usage, revenue, and value-add details at the end of June.

Usage

	Customer					
Small instance-hours used	0	12	2	9	1	24
Large instance-hours used	5	0	6	0	0	11
X-large instance-hours used	5	0	4	11	30	50
GB data uploaded	72	43	22	112	68	317
GB data downloaded	35	10	19	14	44	122

Revenue

The next table shows your revenue as of the end of June. This includes *actual* revenue and *expected* revenue. The actual revenue covers the prorated monthly fees your customers already paid at sign-up. The expected revenue covers their usage through the end of June.

	Customer					
June prorated monthly fee	\$18.67	\$18.00	\$17.33	\$10.67	\$7.33	\$72.00
Small instance-hours used	\$0.00	\$2.40	\$0.40	\$1.80	\$0.20	\$4.80
Large instance-hours used	\$2.50	\$0.00	\$3.00	\$0.00	\$0.00	\$5.50
X-large instance-hours used	\$4.50	\$0.00	\$3.60	\$9.90	\$27.00	\$45.00
Total	\$25.67	\$20.40	\$24.33	\$22.37	\$34.53	\$127.30

AWS Costs

The next table shows the AWS costs you owe based on the usage amounts shown in the preceding usage table.

Amazon DevPay Developer Guide
June

	Customer					
Small instance-hours used	\$0.00	\$1.20	\$0.20	\$0.90	\$0.10	\$2.40
Large instance-hours used	\$2.00	\$0.00	\$2.40	\$0.00	\$0.00	\$4.40
X-large instance-hours used	\$4.00	\$0.00	\$3.20	\$8.80	\$24.00	\$40.00
GB data uploaded	\$7.20	\$4.30	\$2.20	\$11.20	\$6.80	\$31.70
GB data downloaded	\$5.95	\$1.70	\$3.23	\$2.38	\$7.48	\$20.74
Total	\$19.15	\$7.20	\$11.23	\$23.28	\$38.38	\$99.24

Value-Add

At the end of June, we again calculate the individual value-add amount for each customer (as shown in the following table). Note that some of the amounts are less than zero, meaning you have no value-add for that particular customer (and therefore pay no 3% DevPay fee for that customer; you still, however, pay the \$0.30 transaction fee).

	Customer				
Revenue	\$25.67	\$20.40	\$24.33	\$22.37	\$34.53
AWS Costs	\$19.15	\$7.20	\$11.23	\$23.28	\$38.38
Value-Add	\$6.52	\$13.20	\$13.10	- \$0.91	- \$3.85
3% DevPay Fee on Value-Add	\$0.20	\$0.40	\$0.39	N/A	N/A

The value-add amounts for Customer D and E are less than zero because based on the price you set for ABC AMI, the prorated monthly fee they paid plus the usage-based revenue wasn't enough to cover the AWS costs they incurred. Depending on their usage in July (a month where they pay the full \$20.00 monthly fee), you might have a value-add amount that is greater than zero for each of them.

The following table compares the value-add amounts for Customers A–C on June 14 and at the end of June.

	Customer		
Value-add on June 14	\$7.12	\$13.10	\$17.33
Value-add at end of June	\$6.52	\$13.20	\$13.10

We use the final sum at the end of June as the basis for the 3% DevPay fee that you pay after July 1. However, the DevPay Activity page is updated daily with the current sum of all value-add amounts greater than zero.

DevPay Activity Page

Following is the final June 2009 DevPay Activity page that summarizes the details in the preceding tables.

DevPay Activity		
View:	June 2009 Statement	
Summary (June 1 - June 30, 2009)		
	1 Billed (\$)	Collected (\$)
Total Revenue	127.30	72.00
DevPay Fee	-3.98	-1.50
AWS Costs	-99.24	-0.00
Total Net Proceeds	\$24.08	\$70.50
ABC AMI Activity (June 1 - June 30, 2009)		
View Product Details Change Pricing View Customer Reports		
Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	2 72.00
Amazon EC2 small instance-hours used	\$0.200 × 24 hours	4.80
Amazon EC2 large instance-hours used	\$0.500 × 11 hours	5.50
Amazon EC2 x-large instance-hours used	\$0.900 × 50 hours	45.00
		127.30
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.100 × 24 hours	-2.40
Amazon EC2 large instance-hours used	\$0.400 × 11 hours	-4.40
Amazon EC2 x-large instance-hours used	\$0.800 × 50 hours	-40.00
Amazon EC2 data uploaded	\$0.100 × 317 GB	-31.70
Amazon EC2 data downloaded	\$0.170 × 122 GB	-20.74
DevPay Fee	3 (3% × \$32.82) + (10 transactions × \$0.30)	-3.98
		Totals \$99.24 -103.22

- | | |
|----------|---|
| 1 | <p>In the summary, the Expected label has changed to Billed to reflect that the month is closed and we're moving on to the end-of-month billing cycle.</p> <p>Your total revenue for June is \$127.30. You've already received \$72.00 in the form of prorated monthly fees (shown in the Collected column).</p> <p>You're expected to pay \$3.98 in DevPay fees for June. You've already paid \$1.50 of that in the form of the \$0.30 transaction fees we subtracted when your five customers signed up and paid the prorated monthly charges. After July 1, we'll bill you for any remaining fees you owe.</p> <p>Finally, you're expected to pay \$99.24 in AWS costs for June. However, we won't bill you for that until after July 1, so 0.00 is displayed in the Collected column for now.</p> |
| 2 | <p>The additional two customers have signed up since June 14, so the \$72.00 is the total of the prorated monthly charges all five customers paid when they signed up.</p> |
| 3 | <p>The DevPay fee includes 3% of the sum of the individual value-add amounts that are greater than zero. These were shown in a preceding table to be \$6.52 for Customer A, \$13.20 for Customer B, and \$13.10 for Customer C, and they total \$32.82.</p> <p>The DevPay fee also includes the \$0.30 transaction fee we collect each time we charge your customers. Therefore we include 10 instances of the \$0.30 fee because we charged each of your five customers when they signed up during June, and we will charge them on July 1 for their June usage.</p> <p>Note
The July 1 bill will cover your revenue for both June (usage) and July (monthly fees); however, the \$0.30 transaction fee per customer collected on July 1 is reported on the June statement.</p> |

