black hat

USA 2015

DOM FLOW
UNTANGLING THE DOM FOR EASY BUGS

#whoami

Ahamed Nafeez (@skeptic_fx)

Security Engineer with interest in browsers

Speaker at BlackHat Asia, Hack In The Box, nullc0n, c0c0n.

Overview

Modern web apps and their problems w.r.t pen tests

Hookish! tool and how it works

Dom Flow and its techniques

Few JavaScript / DOM nuances and how to catch them

Today's web apps

Today’s state

Classic XSS is already fading away

Static analysis is becoming harder for client side JS code

Frameworks are getting more complex (JSX?)

DOM XSS / Javascript injection

XSS triggered due to client side code (Mostly..)

Most generic class of vulnerability on browser.

Sources - Entry point for untrusted data

Sinks - Executes untrusted data

The hello world of DOM XSS

https://damnvulnerable.me/domxss/location_hash_to_window_eval#firstName

```
var hash = document.location.hash //source

firstName=hash.slice(1)

document.write(firstName) //sink
```

Common Sources / Sinks

Sources

Sinks

	URL	Cookie	referrer	name	postMessage	WebStorage	Total
HTML	1356796	1535299	240341	35466	35103	16387	3219392
JavaScript	22962	359962	511	617743	448311	279383	1728872
URL	3798228	2556709	313617	83218	18919	28052	6798743
Cookie	220300	10227050	25062	1328634	2554	5618	**11809218**
WebStorage	41739	65772	1586	434	194	105440	215165
postMessage	451170	77202	696	45220	11053	117575	702916
Total	**5891195**	14821994	581813	2110715	516134	552455	24474306

25 Million Flows Later - Large-scale Detection of DOM-based XSS (2013)

Sebastian Lekies, Ben Stock, Martin Johns

String into Code

Everyone(Frameworks, Developers, . .) use ‘**strings**’ in a way that directly or indirectly turns into code

The DOM specification is rich in doing that

Direct

eval()

setTimeout

Function(x)()

execScript(x)

Indirect

jQuery’s - $(x)

document.write

Element.setAttribute(x)

Element.innerHTML=x

jQuery - $(x)

$(‘#id’), $(‘.class’), $(‘a’) - Acts as a query selector

$(‘<img src=“1.png”>’) - Creates a new IMG element

So why is it hard to pen test them?

