



Possessive: IE Apostrophes⁰
infectionvectors.com
September 2007

Overview

In a March 2006 feature¹ infectionvectors.com discussed how a patch is classified as a “security fix” instead of just a “fix.” Performance patches are a well-established means of improving a product’s usability, as are the similarly used “bug fixes.” Given the same content, however, as was noted in the report, a patch classified as a “recommended performance update” will get less attention than anything called a “security patch.” Moreover, the often fuzzy definition of security is called into question with every software patch, as the “eye of the beholder” is often the most important distinguishing characteristic in determining whether the flaw being corrected is security-related or not.

In August of 2007, infectionvectors.com became aware of a good test case for what is considered “security-relevant” with a rendering issue that affects Microsoft’s Internet Explorer.² This paper examines that issue, and why it can be seen as both a harmless, possibly annoying glitch, or as a security concern, depending on one’s point of view. The “security relevance” of any issue can be debated, as can its ownership. Both of these qualities affect the mitigation strategy of an organization, meaning they can cost a great deal of money and should be considered as part of any thorough security policy.

&apos

A character that is often escaped or encoded for obvious reasons when part of SQL-supported applications, the apostrophe is used by many people without injection attacks on their minds. When fed back into dynamically created URLs, applications generally encode the apostrophe (for both security and functionality reasons – one would not want executable code or markup that would alter the look of a page from entering a document).

XML (and XHTML) require all attributes to be enclosed in quotation marks, regardless of the content specified.³ There are certain situations where an HREF anchor does not require double quotes (where only letter and numbers are used), but the bulk of “best practices” literature would support using the quotes for everything.⁴ To say, then, that an error caused by coding one’s HTML out of compliance with the standards is the fault of the browser would seem ridiculous. However, is this always the case, is this how it is perceived in the marketplace?

When any application encounters unexpected input, its reaction is of special interest to security researchers. Indeed, there is little that interests many vulnerability researchers more. That is the very nature of fuzzing, an advanced means of testing program limits.

Browsers are often put through fuzzing and related stress tests. On a much simpler plane, however, consider the lines of HTML below:

```
<html>
<head>
<title>SINGLE_DOUBLE</title>
<body>
<a href=http://www.testurl.tld/images/sl='quote>Pictures</a></div>
TEST LINK GOES HERE: <a
href=http://www.infectionvectors.com/'quote2>Click here</a> to visit to
the site immediately.
</body>
</html>
```

The unquoted `<a href>` is trouble; it is not compliant with the HTML specifications. But, one might not expect that the URL from the first HREF is associated with the tag in the second and that all content in between is missing. The result of this code is shown below:



The mouse pointer is hovering over the “Click here,” and shows the link in the bottom left corner. This issue was seen in IE 6 & 7 and was not observed with Firefox or Opera iterations that were available. But that doesn’t make it a problem with Internet Explorer, there is no agreed upon means of handling non-compliant HTML. Every poor coding technique cannot be considered an issue with the rendering agent – to be sure, every time a XSS bug is reported, it does not come back to Microsoft for correction, that action is incumbent on the site owner.

Rendered

After tests of all varieties with IE 6, we found that non-encoded single instances of the apostrophe in a URL caused some odd behavior.⁵ When delivered as part of a URL, it causes IE to mismatch URLs with their tags. But, how does one find a site to output a single, unencoded quotation mark? The first thought that came to mind was the use of “autocrafted” URLs within search engine results to feed one’s search criteria to other parts of the site. For example, if one searches for “panda bears” through MSN’s search

page, when the results are displayed there will be a link for “Images” that has the search string set with the argument “panda bears” for the image search application.

That being the case, we used the big search engines as tests. MSN Search Live uses double quotes around everything, so when IE displays it, everything works as expected. For each of the tests, we used the string, “‘single quote’.” For clarity, that is, apostrophe, the word single, the word quote, and then a double quote (not double apostrophes, which are tested later). How about Google? The same string produced this “Images” link prior to September 2007⁶:

```
<div class=gb1><a href=http://images.google.com/images?hl=en&q='single+quote%22&um=1&sa=N&tab=wi>Images</a></div>
```

No quotation marks around the string. How does this look in IE? Not great. The links (including those for the bar placed at the top of the search results, with the links to related searches in “Images,” “Maps,” etc.) are swallowed by the next HREF on the page. This behavior is observed for any set of links where the following HREF contains one apostrophe (unencoded and not enclosed in quotation marks). In the case of Google’s results, the “Images” link was displayed for the “Video” tag (the word “Images” was not visible at all).

The point of this is in no way meant to say that Google has a problem (outside of being out of compliance with specifications) but to show that there are very large web spaces that may employ HTML that asks IE to render unencoded, unquoted HREFs. By the same token, it is not an indictment of IE, there is no reason to find fault exclusively (if at all) with Microsoft. Again, that is a question that the authors believe can be answered many ways and likely has a contextual element for each organization considering it.