Your DevPay Transaction History in June

At the bottom of the DevPay Activity page is a link to your **DevPay Transaction History** page. Following is the page at the end of June. It shows each deposit and charge related to DevPay. We aggregate all the deposits that occur on a single day, and all charges that occur on a single day.

DevPay Transaction History

Your Amazon Payments Balance (as of June 30, 2009) [Withdraw to Bank Account](#)

Total Balance: \$70.50

Balance Available for Withdrawal: \$70.50

DevPay Transaction History (as of June 30, 2009)

View: for:

Date (MM/DD/YYYY)	Description	Amount (\$)
06/20/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	7.03
06/15/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	10.37
06/05/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	17.03
06/04/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	17.70
06/03/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	18.37

The deposits are the prorated monthly charges your customers paid when they signed up during June. The sum total of their payments is \$72.00 (which is shown on the preceding Account Activity page). If you subtract the \$0.30 transaction fees (5 x \$0.30), the result is \$70.50. This is the current total balance displayed on the page.

Note

The DevPay transaction history is updated continuously as deposits or withdrawals are made during the day. However, the DevPay Activity page is updated once each day. Because of this, the DevPay transaction history might show entries that are not yet reflected in the DevPay Activity page.

The following table shows the entries you can expect to see in the DevPay transaction history. For all entries where we collect money from you, we indicate whether we charge your Amazon Payments account or your credit card.

Entry	Description
Deposit (Revenue collected minus \$0.30 fee per customer)	A deposit we make into your account when we collect money from your customers. For more information, see When and How You Get Paid (p. 37) .
Charge (Revenue collected minus \$0.30 fee per customer)	A charge we make for the net amount you owe if the revenue collected from your customer is less than the \$0.30 fee you owe for that customer. We typically aggregate these types of charges because they're very small (less than \$0.30), so the listed amount might cover multiple charges. For more information, see When You Have a Small Monthly Charge (p. 38) .

Entry	Description
AWS Costs and 3% DevPay Fees for MM/YYYY	The monthly charge we make for: <ul style="list-style-type: none">• The cost of AWS services used by your customers (for more information, see The Cost of AWS Services (p. 24))• The 3% DevPay fees on each individual value-add amount greater than zero (for more information, see DevPay Fees (p. 20))
Customer Refunds	A charge we make for any recent refunds we have processed for your customers. For more information, see Customer Refunds and Chargebacks (p. 57) .
Withdrawal	A withdrawal you make from your Amazon Payments account to your bank account. For more information, see Your Amazon Payments Account (p. 15) .

Note

The account balance displayed on the DevPay Transaction History page does not necessarily reflect all the items displayed in the DevPay transaction history. For more information, see [Your Amazon Payments Account \(p. 15\)](#).

Your June DevPay Activity Page in Early July

You can view your June DevPay Activity Page in July to see how much of the June revenue you've received. We update the numbers in the **Collected** column daily with the latest information.

Following is the June DevPay Activity page as displayed in July after we've billed your customers and charged you for the AWS costs and DevPay fees. We've successfully collected the remaining revenue (and deposited it into your Amazon Payments account) and successfully charged you for all the costs and remaining fees you owe. Therefore, the amounts in the **Collected** column match the amounts in the **Billed** column. To see how those remaining deposits and charges appear in your DevPay Transaction History, see [Your DevPay Transaction History for July](#) (p. 255).

DevPay Activity		
View:	June 2009 Statement	
<hr/>		
Summary (June 1 - June 30, 2009)		
	Billed (\$)	Collected (\$)
Total Revenue	127.30	127.30
DevPay Fee	-3.98	-3.98
AWS Costs	-99.24	-99.24
Total Net Proceeds	\$24.08	\$24.08
<hr/>		
ABC AMI Activity (June 1 - June 30, 2009)		
View Product Details Change Pricing View Customer Reports		
Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	72.00
Amazon EC2 small instance-hours used	\$0.200 x 24 hours	4.80
Amazon EC2 large instance-hours used	\$0.500 x 11 hours	5.50
Amazon EC2 x-large instance-hours used	\$0.900 x 50 hours	45.00
		127.30
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.100 x 24 hours	-2.40
Amazon EC2 large instance-hours used	\$0.400 x 11 hours	-4.40
Amazon EC2 x-large instance-hours used	\$0.800 x 50 hours	-40.00
Amazon EC2 data uploaded	\$0.100 x 317 GB	-31.70
Amazon EC2 data downloaded	\$0.170 x 122 GB	-20.74
DevPay Fee	(3% x \$32.82) + (10 transactions x \$0.30)	-3.98
		-103.22

July

Topics

- [Your July DevPay Activity Page at the End of July \(p. 251\)](#)
- [Your DevPay Transaction History for July \(p. 255\)](#)
- [Your July DevPay Activity Page on August 4 \(p. 256\)](#)
- [Your July DevPay Activity Page after August 8 \(p. 258\)](#)

On July 1 we bill your five customers for their June usage and the July monthly fees. In this example, their payments all come in immediately. In the next day or so, we charge you for the costs of Amazon EC2 used by your customers in June, and for the 3% DevPay fee on the value-add amount per customer. For more information about how we collect the 3% fee, see [When and How You Get Paid \(p. 37\)](#).