Usually they look like this!

```
cc=function(a){if(a&&"number"==typeof a.length){if(_.bc(a))return"function"==typeof a.item||"string"==typeof a.item;if(_.fa(a))return"function"==typeof a.item}return!1};dc=
{cellpadding:"cellPadding",cellspacing:"cellSpacing",colspan:"colSpan",frameborder:"frameBorder",height:"height",maxlength:"maxLength",role:"role",rowspan:"rowSpan",type:"type"
emap:"useMap",valign:"vAlign",width:"width"};ec="constructor hasOwnProperty isPrototypeOf propertyIsEnumerable toLocaleString toString valueOf".split(" ");
_.fc=function(a,c,d){for(var e in a)c.call(d,a[e],e,a)};_.gc=function(a){var c=a.length;if(0<c){for(var d=Array(c),e=0;e<c;e++)d[e]=a[e];return d}return[]};_.hc=function(a,c)
{return 0<=(0,_.qa)(a,c)};_.ic=function(a){var c=_.da(a);return"array"==c||"object"==c&&"number"==typeof a.length};jc=function(a,c,d){function e(d){d&&c.appendChild(_.u(d)?
a.createTextNode(d):d)}for(var f=2;f<d.length;f++){var g=d[f];!_.ic(g)||_.bc(g)&&0<g.nodeType?e(g):(0,_.ra)(cc(g)?_.gc(g):g,e)}};
_.kc=function(a,c){_.fc(c,function(c,e){"style"==e?a.style.cssText=c:"class"==e?a.className=c:"for"==e?a.htmlFor=c:dc.hasOwnProperty(e)?
a.setAttribute(dc[e],c):0==e.lastIndexOf("aria-",0)||0==e.lastIndexOf("data-",0)?a.setAttribute(e,c):a[e]=c})};_.lc=function(a,c){for(var d,e,f=1;f<arguments.length;f++)
{e=arguments[f];for(d in e)a[d]=e[d];for(var g=0;g<ec.length;g++)d=ec[g],Object.prototype.hasOwnProperty.call(e,d)&&(a[d]=e[d])}};_.mc=function(a){return"number"==typeof a};
_.nc=function(a,c){var d=c[0],e=c[1];if(!_.Lb&&e&&(e.name||e.type)){d=["<",d];e.name&&d.push(' name="',_.ac(e.name),'"');if(e.type){d.push(' type="',_.ac(e.type),'"');var f=
{};_.lc(f,e);delete f.type;e=f}d.push(">");d=d.join("")}d=a.createElement(d);e&&(_.u(e)?d.className=e:_.ea(e)?d.className=e.join(" "):_.kc(d,e));2<c.length&&jc(a,d,c);return
d};_.oc=function(a){return/^[\s\xa0]*$/.test(a)};pc=function(a){var c=[],d=0,e;for(e in a)c[d++]=e;return c}; _.qc=function(a){var c=[],d=0,e;for(e in a)c[d++]=a[e];return
c};_.rc=function(a){return _.pa.concat.apply(_.pa,arguments)};_.sc=function(a,c){var d=(0,_.qa)(a,c),e;(e=0<=d)&&_.pa.splice.call(a,d,1);return e};_.tc=function(){};
_.uc=function(a){_.uc[" "](a);return a};_.uc[" "]=_.tc;_.vc=function(a,c){try{return _.uc(a[c]),!0}catch(d){}return!1};var
xc,yc;_.wc=!_.L||9<=_.Kb;xc=!_.L||9<=_.Kb;yc=_.L&&!_.N("9");!_.M||_.N("528");_.ub&&_.N("1.9b")||_.L&&_.N("8")||_.sb&&_.N("9.5")||_.M&&_.N("528");_.ub&&!_.N("8")||_.L&&_.N("9");
c=function(a,c){this.type=a;this.o=this.target=c;this.C=!1;this.Re=!0};_.zc.prototype.stopPropagation=function(){this.C=!0};_.zc.prototype.preventDefault=function()
{this.Re=!1};_.Ac=_.L?"focusin":"DOMFocusIn";_.Bc=_.M?"webkitTransitionEnd":_.sb?"otransitionend":"transitionend";_.Cc=function(a,c){_.zc.call(this,a?
a.type:"");this.A=this.o=this.target=null;this.D=this.keyCode=this.button=this.clientY=this.clientX=0;this.F=this.G=this.w=this.B=!1;this.b=this.state=null;a&&this.init(a,c)};_
_.Cc,_.zc);
_.Cc.prototype.init=function(a,c){var d=this.type=a.type;this.target=a.target||a.srcElement;this.o=c;var e=a.relatedTarget;e?_.ub&&(_.vc(e,"nodeName")||(e=null)):"mouseover"==c
e=a.fromElement:"mouseout"==d&&(e=a.toElement);this.A=e;this.clientX=void 0!==a.clientX?a.clientX:a.pageX;this.clientY=void 0!==a.clientY?
