

# Prevent HTML5 video from being downloaded (right-click saved)?

Asked 12 years, 9 months ago   Modified 2 years, 5 months ago

Viewed 376k times



243

How can I disable "Save Video As..." from a browser's right-click menu to prevent clients from downloading a video?



Are there more complete solutions that prevent the client from accessing a file path directly?



javascript

html

html5-video

Share

Improve this question

Follow

edited Sep 7, 2020 at 19:58



Braiam

4,484 ● 11 ● 49 ● 81

asked Mar 18, 2012 at 7:53



python

2,825 ● 3 ● 18 ● 15

- 
- 4 I up-voted this question because it only *absolutely asks* for how to "disable the right-click" for an HTML5 video. I am not sure if it is similar to right-click disabling for normal images or if there are other overlay tricks, etc., that can be applied.  
– user166390 Mar 18, 2012 at 8:35
-

13 Even if you disable right-click, they can still save it from the browser menu ( `File→Save As` ). Even if you could somehow block that, they can *view-source* to find the URL of the file. Even if you could obscure that a bit, they can rip it from the cache. Even if you could complicate that (e.g., stream), they can capture the network traffic with a sniffer or something. The fact is, if you send it to a user, they can save it. No way around that. The question you need to ask is why you need to stop it so badly. Is it really even that necessary? Is it worth the effort and user-unfriendly-ness? – [Synetech](#) Jul 25, 2015 at 0:43

---

I'm going to look pedantic here, but you're overloading the term "download". You of course *do want* to allow the video to be downloaded. – [Johan Boulé](#) Jul 13, 2018 at 14:11

---

28 Answers

Sorted by:

Highest score (default)



## You can't.

342



That's because that's what browsers were designed to do: *Serve content*. But you can *make it harder to download*.

---



## Convenient "Solution"



I'd just upload my video to a third-party video site, like YouTube or Vimeo. They have good video management tools, optimizes playback to the device, and they make efforts in preventing their videos from being ripped with zero effort on your end.

---

## Workaround 1, Disabling "The Right Click"

You *could* disable the `contextmenu` [event](#), aka "the right click". That would prevent your regular skiddie from blatantly ripping your video by right clicking and Save As. But then they could just disable JS and get around this or find the video source via the browser's debugger. Plus this is bad UX. There are lots of legitimate things in a context menu than just Save As.

---

## Workaround 2, Video Player Libraries

Use custom video player libraries. Most of them implement video players that customize the context menu to your liking. So you don't get the default browser context menu. And if ever they do serve a menu item similar to Save As, you can disable it. But again, this is a JS workaround. Weaknesses are similar to Workaround 1.

---

## Workaround 3, HTTP Live Streaming

Another way to do it is to serve the video using [HTTP Live Streaming](#). What it essentially does is chop up the video into chunks and serve it one after the other. This is how most streaming sites serve video. So even if you manage to Save As, you only save a chunk, not the whole video. It would take a bit more effort to gather all

the chunks and stitch them using some dedicated software.

---

## Workaround 4, Painting on Canvas

Another technique is to [paint <video> on <canvas>](#). In this technique, with a bit of JavaScript, what you see on the page is a `<canvas>` element rendering frames from a hidden `<video>`. And because it's a `<canvas>`, the context menu will use an `<img>`'s menu, not a `<video>`'s. You'll get a Save Image As instead of a Save Video As.

---

## Workaround 5, CSRF Tokens

You could also use [CSRF tokens](#) to your advantage. You'd have your sever send down a token on the page. You then use that token to fetch your video. Your server checks to see if it's a valid token before it serves the video, or get an [HTTP 401](#). The idea is that you can only ever get a video by having a token which you can only ever get if you came from the page, not directly visiting the video url.

---

Share Improve this answer

edited Apr 28, 2022 at 19:00

Follow



ioannis-kokkalis

454 ● 1 ● 3 ● 14

answered Mar 18, 2012 at 8:05



Joseph

120k ● 30 ● 183 ● 237

---

1 thank you for the detail answer, is it possible at least disable the save as option from the right click menu? it will cover most basic knowledge cases – [python](#) Mar 18, 2012 at 8:43

---

3 that depends on the browser. i have seen times (especially firefox and chrome) that if the video is fully loaded, when you hit "save" they just pick the video from the cache instead of re-downloading (the video is already downloaded in the cache, why download it again?), thus there is no second request. the method above is applicable only when the link is reused. – [Joseph](#) Mar 18, 2012 at 8:49 ✎

