2016 年 7 月 21 日，星期四

漏洞聚焦：OpenOffice Impress MetaActions 任意读写漏洞

此漏洞的发现者为思科 Talos 的 Richard Johnson 和 Yves Younan。

Talos 发布了有关 OpenOffice Impress 漏洞的公告。(TALOS-2016-0051/CVE-2016-1513)。Talos 在处理 MetaActions 时，发现 OpenOffice 中存在一个可被利用的越界漏洞。经过特别设计的 OpenDocument 演示文稿 .ODP 文件或演示文稿模板 .OTP 文件可造成越界读取/写入，从而导致拒绝服务（内存损坏和应用程序崩溃），并且可能执行导致任意代码。

概述

OpenOffice 是一款适用于文字处理、电子表格、演示文稿、图形、数据库和其他办公功能的开源办公软件套件。它可在多种操作系统中运行，并提供多种语言选择。它采用通用文件类型国际开放标准格式，也可对其他常见办公软件套件（例如 Microsoft Office）的文件进行读取和写入操作。它具有灵活性和开源特性，因而被广泛采用。

根据目前的报告，OpenOffice 已拥有 8400 多万用户和 1.25 亿次下载量，其中 87% 的用户运行 Microsoft Windows。攻击者可利用此漏洞诱使最终用户打开特别设计的恶意文件，从而触发此漏洞。攻击者可以通过钓鱼邮件附件向用户发送托管于 Web 服务器的文件并诱导用户打开文件，或者采用任何其他方法诱使用户打开恶意文件，从而实施漏洞攻击。

详细信息

在附件示例中，在执行 MetaPolyPolygonAction 并替换 PolyPolygon 对象中的 Polygon 时会发生此越界漏洞。在这种情况下，阵列中的位置是 512，而包含 Polygon 的阵列 (mpPolyAry) 的大小仅为 2。这将导致删除 main\tools\source\generic\poly2.cxx 文件第 228 行中越界读取的指针。接着会立即执行越界写入，从而写入通过在此位置创建新的 Polygon 获取的新指针。这可为攻击者提供多种利用此漏洞的方式：首先可以释放无效指针；而如果该方式失败，越界写入新的指针可提供第二次利用此漏洞的机会。main\tools\source\generic\poly2.cxx 的第 217-230 行如下：

```
void PolyPolygon::Replace( const Polygon& rPoly, saluInt16 nPos )
{
    DBGCHKTHIS( PolyPolygon, NULL );
    DBG_ASSERT( nPos < Count(), "PolyPolygon::Replace(): nPos >= nSize" );

if ( mpImplPolyPolygon->mnRefCount > 1 )
{
    mpImplPolyPolygon->mnRefCount--;
    mpImplPolyPolygon = new ImplPolyPolygon( *mpImplPolyPolygon );
}

delete mpImplPolyPolygon->mpPolyAry[nPos];
mpImplPolyPolygon->mpPolyAry[nPos] = new Polygon( rPoly );

}
```

尽管会在第 220 行中检查确认 npos 小于阵列大小，但这种断言仅可在调试模式下启用。

在示例文件中，通过 MetaPolyPolygonAction::Read 函数在 main\vcl\source\gdi\metaact.cxx 文件第 1189 行读取的值如下：

```
rIStm >> nNumberOfComplexPolygons;
for ( i = 0; i < nNumberOfComplexPolygons; i++ )
{
    rIStm >> nIndex;
    Polygon aPoly;
    aPoly.Read( rIStm );
    maPolyPoly.Replace( aPoly, nIndex );
}
```

