9-1-2013

Finding Information Leaks in JavaScript

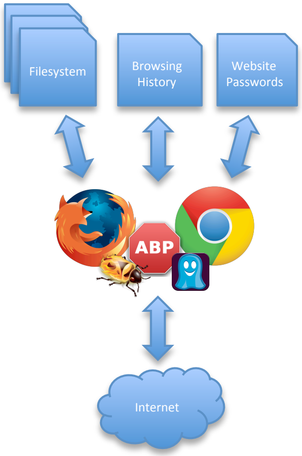
Thomas Ashmore
Harvey Mudd College

HARVEY MUDD COLLEGE

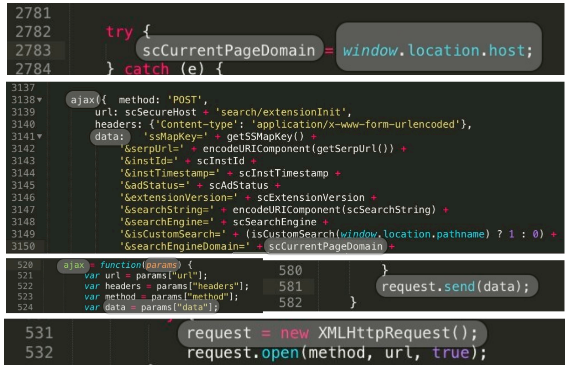
Finding Information Leaks in Javascript

Tommy Ashmore

Can browser add-ons leak your personal information?



Yes, some do!



```
    try {
        scCurrentPageDomain = window.location.host;
    } catch (e) {
```

```
    ajax({  method: 'POST',
        url: scSecureHost + 'search/extensionInit',
        headers: {'Content-type': 'application/x-www-form-urlencoded'},
        data:   'ssMapKey=' + getSSMapKey() +
            '&serpUrl=' + encodeURIComponent(getSerpUrl()) +
            '&instId=' + scInstId +
            '&instTimestamp=' + scInstTimestamp +
            '&adStatus=' + scAdStatus +
            '&extensionVersion=' + scExtensionVersion +
            '&searchString=' + encodeURIComponent(scSearchString) +
            '&searchEngine=' + scSearchEngine +
            '&isCustomSearch=' + (isCustomSearch(window.location.pathname) ? 1 : 0) +
            '&searchEngineDomain=' + scCurrentPageDomain +
```

```
    ajax = function(params) {
        var url = params["url"];
        var headers = params["headers"];
        var method = params["method"];
        var data = params["data"];
```

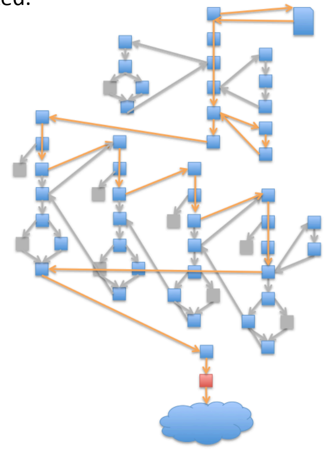
```
            }
            request.send(data);
        }
```

```
            request = new XMLHttpRequest();
            request.open(method, url, true);
```

How can we detect information leaks?

Currently, Mozilla volunteers manually inspect the source of popular addons for leaks. Our tool helps automate this process.

We create a graph of information flow between program statements. By tracing backwards from a call that sends data to the internet, we can determine whether sensitive information may be leaked.

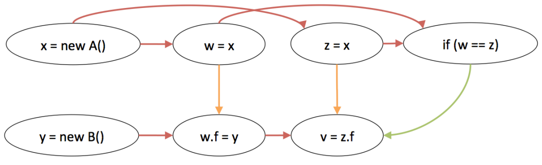


The analysis requires us to approximate the program's behavior. We use abstract interpretation to determine when two pointers could reference the same memory location.