---


1 well, i found an article talking about overlaying the video tag with a div. updated my answer – [Joseph](#) Mar 18, 2012 at 9:25 ✎

---

3 Thanks. I Just read [craftymind.com/factory/html5video/CanvasVideo.html](http://craftymind.com/factory/html5video/CanvasVideo.html). The idea is almost same as your answer. – [Trung](#) Jun 19, 2013 at 4:26

---

1 @Cupidvogel The "onetime use url" is a server endpoint which accepts a server generated token. The token is generated upon page generation, and saved to the db. It is also shipped with the page as `src` of the `<video>`. By the time your page has loaded, the db has the token, the page has the token. Once `<video>` starts to load (Accesses the endpoint), server checks if token is in the db, deletes it and streams the file. If the token isn't there *as a result of second*

access, then don't stream the file. – [Joseph](#) Sep 4, 2014 at 2:27 

---



137



This is a simple solution for those wishing to simply remove the right-click "save" option from the html5 videos

```
$(document).ready(function(){  
    $('#videoElementID').bind('contextmenu',function()  
    });
```

Share Improve this answer

answered Feb 6, 2013 at 18:39

Follow



[Clayton Graul](#)

1,495 ● 1 ● 10 ● 7

- 
- 1 That is fantastic ! It does a great job from preventing ordinary people from downloading the video ! – [Etienne Noël](#) Jun 7, 2013 at 20:46
- 
- 4 This does not help however if JavaScript is disabled in the browser. – [mvark](#) May 2, 2014 at 16:10
- 
- 3 Thanks, this solution is sufficient for 90 % of all our visitors. – [Avatar](#) Jun 5, 2014 at 6:12
- 
- 5 Bleh. Just inspect the element in Firebug, see the `src` attribute, and open that in another tab or use `wget` to download it! – [NedStarkOfWinterfell](#) Sep 3, 2014 at 11:31
- 
- 22 I think the main aim of this is to avoid "normal" users to download the video. This is a good solution to solve this situation. – [Unapedra](#) Sep 5, 2014 at 10:30
-



59



# Yes, you can do this in three steps:

1. Place the files you want to protect in a subdirectory of the directory where your code is running.

```
www.foo.com/player.html  
www.foo.com/videos/video.mp4
```

2. Save a file in that subdirectory named ".htaccess" and add the lines below.

```
www.foo.com/videos/.htaccess
```

```
#Contents of .htaccess
```

```
RewriteEngine on  
RewriteCond %{HTTP_REFERER} !^http://foo.com/.*$ [  
RewriteCond %{HTTP_REFERER} !^http://www.foo.com/.  
RewriteRule .(mp4|mp3|avi)$ - [F]
```

**Now the source link is useless**, but we still need to make sure any user attempting to download the file cannot be directly served the file.

3. For **a more complete solution**, now serve the video with a flash player (or html canvas) and never link to the video directly. To just remove the right click menu, add to your HTML:

```
<body oncontextmenu="return false;">
```

## The Result:

www.foo.com/player.html **will correctly play video**, but if you visit www.foo.com/videos/video.mp4:

Error Code 403: FORBIDDEN

**This will work for direct download, cURL, hotlinking, you name it.**

*This is a complete answer to the two questions asked and not an answer to the question: "can I stop a user from downloading a video they have already downloaded."*

Share Improve this answer

Follow

edited Aug 28, 2017 at 14:35



Timo Schwarzer

409 ● 5 ● 17

answered Dec 25, 2016 at 21:58



Tzshand

1,613 ● 12 ● 14



---

2 Great answer, but you have a ` that you should remove it from your `.htaccess` content – [MAZux](#) Jul 6, 2017 at 8:22



---

6 You can still fake the HTTP Referer, which will allow a person to download. However, this is a very clever solution. If you club this with a one-time code on the file, you're good to go! – [Shiroy](#) Sep 1, 2017 at 18:34

---

2 It seems still IDM can download it! – [PersianMan](#) Jun 24, 2018 at 5:42

---

@PersianMan Correct - I encourage you to read the first answers – [Tzshand](#) Jun 28, 2018 at 1:59

---

1 If you disable the javascript from browser, then this trick won't work, since then right click gets enabled. To avoid that too, you should fetch and load the video element dynamically using jquery. – [Anindya Sankar Dasgupta](#) Apr 10, 2019 at 16:04

---



Simple answer,

49

## YOU CAN'T



If they are watching your video, they **have it already**



You can slow them down but can't stop them.