a.clientY:a.pageY;this.button=a.button;this.keyCode=a.keyCode||0;this.D=a.charCode||("keypress"==d?
a.keyCode:0);this.B=a.ctrlKey;this.w=a.altKey;this.G=a.shiftKey;this.F=a.metaKey;this.state=
a.state;this.b=a;a.defaultPrevented&&this.preventDefault()};_.Cc.prototype.stopPropagation=function(){_.Cc.H.stopPropagation.call(this);this.b.stopPropagation?
this.b.stopPropagation():this.b.cancelBubble=!0};_.Cc.prototype.preventDefault=function(){_.Cc.H.preventDefault.call(this);var a=this.b;if(a.preventDefault)a.preventDefault();e
if(a.returnValue=!1,yc)try{if(a.ctrlKey||112<=a.keyCode&&123>=a.keyCode)a.keyCode=-1}catch(c){}};_.Cc.prototype.U=function(){return this.b};
var Fc;_.Dc="closure_listenable_"+(1E6*Math.random()|0);_.Ec=function(a){return!(!a||!a[_.Dc])};Fc=0;var Gc=function(a,c,d,e,f)
{this.listener=a;this.b=null;this.src=c;this.type=d;this.ic=!!e;this.Cc=f;this.key=++Fc;this.Hb=this.hc=!1},Hc=function(a)
{a.Hb=!0;a.listener=null;a.b=null;a.src=null;a.Cc=null};var Ic=function(a){this.src=a;this.b={};this.o=0},Kc,Jc;Ic.prototype.add=function(a,c,d,e,f){var
g=a.toString();a=this.b[g];a||(a=this.b[g]=[],this.o++);var h=Jc(a,c,e,f);-1<h?(c=a[h],d||(c.hc=!1)):(c=new Gc(c,this.src,g,!!e,f),c.hc=d,a.push(c));return
c};Ic.prototype.remove=function(a,c,d,e){a=a.toString();if(!(a in this.b))return!1;var f=this.b[a];c=Jc(f,c,d,e);return-1<c?(Hc(f[c]),_.pa.splice.call(f,c,1),0==f.length&&(dele
this.b[a],this.o--),!0):!1};
Kc=function(a,c){var d=c.type;if(!(d in a.b))return!1;var e=_.sc(a.b[d],c);e&&(Hc(c),0==a.b[d].length&&(delete a.b[d],a.o--));return e};_.Lc=function(a,c,d,e,f)
{a=a.b[c.toString()];c=-1;a&&(c=Jc(a,d,e,f));return-1<c?a[c]:null};Jc=function(a,c,d,e){for(var f=0;f<a.length;++f){var g=a[f];if(!g.Hb&&g.listener==c&&g.ic==!!d&&g.Cc==e)retur
f}return-1};
var Mc,Nc,Qc,Sc,Uc,Vc,ad,$c,Wc,bd;Mc="closure_lm_"+(1E6*Math.random()|0);Nc={};Qc=0;_.P=function(a,c,d,e,f){if(_.ea(c)){for(var g=0;g<c.length;g++)_.P(a,c[g],d,e,f);return
null}d=_.Rc(d);return _.Ec(a)?a.$a(c,d,e,f):Sc(a,c,d,!1,e,f)};
Sc=function(a,c,d,e,f,g){if(!c)throw Error("x");var h=!!f,l=_.Tc(a);l||(a[Mc]=l=new Ic(a));d=l.add(c,d,e,f,g);if(d.b)return
d;e=Uc();d.b=e;e.src=a;e.listener=d;if(a.addEventListener)a.addEventListener(c.toString(),e,h);else if(a.attachEvent)a.attachEvent(Vc(c.toString()),e);else throw
Error("y");Qc++;return d};Uc=function(){var a=Wc,c=xc?function(d){return a.call(c.src,c.listener,d)}:function(d){d=a.call(c.src,c.listener,d);if(!d)return d};return c};
_.Xc=function(a,c,d,e,f){if(_.ea(c)){for(var g=0;g<c.length;g++)_.Xc(a,c[g],d,e,f);return null}d=_.Rc(d);return _.Ec(a)?a.xc(c,d,e,f):Sc(a,c,d,!0,e,f)};_.Yc=function(a,c,d,e,f
{if(_.ea(c))for(var g=0;g<c.length;g++)_.Yc(a,c[g],d,e,f);else d=_.Rc(d),_.Ec(a)?a.gd(c,d,e,f):a&&(a=_.Tc(a))&&(c=_.Lc(a,c,d,!!e,f))&&_.Zc(c)};
_.Zc=function(a){if(_.mc(a)||!a||a.Hb)return!1;var c=a.src;if(_.Ec(c))return c.fc(a);var d=a.type,e=a.b;c.removeEventListener?
c.removeEventListener(d,e,a.ic):c.detachEvent&&c.detachEvent(Vc(d),e);Qc--;(d=_.Tc(c))?(Kc(d,a),0==d.o&&(d.src=null,c[Mc]=null)):Hc(a);return!0};Vc=function(a){return a in Nc?
Nc[a]:Nc[a]="on"+a};ad=function(a,c,d,e){var f=!0;if(a=_.Tc(a))if(c=a.b[c.toString()])for(c=c.concat(),a=0;a<c.length;a++){var g=c[a];g&&g.ic==d&&!g.Hb&&
(e=fa(g,e),f=f&&!1!==e)}return f};
```

