PAYPAL HACKS™

100 Industrial-Strength Tips & Tools

Shannon Sofield,
Dave Nielsen & Dave Burchell

O'REILLY®

HACK #61 Sell Subscriptions to Your Online Content

Combine a database, PayPal subscriptions, and the IPN system to manage subscriber accounts.

If your web site offers something special that people are willing to pay for, such as access to a technical information database or specialized business-to-business commerce site, you might want to offer subscriptions. PayPal makes it easy. Using IPN, your web server, and your online database, you can easily create an entirely automated system.

Many adult sites on the Internet are available on a subscription basis. Don't offer subscriptions to these sorts of sites with PayPal. Your site's content must be allowed under PayPal's Acceptable Use Policy; otherwise, you might find that your account has been limited **[Hack #5]**.

For the purposes of this example, let's say you offer access to a Rhesus monkey marketing database for the low, low price of $30 per month. This opt-in database contains the monkey name, monkey age, caregiver name, and mailing address of over 10,000 monkeys across North America. You offer your subscribers, typically Rhesus monkey supply vendors, access to this information for marketing purposes.

You'll need four things to implement your subscription business model:

- A Subscribe button on your web site
- An online database that includes a subscribers table
- An IPN script to keep tabs on new, renewed, and expired subscriptions
- Dynamic pages that check a visitor's status before allowing access

Creating a Subscribe Button

The Subscribe button for your site can come straight from PayPal's button generator on the Merchant Tools page (log into PayPal and click the Merchant Tools tab). This example (created without encryption) should look familiar if you have created any unencrypted Buy Now or Donate Now buttons. The variables `a3`, `p3`, and `t3` set the amount, period, and time unit of the subscription, respectively:

```
<html>
<head><title>Monkey Market Database</title></head>
<body>

<form action="https://www.paypal.com/cgi-bin/webscr" method="post">
```

```
<input type="image" src="https://www.paypal.com/en_US/i/btn/x-click-but20.gif"
      border="0" name="submit" alt="Make payments with PayPal - it's fast,
      free and secure!">
<input type="hidden" name="cmd" value="_xclick-subscriptions">
<input type="hidden" name="business" value="burchell@inebraska.com">
<input type="hidden" name="item_name" value="Monkey Market">
<input type="hidden" name="item_number" value="mm-1">
<input type="hidden" name="no_note" value="1">
<input type="hidden" name="currency_code" value="USD">
<input type="hidden" name="a3" value="30.00">
<input type="hidden" name="p3" value="1">
<input type="hidden" name="t3" value="M">
<input type="hidden" name="src" value="1">
</form>

</body>
</html>
```

Setting Up Your Database

Your access control database can be simple. A single table, shown in Table 6-1, containing the email address and the password of your subscriber is all you need. For this example, the table `subscribers` contains two alphanumeric fields: `email` and `password`. You could issue customer usernames to your subscribers, but you might be better served if you follow PayPal's example and use email addresses to identify users. Passwords can be stored as plain text.

Table 6-1. A database to keep track of your subscribers

ID	email	password
4005	shannon@paypalhacks.com	sR3Du4#m77ca
4006	dave@paypalhacks.com	go3@c23-dad43
4007	david@paypalhacks.com	fae0v32c&ewf2