Share Improve this answer

edited Jun 20, 2020 at 9:12

Follow



Community Bot

1 • 1

answered Mar 18, 2012 at 8:15



Starx

78.8k ● 50 ● 186 ● 265

- 
- 2 By the way, this answer apply with HTML5 videos, flash videos, or any technology you can imagine in the future. It's simple: it's how it works. – [Gustavo Rodrigues](#) Aug 24, 2015 at 22:06
- 
- 4 That is not an answer to either of the questions. – [Tzshand](#) Dec 29, 2016 at 3:14
- 
- 6 People could record their entire screen and audio and fool all workarounds, that is why they can only be slowed down. – [kintsukuroi](#) Apr 28, 2020 at 6:11
- 
- 3 Everytime I searched about techniques to make it harder for my users to download or copy our copyrighted contents (which is expensive to make, sold at expensive price and is exclusive content) some people post "you can't". Obviously technically you could always imagine a way to hack around protections but in practice if one/two users hack and share our stuff it doesn't matter but if 100% do it our company is over. – [Ilan Schemoul](#) Jul 3, 2020 at 12:06
- 
- 1 @IlanSchemoul Interesting. Thanks for sharing :) – [Starx](#) Jul 6, 2020 at 18:15
- 



28

As a client-side developer I recommend to use blob URL, blob URL is a client-side URL which refers to a binary object



```
<video id="id" width="320" height="240" type='video/m
```





in HTML leave your video `src` blank, and in JS fetch the video file using AJAX, make sure the response type is *blob*

```
window.onload = function() {
    var xhr = new XMLHttpRequest();
    xhr.open('GET', 'mov_bbb.mp4', true);
    xhr.responseType = 'blob'; //important
    xhr.onload = function(e) {
        if (this.status == 200) {
            console.log("loaded");
            var blob = this.response;
            var video = document.getElementById('id');
            video.oncanplaythrough = function() {
                console.log("Can play through video wi
                URL.revokeObjectURL(this.src);
            };
            video.src = URL.createObjectURL(blob);
            video.load();
        }
    };
    xhr.send();
}
```

Note: This method is not recommended for large file

## EDIT

- Use cross-origin blocking and header token checking to prevent direct downloading.
- If the video is delivered via an API; Use a different http method (PUT / POST) instead of 'GET'

Share Improve this answer

edited Jun 30, 2021 at 9:21

Follow

answered Jan 18, 2017 at 6:20



Sajan

821 ● 9 ● 15

- 
- 4 YouTube uses Blob now too i think :) ? – [C0nw0nk](#) Jul 20, 2017 at 22:13
- 
- 1 Can you explain what's going on here more clearly and how to set up the server for this? – [Anthony](#) Dec 1, 2017 at 1:21
- 
- 1 @nerdofcode how does that behave when users try to forward the video? Will they need to wait until all the video download? – [John Balvin Arias](#) Jun 7, 2018 at 4:31
- 
- 1 @JohnBalvinArias! I've not tested this 100%, but I'm going to say that it only needs a quick buffer... Don't quote me on this though... – [NerdOfCode](#) Jun 7, 2018 at 4:48
- 
- 3 If I inspect the page, in the Network tab I get a request for the video that I can just open in a new tab. – [Simone](#) Feb 26, 2019 at 8:25
- 



27

The best way that I usually use is very simple, I fully disable context menu in the whole page, pure html+javascript:



```
<body oncontextmenu="return false;">
```



That's it! I do that because you can always see the source by right click.

Ok, you say: "I can use directly the browser view source" and it's true but we start from the fact that you **CAN'T** stop downloading `html5` videos.

Share Improve this answer

edited Aug 15, 2016 at 9:21

Follow



Christian Giupponi

7,618 ● 11 ● 74 ● 118

answered Jul 17, 2014 at 14:16



Daniele Cannova

307 ● 3 ● 5

I think the solución must be one that does not disturb "normal" users, disable right click will prevent users to copy and paste some text, or search a word they are intersested in, for example un the title of the video,of course not all users will likely do that but It can be anoying for some of them

– John Balvin Arias Jun 7, 2018 at 4:17



You can use

15

```
<video src="..." ... controlsList="nodownload">
```



<https://developer.mozilla.org/en-US/docs/Web/API/HTMLMediaElement/controlsList>



It doesn't prevent saving the video, but it does remove the download button and the "Save as" option in the context menu.

Share Improve this answer

answered Feb 16, 2020 at 5:09

Follow



clickbait

2,988 ● 1 ● 30 ● 65



We could make that not so easy by hiding context menu, like this:

14



```
<video oncontextmenu="return false;" controls>
  <source src="https://yoursite.com/yourvideo.mp4" >
</video>
```



Share Improve this answer

answered Jan 4, 2019 at 12:09

Follow



Jcyrss

1,748 ● 3 ● 22 ● 36



PHP sends the html5 video tag together with a session where the key is a random string and the value is the filename.