During July, two more people sign up for ABC AMI (for simplicity, we'll say they both sign up on July 16). You receive the new customers' prorated monthly fees to cover the rest of July when they sign up. Also, Customer B cancels on July 21, and we issue the customer a prorated refund for the remaining days in July. We still bill the customer on August 1 for any pre-cancellation usage. At the end of July, you have six customers signed up for ABC AMI.

Your July DevPay Activity Page at the End of July

Your customers use ABC AMI during July. The following tables (which are similar in format to the preceding ones for June) show your customers' usage, revenue, and value-add details at the end of July.

Usage

	Customer							
Small instance-hours used	0	10	9	13	0	0	49	81
Large instance-hours used	0	0	0	0	0	12	0	12
X-large instance-hours used	8	0	4	3	155	0	0	170
GB data uploaded	118	14	34	98	82	57	76	479
GB data downloaded	44	19	60	120	21	49	78	391

Revenue

The next table shows your revenue at the end of July. This includes *actual* revenue and *expected* revenue.

The actual revenue covers the full July monthly fees your first five customers paid on July 1, plus the prorated monthly fees the two new customers paid at sign-up. For Customer B, the July monthly fee in the table is still listed as \$20.00. We'll account for the cancellation refund we give Customer B elsewhere.

Amazon DevPay Developer Guide
July

The expected revenue covers all the customers' usage during July (they don't actually pay that money until we bill them on August 1).

	Customer							
July monthly fee (full or prorated)	\$20.00	\$20.00	\$20.00	\$20.00	\$20.00	\$10.32	\$10.32	\$114.19
Small instance-hours used	\$0.00	\$2.00	\$1.80	\$2.60	\$0.00	\$0.00	\$9.80	\$16.20
Large instance-hours used	\$0.00	\$0.00	\$0.00	\$0.00	\$0.00	\$6.00	\$0.00	\$6.00
X-large instance-hours used	\$7.20	\$0.00	\$3.60	\$2.70	\$139.50	\$0.00	\$0.00	\$153.00
Total	\$27.20	\$22.00	\$25.40	\$25.30	\$159.50	\$16.32	\$20.12	\$295.84

AWS Costs

The next table shows the AWS costs you owe based on the usage amounts shown in the preceding usage table.

	Customer							
Small instance-hours used	\$0.00	\$1.00	\$0.90	\$1.30	\$0.00	\$0.00	\$4.90	\$8.10
Large instance-hours used	\$0.00	\$0.00	\$0.00	\$0.00	\$0.00	\$4.80	\$0.00	\$4.80
X-large instance-hours used	\$6.40	\$0.00	\$3.20	\$2.40	\$124.00	\$0.00	\$0.00	\$136.00
GB data uploaded	\$11.80	\$1.40	\$3.40	\$9.80	\$8.20	\$5.70	\$7.60	\$47.90
GB data downloaded	\$7.48	\$3.23	\$10.20	\$20.40	\$3.57	\$8.33	\$13.26	\$66.47
Total	\$25.68	\$5.63	\$17.70	\$33.90	\$135.77	\$18.83	\$25.76	\$263.27

Value-Add

The next table shows your value-add for each customer, which is the difference between the revenue from the customer and the AWS costs the customer incurred. Note that the table also has a row to show the refund we gave Customer B for the unused portion of the \$20.00 monthly fee.

Amazon DevPay Developer Guide
July

	Customer						
Revenue	\$27.20	\$22.00	\$25.40	\$25.30	\$159.50	\$16.32	\$20.12
Refund	\$0.00	\$6.45	\$0.00	\$0.00	\$0.00	\$0.00	\$0.00
AWS Costs	\$25.68	\$5.63	\$17.70	\$33.90	\$135.77	\$18.83	\$25.76
Value-Add	\$1.52	\$9.92	\$7.70	- \$8.60	\$23.73	- \$2.51	- \$5.64
3% DevPay Fee on Value-Add	\$0.05	\$0.30	\$0.23	N/A	\$0.71	N/A	N/A

At the end of July, only some of the individual value-add amounts that you can expect to receive from your customers are greater than zero. In terms of the 3% DevPay fees, you will owe \$0.05 + \$0.30 + \$0.23 + \$0.71 = \$1.29 when we charge you after August 1.

DevPay Activity Page

Following is the final July 2009 DevPay Activity page that summarizes the details in the preceding tables.

