

Set-Cookie Kung-Fu

"bypass path restrictions"

Ivan Bütler – 17. Sept. 2009



Path is Not a Security Boundary

In addition to the name and value attributes, the server can specify several attributes for a cookie which affect how the browser will use it. Attributes can affect how long the browser stores the cookie, if it persistently stores the cookie at all, whether the browser will send it over any connection or only over HTTPS connections, and what server hostnames or **Path** it will send the cookie back to, among other things. Do you think it is safe to run two different applications on the same hostname, where the first cookie is bounded to <http://www.csnc.ch/public> and the second cookie linked to <http://www.csnc.ch/private>?

It is a common understanding that a malicious web app on <http://www.csnc.ch/public> cannot read out the <http://www.csnc.ch/private> cookie. Does this mean we can happily share **public** and **private** web apps on the same hostname? Hey dude, this is not true!

While it might seem that **Path** should be a security boundary — i.e. that applications accessed via different paths on the same server should not be able to set or get each other's cookies— in fact it is not. The reason is that the same-origin policy applies only to the domain name, not to the path. Although **document.cookie** in the context of a page from either application will not contain the other's cookies, a malicious application can craft a page that contains an iframe which in turn contains a page from the other application. Script in the enclosing page can then access the other application's cookie. The browser allows this because both applications are in the same domain.

Example iframe snippet from <http://www.csnc.ch/public/somepage.html>

```
<script>

function bru() {
    var cookies = document.getElementById('f').cookie;
    alert(cookies);
}

</script>

<iframe id="f" onload="bru()" style="display:none"
src="http://www.csnc.ch/private/anywhere.html" >
</iframe>
```

The cookie attribute "HttpOnly" can help provide some marginal defense against cross-site scripting (XSS) attacks. It is recommended that you use it where possible (i.e. whenever the client-side portion of the application does not need read access to the cookie), but that you don't depend on it as your sole means of XSS defense. When an XSS vulnerability is present in your application, an attacker can still do plenty of damage without reading a user's session cookie.

Thank You

Thanks to Cyrill Brunschwiler from Compass Security AG for coming up with this issue.

Ivan Bütler, E1
Compass Security AG

ivan.buetler@csnc.ch

Thanks to Chris Palmer

Chris Palmer <chris@isecpartners.com>
<https://www.isecpartners.com/files/web-session-management.pdf>