13



```
ini_set('session.use_cookies',1);
session_start();
$ogv=uniqid();
$_SESSION[$ogv]='myVideo.ogv';
$webm=uniqid();
$_SESSION[$webm]='myVideo.webm';
echo '<video autoplay="autoplay">'
    .'<source src="video.php?video='.$ogv.'" type="vide
    .'<source src="video.php?video='.$webm.'" type="vid
    .'</video>';
```



Now PHP is asked to send the video. PHP recovers the filename; deletes the session and sends the video

instantly. Additionally all the 'no cache' and mime-type headers must be present.

```
ini_set('session.use_cookies',1);
session_start();
$file='myhiddenvideos/'.$_SESSION[$_GET['video']];
$_SESSION=array();
$params = session_get_cookie_params();
setcookie(session_name(),'', time()-42000,$params["path"],
                                           $params["secure"],
                                           $params["httponly"]);
if(!file_exists($file) or $file===' or !is_readable($file))
    header('HTTP/1.1 404 File not found',true);
    exit;
}
readfile($file);
exit;
```

Now if the user copy the url in a new tab or use the context menu he will have no luck.

Share Improve this answer

edited Nov 12, 2014 at 10:02

Follow

answered Nov 4, 2014 at 9:07



B.F.

477 ● 6 ● 9

- 
- 2 I like the solution- it solves OPs question. One unfortunate thing is, when the checks the source code in Chrome and right-clicks on the link. The user will download a html file, which in fact will be the video file. – [user1252280](#) Dec 21, 2017 at 20:48
-



9



We ended up using AWS CloudFront with expiring URLs. The video will load, but by the time the user right clicks and chooses Save As the video url they initially received has expired. Do a search for CloudFront Origin Access Identity.

Producing the video url requires a key pair which can be created in the AWS CLI. FYI this is not my code but it works great!

```
$resource = 'http://cdn.yourwebsite.com/videos/yourvid
$timeout = 4;

//This comes from key pair you generated for cloudfront
$keyPairId = "AKAJSDHFKASWERASDF";

$expires = time() + $timeout; //Time out in seconds
$json = '{"Statement":[{"Resource":'."'$resource.'", "Co
{"AWS:EpochTime":'."'$expires.'"}]}]}'

//Read Cloudfront Private Key Pair
$fpr=fopen("/absolute/path/to/your/cloudfront_privateke
$priv_key=fread($fp, 8192);
fclose($fp);

//Create the private key
$key = openssl_get_privatekey($priv_key);
if(!$key)
{
    echo "<p>Failed to load private key!</p>";
    return;
}

//Sign the policy with the private key
if(!openssl_sign($json, $signed_policy, $key, OPENSSL_
{
    echo '<p>Failed to sign policy: ' .openssl_error_st
    return;
}
```



```
//Create url safe signed policy
$base64_signed_policy = base64_encode($signed_policy);
$signature = str_replace(array('+','='),array('-',_'), array('-',_')
$base64_signed_policy);

//Construct the URL
$url = $resource.'?Expires='.$expires.'&Signature='.$s
Id='.$keyPairId;

return '<div class="videowrapper" ><video autoplay con
style="width:100%!important;height:auto!important;"><s
type="video/mp4">Your browser does not support the vid
```

Share Improve this answer

answered Sep 28, 2016 at 18:32

Follow



prophoto

362 ● 4 ● 9

- 
- 1 Highly underrated comment. I'd advise using [docs.aws.amazon.com/sdk-for-php/v3/developer-guide/...](https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/) nowadays however. – Zmart Oct 18, 2018 at 3:29
  - 2 If the token expires, does that mean they cannot navigate around the video either? As this seems to contact the video URL again. – Chud37 Apr 5, 2019 at 12:27
- 

If the video is expired, the user can still watching the video to the end. But I don't think they can jump to different time stamp. When they jump, that's another GET request, which will fail. – iouzzr May 28, 2022 at 3:28

---



7

You can at least stop the the non-tech savvy people from using the right-click context menu to download your video. You can disable the context menu for any element using the oncontextmenu attribute.



```
oncontextmenu="return false;"
```



This works for the body element (whole page) or just a single video using it inside the video tag.

```
<video oncontextmenu="return false;" controls>...</vid
```

Share Improve this answer

edited Jun 2, 2016 at 15:59

Follow

answered Jun 2, 2016 at 15:49



[TxRegex](#)

2,415 ● 23 ● 20



7

First of all realise **it is impossible to completely prevent a video being downloaded, all you can do is make it more difficult**. I.e. you hide the source of the video.