DevPay Activity

View: July 2009 Statement GO

Summary (July 1 - July 31, 2009)

	1 Billed (\$)	Collected (\$)
Total Revenue	295.84	120.64
DevPay Fee	-3.99	-0.60
AWS Costs	-263.27	-0.00
Customer Refunds	-6.45	-6.45
Total Net Proceeds	22.13	113.59

ABC AMI Activity (July 1 - July 31, 2009)

[View Product Details](#) |
 [Change Pricing](#) |
 [View Customer Reports](#)

Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	2 120.64
Amazon EC2 small instance-hours used	\$0.200 x 81 hours	16.20
Amazon EC2 large instance-hours used	\$0.500 x 12 hours	6.00
Amazon EC2 x-large instance-hours used	\$0.900 x 170 hours	153.00
		295.84
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.100 x 81 hours	-8.10
Amazon EC2 large instance-hours used	\$0.400 x 12 hours	-4.80
Amazon EC2 x-large instance-hours used	\$0.800 x 170 hours	-136.00
Amazon EC2 data uploaded	\$0.100 x 479 GB	-47.90
Amazon EC2 data downloaded	\$0.170 x 391 GB	-66.47
DevPay Fee	3 (3% x \$42.87) + (9 transactions x \$0.30)	-3.99
		-267.26

Amazon DevPay Developer Guide
July

1	<p>Your total revenue for July is \$295.84. You've already received \$120.64 in the form of full and prorated monthly fees (shown in the Collected column).</p> <p>You're expected to pay \$3.99 in DevPay fees for July. You've already paid \$0.60 of that in the form of the \$0.30 transaction fees we subtracted when your two new customers paid the prorated monthly charges at sign-up.</p> <p>You're expected to pay \$263.27 in AWS costs for July. However, we won't bill you for that until after August 1, so 0.00 is displayed in the Collected column for now.</p> <p>Finally, Customer B canceled on August 21. We refunded the customer the prorated amount to cover the 10 remaining days in July (\$6.45). We then charged you that amount.</p>
2	<p>The \$120.64 is the sum of:</p> <ul style="list-style-type: none">• The \$100.00 in monthly charges for July for the five existing customers (billed on July 1)• The \$20.64 in prorated monthly charges for July for your two new customers (billed when they signed up) <p>The five existing customers were billed on July 1 for the \$100.00, along with the usage charges for June (discussed in the section for June). In this example, the payments came in immediately on July 1 (shown on the following DevPay Transaction History page).</p> <p>On July 16 you received the money you were due from the \$20.64 prorated monthly charges.</p>
3	<p>The DevPay fee includes 3% of the sum of the individual value-add amounts that are greater than zero. These were shown in a preceding table to be \$1.52, \$9.92, \$7.70, and \$23.73, and they total \$42.87.</p> <p>The DevPay fee also includes the \$0.30 transaction fee we collect each time we charge your customers. Therefore we're charging nine instances of the \$0.30 fee to cover:</p> <ul style="list-style-type: none">• When we billed the two new customers for the sign-up charges• When we later bill all seven customers for their July usage (on August 1). Even though one of your customers canceled in July, we still charge that customer for any pre-cancellation usage. <p>Note The August 1 bill will cover your revenue for both July (usage) and August (monthly fees); however, the \$0.30 transaction fee per customer collected on August 1 is reported on the July statement.</p>

Your DevPay Transaction History for July

Following is the DevPay Transaction History page at the end of July.

DevPay Transaction History

Your Amazon Payments Balance (as of July 31, 2009) [Withdraw to Bank Account](#)

Total Balance: \$137.66

Balance Available for Withdrawal: \$137.66

DevPay Transaction History (as of July 31, 2009)

View: for:

Date (MM/DD/YYYY)	Description	Amount (\$)
07/21/2009	Customer Refunds (DevPay Account charged)	-6.45
07/16/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	20.04
07/02/2009	AWS Costs and 3% DevPay Fees for 06/2009 (DevPay Account charged)	-100.22
07/01/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	153.80

Starting at the bottom of the page:

- The July 1 deposit of \$153.80 corresponds to the July 1 billing of your customers for their June usage and the July monthly fees. To recap: On July 1 your customers paid \$55.30 for their usage in June and \$100.00 for the July monthly fee, so the total they paid was $\$55.30 + \$100.00 = \$155.30$. If you subtract the \$0.30 transaction fee per customer, the result is \$153.80.
- On July 2 we charged you \$99.24 for the cost of Amazon EC2 your customers used in June, plus \$0.98, which is the 3% DevPay fee on the total of the value-add amounts greater than zero. The total charge is \$100.22.
- On July 16 you received the \$20.04 in prorated payments from the two new customers who signed up that day. To recap: Those two customers paid \$20.64 when they signed up. If you subtract the \$0.30 transaction fee per customer, the result is \$20.04.
- On July 21, Customer B canceled, and we charged you for that refund.

Your July DevPay Activity Page on August 4

On August 1 we bill your customers for their July usage and the August monthly fees. A day or so later, we bill you for the costs of Amazon EC2 used by your customers in July, and for the DevPay fees you owe for July. Customer A's payment of \$27.20 on August 1 fails, but it succeeds when we retry the payment on August 7. Following is the July DevPay Activity page as displayed on August 4. The summary table at the top has been updated to reflect the latest amounts collected.

Note

If we initially can't collect a customer's payment, we contact the customer by e-mail to provide a valid credit card. We then retry the collection three times, once every 7 days. If after 21 days our retries are still unsuccessful, that payment remains uncollected. We cancel the customer's access to your product. For more information about what happens when a customer doesn't pay, see [If a Customer Doesn't Pay the Monthly Bill](#) (p. 42).

