# CWE Detail – CWE-82

## Description

The web application does not neutralize or incorrectly neutralizes scripting elements within attributes of HTML IMG tags, such as the src attribute.

## Extended Description

Attackers can embed XSS exploits into the values for IMG attributes (e.g. SRC) that is streamed and then executed in a victim's browser. Note that when the page is loaded into a user's browsers, the exploit will automatically execute.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2006-3211: Stored XSS in a guestbook application using a javascript: URI in a bbcode img tag.

**•** CVE-2002-1649: javascript URI scheme in IMG tag.

**•** CVE-2002-1803: javascript URI scheme in IMG tag.

**•** CVE-2002-1804: javascript URI scheme in IMG tag.

**•** CVE-2002-1805: javascript URI scheme in IMG tag.

**•** CVE-2002-1806: javascript URI scheme in IMG tag.

**•** CVE-2002-1807: javascript URI scheme in IMG tag.

**•** CVE-2002-1808: javascript URI scheme in IMG tag.

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Read Application Data, Execute Unauthorized Code or Commands — Notes:

## Potential Mitigations

**•** Implementation: Use and specify an output encoding that can be handled by the downstream component that is reading the output. Common encodings include ISO-8859-1, UTF-7, and UTF-8. When an encoding is not specified, a downstream component may choose a different encoding, either by assuming a default encoding or automatically inferring which encoding is being used, which can be erroneous. When the encodings are inconsistent, the downstream component might treat some character or byte sequences as special, even if they are not special in the original encoding. Attackers might then be able to exploit this discrepancy and conduct injection attacks; they even might be able to bypass protection mechanisms that assume the original encoding is also being used by the downstream component. The problem of inconsistent output encodings often arises in web pages. If an encoding is not specified in an HTTP header, web browsers often guess about which encoding is being used. This can open up the browser to subtle XSS attacks. (Effectiveness: N/A)

**•** Implementation: To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set. (Effectiveness: Defense in Depth)

## Applicable Platforms

**•** None (Class: Not Language-Specific, Prevalence: Undetermined)