A web browser temporarily downloads the video in a buffer, so if could prevent download you would also be preventing the video being viewed as well.

You should also know that <1% of the total population of the world will be able to understand the source code making it rather safe anyway. That does not mean you should not hide it in the source as well - *you should*.

You should **not** disable right click, and even less you should display a message saying "You cannot save this

video for copyright reasons. Sorry about that." . As suggested in [this answer](#).

This can be very annoying and confusing for the user. Apart from that; if they disable JavaScript on their browser they *will* be able to right click and save anyway.

Here is a CSS trick you could use:

```
video {  
    pointer-events: none;  
}
```

CSS cannot be turned off in browser, protecting your video without actually disabling right click. However one problem is that `controls` cannot be enabled either, in other words they must be set to `false` . If you are going to implement your own Play/Pause function or use an API that has buttons separate to the `video` tag then this is a feasible option.

`controls` also has a download button so using it is not such a good idea either.

Here is a [JSFiddle](#) example.

---

If you are going to disable right click using JavaScript then also store the source of the video in JavaScript as well. That way if the user disables JavaScript (allowing right click) the video will not load (it also hides the video source a little better).

From [TxRegex answer](#):

```
<video oncontextmenu="return false;" controls>
  <source type="video/mp4" id="video">
</video>
```

Now add the video via JavaScript:

```
document.getElementById("video").src =
  "https://www.w3schools.com/html/mov_bbb.mp4";
```

Functional [JSFiddle](#)

---

Another way to prevent right click involves using the `embed` tag. This does not however provide the controls to run the video so they would need to be implemented in JavaScript:

```
<embed src="https://www.w3schools.com/html/mov_bbb.mp4"
```

Share Improve this answer

answered Mar 17, 2018 at 0:14

Follow




**Xantium**

11.6k ● 12 ● 70 ● 93

- 
- 1 Adding the src URL via JavaScript is not much useful. Inspecting the DOM will show the URL in plain sight after the script set it. – [Simone](#) Feb 26, 2019 at 8:14
- 

@Simone I agree, however it is better than showing it directly in the html source, where anybody that right clicks it can see it immediatly. You could always split up the url or encrypt it.

But remember that is additional processing – [Xantium](#) Feb 26, 2019 at 11:37 

---

- 2 "CSS cannot be turned off in browser" - Technically you can modify the CSS in web browser developer tools to disable certain CSS rules, so a more tech-savvy person could remove `pointer-events: none;` – [Phil Gibbins](#) Apr 21, 2020 at 9:26
- 



6



well, you can't protect it 100% but you can make it harder. these methods that I'm explaining, I faced them during studying protection methods in [PluralSight](#) and [BestDotNetTraining](#). nevertheless, none of these methods stopped me from downloading what I want, but I had a hard time to curate the downloader to pass their protection.

In addition to other mentioned methods to disable the context menu. the user still is able to use third-party tools like InternetDownload manager or other similar software to download the videos. the protection method that I'm explaining here is to mitigate those 3rd party software.

the requirement of all of these methods is to block a user when you identify someone is downloading your videos. in this way they are able to download only one or two videos only before you banned them from accessing to your website.

## disclaimer

I will not accept any responsibility if someone abuses these methods or use it to harm others or the websites that I mentioned as an example. it's just for sharing knowledge to help you to protect your intellectual product.

## **generate links with an expiry**

the requirement for this is to create a download link per user. that one can easily be handled by azure blob storage or amazon s3. you can create a download link with twice of the video length expiry timestamp. then you need to capture that video link and the time that is requested. this is necessary for the next method. the catch for this method is you are generating the download link when the user click the play button.

on play button event you will send a request to the server and get the link and update the source.

## **throttle the video request rate**

then you monitor how fast the user request for the second video. if the user request for a download link too fast, then you block them right away. you can't put this threshold too big because you can mistakenly block users that are just browsing or skimming through the videos.