Existing tools (DOM XSS)

Dominator Pro - Dynamic taint tracking using Firefox.

Plethora of static analysis tools - Regex pattern match, Parse JS code and analyse.

What can we look for?

All cases of DOM Injection

DOM XSS / Javascript injection

DOM based open redirection

Second order DOM injection (XHR, WebSocket)

WebStorage manipulation

Quirky DOM behaviour

Globally exposed variables in the DOM

DOM Clobbering

Usage of certain methods which could have unforeseen security implications

damnvulnerable.me

DamnVulnerable.me is a webapp that is deliberately vulnerable to DOM based attacks.

Its goal is to provide a platform to learn, test and practice DOM based bugs and other exotic cases.

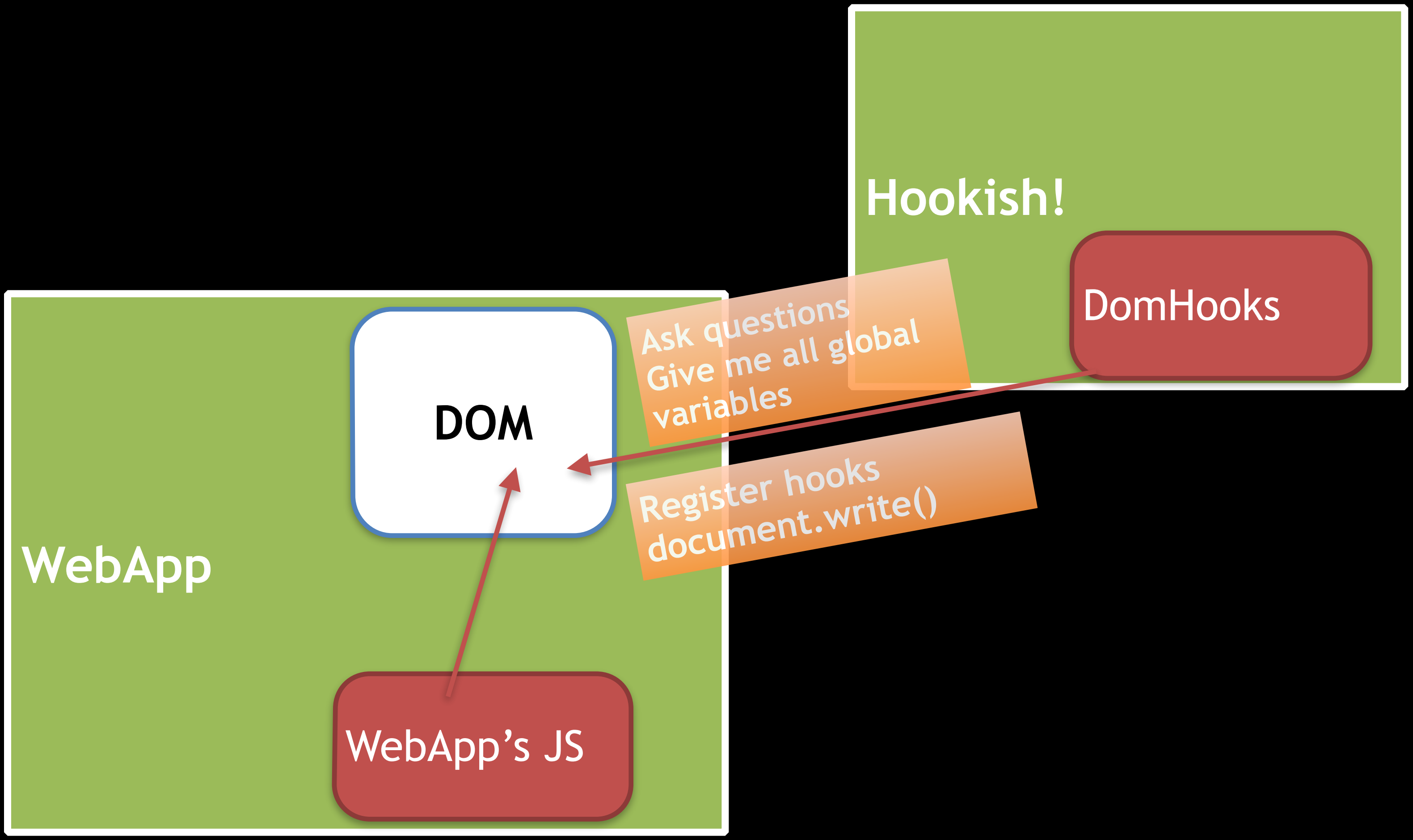
How Hookish works

Inject DomHooks for sources and sinks

Wait for page to load

Track all sources and sinks

Injecting DomHooks



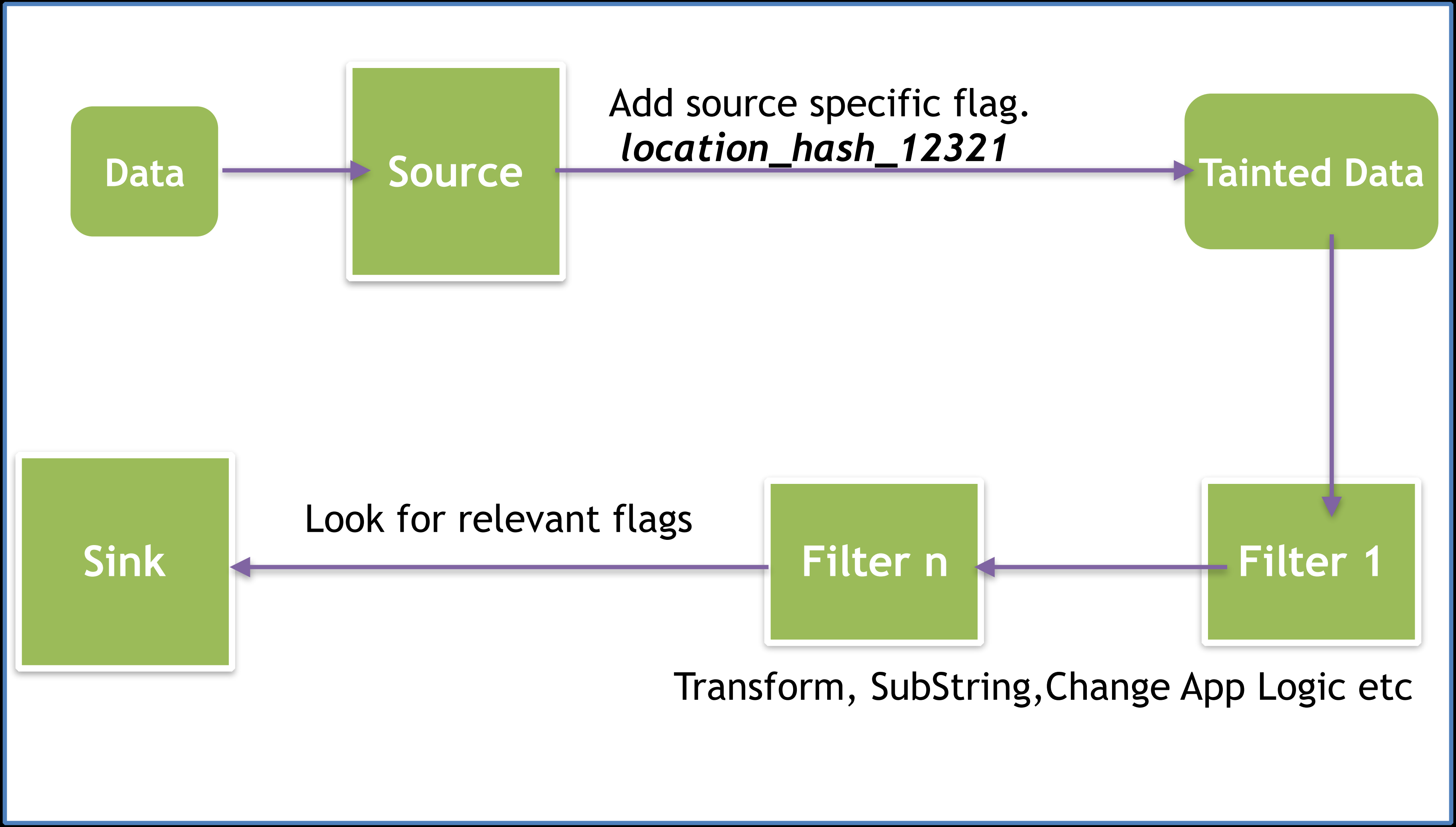
domhooks.js

Standalone library which selectively registers required DOM properties & methods.

https://github.com/skepticfx/hookish/blob/master/src/js/domHooks.js

Can be used in other tools for performance analysis, hardening DOM, DOM based IDS etc.

DomFlow



Data

Source

Add source specific flag.
location_hash_12321

Tainted Data

Filter 1

Filter n

Look for relevant flags

Sink

Transform, SubString,Change App Logic etc

DomFlow- cookie to innerHTML

Every time a cookie is accessed, the data is tagged with a unique flag - ***doc_cookie_12391***

This data may go through various transformations.

When a registered innerHTML receives data with this tag, it marks that as a possible DOM XSS.