以下是发生此问题时的调用栈：

```
00afe04c 68c2109f tl!Polygon::~Polygon+0x48 [d:\aoo\main\tools\source\generic\poly.cxx @ 667]
00afe058 68c2cb8b tl!Polygon::`scalar deleting destructor'+0xf
00afe0b0 67b3be7e tl!PolyPolygon::Replace+0x10b [d:\aoo\main\tools\source\generic\poly2.cxx @ 228]
00afe0f4 67b374ac vcl!MetaPolyPolygonAction::Read+0xce [d:\aoo\main\vcl\source\gdi\metaact.cxx @ 1193]
00afe3c0 67aee49d vcl!MetaAction::ReadMetaAction+0x144c [d:\aoo\main\vcl\source\gdi\metaact.cxx @ 247]
00afe43c 67b1944d vcl!operator>>+0x19d [d:\aoo\main\vcl\source\gdi\gdimtf.cxx @ 2918]
00afe804 67afc9fb vcl!operator>>+0x4ad [d:\aoo\main\vcl\source\gdi\impgraph.cxx @ 1826]
00afe814 66e97234 vcl!operator>>+0x1b [d:\aoo\main\vcl\source\gdi\graph.cxx @ 818]
00afebcc 665dde56 svt!GraphicFilter::ImportGraphic+0x9b4 [d:\aoo\main\svtools\source\filter\filter.cxx @ 1637]
00afecb4 665dd95f svxcore!SdrGrafObj::ImpSwapHdl+0x4e6 [d:\aoo\main\svx\source\svdraw\svdograf.cxx @ 1557]
00afecc0 68bceb64 svxcore!SdrGrafObj::LinkStubImpSwapHdl+0xf [d:\aoo\main\svx\source\svdraw\svdograf.cxx @ 1481]
00afecd8 66ef08f8 tl!Link::Call+0x24 [d:\aoo\main\solver\411\wntmsci12\inc\tools\link.hxx @ 135]
00afecec 66eef8aa svt!GraphicObject::GetSwapStream+0x28 [d:\aoo\main\svtools\source\graphic\grfmgr.cxx @ 480]
00afed44 66ef105f svt!GraphicObject::ImplAutoSwapIn+0xca [d:\aoo\main\svtools\source\graphic\grfmgr.cxx @ 264]
00afed50 665da3fa svt!GraphicObject::FireSwapInRequest+0xf [d:\aoo\main\svtools\source\graphic\grfmgr.cxx @ 598]
00afed80 664b6b70 svxcore!SdrGrafObj::ForceSwapIn+0x10a [d:\aoo\main\svx\source\svdraw\svdograf.cxx @ 706]
00afed94 664b67e2 svxcore!sdr::contact::ViewObjectContactOfGraphic::doAsynchGraphicLoading+0x50 [
d:\aoo\main\svx\source\sdr\contact\viewobjectcontactofgraphic.cxx @ 218]
00afeda0 664c0449 svxcore!sdr::event::AsynchGraphicLoadingEvent::ExecuteEvent+0x12 [
d:\aoo\main\svx\source\sdr\contact\viewobjectcontactofgraphic.cxx @ 72]
00afedbc 664c0688 svxcore!sdr::event::EventHandler::ExecuteEvents+0x29 [
d:\aoo\main\svx\source\sdr\event\eventhandler.cxx @ 114]
00afedc8 679bc1f1 svxcore!sdr::event::TimerEventHandler::Timeout+0x18 [
d:\aoo\main\svx\source\sdr\event\eventhandler.cxx @ 147]
00afedf4 6790c1a8 vcl!Timer::ImplTimerCallbackProc+0xd1 [d:\aoo\main\vcl\source\app\timer.cxx @ 142]
00afee00 6790c0a9 vcl!SalTimer::CallCallback+0x18 [d:\aoo\main\vcl\inc\saltimer.hxx @ 62]