# Enable HTTP Range

use some js library like [videojs](#) to play your video, also you need to return an AcceptRange in your header. Azure blob storage supports this out of the box. this way the browser starts to download the video chunk by chunk. usually, 32byte by 32byte. then you need to listen to videojs `timeupdate` change and update your server about the percentage that the video is watched. the percentage that the video is watched can't be more than the percentage that video is delivered. and if you are delivering a video content without receiving any percentage change, then you can block the user. because for sure they are downloading.

implementing this is tricky because the user can skip the video forward or backwards so be conscious about this when you are implementing this.

this is how BestDotnetTraining is handling the

`timeupdate`

```
myPlayer.ready(function () {
    //var player = this;
    this.src({
        type: "video/mp4",
        src: videoURL
    });
    if (videoId) {
        myPlayer.play();
        this.on('timeupdate', function () {
            var currentPercent = parseInt(100 * myPlay
myPlayer.duration()); //calculate as percentage
            if (currentPercent % 5 == 0) {
                //send percentage to server
            }
        });
    }
});
```

```

        SaveVideoDurationWatched(currentPercen
    }
    });
}
});

```

anyway, the user is able to work around this by using some download method that downloads a file through streaming. almost c# do it out of the box and for nodejs, you can use `request` module. then you need to start a stopwatch, listen to a package received and compare the total byte received compare to the total size. this way you can calculate a percentage and the time spent to get that amount of percentage. then use the `Thread.Sleep()` or something like that to delay the thread the amount that you have to wait if you watch the video normally. also before the sleep the user can call the server and update the percentage that is received. so the server thinks that the user is actually watching a video.

the calculation will be something like this, for example, if you calculate that you received 1 per cent so far, then you can calculate the amount that you should wait to sleep the download thread. in this way you can't download a video faster than what it's actual length is. if a video is 24 min. it will takes 24 min to download it. (plus the threshold we put in the first method)

```

original video length 24 minute
24 min * 60000 = 1,440,000 milliseconds
1,440,000 % 100 = 14,400 milisecond is needed to downl

```



# check the browser agent

when you are serving a webpage and serving the video link or accepting the progress update request you can look at the browser agent. if it's different then ban the user.

just be aware that some old browser doesn't pass this information. so you should ignore this when there is no browser agent in both video request and webpage request. but if one request has it and another one doesn't, then you should ban the user.

to work around this the user can set the browser agent header manually same as the headless browser that they are using to capture the download link.

# check the referer header

when the referer is something other than your host URL or the page URL that you are serving the video, you can ban the user, because they put the download link in another tab or another application. even you can do that for the progress update request.

the requirement for this is to has a mapping of video and the page that shows that video. you can create some convention or pattern to understand what the URL should be, it's up to your design.

to work around it the user can set the referrer header manually equal to the download page URL when downloading the videos.

## Calculate the time between request

if you receive so many requests that the time between them is the same, then you should block the user. you should put this to capture how much is time between the video link generation request. if they are the same (plus/minus some threshold) and it happens more than a number of times, then you can ban the user. because if there is a bot that is going to crawl your website or videos, then usually they have the same sleep time between their request. so if you receive each request, for example, every 1.3(minus some deviation) minutes. then you raise an alarm. for this, you can use some statistic calculation to know the deviation between the requests.

to workaround this, the user can put a random sleep time between the requests.

## sample code

I have a repo [PluralSight-Downloader](#) that is doing it halfway. I created this repo almost 5 years ago. because I

wrote it for study purpose and own personal use only, the repo isn't received any update so far and I'm not going to update or make it easy to work with. it's just an example of how it can be done.

Share Improve this answer

edited Nov 22, 2019 at 4:24

Follow

answered Nov 22, 2019 at 4:03



**Mohammad Hossein Amri**

1,986 ● 2 ● 28 ● 49



The

5

```
<body oncontextmenu="return false;">
```



no longer works. Chrome and Opera as of June 2018 has a submenu on the timeline to allow straight download, so user doesn't need to right click to download that video. Interestingly Firefox and Edge don't have this ...



Share Improve this answer

answered Jun 18, 2018 at 10:45

Follow



**kendolew**

506 ● 6 ● 6

Chrome+Opera+Edge now supports

`controlsList="nodownload"` – [oriadam](#) Dec 16, 2020 at 15:03



3



Using a service such as Vimeo: Sign in [Vimeo](#) > [Goto Video](#) > [Settings](#) > [Privacy](#) > [Mark as Secured](#), and also select embed domains. Once the embed domains are set, it will not allow anyone to embed the video or display it from the browser unless connecting from the domains specified. So, if you have a page that is secured on your server which loads the Vimeo player in iframe, this makes it pretty difficult to get around.

[Share](#) [Improve this answer](#)

[Follow](#)

edited Sep 5, 2014 at 12:15



[user2771704](#)

6,194 ● 6 ● 40 ● 42

answered Sep 5, 2014 at 11:54



[CommentUser](#)

39 ● 1



3



+1 simple and cross-browser way: You can also put transparent picture over the video with css z-index and opacity. So users will see "save picture as" instead of "save video" in context menu.

