作为普通的网民来说，一般不需要知道也不用关心什么是盗链，不过如果你是网站的开发者或维护者，就不得不重视盗链的问题了。如果你刚刚开发完一个没有防盗链的带有文件下载功能的网站，挂上internet，然后上传几个时下非常热门的软件或电影并在网站内公布下载地址，让MSN上的所有好友都来体验一下你的杰作。

不用多久就会发现网速出奇地变慢，甚至服务器托管中心的服务员会热情地打电话告诉你的网站流量很大，估计是网站受欢迎起来了，问你是不是该考虑加钱租用带宽更宽但价格更贵的网线了。在这个值得庆祝的时候赶快打开Google Analytics看看有多少人来光顾你的网站了吧，如果发现访客每天才十来个人，很遗憾地告诉你：你的网站资源不幸地被人盗链了。

而且更糟糕的是，当你把网站上的文件和电影通通删光之后，网站仍然没有变快多少，从web服务器的访问日志里会发现疯狂的访问请求正从四面八方涌过来，web服务器为了迎接这批访客而没有时间处理正常的页面，这种状况可能会一直持续好几个周时间。

网站资源被盗链简单来说就是别人不是从你的网站通过下载资源，被盗链的几种可能情况：

1、人气非常旺的网站、论坛、社区的网页里直接引用了（使用标记）你网站上的图片，或者直接在其他网页（使用flash或媒体播放插件）里嵌入了你网站上的mp3。

2、在人气非常旺的网站、论坛、社区里提供了你的资源的下载地址。

3、你网站的资源可能被一些下载软件列入了“资源候选名单”，当其他人用下载工具下载相同的文件时，下载软件会自动找上门并且从你的服务器下载。

既然被盗链的后果这么可怕，那有哪些方法可以防止盗链呢下面从简到繁总结一下常见的以及自己实践过的一些方法，并简单分析一下。不过很遗憾地，这些方法都没法完全杜绝被盗链，并且防盗链的目的应该是从一定的程度上减少被盗链所产生的影响，同时能让合法的用户能够以自然的方式、顺畅地从你的网站下载资源。

有几种方法防止盗链：

方法1：判断引用地址

这个方法是最早及最常见的方法。所谓判断引用地址，就是判断浏览器请求时HTTP头的Referer字段的值，这个值在asp[.NET](http://lib.csdn.net/base/dotnet)里面可以用 Request.UrlReferrer属性取得。几个例子来说，在正常情况下当用户在浏览 http://uushare.com/abc.html 时点击一个链接去到 http://uushare.com/jacky.mp3 文件时，浏览器在发出请求jacky.mp3 资源时还会附带当刻浏览器所处的页面地址（即http://uushare.com/abc.html），所以当你的网站程序接收到下载 jacky.mp3 资源请求的时候，先判断http的referer字段的值，如果是从 自己的域名（uushare.com）过来的，则可以认为是合法的连接请求，否则就返回一个错误的提示信息。

这种方法通常用于图片、 mp3这种容易被人用html“嵌入”到其他网站的资源，使用这种方法可以防止你的图片直接出现在别人的网页里（或者防止mp3直接被其他网站嵌入到 flash播放器里），不过访客使用下载工具还是可以轻松下载，因为现在的下载工具一般会自动用你的域名构造一个引用地址，所以如果想再进一步防范的话，可以使用一个对应表限制每个资源的引用地址，例如将 jacky.mp3 的引用地址限制为 http://uushare.com/abc.htmlid=12345，这样下载工具就不太可能构造一个“正确”的引用地址了。

方法2：使用登录验证

这个方法常见于论坛、社区。当访客请求网站上的一个资源时，先判断此请求是否通过登录验证（在asp[.net](http://lib.csdn.net/base/dotnet)里常用session或form验证来记录登录状态），如果尚未登录则返回一个错误提示信息。使用这个方法还可以进一步判断登录的用户的权限是否足够，以实现带“权限”的下载。

不过因为登录状态依赖于会话id，而会话id往往储存于http请求的cookie字段里，下载工具一般没法获得浏览器的cookie字段，所以这些资源往往无法使用下载工具来下载，给正常合法用户带来诸多不便（因为大部分网民的系统都安装了下载工具，一点击下载链接一般会被下载工具拦截，导致无法使用浏览器本身的下载功能）。简单的解决方法是将这个session id放到URL中。  
这种方法的另外一个缺点是访客无法匿名下载，所以这个方法一般只用于论坛和社区网站。

