**数字版权管理**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/digital-rights-management.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/digital-rights-management.md" \o "1个贡献者)

数字版权管理（DRM）为内容提供商提供了保护其专有音乐或其他数据免遭未经授权的复制和其他非法使用的手段。DRM技术通过对数字内容进行加密并附加使用规则来保护数字内容，该使用规则确定了用户可以播放内容的条件。使用规则通常会阻止复制或限制内容播放的次数。操作系统与驱动程序一起执行这些规则。

DRM旨在对用户透明，除非用户试图违反购买数字内容时同意的使用规则。

受DRM保护的任何数字音频内容只能由受信任的音频驱动程序播放。这些驱动程序已通过Microsoft的硬件兼容性测试，以验证它们是否兼容DRM，并且没有漏洞可以规避DRM安全措施。

此外，将调试器连接到驱动程序后，无法播放受保护的内容。

对于Microsoft Windows Me驱动程序，WHQL（Microsoft Windows硬件质量实验室）的DRM符合性测试是可选的。但是，对于Windows XP和更高版本中的驱动程序，必须符合DRM。有关详细信息，请参阅以下列表中的“ **DRM要求”**主题。

本节介绍以下主题：

[DRM概述](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/drm-overview)

[内容ID和内容权利](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/content-ids-and-content-rights)

[转发DRM内容ID](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/forwarding-drm-content-ids)

[DRM要求](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/drm-requirements)

[开发和调试DRM驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/developing-and-debugging-drm-drivers)

[DRM功能和接口](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/drm-functions-and-interfaces)

**DRM概述**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/drm-overview.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/drm-overview.md" \o "1个贡献者)

用于数字音频的DRM在Microsoft Windows 2000和更高版本以及Windows Me / 98上实现。但是，只有Microsoft Windows XP和更高版本以及Windows Me在内核中实现DRM安全。当前，Windows不为MIDI流或DLS集提供DRM安全性。

受DRM保护的数字内容以加密形式存储在磁盘或其他存储介质类型上。加密算法对内容进行加密，以使其变得难以理解，直到对其进行解密为止。在播放过程中，从磁盘读取内容并将其缓冲在内存中时，内容仍保持混乱状态。在数据路径的末端附近，[DRMK系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components#drmk_system_driver)（Drmk.sys）对数据进行解密，并将其直接馈送到要播放的音频驱动程序。通过限制传输未加密内容的数据路径的范围，DRMK使得内容更不容易受到未经授权的复制的影响。

在Windows 2000和Windows 98中，安全漏洞允许用户轻松加载流氓驱动程序，该流氓驱动程序将安全内容的播放以未加密的形式路由到磁盘。Windows XP和更高版本以及Windows Me通过仅允许受信任的音频驱动程序播放受DRM保护的内容来消除此漏洞。

在Windows XP和更高版本以及Windows Me中，安全内容在遍历音频数据路径的过程中一直保持混乱状态，直到它进入内核的受保护环境为止。在内核中，受保护的组件对数据进行解密，并将解密后的数据馈送到受信任的驱动程序以进行回放。当配置过滤器图表以播放未加密的音频流时，DRMK会为放置在图表中的每个KS过滤器验证适配器驱动程序。系统通知驱动程序受保护内容的使用规则。反过来，驱动程序会将内容路由到的任何下游过滤器通知DRMK，并且系统也对这些过滤器进行身份验证。该过程一直持续到图形完成为止。如果数字播放流通过任何不兼容DRM的组件，则系统将拒绝整个图形。

兼容DRM的驱动程序必须在播放数字内容时防止未经授权的复制。另外，驱动程序必须禁用所有可以通过标准接口（例如S / PDIF）传输内容的数字输出，通过该接口可以捕获解密的内容。请注意，此要求不适用于USB设备。当前，DRMK仅通过没有数字输出的USB扬声器设备播放安全内容。

**内容ID和内容权利**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/content-ids-and-content-rights.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/content-ids-and-content-rights.md" \o "1个贡献者)

内容ID（标识符）是[DRMK系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components" \l "drmk_system_driver)在运行时生成的ULONG值，用于识别馈入特定引脚的音频数据流中受DRM保护的内容。

内容权利是内容提供商授予用户播放和复制受DRM保护的内容的权利的数字表示。内容权限以DRMK传递给音频驱动程序的**[DRMRIGHTS](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/ns-drmk-tagdrmrights)**结构的形式指定。