DevPay Activity		
View:	July 2009 Statement	
Summary (July 1 - July 31, 2009)		
	Billed (\$)	Collected (\$)
Total Revenue	295.84	288.64
DevPay Fee	-3.99	-3.64
AWS Costs	-263.27	-257.59
Customer Refunds	-6.45	-6.45
Total Net Proceeds	22.13	20.96
ABC AMI Activity (July 1 - July 31, 2009)		
View Product Details Change Pricing View Customer Reports		
Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	120.64
Amazon EC2 small instance-hours used	\$0.200 × 81 hours	16.20
Amazon EC2 large instance-hours used	\$0.500 × 12 hours	6.00
Amazon EC2 x-large instance-hours used	\$0.900 × 170 hours	153.00
		295.84
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.100 × 81 hours	-8.10
Amazon EC2 large instance-hours used	\$0.400 × 12 hours	-4.80
Amazon EC2 x-large instance-hours used	\$0.800 × 170 hours	-136.00
Amazon EC2 data uploaded	\$0.100 × 479 GB	-47.90
Amazon EC2 data downloaded	\$0.170 × 391 GB	-66.47
DevPay Fee	(3% × \$42.87) + (9 transactions × \$0.30)	-3.99
		-267.26

- 1** Customer A's payment of \$27.20 failed, so out of the \$295.84 in revenue you're due, you're missing \$7.20 in Customer A's usage-based revenue (leaving \$288.64 collected so far). The other \$20.00 Customer A was supposed to pay on August 1 was to cover the August monthly payment. The August DevPay Activity page would reflect that some of the August revenue was not yet collected.
- 2** Because Customer A's payment failed on August 1, you haven't yet paid all the DevPay fees you owe. Specifically, you haven't paid the \$0.30 transaction fee for Customer A's payment, and you haven't paid the 3% DevPay fee (of \$0.05) on Customer A's \$1.52 value-add. Therefore, of the \$3.99 in DevPay fees you owe, you've paid only \$3.64 of it as of August 4. After Customer A's \$27.20 payment succeeds, we'll subtract the \$0.30 from the payment and later charge you the \$0.05.

3

Because Customer A's payment failed, you don't have enough July revenue from Customer A to cover the AWS costs that Customer A incurred in July (\$25.68). So far, Customer A has paid you only \$20.00 (the July monthly fee paid on July 1). Therefore, when we collect the AWS costs for July from you on August 2, for Customer A, we collect only \$20.00. Therefore, of the \$263.27 in AWS costs you owe for July, you've paid only \$257.59 as of August 4. Later, if Customer A's \$27.20 payment goes through, then we can collect the remaining \$5.68 in AWS costs (and you'll get your remaining \$7.20 in revenue).

To see how the August 1 deposits and August 2 charges appear in your DevPay Transaction History, see [August \(p. 259\)](#).

Your July DevPay Activity Page after August 8

We're able to collect Customer A's payment of \$27.20 on August 7. Following is the July DevPay Activity page after August 8 (when we collected the remaining AWS costs and DevPay fees). The summary table at the top has been updated to reflect the latest amounts collected.

DevPay Activity		
View:	July 2009 Statement	<input type="button" value="GO"/>
<hr/>		
Summary (July 1 - July 31, 2009)		
	Billed (\$)	Collected (\$)
Total Revenue	295.84	295.84
DevPay Fee	-3.99	-3.99
AWS Costs	-263.27	-263.27
Customer Refunds	-6.45	-6.45
Total Net Proceeds	22.13	22.13
<hr/>		
ABC AMI Activity (July 1 - July 31, 2009)		
View Product Details Change Pricing View Customer Reports		
Description	Details	Total (\$)
Revenue		
Recurring Monthly Charges	Monthly charges including prorated amounts	120.64
Amazon EC2 small instance-hours used	\$0.200 × 81 hours	16.20
Amazon EC2 large instance-hours used	\$0.500 × 12 hours	6.00
Amazon EC2 x-large instance-hours used	\$0.900 × 170 hours	153.00
		295.84
AWS Costs & DevPay Fee		
Amazon EC2 small instance-hours used	\$0.100 × 81 hours	-8.10
Amazon EC2 large instance-hours used	\$0.400 × 12 hours	-4.80
Amazon EC2 x-large instance-hours used	\$0.800 × 170 hours	-136.00
Amazon EC2 data uploaded	\$0.100 × 479 GB	-47.90
Amazon EC2 data downloaded	\$0.170 × 391 GB	-66.47
DevPay Fee	(3% × \$42.87) + (9 transactions × \$0.30)	-3.99
		-267.26

Customer A's payment of \$27.20 succeeded, so you finally received the remaining \$7.20 in revenue. We subtracted the \$0.30 transaction fee from the payment and deposited the remainder in your Amazon Payments account. On August 8, we charged your account the remaining \$5.68 in AWS costs you owe and the 3% DevPay fee of \$0.05 you owe. At this point, you've received all of your July revenue, and we've collected all the AWS costs and DevPay fees for July.

To see how the August 7 deposits and August 8 charges appear in your DevPay Transaction History, see [August \(p. 259\)](#).

August

In this section, we'll just look at the DevPay Transaction History page so you can see how the charges for July are collected given Customer A's initial failed payment.