方法3：使用cookie

其实这种方法原理上跟方法2差不多。就是在显示“下载”链接的页面里产生一个动态值的cookie，然后在处理资源下载请求时先判断cookie里有没有正确的cookie，如果没有则返回错误提示信息。至于这个动态值如何产生，只要能逆向判断动态值是否合法的都可以，例如将当前的时间去除秒数取哈希值（也叫散列值）。如果网页程序是asp[.Net](http://lib.csdn.net/base/dotnet)则更简单，可以往Session里随便存一个字符串或数字，然后在处理下载请求时先检查Session 里是否存在这个字符串或数字。使用这个方法的缺点跟方法2一样。

方法4：使用POST下载

客户端浏览器请求资源都是使用HTTP的GET方法的，其实使用POST方法也可以往客户端返回数据。所以可以将下载链接换成一个表单（Form）和一个按钮(Submit)，将待下载的文件的名称或id放到表单的一个隐藏文本框（Input）里，当用户点击提交按钮时，服务程序先判断请求是否为 POST方式，如果是则读取目标资源的二进制数据并写入响应对象（在asp.net里是respone.BinaryWrite方法）。

使用这个方法的缺点同样是无法使用下载工具，更没法实现断点续传。 不过比方法2，3好一点的是，下载工具不会拦截你的下载动作，所以正常用户还是比较顺畅地下载到文件。这个方法比较适合小文件的下载。

方法5：使用图形验证码

使用这个方法可以保证每次下载都是“人”在你的网站上下载，而不是下载工具。因为网上很多介绍使用图形验证码的方法，所以这里就不再重复了。这个方法的缺点是比较容易让正常的用户感到麻烦。

方法6：使用动态文件名

也叫动态钥匙法，当用户点击一个下载链接时，先在程序端计算一个Key（使用一定规律产生的Key，最好不要使用随机字符串例如GUID，并且这个 Key必须有一定时效的），然后在[数据库](http://lib.csdn.net/base/mysql)或Cache里记录这个Key以及它所对应的资源ID或文件名，最后让网页重定向一个新的URL地址，这个新 URL地址里需要包含这个Key。当浏览器或下载工具发出下载请求时，程序先检测这个Key是否存在，如果存在则返回对应的资源数据。

使用这个方法的好处是下载工具也可以下载，并且在Key失效前可以断点续传，并且可以通过Key来控制下载的线程数。

使用这个方法（包括以上所有支持下载工具的方法）的缺点是：当任意一个用户下载成功之后，你的资源就会被一些下载工具列入“资源候选名单”，以后其他人在其他地方下载同样的文件时，下载工具会不断连接你的服务器，即使你的文件已经删除或者Key已经失效了，这样会造成类DDos攻击的后果，下面再介绍两个即可以让下载工具下载，又可以防止盗链的方法。

方法7：擅改资源的内容

一般热门的资源都是电影、mp3、较大的压缩包等，这些文件都是有很多可以插入数据的地方的，例如mp3有一个tag区，rar/zip有一个备注区，电影的内容随便一个地方，只要在下载过程当中，动态地往这些地方注入一些随机的字节（几个字节即可），就可以达到让整个文件的哈希值（即散列值、指纹值）发生改变，让从你网站下载的文件的哈希值跟别人的不一样，就可以防止下载工具主动找上门了。用这个方法配合方法6，可以达到较好的防盗链的效果。缺点是，虽然文件被修改的部分不会被“看”、“听”出来，不过多多少少让知道的人觉得不爽。另外就是如果别人把从你网站下载的文件放到其他网站，那么仍然存在下载工具主动找上门的情况（虽然实际上它下载不了内容）。

方法8：打包下载

这个方法跟方法7的道理是一样的，只不过这次不是往原始文件里修改，而是在原始的文件基础上再加个“外壳”，让资源的哈希值跟别人的不一样。使用这个方法可以在不擅改资源原始的内容基础上实现方法6同样的效果，并且狠一点的话，甚至可以在打包的时候放入自己的一些广告。缺点是用户每次下载都得加压缩，不过目前大部分人都懂得解压，所以这个缺点有时可以忽略不计。