DRMRIGHTS包含两个标志：**DigitalOutputDisable**和**CopyProtect**。如果设置了**DigitalOutputDisable**标志，则驱动程序必须禁用任何连接到外部设备的数字输出（例如，通过S / PDIF连接器）。如果设置了**CopyProtect**标志，则驱动程序必须禁用可能允许将安全内容的持久副本保存到磁盘或任何其他形式的非易失性存储的功能。例如，典型的音频硬件允许将回放信号路由到捕获通道。如果此信号为数字形式，则捕获的信号可能是输入信号的完美数字副本。如果播放混音包含来自具有**CopyProtect的**任何流的数据 如果设置了此标志，则驱动程序必须使回放捕获路径静音。

兼容DRM的音频驱动程序必须在其WaveCyclic和WavePci微型端口驱动程序对象上支持[IDrmAudioStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nn-drmk-idrmaudiostream)接口，该对象公开用于渲染音频数据的接收器引脚。为了从驱动程序获得对**IDrmAudioStream**对象的引用，DRMK 在引脚上调用**QueryInterface**方法。该引脚具有[IMiniportWaveCyclicStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-iminiportwavecyclicstream)或[IMiniportWavePciStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-iminiportwavepcistream)类型的接口。所述**IDrmAudioStream**接口仅支持一个方法，**[IDrmAudioStream :: SetContentId](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-idrmaudiostream-setcontentid)**（除了三个**IUnknown的**方法）。当DRMK调用**SetContentId时**，它传入一个内容ID和内容权限，驱动程序将其与引脚的数据流相关联。

WaveCyclic或WavePci微型端口驱动程序可以通过[IDrmPort2](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport2)接口（**IDrmPort2**派生自基类[IDrmPort](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport)）来访问DRM功能，而不是直接在Drmk.sys中调用DRM函数。在Microsoft Windows XP和更高版本中，WaveCyclic和WavePci端口驱动程序支持**IDrmPort2**。微型端口驱动程序通过使用**REFIID IID\_IDrmPort2**调用端口对象的**QueryInterface**方法来获取对端口驱动程序**IDrmPort2**接口的引用。

某些音频驱动程序支持硬件混合，并且可以同时处理多个输入数据流。这种类型的驱动程序必须同时跟踪各个流的内容ID和所有流的复合内容权限。驱动程序调用**[IDrmPort :: CreateContentMixed](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-createcontentmixed)**来确定混合流的复合权限，并创建内容ID来标识该流。驱动程序完成使用内容ID时，必须调用**[IDrmPort :: DestroyContent](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-destroycontent)**删除内容ID。

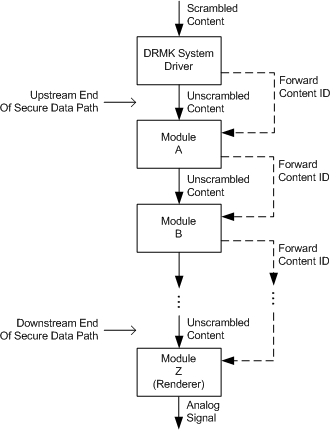
每次将输入流添加到混音器中或从混音器中删除时，驱动程序必须删除旧混音的内容ID，并为新混音创建新的内容ID。在删除旧的内容ID之前，驱动程序必须首先将新的内容ID成功转发到它先前已将旧的内容ID转发到的所有流。有关更多信息，请参阅[转发DRM内容ID](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/forwarding-drm-content-ids)。

**转发DRM内容ID**

* 2017/04/20
* 5分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/forwarding-drm-content-ids.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/forwarding-drm-content-ids.md" \o "1个贡献者)

所述[DRMK系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components" \l "drmk_system_driver)解读包含受保护内容的音频回放流。DRMK实现了一个KS过滤器，该过滤器获取包含加扰数据的输入流，对其进行加扰，然后将未加扰的流馈送到由一定数量的内核驻留模块组成的数据路径中。这些模块可以是KS过滤器或其他类型的驱动程序。数据路径通常在音频渲染设备中结束，该设备将数字内容转换为可以通过扬声器播放的模拟信号。

在允许未加密的内容进入数据路径之前，DRMK会验证数据路径是否安全。为此，DRMK对数据路径中的每个模块进行身份验证，从位于数据路径上游端的模块开始，向下游移动至数据路径的另一端。下图说明了此过程。