	Read	Write
x = new A()		x
z = x	x	z
y = new B()		y
w = x	x	w
w.f = y	w, y	o.f
if (w == z) {	w, z	
v = z.f	z, o.f	v
}		

Can the tool be improved?

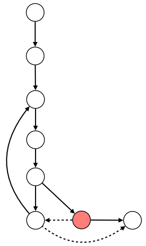
Our analysis often reports potential leaks where none exist. We can help users identify false positives by classifying different types of data dependence.



Exceptions and goto statements alter the order of statement execution. It's tricky to determine how they affect data flow. Our current method could be improved.

```
i = 10
sum = 0

while (true) {
    sum += i
    if (i == 0)
        break
    i--
}
print(sum)
```



Acknowledgements & Citations

Thanks to Vineeth Kashyap and the rest of Ben Hardekopf's team at UCSB!

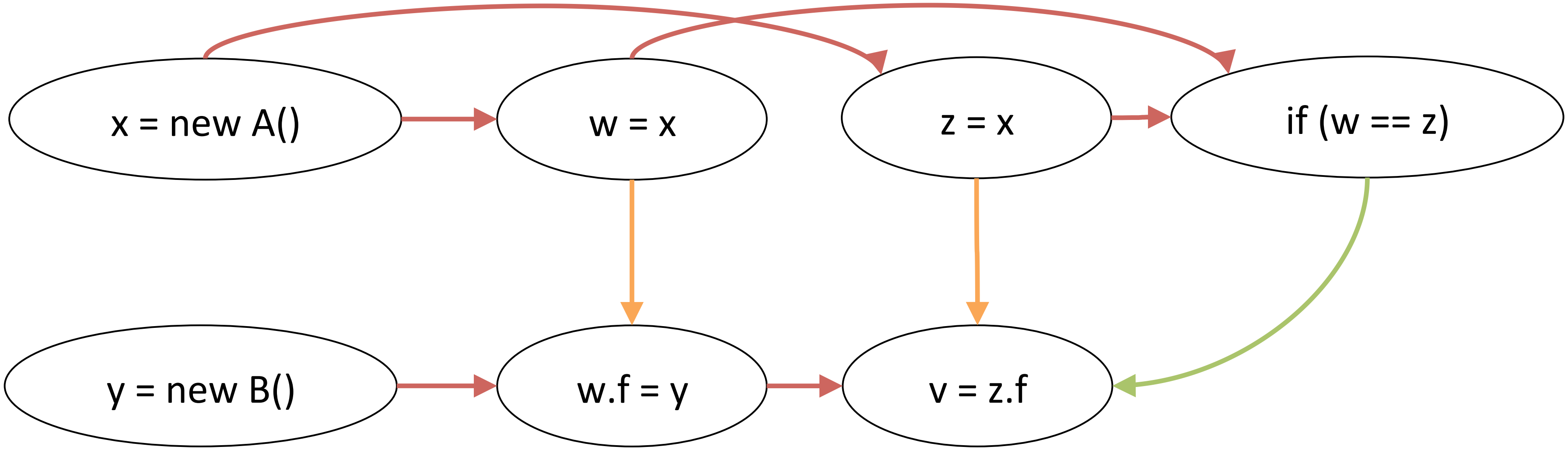
Ball, Thomas, and Susan Horwitz. “Slicing programs with arbitrary control-flow.” Springer Berlin Heidelberg, 1993.

Sridharan, Manu et al. "Thin slicing." *ACM SIGPLAN Notices* 42.6 (2007): 112-122.

Tip, Frank. "A survey of program slicing techniques." *Journal of programming languages* 3.3 (1995): 121-189.

All images belong to T. Ashmore and V. Kashyap, with the exception of company logos. The FireFox logo, Chrome logo, SurfCanyon logo, AdBlock Plus logo, and Ghostery logo belong to Mozilla Foundation, Google, Surf Canyon, Imprint, and Evidon, respectively.

Advisor: Ben Wiedermann



x = new A()
w = x
z = x
if (w == z)
y = new B()
w.f = y
v = z.f