[Share](#) [Improve this answer](#)

[Follow](#)

answered Jul 31, 2015 at 22:24



[Alex Babak](#)

489 ● 3 ● 15

---

2 why pic it will take time to load so only put the div tag and they will get a big menu of chrome like back reload etc  
– [Waqas Tahir](#) Dec 31, 2015 at 5:57

---

I am not sure but still video can be downloaded via  
File>SaveAs – [Arun Kumar](#) Aug 10, 2016 at 11:13

---



Here's a complete solution for disabling download  
**including right click > Save as...** in the context menu:

3



```
<video oncontextmenu="return false;" controlsList="nodownload">
</video>
```



Share Improve this answer

answered May 26, 2021 at 5:51



Follow



[Koby Douek](#)

16.7k ● 20 ● 78 ● 110



Try this for disable download Video options

3



```
<video src="" controls controlsList="nodownload"></video>
```



Share Improve this answer

answered Feb 2, 2022 at 14:44



Follow



[Muthulakshmi M](#)

777 ● 7 ● 9



Here's what I did:

2



```
function noRightClick() {
    alert("You cannot save this video for copyright that.");
}
```



```
<body oncontextmenu="noRightClick();">
<video>
<source src="http://calumchilds.com/videos/big_b
type="video/mp4">
</video>
</body>
```

[Run code snippet](#)[Expand snippet](#)

This also works for images, text and pretty much anything. However, you can still access the "Inspect" and the "View source" tool through keyboard shortcuts. (As the answer at the top says, you can't stop it entirely.) But you can try to put barriers up to stop them.

[Share](#) [Improve this answer](#)

answered Mar 9, 2018 at 16:37

[Follow](#)

[Calum Childs](#)

337 ● 2 ● 13



1



controlsList Prevent action such as download begin fullscreen without adding any other JavaScript function

```
<video width="400" controlsList="nofullscreen nodo
```

[Share](#) [Improve this answer](#)

answered Oct 15, 2021 at 5:56

[Follow](#)

[Arun AL](#)

138 ● 2 ● 5

doesn't work, still allows downloading – [Zimba](#) Oct 21, 2021 at 7:41



0

It seems like streaming the video through websocket is a viable option, as in stream the frames and draw them on a canvas sort of thing.



[Video streaming over websockets using JavaScript](#)



I think that would provide another level of protection making it more difficult for the client to acquire the video and of course solve your problem with "Save video as..." right-click context menu option ( overkill ?! ).

Share Improve this answer

edited May 23, 2017 at 12:34

Follow



Community Bot

1 • 1

answered Dec 11, 2015 at 2:38



Guy

1,322 • 18 • 19



0

**Short Answer:** Encrypt the link like youtube does, don't know how than ask youtube/google of how they do it. (Just in case you want to get straight into the point.)



I would like to point out to anyone that this is possible because youtube does it and if they can so can any other website and it isn't from the browser either because I tested it on a couple browsers such as microsoft edge



and internet explorer and so there is a way to disable it and seen that people still say it...I tried looking for an answer because if youtube can then there has to be a way and the only way to see how they do it is if someone looked into the scripts of youtube which I am doing now. I also checked to see if it was a custom context menu as well and it isn't because the context menu is overflowing the inspect element and I mean like it is over it and I looked and it never creates a new class and also it is impossible to actually access inspect element with javascript so it can't be. You can tell when it double right-clicks a youtube video that it pops up the context menu for chrome. Besides...youtube wouldn't add that function in. I am doing research and looking through the source of youtube so I will be back if I find the answer...if anyone says you can't then, well they didn't do research like I have. The only way to download youtube videos is through a video download.

Okay...I did research and my research says that you can disable it except there is no javascript to it...you have to be able to encrypt the links to the video for you to be able to disable it because I think any browser won't show it if it can't find it and when I opened a youtube video link it showed as this "blob:<https://www.youtube.com/e5c4808e-297e-451f-80da-3e838caa1275>" without quotes so it is encrypting it so it cannot be saved...you need to know php for that but like the answer you picked out of making it harder, youtube makes it the hardest of heavy encrypting it, you need to be an advanced php programmer but if you don't know that then take the



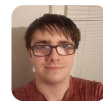
person you picked as best answer of making it hard to download it...but if you know php than heavy encrypt the video link so it only is able to be read on yours...I don't know how to explain how they do it but they did and there is a way. The way youtube Encrypts there videos is quite smart so if you want to know how to than just ask youtube/google of how they do it...hope this helps for you although you already picked a best answer. So encrypting the link is best in short terms.