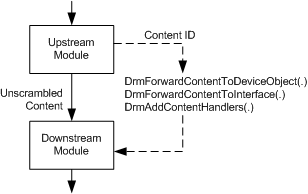


在上图中，实线箭头表示数据路径，而虚线箭头表示验证数据路径安全所需的通信。仅在DRMK已完成对该路径中所有模块的身份验证之后，未加密的数据才进入该路径。

在DRMK对每个模块进行身份验证之后，该模块将为DRMK提供有关数据路径中下一个模块的信息，以便对其进行身份验证。在对每个模块进行身份验证时，它会接收标识该流的DRM内容ID。

从安全数据路径的上游端开始，DRMK将内容ID转发到模块A，然后模块A将内容ID转发到模块B。此过程一直进行到内容ID转发到模块Z（安全模块中的最后一个模块）为止。数据路径。

下图显示了数据路径中的一对相邻模块。



上游侧的模块调用以下[DRM功能](https://docs.microsoft.com/windows-hardware/drivers/audio/drm-functions)之一，以向DRMK提供有关下游模块的信息，并将内容ID转发到该模块：

[**DrmForwardContentToDeviceObject**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmforwardcontenttodeviceobject)

[**DrmForwardContentToInterface**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmforwardcontenttointerface)

[**DrmAddContentHandlers**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmaddcontenthandlers)

这些“转发”功能中的每一个均向DRMK提供标识受保护流的DRM内容ID，以及DRMK认证下游模块所需的信息。调用这三个函数中的哪一个取决于两个相邻模块在管理受保护内容的传输时用来相互通信的接口类型：

1. 如果上游模块调用**[IoCallDriver](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/wdm/nf-wdm-iocalldriver)**与下游模块进行通信，则下游模块是WDM驱动程序的一部分。在这种情况下，上游模块调用**DrmForwardContentToDeviceObject**来向DRMK提供代表下游模块的设备对象。DRMK使用设备对象来认证下游模块。
2. 如果两个模块通过下游模块实现的COM接口进行通信，则上游模块将调用**DrmForwardContentToInterface**。该调用为DRMK提供了指向下游模块的COM接口的指针。尽管两个模块本身必须就这些方法的作用达成共识，但DRMK 在此接口中仅调用**IUnknown**方法，并且不对其他方法做任何假设。DRMK验证接口中每个方法的入口点都属于经过身份验证的模块。如果入口点分布在多个模块之间，则DRMK将对所有这些模块进行身份验证。
3. 如果两个模块都不使用COM接口也不使用**IoCallDriver**函数进行通信，则上游模块将调用**DrmAddContentHandlers**来为DRMK提供在下游模块中实现的“内容处理程序”的入口点列表。DRMK不会调用内容处理程序，也不会对它们执行的功能做任何假设。但是，DRMK确实会对入口点所在的一个或多个模块进行身份验证。

认证通过后，下游模块需要以下信息：

* DRM内容ID，用于标识包含受保护内容的流。模块需要此ID才能将计划向其发送受保护的内容的任何模块（更下游）通知DRMK。
* 与受保护内容关联的DRM内容权限。该模块需要内容权限，才能实施适当的安全级别。

三种转发功能均以略有不同的方式向模块提供此信息：

1. 所述**DrmForwardContentToDeviceObject**功能发送**[KSPROPERTY\_DRMAUDIOSTREAM\_CONTENTID](https://docs.microsoft.com/previous-versions/ff537351(v=vs.85))**设置属性请求向下游模块的设备对象。该请求将流的内容ID和内容权限转发到下游模块。
2. 该**DrmForwardContentToInterface**函数查询下游模块的COM接口[IDrmAudioStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nn-drmk-idrmaudiostream)接口。如果查询成功，该函数将调用**[IDrmAudioStream :: SetContentId](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-idrmaudiostream-setcontentid)**方法将内容ID和内容权限转发给下游模块。
3. 对于**DrmAddContentHandlers**函数，调用者（上游模块）负责将流的内容ID和内容权限转发到下游模块。一旦**DrmAddContentHandlers**返回并带有指示下游模块已通过身份验证的成功代码，上游模块将通过调用其内容处理程序之一将内容ID和内容权限传递给下游模块。