Overriding filters

Example: **XHR to innerHTML**

XHR responses are usually JSON content

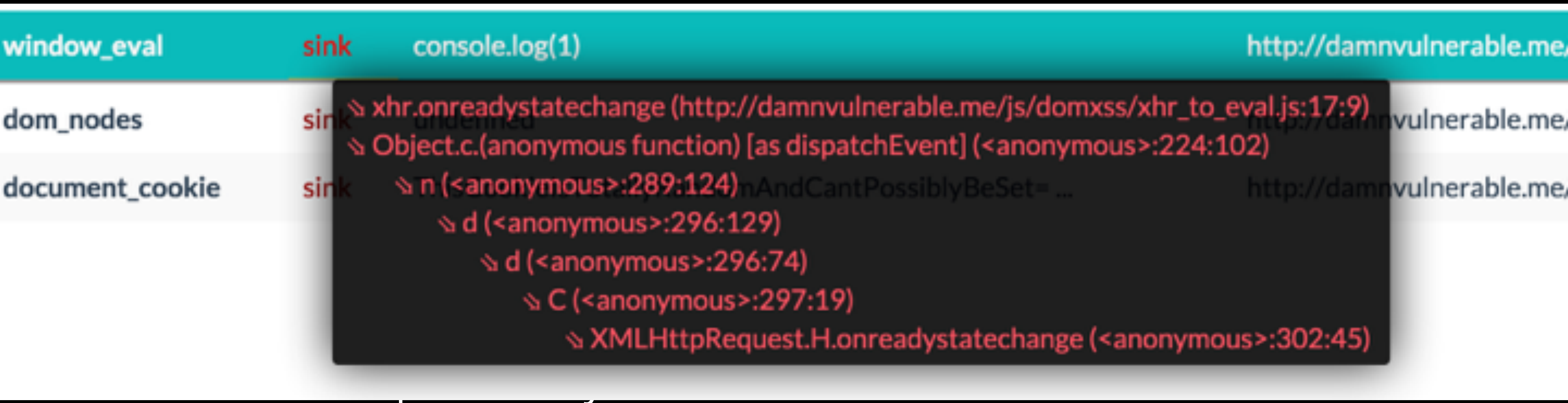
JSON.parse(**{‘data1’: ‘value1’, ‘data2’: ‘value2’}**)

Object.Stringify(**{‘data1’: ‘value1Flag’, ‘data2’: ‘value2Flag’}**)

Boxing strings in JS

```
var str = “hello”

typeof str; // string

str.flag = true;

// JS propagates this string flag in most cases
```

Navigating across the flows



window_eval	sink	console.log(1)	http://damnvulnerable.me/
dom_nodes	sink	undefined	http://damnvulnerable.me/
document_cookie	sink	mAndCantPossiblyBeSet= ...	http://damnvulnerable.me/

```
↘ xhr.onreadystatechange (http://damnvulnerable.me/js/domxss/xhr_to_eval.js:17:9)
↘ Object.c.(anonymous function) [as dispatchEvent] (<anonymous>:224:102)
    ↘ n (<anonymous>:289:124)
        ↘ d (<anonymous>:296:129)
            ↘ d (<anonymous>:296:74)
                ↘ C (<anonymous>:297:19)
                    ↘ XMLHttpRequest.H.onreadystatechange (<anonymous>:302:45)
```

Getting the stack trace in V8 Engine

Dynamically throw the error and filter to remove Hookish! specific stacks

Easily integrates with Chrome’s dev tools and helps analyse vulnerable lines of code

Tracking status of all hooks

domstorm.skepticfx.com

Hooking Storage objects,
http://domstorm.skepticfx.com/modules?id=529d4f84090faf0000000002

Second order DOM injection

DOM injection where the sources doesn’t flow directly.

Rather, they are fetched from a persistent storage at some point.

XHR/WS response flowing in to sinks

Four Scenarios

The following 4 scenarios talks about bugs/special cases that are often missed while security testing a web app.

Hookish! tool is built to easily find / analyse such bugs

1. Do you check how XHR responses are handled in your application?

Most common issue which pen testers miss / scanners usually ignore.

The choke point is how you treat these data before populating into the DOM (regardless of how you store untrusted input)

