JQuery XSS DOM:

JqueryXSS.html Finding

Another approach a friend of mine once used was this:

var hash = location.hash.split('#')[1];

This also has its issues, which are ironically less than the first one, even though it seems a far more naive approach.

* With the same test hash, it would at least get the "foo@o" part, which means it only fails when the hash contains a pound sign
* When there’s no hash, it doesn’t throw an error, although it returns undefined instead of the empty string.

JqueryXSS2.html Finding

if you try to input <script>alert('xss')</script>, only one alert box will be thrown: the one inside `test-html' div (with html() function). as you can see test-html div is empty, and test-innerhtml div contans the script.

.html() strips out the script tags before it inputs the HTML and executes them separately.

 combines malicious JavaScript with an iframe that loads a legitimate page in an effort to steal data from an unsuspecting user.

The iframe loads the flawed example.com page, and injects some script into it through the XSS flaw.

 this attack is effectively the same as a conventional XSS attack; the attacker simply uses the src attribute of the injected iframe element as a vehicle to run some javascript code in the attacked page.

Script pseudo-protocols can be used in various locations to execute inline script within an attribute that expects a URL. Here are some examples:

<object data=javascript:alert(1)>

<iframe src=javascript:alert(1)>

<embed src=javascript:alert(1)>

AngularJS XSS

Angular.html and angular2.html

Pay attention to the HTML code entered in the text field. It is printed as it is, on to the HTML page. Having a bit of trouble making it work.