如果上游模块是WaveCyclic或WavePci微型端口驱动程序，则它可以通过以下方法之一间接调用适当的DRM函数：

[**IDrmPort2 :: ForwardContentToDeviceObject**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport2-forwardcontenttodeviceobject)

[**IDrmPort :: ForwardContentToInterface**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-forwardcontenttointerface)

[**IDrmPort2 :: AddContentHandlers**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport2-addcontenthandlers)

有关更多信息，请参见[DRM功能](https://docs.microsoft.com/windows-hardware/drivers/audio/drm-functions)。

为简单起见，前面的讨论假定数据路径中的每个模块都接受来自单个源的流，并将该流转发到最多一个下游模块。实际上，模块可以将流转发到两个或多个下游模块，但是它必须首先通过调用三个转发功能之一来对每个下游模块进行身份验证。类似地，模块可以将几个输入流混合在一起，但是它必须通过为混合输出流提供适当级别的保护来尊重输入流的内容权限。有关更多信息，请参见[Content ID和Content Rights](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/content-ids-and-content-rights)中有关**[DrmCreateContentMixed](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmcreatecontentmixed)**函数的讨论。

典型的安全数据路径包括[KMixer系统驱动程序，](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components" \l "kmixer_system_driver)后跟代表音频渲染设备的波形滤波器。该过滤器可与相应的端口驱动器结合起来实现为WaveCyclic或WavePci微型端口驱动器。为了验证数据路径是否安全，DRMK将内容ID转发到KMixer，后者又将内容ID转发到过滤器。实现通用筛选器功能的端口驱动程序接收内容ID，并将其转发给微型端口驱动程序。具体来说，端口驱动程序调用**DrmForwardContentToInterface**用于将内容ID转发到微型端口驱动程序已实例化以表示音频呈现设备上的wave输出引脚的流对象的功能。此调用的参数值之一是指向流对象的[IMiniportWaveCyclicStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-iminiportwavecyclicstream)或[IMiniportWavePciStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-iminiportwavepcistream)接口的指针。通过此接口，该函数在流对象中查询其**IDrmAudioStream**接口，并调用该接口的**SetContentId**方法。

有关更多信息，请参见Microsoft Windows驱动程序工具包（WDK）中sb16和msvad示例驱动程序中**SetContentId**方法的实现。

**DRM要求**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/drm-requirements.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/drm-requirements.md" \o "1个贡献者)

本节介绍了音频微型端口驱动程序必须满足的条件，才能通过Microsoft Windows硬件质量实验室（WHQL）的DRM兼容性测试。这些要求专门适用于WaveCyclic和WavePci [音频微型端口驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/audio-miniport-drivers)，它们是端口类库（Portcls.sys）中与WavePci和WaveCyclic端口驱动程序相对应的特定硬件。DRM兼容性测试当前不适用于USB驱动程序。

在Windows Me和Windows XP及更高版本中，只有受信任的音频驱动程序才能播放受DRM保护的内容。Windows通过存储在驱动程序的.cat（目录）文件中的DRM特定的数字签名来识别受信任的驱动程序。Microsoft仅为通过DRM兼容性测试的驱动程序发布DRM签名，该驱动程序是WHQL管理的硬件兼容性测试的一部分。

对于Windows Me驱动程序，DRM兼容性测试是可选的，并且仅在硬件供应商的要求下执行。DRM签名与Windows徽标签名是分开的，并且是Windows徽标签名的补充。请注意，通过Windows徽标测试但未通过DRM兼容性测试的驱动程序仍然可以播放不受DRM安全保护的内容。

但是，对于Windows XP和更高版本，DRM兼容性测试是WHQL测试的必需部分。驱动程序必须通过DRM兼容性测试，才能获得“ Designed for Windows XP”徽标。