XHR response - innerHTML

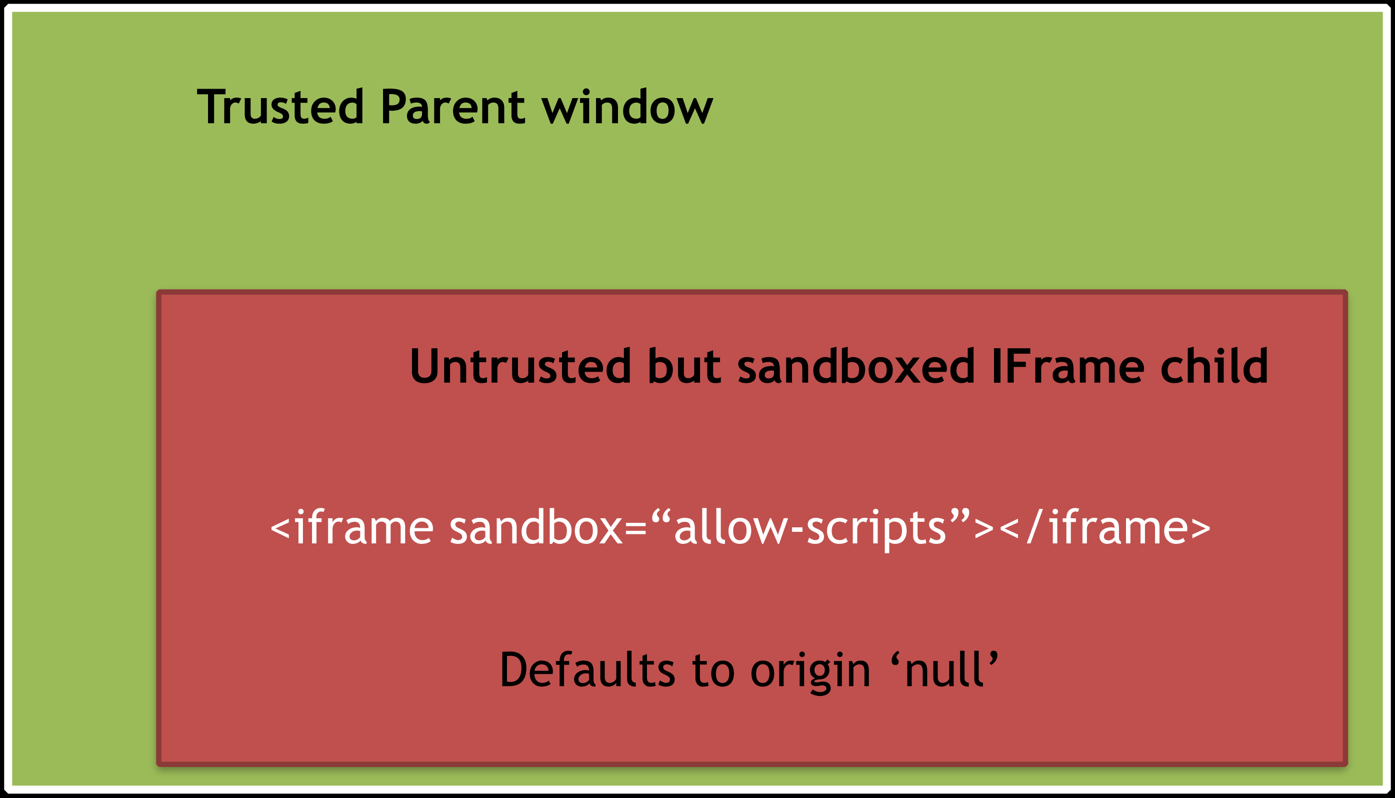
```
var response = JSON.parse(xhr.responseText);

var description = response.description;

var div = document.getElementById('vulnerableDiv');

div.innerHTML = description;
```

2. DOM Clobbering using Global Variables

Consider an IFrame sandbox which executes arbitrary code.

Exposed global variables can change logic in parent window.

Classic Iframe sandboxing



Trusted Parent window

Untrusted but sandboxed IFrame child

<iframe sandbox=“allow-scripts”></iframe>

Defaults to origin ‘null’

About this sandbox

IFrame sandboxes have ‘null’ origin.

The JS in sandboxed IFrame should not interact with the parent Window’s DOM.

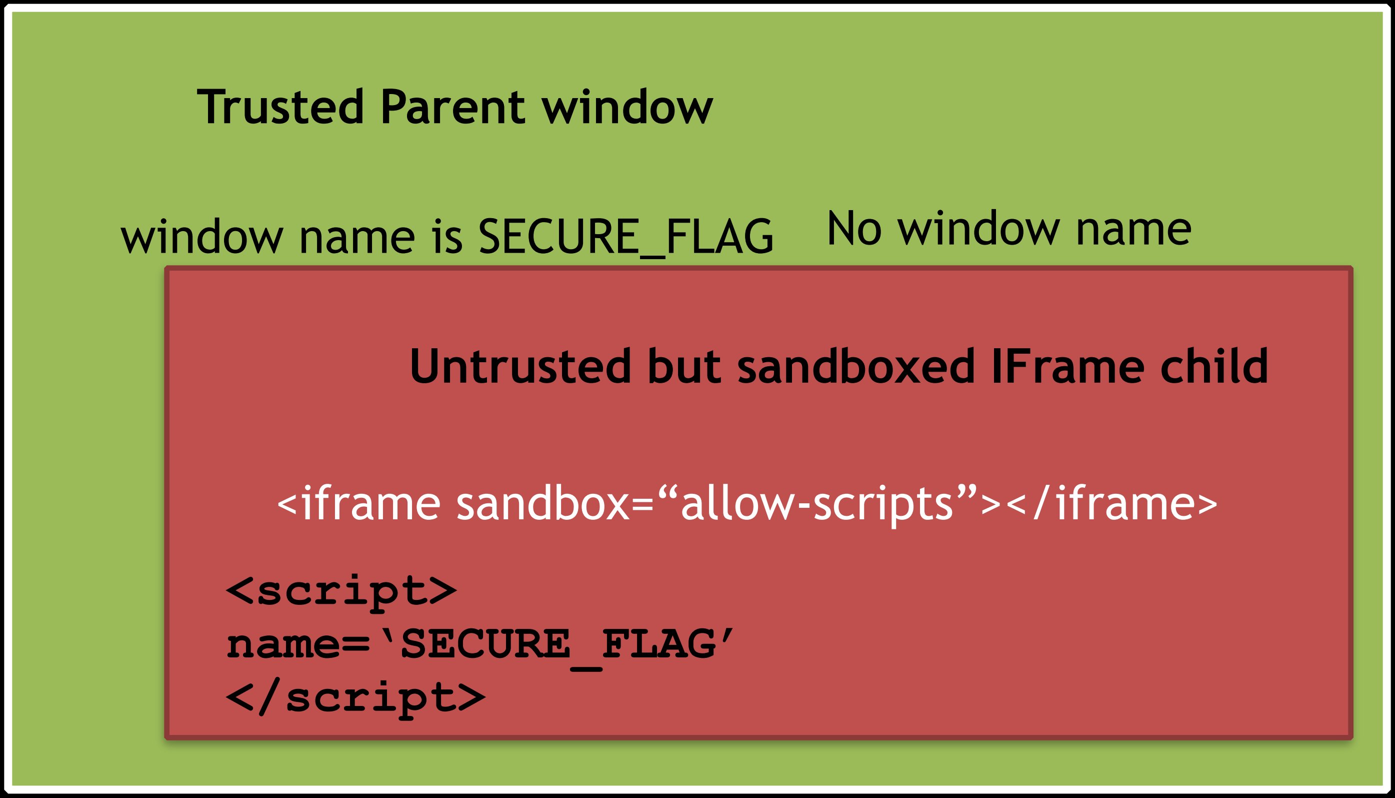
http://www.html5rocks.com/en/tutorials/security/sandboxed-iframes/