Processing Subscriber Notifications

You need to handle two kinds of notifications from PayPal: the addition of new subscribers to your database when they sign up and removal of subscribers whose subscriptions lapse or are cancelled. Here's a snippet of ASP that does this (see the "Database Coding and Platform Choices" section of the Preface for database considerations):

```
<!-- Standard IPN processing here -->

<%

if Request.Form("txn_type") == "subscr_signup" then
```

```
 ' Add this subscriber to the database
 ' Use SQL like this:
  set cInsSubscr = Server.CreateObject("ADODB.Command")
  cInsSubscr.ActiveConnection = "DRIVER={Microsoft Access Driver
          (*.mdb)};DBQ="C:/InetPub/wwwroot/database/dbPayPal.mdb")
  cInsSubscr.CommandText = "INSERT INTO subscriber (email, password) VALUES
      ( '" & Request.Form("payer_email") & "', 'drowssap')"
  cInsSubscr.CommandType = 1
  cInsSubscr.CommandTimeout = 0
  cInsSubscr.Prepared = true
  cInsSubscr.Execute( )

 ' Email the password to the new subscriber
elsif
 Request.form("txn_type") == "subscr_cancel" then

 ' Remove a subscriber from the database
 ' Use SQL like this:
  set cDelSubscr = Server.CreateObject("ADODB.Command")
  cDelSubscr.ActiveConnection = "DRIVER={Microsoft Access Driver
          (*.mdb)};DBQ="C:/InetPub/wwwroot/database/dbPayPal.mdb")
  cDelSubscr.CommandText = "DELETE * FROM subscriber WHERE email =
           '" & Request.Form("payer_email") & "'"
  cDelSubscr.CommandType = 1
  cDelSubscr.CommandTimeout = 0
  cDelSubscr.Prepared = true
  cDelSubscr.Execute( )

end

%>
```

Don't really give every one of your subscribers the same password (`drowssap` in this example). Instead, use an algorithm for generating a password or let them choose a password for themselves in the subscription process.

Don't forget to turn on IPN in your PayPal account and point it at your IPN processing script **[Hack #65]**.

Controlling Access to Your Valued Content

Now you have a list of valid subscribers that is automatically updated by PayPal and your IPN script. Next, you'll need to make use of this information by ensuring that visitors to your site are on the current subscriber list. In this example, all the members-only pages are dynamic ASP pages. The first thing the code does is check that the user is properly logged in. If not, the premium content is not displayed and the user is redirected to a Sign In page. You know the user is signed in if the *magic cookie* has been set.

```
<%
'content.asp
'Check for the magic cookie.
'If not found, redirect
if Response.Cookies("MagicMonkey) != "swordfish" then
 Response.Print("Please log in before accessing this page.")
 Response.Redirect("login.asp")
end
%>

<!-- Put your content here -->
```

The Sign In page simply asks for the user's email address and password. If this information shows the visitor is a valid subscriber, a cookie is set on the user's browser. The cookie contains the magic word that allows your subscribers access. Without this cookie, set to the proper magic word, no one can access subscriber-only content.

```
<%
'Sign in page: sign_in.asp
'Database connection code goes here
'Connect to database and create recordset
connStore = "DRIVER={Microsoft Access Driver (*.mdb)};
          DBQ="C:/InetPub/wwwroot/database/dbPayPal.mdb")
set rsCookies = Server.CreateObject("ADODB.Recordset")
rsCookies.ActiveConnection = connStore
rsCookies.Source = "SELECT * from subscribers WHERE email =
          '" & Request.Form("email") & "' AND password =
          '" & Request.Form("password") & "'"
rsCookies.Open( )

'IF the query turns up a match, execute this code:

'Set new cookie session in MagicMonkey
' "swordfish" happens to be today's magic cookie word
Response.Cookies("MagicMonkey") = "swordfish"

'Set cookie expiration
Response.Cookies("MagicMonkey").Expires = Now( ) + 1 'one day

Response.Print("Thank you for logging in. <a href="content.asp">Click
          here</a> to start selling stuff to a bunch of monkey lovers.")

'ELSE do this:

Response.Redirect("login.asp")
%>
```

Your page, *login.asp*, should contain an HTML form that asks for each customer's email address and password. Its data is posted to *sign_in.asp*.

Hacking the Hack

This example is purposefully simplistic. If the cookie is always the same, all a nonsubscriber needs to do to gain access is manually set the browser's cookies to include your magic word. In practice, you will want to change your magic cookie daily. Users will need to visit the Sign In screen each day and provide their email address and password to get that day's magic cookie. Better yet, use a one-way encryption algorithm to create a unique cookie each day for each subscriber.