POODLE SSL 3.0 Vulnerability

Merchant Response Guide

Overview

What is POODLE?

POODLE is an internet security vulnerability that impacts the Secure Sockets Layer (SSL) 3.0 protocol, which was designed to ensure secure connections when surfing on the Internet. When exploited, this vulnerability enables a cyber criminal to gain access to connections considered secure via this widespread (but 15-year-old) security protocol.

How is PayPal responding?

PayPal will completely disable SSL 3.0 support on **December 3, 2014**. Unfortunately, we realize shutting off SSL 3.0 may cause compatibility problems for a few of our customers resulting in the inability to pay with PayPal on some merchant sites or other processing issues that we are still identifying. To enable your assessment and potential remediation, we've put together this *Merchant Response Guide* to help ensure your integration is secure from this vulnerability.

We plan to keep our customers up to date on how we are addressing this issue via the appropriate channels, including PayPal Forward, our Twitter handle, Customer Service and for merchants, through our Merchant Services team. We appreciate your patience and understanding as we work around the clock to better serve you and keep you and our consumers safe.

What you need to do…

If you do not manage your website or the PayPal Integration, we strongly encourage you to work with your website service partner (developer, hosting company, e-commerce platform, etc.) and share this *Merchant Response Guide*, which provides the basic guidelines on how to update to TLS. If your website service partner has any questions or needs support, they can get in touch with our Merchant Technical Support team at www.paypal.com/mts.

1. Test your current integration against the PayPal Sandbox

If you are directly integrated with PayPal, follow the steps below:

NOTE: We are working with our Partners to resolve the SSL 3.0 issue. If you are integrated through a Partner or third-party system, we recommend you work with your Partner to ensure they are no longer using SSL 3.0. If you are using downloadable components or a non-hosted solution, you may need to upgrade or download the latest version.

a. Point your test environment to our Sandbox: https://developer.paypal.com/docs/classic/lifecycle/ug_sandbox/

   - SSL 3.0 has already been disabled on the PayPal Sandbox, so if you can successfully make an application programming interface (API) request you are not using SSL 3.0.

b. If your request fails, check your logs to see why.

   - If you see an error similar to those shown below, then you are using SSL 3.0 and will need to configure your secure connection to use Transport Layer Security (TLS).

```
* Unknown SSL protocol error in connection to api-3t.sandbox.paypal.com:-9824
```

OR

```
140062736746144:error:1408F10B:SSL routines:SSL3_GET_RECORD:wrong version
number:s3_pkt.c:337:
...
New, (NONE), Cipher is (NONE)
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
SSL-Session:
Protocol: SSLv3
...
```

2. Update to TLS

All PayPal customers are required to disable SSL 3.0 for client interactions and upgrade to TLS by December 3, 2014. The following table provides basic guidelines on how to update to TLS using common languages and connection methods. Your exact settings may vary…

NOTE: For detailed upgrade instructions for the SDKs and languages listed below, please reference https://ppmts.custhelp.com/app/answers/detail/a_id/1182.

<table>
<tr><th>Connection Method</th><th colspan="2">Action</th></tr>
<tr><td><b>PayPal SDK</b></td><td colspan="2">No current PayPal Software Development Kit (SDK) versions or languages use SSL 3.0. However, since the Java and PHP SDKs were recently updated to address this issue, all merchants using these SDKs or legacy SDKs (older than October 21, 2014) will need to update to the latest version. If you're unsure if you're using the latest SDK, test your integration against the Sandbox environment as explained in Step 1.
• For information on the latest SDK versions, see: http://paypal.github.io/sdk/#merchant</td></tr>
<tr><td rowspan="8"><b>API Endpoint</b></td><td colspan="2">Ensure you are connecting to PayPal endpoints using TLS. See the table below to set the TLS protocol for the language you are using. If your environment supports it, do not hardcode to a specific TLS version as the protocol will decide the highest possible version automatically.</td></tr>
<tr><th>Language</th><th>Action</th></tr>
<tr><td><b>Ruby</b></td><td>Set the TLS protocol in the OpenSSL::SSL::SSLContext.
• For more details, see: http://ruby-doc.org/stdlib-1.9.3/libdoc/openssl/rdoc/OpenSSL/SSL/SSLContext.html</td></tr>
<tr><td><b>Python</b></td><td>Set the TLS protocol in the ssl.SSLContext.
• For more details, see: https://docs.python.org/2/library/ssl.html#ssl.SSLContext</td></tr>
<tr><td><b>Node.js</b></td><td>Use the correct renegotiation limit as specified here:
• http://nodejs.org/api/tls.html#tls_client_initiated_renegotiation_attack_mitigation</td></tr>
<tr><td><b>PHP</b></td><td>Set CURLOPT_SSLVERSION to CURL_SSLVERSION_TLSv1 in your Curl options.
• For more details, see: http://curl.haxx.se/libcurl/c/CURLOPT_SSLVERSION.html</td></tr>
<tr><td><b>Java</b></td><td>Set the TLS protocol in the javax.net.ssl.SSLContext.
• For more details, see: http://docs.oracle.com/javase/7/docs/technotes/guides/security/jsse/JSSERefGuide.html</td></tr>
<tr><td><b>C#</b></td><td>Use SecurityProtocolType Tls.
• For more details, see: http://msdn.microsoft.com/en-us/library/system.net.securityprotocoltype%28v=vs.110%29.aspx</td></tr>
</table>

3. Issue new credentials (optional)

After you've successfully tested and upgraded to TLS, you may want to reissue and download new API credentials for any PayPal API requests. This step is strongly recommended, but not required. Please make a risk-based decision for your business and customers.

- If you are using **Certificate** authentication, no action is required because the vulnerability is in the SSL 3.0 protocol and not in the design of the SSL Certificates.
- If you are using **Signature** authentication, see: https://developer.paypal.com/docs/classic/api/apiCredentials/
- If you are using **OAuth** authentication, see: https://developer.paypal.com/docs/integration/admin/manage-apps/

Thank You

Thank you for your prompt attention to this issue and understanding of our approach. Though we recognize this necessary step may cause compatibility issues, we can't stress enough that this short-term inconvenience is heavily outweighed by our joint promise to our respective customers that we will keep their accounts and financial details safe.