DevPay Transaction History		
Your Amazon Payments Balance (as of August 31, 2009)		Withdraw to Bank Account
Total Balance: \$66.20		
Balance Available for Withdrawal: \$66.20		
DevPay Transaction History (as of August 31, 2009)		
		View: All transactions <input type="button" value="v"/> for: Last 30 days <input type="button" value="v"/>
Date (MM/DD/YYYY)	Description	Amount (\$)
08/14/2009	Withdrawal (DevPay Account to bank account [ending in 2533])	-100.00
08/08/2009	AWS Costs and 3% DevPay Fees for 07/2009 (DevPay Account charged)	-5.73
08/07/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	26.90
08/02/2009	AWS Costs and 3% DevPay Fees for 07/2009 (DevPay Account charged)	-258.83
08/01/2009	Deposit (Revenue collected minus \$0.30 fee per customer)	266.20

Starting from the bottom of the page:

- The August 1 deposit corresponds to the August 1 billing of your customers for their July usage and the August monthly fees. The total billed amount includes \$175.20 for the July usage and \$120.00 for the August monthly fees (totaling \$295.20). However, Customer A's payment of \$27.20 fails. Therefore, you receive the total amount minus Customer A's payment, minus the \$0.30 transaction fee per customer payment collected ($\$295.20 - \$27.20 - (\$0.30 * 6) = \266.20).
- On August 2 we charge you \$258.83. This includes two amounts:
 - \$257.59 for the cost of Amazon EC2 your customers used in July. For Customer A, we collect only the EC2 costs up to the amount of revenue we've already collected. Therefore, we collect \$20.00, which is equivalent to the July monthly fee we've already collected from Customer A. The total amount of AWS costs for Customer A is \$25.68, so you still owe \$5.68.
 - \$1.24 for the 3% DevPay fee on the value-add amounts we've collected from your customers (all except Customer A, whose payment failed). The total 3% fee you owe for *all* your customers is \$1.29, but we won't collect the 3% fee for Customer A (\$0.05) until that payment succeeds.
- On August 7, we retry Customer A's \$27.20 payment and it succeeded. We deposited \$26.90 into your account (\$27.20 minus the \$0.30 fee).
- On August 8, we charge you \$5.73. This includes two amounts:
 - \$5.68 for the remaining AWS costs for Customer A
 - \$0.05 for the 3% DevPay fee on Customer A's value-add for July ($3\% * \$1.52$)
- On August 14, you withdraw \$100.00 to your bank account.

Important

If you had decided to withdraw money from your account in June or July, it's possible the balance on August 2 wouldn't have covered the \$257.59 charge. If that had been the case, we would have charged the remaining amount to your credit card. We recommend you plan ahead and

leave sufficient funds in your Amazon Payments account to cover the charge for AWS costs and DevPay fees.

Appendix: Risks to DevPay Products

All Internet-based products are subject to risk. This appendix outlines several possible threats to Amazon S3 DevPay products (they're not applicable to paid or supported AMLs). Keep these in mind when designing your product.

Malicious User Impersonates a Desktop Product

Who experiences the risk: You

A malicious user who understands how DevPay works signs up to use your DevPay desktop product. The user then extracts the product token from the product and uses the token with another product. You become liable for the charges incurred by the other product.

You should design your desktop product to guard your product token.

Malicious User Steals a Customer's Credentials

Who experiences the risk: Your customer

A malicious user steals customer credentials from an insecure desktop or database and impersonates the customer. The customer is then billed for the malicious user's use of the product.

You should harden or obfuscate all stored credentials. Encourage your customers to closely monitor their bills and usage.

Malicious User Re-Signs Requests

Who experiences the risk: You

This is a classic "man-in-the-middle" attack. A malicious user who understands how DevPay works doesn't want to pay the prices required by a DevPay product. He signs up for his own AWS developer account and signs up to use the AWS service your DevPay product uses. He inserts a man in the middle attack, whereby he removes the DevPay product token from the request, inserts his own AWS developer credentials, and re-signs the request. He receives your product's functionality at the normal AWS service price (instead of the higher price charged by you), and you receive no revenue for his use of your product.

Amazon DevPay Resources

The following table lists related resources that you'll find useful as you work with DevPay.

Resource	Description
Amazon DevPay Getting Started Guide	The getting started guide describes the major tasks required to use DevPay. We recommend you read the getting started guide first. It also shows how to update an example Amazon S3 application to work with DevPay. Instructions for Java, C++, and Ruby are included.
Amazon DevPay Release Notes	The Release Notes give a high-level overview of the current release. They specifically note any new features, corrections, and known issues.
Technical documentation for related AWS services	You might need the documentation for Amazon EC2 or Amazon S3. For those documents, go to the http://aws.amazon.com/documentation .
AWS Developer Resource Center	A central starting point to find documentation, code samples, release notes, and other information to help you build innovative applications with AWS.
Discussion Forums	A community-based forum for developers to discuss technical questions related to Amazon Web Services.
AWS Support Center	The home page for AWS Technical Support, including access to our Developer Forums, Technical FAQs, Service Status page, and AWS Premium Support (if you are subscribed to this program).
Product information about Amazon DevPay	The primary web page for information about Amazon DevPay.
E-mail address for questions related to your DevPay product: <DevPay@amazon.com>	This e-mail address is <i>only</i> for questions about the business side of your DevPay product. For technical questions, use the Discussion Forums.
Contact Us	A central contact point for inquiries concerning AWS billing, account, events, abuse, etc.
Conditions of Use	Detailed information about the copyright and trademark usage at Amazon.com.

Document History

This documentation is associated with the 2007-12-01 release of Amazon DevPay. This guide was last updated on 16 May 2013.