Share Improve this answer

edited Nov 4, 2018 at 16:10

Follow

answered Nov 4, 2018 at 15:57



Jonathan J. Pecany

313 ● 2 ● 17



If you are looking for a complete solution/plugin, I've found this very useful

0

<https://github.com/mediaelement/mediaelement>



Share Improve this answer

answered Dec 30, 2020 at 12:15

Follow



Matheus Barbosa

2,725 ● 1 ● 21 ● 22





0

**Prevent HTML5 video from being downloaded (right-click saved)**



```
<video type="video/mp4" width="330" height="300" controls oncontextmenu="return false;"></video>
```



Share Improve this answer

answered May 8, 2021 at 13:54



Follow



[dammika rajapaksha](#)

22 ● 6



0

**You can't.**

For example, people can use some API<sup>for example</sup> [desktopCapture](#), [getUserMedia](#) that allows users to record [screen, window, tab](#).



People can use it and write it to the [canvas](#) and then concatenate all the [chunks](#) together to get the video,



So there is no way to stop them from downloading the video if they really want it.

Share Improve this answer

answered Aug 21, 2021 at 15:59

Follow



[Carson](#)

7,820 ● 2 ● 56 ● 54



0

I found a good answer to a similar problem, using PHP instead of JavaScript for better security.



I want to play test.mp4 in the user's browser using the browser's default player (just as though URL/test.mp4 had been clicked on a Web page), but requiring a password, which is either supplied by the user or internally by software.

Here is a brief sketch of the idea. It starts with the user going to (running) a program I wrote called secure.php to play test.mp4.

The file test.mp4 is in a subdirectory ("secureSubdirectory") that contains a .htaccess containing "Require all denied". This immediately prevents any direct access through a URL.

When secure.php is run, it supplies a password (or queries the user for a password), then does a POST to itself that includes the password, verifies it using a salt, using the PHP commands:

```
$Hash=base64_encode(hash_hmac("sha256",$Pwd,$Salt,true));  
$HashesAreSame=hash_equals($Hash,$GoalHash);
```

then tests for test.mp4 existing, and executes the following PHP code to return the test.mp4 file as a byte stream to the user's browser:

```
header("Content-Type: video/mp4");  
echo file_get_contents("secureSubdirectory/$path");  
exit;
```

The video shows as expected. If I then right-click on the page showing the video and try saving the video, the resulting file will just contain an error string, like "Error: password not found", since test.mp4 is being queried using the plain secure.php URL, not through POST with the correct password.

Of course, you can obtain the response payload (the video bytes) using the Network option of the browser debugging tools, but this could be prevented by the PHP program or the .htaccess file if the browser provided an option to prevent access to the debugging tools.

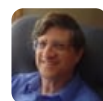
I can't imagine a failure case, but I'd be very interested if one exists, as simple but perfect authorization is a very rare thing. (Note that, since this method relies on a password, associating it with the user is not a secure way to authenticate, since the user can accidentally or deliberately publish or share the password.)

Share Improve this answer

edited Jul 10, 2022 at 13:29

Follow

answered Jul 9, 2022 at 16:42



David Spector

1,668 ● 16 ● 24



@Clayton-Graul had what I was looking for, except I needed the CoffeeScript version for a site using

-1



AngularJS. Just in case you need that too, here's what you put in the AngularJS controller in question:



```
# This is how to we do JQuery ready() dom stuff
$ ->
    # let's hide those annoying download video opt
    # of course anyone who knows how can still dow
    # the video, but hey... more power to 'em.
    $('#my-video').bind 'contextmenu', ->
        false
```

"strange things are afoot at the circle k" (it's true)

Share Improve this answer

answered Oct 4, 2014 at 5:01

Follow



Ryan Crews

3,033 ● 1 ● 33 ● 28



-2

Everything you see in the browser is downloaded content. The question being alluded to is how to save that content in the browser. To view content, client browsers download from content servers and make it available locally.



One solution becoming popular is to save (ephemeral) content in browser only, and for a limited time, in a way that cannot be saved directly. Blobs are one implementation of this with the added benefit of reducing bandwidth & storage overheads, since the content is stored in binary objects.

The short expiry of content makes persistent storage almost impossible to ordinary users since new content is

displayed before user can attempt to save expired content.

Share Improve this answer

answered Oct 21, 2021 at 12:07

Follow



Zimba

3,643 ● 23 ● 27



**Highly active question.** Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.