Spot the bug and break out of this sandbox

https://damnvulnerable.me/misc/insecure_global_variable

Setting global variables using window.name

DOM sets the name of iframe windows to the window object (DOM CLOBBERING)



Trusted Parent window

window name is SECURE_FLAG

No window name

Untrusted but sandboxed IFrame child

<iframe sandbox=“allow-scripts”></iframe>

```
<script>
name=‘SECURE_FLAG’
</script>
```

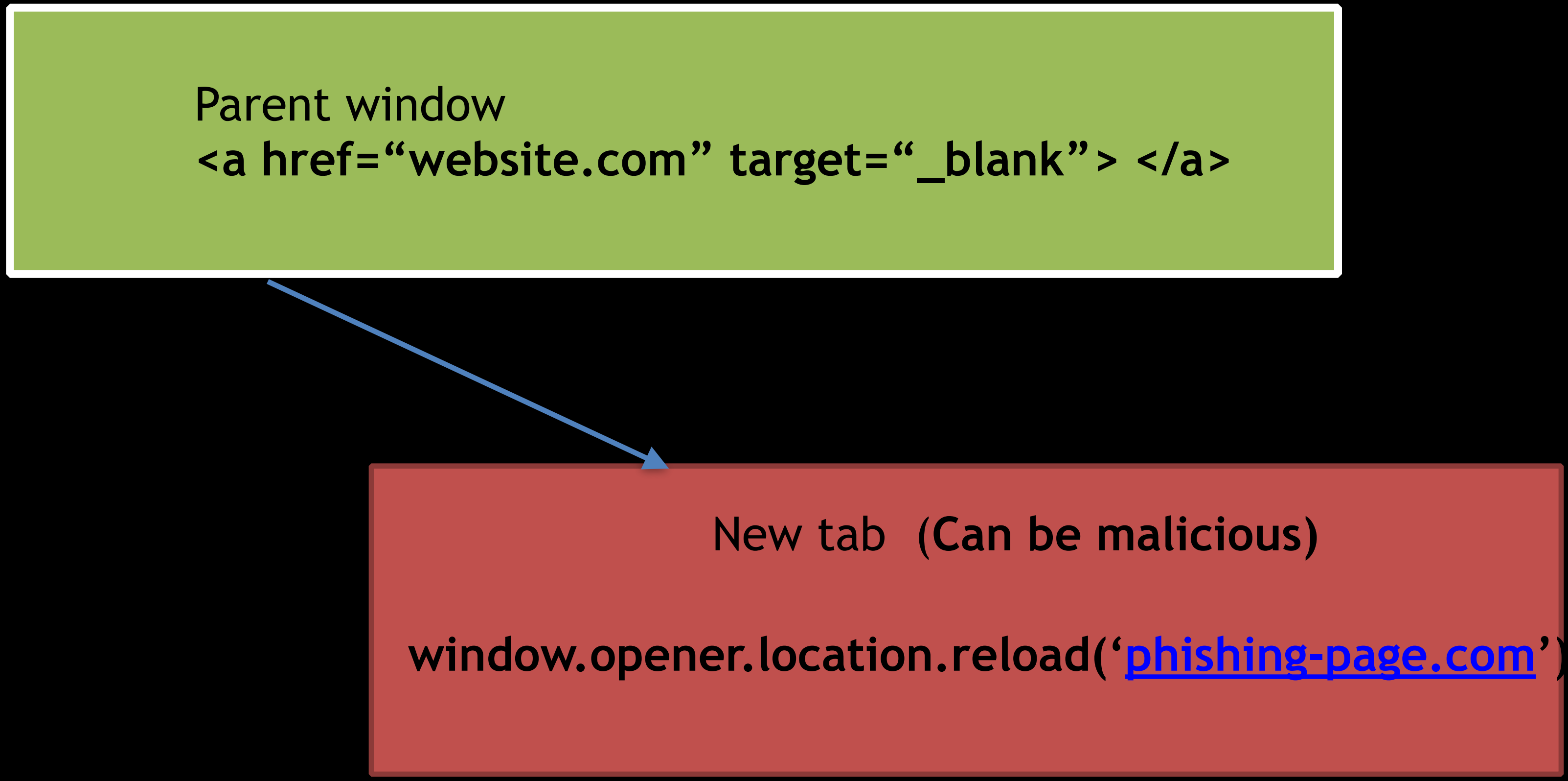
This sets the global variable SECURE_FLAG in the parent window’s DOM and bypassese the security check

3. Redirect parent window while opening links in new tab

https://hackerone.com/reports/23386

Works on Chrome and Firefox.

Opening links in new tab



Parent window
<a href=“website.com” target=“_blank”> </a>

New tab (Can be malicious)

window.opener.location.reload(‘phishing-page.com’)

window.opener should be null always and should not be accessible by another Cross-Domain window

Finding anchor tags with target=_blank

Easy to find on static HTML pages.