Change	Description	Release Date
General Update	Added an expanded breadcrumb to the HTML version, and reduced the front matter content.	in this release
Amazon EC2 Spot Instances	Updated the information about paid and supported AMIs to reflect that DevPay does not support Spot Instances. For more information, see Current Limitations (p. 86) .	13 December 2009
New Region	Updated images of the product registration process to include the new US-West (Northern California) Region.	2 December 2009
Amazon EBS-Backed AMIs	Updated the information about paid and supported AMIs to reflect that DevPay only supports Amazon S3-backed AMIs (those that use an instance store root device), and not Amazon EBS-backed AMIs (those that use an Amazon EBS root device).	2 December 2009
Activation Timing	Added information about the timing of activation vs. the timing of when the customer is subscribed to DevPay products that use Amazon S3. For more information, see Activation and Subscription Timing (p. 116) for desktop products, and see Activation and Subscription Timing (p. 127) for web products.	3 August 2009
Auto Scaling and Amazon CloudWatch	Added information about Auto Scaling and Amazon CloudWatch and their use with paid or supported AMIs. For more information, see Costs You're Not Responsible For (p. 27) .	18 May 2009
Elastic Load Balancing	Updated the guide to discuss that Elastic Load Balancing is currently not available for use with instances of paid or supported AMIs. For more information, see Allowed Usage-Based Charges (p. 30) .	18 May 2009

Document Conventions

This section lists the common typographical and symbol use conventions for AWS technical publications.

Typographical Conventions

This section describes common typographical use conventions.

Convention	Description/Example
Call-outs	<p>A call-out is a number in the body text to give you a visual reference. The reference point is for further discussion elsewhere.</p> <p>You can use this resource regularly. 1</p>
Code in text	<p>Inline code samples (including XML) and commands are identified with a special font.</p> <p>You can use the command <code>java -version</code>.</p>
Code blocks	<p>Blocks of sample code are set apart from the body and marked accordingly.</p> <pre># ls -l /var/www/html/index.html -rw-rw-r-- 1 root root 1872 Jun 21 09:33 /var/www/html/index.html # date Wed Jun 21 09:33:42 EDT 2006</pre>
Emphasis	<p>Unusual or important words and phrases are marked with a special font.</p> <p>You <i>must</i> sign up for an account before you can use the service.</p>
Internal cross references	<p>References to a section in the same document are marked.</p> <p>See Document Conventions (p. 265).</p>

Convention	Description/Example
Logical values, constants, and regular expressions, abstracta	A special font is used for expressions that are important to identify, but are not code. If the value is <code>null</code> , the returned response will be <code>false</code> .
Product and feature names	Named AWS products and features are identified on first use. Create an <i>Amazon Machine Image</i> (AMI).
Operations	In-text references to operations. Use the <code>GetHITResponse</code> operation.
Parameters	In-text references to parameters. The operation accepts the parameter <i>AccountID</i> .
Response elements	In-text references to responses. A container for one <code>CollectionParent</code> and one or more <code>CollectionItems</code> .
Technical publication references	References to other AWS publications. If the reference is hyperlinked, it is also underscored. For detailed conceptual information, see the <i>Amazon Mechanical Turk Developer Guide</i> .
User entered values	A special font marks text that the user types. At the password prompt, type <code>MyPassword</code> .
User interface controls and labels	Denotes named items on the UI for easy identification. On the File menu, click Properties .
Variables	When you see this style, you must change the value of the content when you copy the text of a sample to a command line. <code>% ec2-register <your-s3-bucket>/image.manifest</code> See also the following symbol convention.

Symbol Conventions

This section describes the common use of symbols.

Convention	Symbol	Description/Example
Mutually exclusive parameters	(Parentheses and vertical bars)	Within a code description, bar separators denote options from which one must be chosen.
		<pre>% data = hdfread (start stride edge)</pre>
Optional parameters XML variable text	[square brackets]	Within a code description, square brackets denote completely optional commands or parameters.
		<pre>% sed [-n, -quiet]</pre>
		Use square brackets in XML examples to differentiate them from tags. <pre><CustomerId>[ID]</CustomerId></pre>
Variables	<arrow brackets>	Within a code sample, arrow brackets denote a variable that must be replaced with a valid value. <pre>% ec2-register <your-s3-bucket>/image.manifest</pre>

Index

A

Access Key ID, 108
 account balance, 15, 16, 239
 account history, 16, 17, 239
 account, Amazon Payments, 15
 activate URL, 203
 ActivateDesktopProduct, 114, 173
 ActivateHostedProduct, 125, 177
 activation
 desktop products, 112, 114
 web products, 123, 125
 activation key, 80, 108
 activity page for product owner, 17
 advertising your product, 84
 Amazon CloudWatch, 27
 Amazon EBS, 27
 Amazon EC2 in Europe, 30
 Amazon Elastic Block Store, 27
 Amazon Payments, 4, 15
 Amazon Payments and logo, 4
 Amazon S3, 104
 bucket limits, 105
 copying objects, 137
 data
 accessing, 11
 how affected by cancellation, 11, 53
 URL structure, 104
 desktop products, 112
 logging, 138
 POST
 with desktop products, 118
 with web products, 132
 query string authentication
 with desktop products, 118
 with web products, 130
 Requester Pays, 142
 web products, 123
 AMIs, 85
 and activation key, 99
 and Amazon CloudWatch, 27
 and Amazon Elastic Block Store, 27
 and Auto Scaling, 27
 and elastic IP addresses, 27
 and product code, 94
 and regional data transfer, 27
 FAQs, 97
 in Europe, 30
 product configuration, 88, 92
 rebundling, 97
 selling multiple AMIs, 90
 Windows, 30
 API for License Service, 149
 Application Activation page, 203

Application Billing page, 51
 authentication
 for desktop products, 112
 for License Service requests, 153
 for web products, 123
 Auto Scaling, 27
 AWS costs, 24, 248

