

Remote Code Execution through MitM Attack on Kingsoft Office Application

2014-11-05

Software:	Kingsoft Office
Affected Versions:	5.3.1
CVE Reference:	CVE-2014-2271
Author:	Rob Miller, Nick Walker & Jon Butler - MWR Labs (http://labs.mwrinfosecurity.com/)
Severity:	High
Vendor:	Huawei
Vendor Response:	N/A

Description:

The Kingsoft Office application comes installed on Huawei P2 devices. An attacker that can modify the traffic coming from the application's Google Cloud Print service can gain remote code execution in the context of the application.

Impact:

An attacker can execute code remotely on the victim's device with the permissions of the Kingsoft Office application.

Cause:

The Kingsoft Office application includes functionality to print using Google Cloud Print services. The data sent to the service from the app initially attempts to use a secure channel. However, it was found that the secure channel ignores SSL errors and will fall back to a cleartext channel if a secure channel is not available. The Cloud Print function of the app uses a JavaScript bridge, which is vulnerable to JavaScript bridge remote code execution exploit. The combination of these issues means that an attacker with the ability to manipulate traffic from a user's device can gain remote code execution within the context of the application.

Interim Workaround:

Do not connect to untrusted networks.

Solution:

The Kingsoft Office application should be modified so that only secure connections are used.

Technical Details:

Moffice (cn.wps.moffice), contains functionality to use Google's Cloud Print service via a WebView:

```
cn.wps.moffice.common.beans.print.CloudPrintWebView
public class CloudPrintWebView extends WebView{
    public CloudPrintWebView(Context paramContext, AttributeSet paramAttributeSet,
int paramInt)
    {
        super(paramContext, paramAttributeSet, paramInt);
        setBackgroundColor(-2104085);
        setHorizontalScrollBarEnabled(false);
        setScrollBarStyle(0);
        WebSettings localWebSettings = getSettings();
        localWebSettings.setSupportZoom(true);
        localWebSettings.setJavaScriptEnabled(true);
        localWebSettings.setAllowFileAccess(true);
        localWebSettings.setBuiltInZoomControls(true);
        localWebSettings.setRenderPriority(WebSettings.RenderPriority.HIGH);
        if (Build.VERSION.SDK_INT >= 8)

localWebSettings.setLayoutAlgorithm(WebSettings.LayoutAlgorithm.NARROW_COLUMNS);
        setWebViewClient(new c(0));
        setWebChromeClient(new b(0));
    }
}
```

The WebView used a JavaScript bridge (PrintJSInterface). As this application was not build for API 17 or above, it was therefore vulnerable to a remote code execution exploit through a man-in-the-middle attack:

```
public void setJavaInterface(b paramb)
{
    addJavascriptInterface(paramb, "PrintJSInterface");
}
```

The connection initially tries to use SSL, which would prevent a man-in-the-middle attack. However, the error handling had been overwritten to ignore SSL errors, allowing invalid or self-signed certificates. Additionally, if any error was encountered when trying to connect to the HTTPS URL, the application would fall back to using a cleartext channel:

```
public final void xs()
{
    loadUrl("https://www.google.com/cloudprint/dialog.html");
}

private final class c extends WebViewClient
{
    public final void onReceivedSslError(WebView paramWebView, SslErrorHandler
```

```
paramSslErrorHandler, SslError paramSslError)
{
    paramSslErrorHandler.proceed();
}
public final void onReceivedError(WebView paramWebView, int paramInt, String
paramString1, String paramString2)
{
    if ((paramInt == -6) &&
(paramString2.equals("https://www.google.com/cloudprint/dialog.html")))
    {
        paramWebView.loadUrl("http://www.google.com/cloudprint/dialog.html");
        return;
    }
    super.onReceivedError(paramWebView, paramInt, paramString1, paramString2);
}
```

Detailed Timeline:

Date:	Summary:
2014-01-30	Vulnerability discovered
2014-03-05	Vendor contacted
2014-06-24	Vendor announces vulnerability is fixed
2014-08-28	Vendor notified of intent to disclose
2014-11-05	Vulnerability details released