DRM兼容性测试需要受信任的音频驱动程序才能执行以下操作：

* 音频微型端口驱动程序必须在其流对象中实现[IDrmAudioStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nn-drmk-idrmaudiostream)接口，如果查询IID\_IDrmAudioStream，则该流对象必须返回IDrmAudioStream类型的对象。
* 当请求复制保护（**[DRMRIGHTS](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/ns-drmk-tagdrmrights)**。**副本保护** = **TRUE**），音频驱动器必须禁用捕捉当前正在播放的流的能力。这意味着驱动程序不得将不受保护的数字内容保存到任何形式的非易失性存储器中，包括硬盘，EEPROM，存储卡和存储棒。此外，驱动程序必须禁用输出D / A转换器上的捕获多路复用器，否则将阻止数字内容的环回。
* 当音频驱动器被要求禁用设备（DRMRIGHTS。上的数字音频输出**DigitalOutputDisable** = **TRUE**），则必须禁止所有数字音频输出，其能够通过一个标准的互连方案发送通过标准接口的内容。数字输出包括但不限于S / PDIF，IEEE 1394，并行，串行，调制解调器和网络端口。（此要求当前不适用于USB。）
* 处理安全内容时，音频驱动程序绝不能将不受信任的驱动程序附加到其堆栈中。换句话说，音频驱动程序必须仅依赖于也包含DRM签名的其他组件。驱动程序绝不能促进将音频数据传输到没有DRM签名的任何组件。特别是，如果驱动程序将数字内容传递给另一个组件，则该驱动程序必须使用内核中的DRM API来将这一事实通知[DRMK系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components" \l "drmk_system_driver)。

除了通过DRM兼容性测试外，音频设备和驱动程序还不允许用户选择破坏或破坏内核中DRM组件的操作模式。特别是，驱动程序不得提供注册表设置，用户控制面板或其他禁用DRM功能的方法。

**开发和调试DRM驱动程序**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/developing-and-debugging-drm-drivers.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/developing-and-debugging-drm-drivers.md" \o "1个贡献者)

以下清单可以帮助驱动程序编写者避免一些常见的陷阱：

* 如果驱动程序在播放受DRM保护的内容时禁用了波形捕获和S / PDIF输出，则驱动程序应记住在播放受DRM保护的内容（并且DRM缓冲区被破坏）之后再次启用它们。
* 如果设备执行硬件混合，则驱动程序应跟踪在将流添加到混合中或从混合中删除流时发生的复合使用权更改。例如，任何时候只要混音包含一个或多个受版权保护的DRM流，捕获都应该被静音。如果在播放受保护的混音时打开捕获功能，则应保持静音状态。
* 更改与流关联的过滤器图形或属性设置后，驱动程序可能需要立即更新流的复制保护和输出启用设置。驱动程序应同步其操作，以防止将受保护的内容复制到捕获缓冲区或数字输出。例如，当捕获多路复用器的输入流发生变化时，驱动程序不应在打开和关闭静音所需的时间内允许安全内容变得脆弱。

该[DRMK系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components" \l "drmk_system_driver)防止内核调试器，而DRM保护的内容播放连接。防调试装甲是DRMK用于使受保护内容不透明的几种措施之一。但是，一旦驱动程序准备好进行测试，就可以使用以下技术调试其DRM兼容功能：

* 临时修改波流的**SETSTATE**方法（例如，参照**[IMiniportWavePciStream :: SETSTATE](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-iminiportwavepcistream-setstate)**）来调用**[IDrmAudioStream :: SetContentId](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-idrmaudiostream-setcontentid)**和设置**[DRMRIGHTS](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/ns-drmk-tagdrmrights)**参数的**副本保护**构件**TRUE**。
* 完成调试后，请记住删除**SetContentId**调用。

使用此技术，您可以像播放受DRM保护的内容一样播放不受保护的内容，但可以避免禁用调试器。

例如，您可以使用调试器来验证驱动程序是否阻止了内容的记录。尝试通过更改SndVol32程序的音量和静音设置，诱使驱动程序启用通过捕获MUX记录波形输出流。滑块应该反映您对其设置所做的更改，这些更改是永久的，但是捕获MUX应该继续使波形流静音，直到“受保护”的内容播放完毕。只有这样，新设置才能生效。

**DRM功能和接口**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/drm-functions-and-interfaces.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/drm-functions-and-interfaces.md" \o "1个贡献者)

系统驱动程序组件*Drmk.sys*和*Portcls.sys*实现了DRM功能和接口的集合，驱动程序用于管理内核流音频内容的数字版权。所述*Drmk.sys*部件实现了许多**DrmXxx**功能，并且*Portcls.sys*实现特定DRM集的**PcXxx**功能，并且还对[IDrmPort](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport)和[IDrmPort2](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport2)接口。

以下DRM功能可用：

[**DrmAddContentHandlers**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmaddcontenthandlers)