00afee48 67905335 vcl!SalTimerProc+0xe9 [d:\aoo\main\vcl\win\source\app\saltimer.cxx @ 129]
00afee84 67905621 vcl!SalComWndProc+0x275 [d:\aoo\main\vcl\win\source\app\salinst.cxx @ 837]
00afeed4 75ddc4e7 vcl!SalComWndProcW+0x61 [d:\aoo\main\vcl\win\source\app\salinst.cxx @ 885]
00afef00 75ddc5e7 USER32!InternalCallWinProc+0x23
00afef78 75ddcc19 USER32!UserCallWinProcCheckWow+0x14b
00afefd8 75ddcc70 USER32!DispatchMessageWorker+0x35e
00afefe8 678ec7ed USER32!DispatchMessageW+0xf
00afeff4 67904f35 vcl!ImplDispatchMessage+0xd [d:\aoo\main\vcl\win\source\app\saldata.cxx @ 163]
00aff008 67904e4d vcl!ImplSalDispatchMessage+0x35 [d:\aoo\main\vcl\win\source\app\salinst.cxx @ 663]
00aff038 67905050 vcl!ImplSalYield+0x5d [d:\aoo\main\vcl\win\source\app\salinst.cxx @ 683]
00aff060 679ab4ce vcl!WinSalInstance::Yield+0xe0 [d:\aoo\main\vcl\win\source\app\salinst.cxx @ 745]
00aff078 679ab59f vcl!ImplYield+0x8e [d:\aoo\main\vcl\source\app\svapp.cxx @ 477]
00aff088 679ab3f1 vcl!Application::Yield+0xf [d:\aoo\main\vcl\source\app\svapp.cxx @ 510]
00aff098 69b9bade vcl!Application::Execute+0x31 [d:\aoo\main\vcl\source\app\svapp.cxx @ 453]
00aff734 679b9866 sofficeapp!desktop::Desktop::Main+0x2c8e [d:\aoo\main\desktop\source\app\app.cxx @ 2234]
00aff768 679b9a13 vcl!ImplSVMain+0xa6 [d:\aoo\main\vcl\source\app\svmain.cxx @ 197]
00aff774 69be162a vcl!SVMain+0x23 [d:\aoo\main\vcl\source\app\svmain.cxx @ 238]
00aff7dc 01361098 sofficeapp!sofficemain+0xea [d:\aoo\main\desktop\source\app\sofficemain.cxx @ 47]
00aff7e4 01361039 soffice!salmain+0x8 [d:\aoo\main\desktop\source\app\main.c @ 32]
00aff7f0 01361078 soffice!main+0x19 [d:\aoo\main\desktop\source\app\main.c @ 30]
00aff808 0136125c soffice!WinMain+0x28 [d:\aoo\main\desktop\source\app\main.c @ 30]
00aff898 7622ee1c soffice!tmainCRTStartup+0x140 [f:\dd\vctools\crtbld\selfx86\crt\src\crtexe.c @ 578]
00aff8a4 775437eb kernel32!BaseThreadInitThunk+0xe
00aff8e4 775437be ntdll!RtlUserThreadStart+0x70
00aff8fc 00000000 ntdll!_RtlUserThreadStart+0x1b
```

结论

发现并且负责任地披露零日漏洞可帮助提高人们日常所使用的软件的整体安全性。为此，Talos 致力于通过开发编程方法来识别漏洞，防止其被恶意攻击者所利用。这有助于保护客户所使用的平台以及软件的安全，并提供可以帮助思科改进其流程的洞察力，以开发更优质和更安全的产品。

此外，为了保护我们的客户，Talos 已经发布了相关规则来检测利用此漏洞的攻击尝试。请注意，Talos 未来可能会发布更多规则，当前规则会根据未来得到的更多漏洞信息而有所变更。如需获取有关最新规则的信息，请参阅防御中心、FireSIGHT 管理中心或 Snort.org。

Snort 规则：35828-35829。

如需获取更多有关零日攻击或漏洞的报告和信息，请访问：
http://www.talosintelligence.com/vulnerability-reports/

发布者：William Largent；发布时间：15:41

标签：零日、Impress、OpenOffice、Talos、漏洞、漏洞聚焦