B

balance for Amazon Payments account, 16
 bandwidth charges, 24, 27, 32
 bank account, 4, 15
 billing customers, 37, 48
 browser-based uploads with POST
 with desktop products, 118
 with web products, 132
 bucket limits, 105

C

canceling a customer, 54
 cancellation
 by customer, 53
 for customer non-payment, 56
 of a customer by you, 54
 of DevPay by you, 63
 verifying customer cancellation, 59, 190, 194
 when customer doesn't pay, 56
 changing
 pricing, 64, 213
 product configuration, 64
 product information, 63
 chargebacks, 58
 charges (as listed in your transaction history), 248
 configuration for AMIs
 about, 88
 changing, 64, 92
 copying Amazon S3 objects, 137
 customer
 billing, 48, 209
 cancellation, 53
 experience, 8
 nonpayment, 56
 notification of new customer, 69
 purchase experience, 199
 refunds, 57
 sign-up, 37, 50, 199
 support, 59

D

dashboard
 for customers, 51
 for product developer, 17, 239
 data in S3
 accessing, 11
 how affected by cancellation, 11, 53
 URL structure, 104

data transfer charges, 24, 27, 32
deposits and charges, 239
DevPay Activity page, 17, 239
DevPay fees, 20, 37
DevPay transaction history, 16, 17

E

e-mails
 to customers, 206
 to you, 46
EBS, 27
EC2 paid AMIs, 85
 and activation key, 99
 and Amazon CloudWatch, 27
 and Amazon Elastic Block Store, 27
 and Auto Scaling, 27
 and elastic IP addresses, 27
 and product code, 94
 and regional data transfer, 27
 FAQs, 97
 in Europe, 30
 product configuration, 88, 92
 rebundling, 97
 Reserved Instances, 86, 97
 selling multiple AMIs, 90
 Windows, 30
Elastic Block Store, 27
elastic IP addresses, 27
Elastic Load Balancing, 30
errors for License Service, 169
Europe, Amazon EC2 in, 30

F

fees, DevPay, 20
fraud risks, 261

G

GetActiveSubscriptionsByPid, 182
going live, 82

H

hacking risks, 261

L

License Service
 API reference, 149
 use with desktop products, 115
 use with EC2 paid AMIs, 99
 use with web products, 126
limits on buckets, 105
logging with Amazon S3 DevPay products, 138
logo usage, 4

M

migrating from one product to another, 92, 107

monthly charge for customers, 29
moving from one product to another, 92, 107

N

new customers, notification of, 69
number of customers, 69

P

paid AMIs, 85
 and activation key, 99
 and Amazon CloudWatch, 27
 and Amazon Elastic Block Store, 27
 and Auto Scaling, 27
 and elastic IP addresses, 27
 and product code, 94
 and regional data transfer, 27
 FAQs, 97
 product configuration, 88, 92
 rebundling, 97
 Reserved Instances, 86, 97
 selling multiple AMIs, 90
payments to you, 37
persistent identifier (PID), 69
 desktop products, 120
 EC2 paid AMIs, 99
 GetActiveSubscriptionsByPid, 182
 summary of when used, 108
 VerifyProductSubscriptionByPid, 190
 web products, 135

POST

 with desktop products, 118
 with web products, 132

pre-signed URLs

 with desktop products, 118
 with web products, 130

pricing

 changing, 64, 213
 components, 29
 how to set, 19
 tiers, 31

product code, 60

 summary of when used, 108
 web products, 190

product configuration

 about, 88
 changing, 64, 92

product information, 60, 63

product registration, 217

product token, 60

 desktop products, 112, 194
 summary of when used, 108
 web products, 123

production, 82

purchase URL, 60

Q

- query string authentication
 - with desktop products, 118
 - with web products, 130

R

- rebundling an AMI, 89
- RefreshUserToken, 185
- refunds for customers, 57
- regional data transfer, 27
- registering product, 60, 217
- reports
 - based on Amazon S3 logs, 138
 - DevPay reports, 69
- Requester Pays buckets, 142
- Reserved Instances, 86, 97
- REST-Query requests for License Service, 151
- revenue report, 69
- risks, 261

S

- S3, 104
 - bucket limits, 105
 - copying objects, 137
 - data
 - accessing, 11
 - how affected by cancellation, 11, 53
 - URL structure, 104
 - desktop products, 112
 - logging, 138
 - POST
 - with desktop products, 118
 - with web products, 132
 - query string authentication
 - with desktop products, 118
 - with web products, 130
 - Requester Pays, 142
 - web products, 123
- Secret Access Key, 108
- security risks, 261
- selling your product, 84
- sign-up charge for customers, 29
- signing up for DevPay, 60
- SOAP requests for License Service, 150
- subscription report, 69
- subscription status, 116, 127
- support, customer, 59

T

- taxes, 36
- testing, 82
- tiered pricing, 31
- tiered usage types, 24, 32
- transaction history, 16, 17
- transactions, 239

U

- upgrading from one product to another, 92, 107
- URL structure for S3 data objects, 104
- usage charges for customers, 29
- usage report, 69
- user token
 - desktop products, 112, 194
 - summary of when used, 108
 - web products, 123

V

- value-add, 20, 37, 239
- verifying bank account, 15
- verifying subscription status, 116, 127
- VerifyProductSubscriptionByPid, 127
- VerifyProductSubscriptionByPID, 190
- VerifyProductSubscriptionByTokens, 116, 194

W

- Windows AMIs, 30, 90
- withdrawing money, 4, 16, 17