为系统提供一个驱动程序接口，该接口包含用于处理受保护内容的功能列表。 **[DrmCreateContentMixed](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmcreatecontentmixed)**

创建DRM内容ID，以标识KS音频流，其中包含来自多个输入流的混合内容。 **[DrmDestroyContent](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmdestroycontent)**

删除DRM内容ID。 **[DrmForwardContentToDeviceObject](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmforwardcontenttodeviceobject)**

对驱动程序进行身份验证，并将系统已分配给包含受保护内容的流的DRM内容ID和内容权限发送给驱动程序。 **[DrmForwardContentToFileObject](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmforwardcontenttofileobject)**

功能过时。 **[DrmForwardContentToInterface](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmforwardcontenttointerface)**

对驱动程序对象进行身份验证，并将系统已分配给包含受保护内容的流的DRM内容ID和内容权限发送给该驱动程序对象。 **[DrmGetContentRights](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-drmgetcontentrights)**

检索系统已分配给DRM内容ID的DRM内容权限。此列表中的函数在头文件Drmk.h中声明。内核模式[DRMK系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components" \l "drmk_system_driver) Drmk.sys导出这些功能的入口点。

在Windows XP和更高版本中，PortCls系统驱动程序Portcls.sys为同一组DRM功能导出一组不同的入口点。PortCls函数的名称与上一个列表相似，不同之处在于它们使用前缀Pc代替Drm：

[**PcAddContentHandlers**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pcaddcontenthandlers)

[**PcCreateContentMixed**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pccreatecontentmixed)

[**PcDestroyContent**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pcdestroycontent)

[**PcForwardContentToDeviceObject**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pcforwardcontenttodeviceobject)

[**PcForwardContentToFileObject**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pcforwardcontenttofileobject)

[**PcForwardContentToInterface**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pcforwardcontenttointerface)

[**PcGetContentRights**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-pcgetcontentrights)

这些函数名称在头文件Portcls.h中声明。Portcls.sys中的入口点只需要调用Drmk.sys中的相应函数即可。PortCls入口点只是为了方便起见而提供，因此已经连接到Portcls.sys的音频驱动程序不需要显式加载Drmk.sys。

在Windows XP和更高版本中，相同的功能集也作为[IDrmPort](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport)和[IDrmPort2](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport2)接口中的方法[公开](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nn-portcls-idrmport2)：

[**IDrmPort2 :: AddContentHandlers**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport2-addcontenthandlers)

[**IDrmPort :: CreateContentMixed**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-createcontentmixed)

[**IDrmPort :: DestroyContent**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-destroycontent)

[**IDrmPort2 :: ForwardContentToDeviceObject**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport2-forwardcontenttodeviceobject)

[**IDrmPort :: ForwardContentToFileObject**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-forwardcontenttofileobject)

[**IDrmPort :: ForwardContentToInterface**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-forwardcontenttointerface)

[**IDrmPort :: GetContentRights**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-idrmport-getcontentrights)

该**IDrmPort**和**IDrmPort2**接口在头文件Portcls.h和宣布，以Portcls.sys实现。这些方法无非就是调用Drmk.sys中的相应函数。微型端口驱动程序通过查询该端口的端口驱动程序来获得对**IDrmPort***x*接口的引用。使用**IDrmPort***x*接口代替相应的Drm *Xxx*或Pc *Xxx*的优点功能是驱动程序可以使用此查询在运行时确定操作系统版本是否支持DRM。这简化了编写单个驱动程序的任务，该驱动程序可以在支持DRM的Windows新版本和不支持DRM的旧版本中运行。**IDrmPort2**派生自**IDrmPort，**并提供了两种附加方法。

如果相应的微型端口驱动程序支持，则WaveCyclic和WavePci端口驱动程序将使用[IDrmAudioStream](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nn-drmk-idrmaudiostream)接口。端口驱动程序调用**[IDrmAudioStream :: SetContentId](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/nf-drmk-idrmaudiostream-setcontentid)**方法，以将DRM保护分配给音频流中的数字内容。

在头文件Drmk.h中定义的[**DEFINE\_DRMRIGHTS\_DEFAULT**](https://docs.microsoft.com/previous-versions/ff536254(v=vs.85))宏将[**DRMRIGHTS**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/drmk/ns-drmk-tagdrmrights)结构的成员初始化为其默认值。