In modern apps, usually anchor tags are dynamically inserted in to the DOM.

Hookish! finds these after the DOM is rendered and all anchor tags are populated.

Not a serious issue most of the times, but depends on where you have these new links.

4. Custom templating engines

```
var data = {‘name’: ‘mark’, ‘age’: ‘23’}

Welcome to this page, <%- data.mark %>
```

How would some one write a templating engine using JavaScript?

1. Load the template **data** object and encode it.

2. Find the template pattern

3. Use string.replace(pattern, matching_data)

A simple templating code

```
var inputHTML = "<img src='PLACEHOLDER'>";

function doTemplating(){

  var input = document.getElementById('id_input').value;

  input = filterInput(input);

  var finalHTML = inputHTML.replace("PLACEHOLDER", input);

  console.log(finalHTML);

  document.write('Your input: </br>' + input);

  document.write(finalHTML);

}
```

The bypass

```
$` onerror=alert(1);//
```

String.prototype.replace

ECMAScript’s String.replace is the culprit

http://www.ecma-international.org/ecma-262/5.1/#sec-15.5.4.11

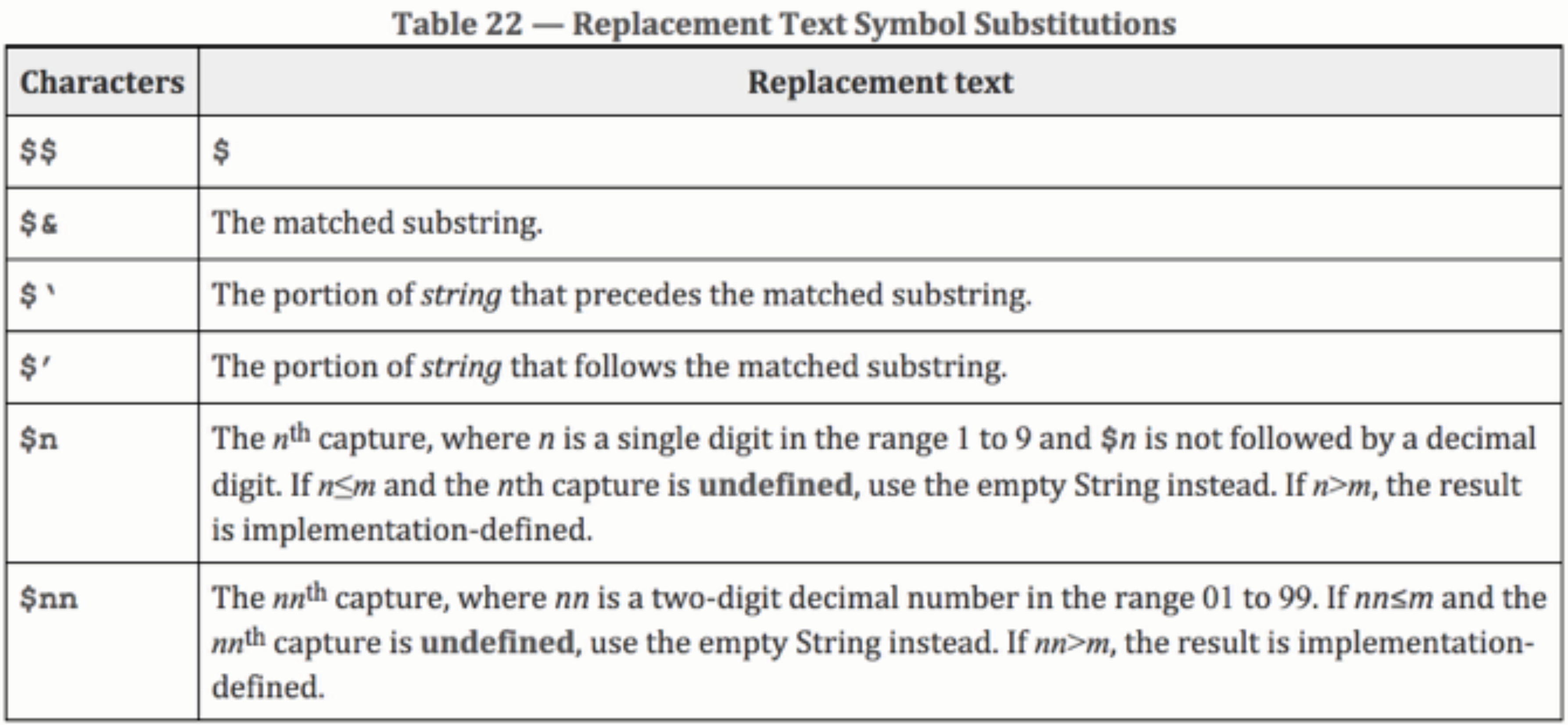


Table 22 — Replacement Text Symbol Substitutions

Characters	Replacement text
$$	$
$&	The matched substring.
$`	The portion of *string* that precedes the matched substring.
$'	The portion of *string* that follows the matched substring.
$n	The *n*th capture, where *n* is a single digit in the range 1 to 9 and $*n* is not followed by a decimal digit. If *n*≤*m* and the *n*th capture is **undefined**, use the empty String instead. If *n*>*m*, the result is implementation-defined.
$nn	The *nn*th capture, where *nn* is a two-digit decimal number in the range 01 to 99. If *nn*≤*m* and the *nn*th capture is **undefined**, use the empty String instead. If *nn*>*m*, the result is implementation-defined.

Work in progress

Patching chromium to have V8 level tainting and enable overriding of Objects that are not possible now.

Thanks

More questions

@skeptic_fx