Trophy

So there is a peculiarity with a browser rendering non-standard HTML, is there any reason to even consider whether this is security related? Certainly, it is a bit of a stretch, but with a little massaging and possibly on a different type of page altogether, it may be fairly simple to make an illegitimate link look “trusted.” For instance, one could make a malicious site appear to be “trusted” - i.e., part of the search engine (in the case of Google, put search result 1, 11, 21, etc. in the “more” drop down menu). As an example, with a non-malicious site, of course:

Google “single” (without the double quotes) and see that a singles meeting site appears in the “more” menu (or, prove that it works for the 11th site in the results by sending: <http://www.google.com/search?q=%27single&hl=en&start=10&sa=N> prior to Google’s correction). It seems that there are a number of other sites, however, that return unquoted HREFs, which is probably no surprise to web security professionals. Is the issue exploitable? Whose issue is it for correction? Unlike XSS, there is some rendering problem with IE, it is doing something the site owner would not expect to be done with HTML which works on other browsers.

There are four distinct possibilities (at least) that are possible results of this discussion. The issue can be considered:

	Security	Non-security
Site Owner		
Microsoft		

This allows for 4 discrete answers:

- 1) A security problem caused by the site owner
- 2) A non-security related problem caused by the site owner
- 3) A security problem caused by Microsoft
- 4) A non-security related problem caused by Microsoft

But, in discussions with security practitioners, anecdotal evidence suggest that some would argue that one could place their “x” on one or more of the lines, meaning the answer could be the site owner and Microsoft share responsibility.

Even with the addition of permutations of multiple agents and degrees of “security relevance,” this represents much of the current thinking on security issues. Is this, at its heart, a false dichotomy of responsible agents versus the security/non-security qualifier? The number of distinct categories for classifying software issues is certainly wider than just security-related or not-security-related; performance, usability, and enhancement are all possible qualifiers. Do these help provide context for a security professional though?

In the example above, imagine how various organizations may respond to having the issue classified as security, performance, bug, or non-issue altogether (by all parties). Is the reaction guided by policy or by the context one has relative to the affected products?

And Yours

If security relevance is subjective, how does that affect responsible disclosure? All bugs would become protected information. That may not be good news for software development as a whole. It would imply that anything that could be exploited by a criminal is a security-related issue.

XSS shows that the security boundary is sometimes hazy – many bugs that a site owner considers trivial are in fact quite serious. The answer appears to be yes, but only until they are wrapped into an exploit. Within any permutation of applications, network ecosystem, and risk acceptance model (many of which are exceptionally complex) there is a chance that any bug is security related.

In the next report, we intend to examine how and when “security relevance” is identified in security policies.

References

0. The use of the apostrophe in identifying possessiveness (i.e., Jim's hat) is the reference for the murky title, which is also intended to be seen as "IE's" (I E apostrophe "s"), maybe not so witty on second thought. See note 2 for clarification.

1. Gordon, Jason. "Non-Security Related: Classifying Fixes." March 2006, infectionvectors.com. http://www.infectionvectors.com/library/notsecurity_iv.pdf.

2. Although IE 6 and 7 were the only browsers found to exhibit the behavior described here, the authors do not intend to label this a security problem with Internet Explorer – on the contrary, that is for the reader to decide. Determining what is "security related," and what is not is an exercise the authors suggest is something quite subjective – and must be defined in a security policy much like the acceptable use, disaster recovery, and enforcement clauses.

3. W3C, updated 29 September 2006, "Extensible Markup Language (XML) 1.0 (Fourth Edition)." <http://www.w3.org/TR/REC-xml/>.

4. The exceptions are in the specification, <http://www.w3.org/TR/REC-html40/intro/sgmltut.html#h-3.2.2>. One source that recommends always using the quotationmarks: Michael Hamm, "HTML Tutorial" Washington University in St. Louis. <http://www.math.wustl.edu/~msh210/html.html>.

5. If you are so inclined, try it out with code similar to the following:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>SINGLE_DOUBLE</title>
<body>
<a
href=http://pictures.test.tld/images?hl=en&q='single+quote%22&um=1&sa=N
&tab=wi>Images</a></div>
TEST LINK GOES HERE: Click <a
href=http://www.infectionvectors.com'>here</a> to visit to the site
immediately.
</body>
</html>
```

6. This paper was not released until after Google changed affected code, sometime around the middle of September 2007. The examples using the domain google.com should not produce unusual results (unless the code is changed back to employ unquoted hrefs sometime after this publication, that is). The use of Google as the sample site is meant only to point out that these types of issues can affect any site, even the most popular domain on the web (reference: NetCraft site rankings report: http://toolbar.netcraft.com/site_report?url=http://www.google.com).

Copyright © 2007 infectionvectors.com. All rights reserved.