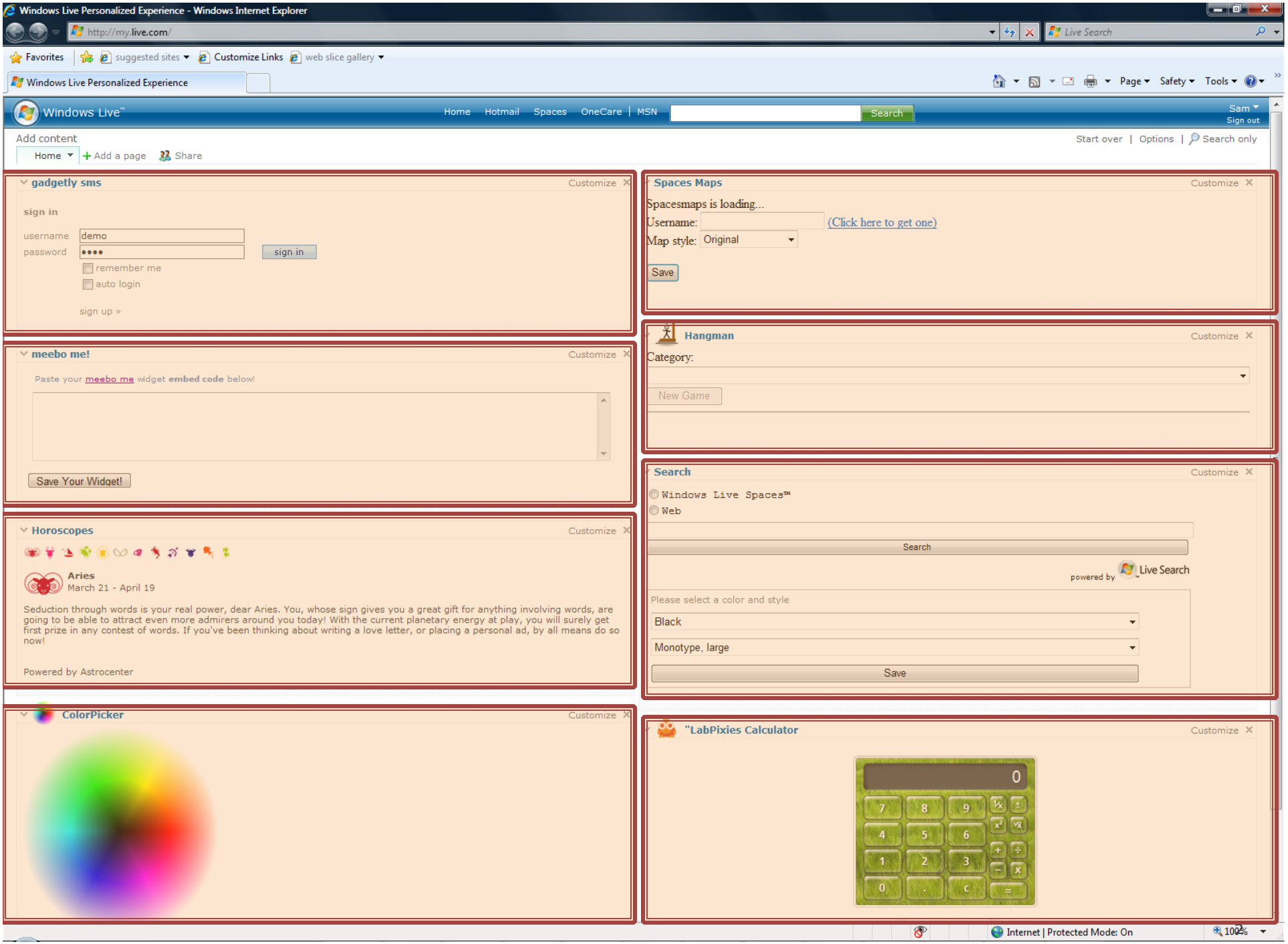
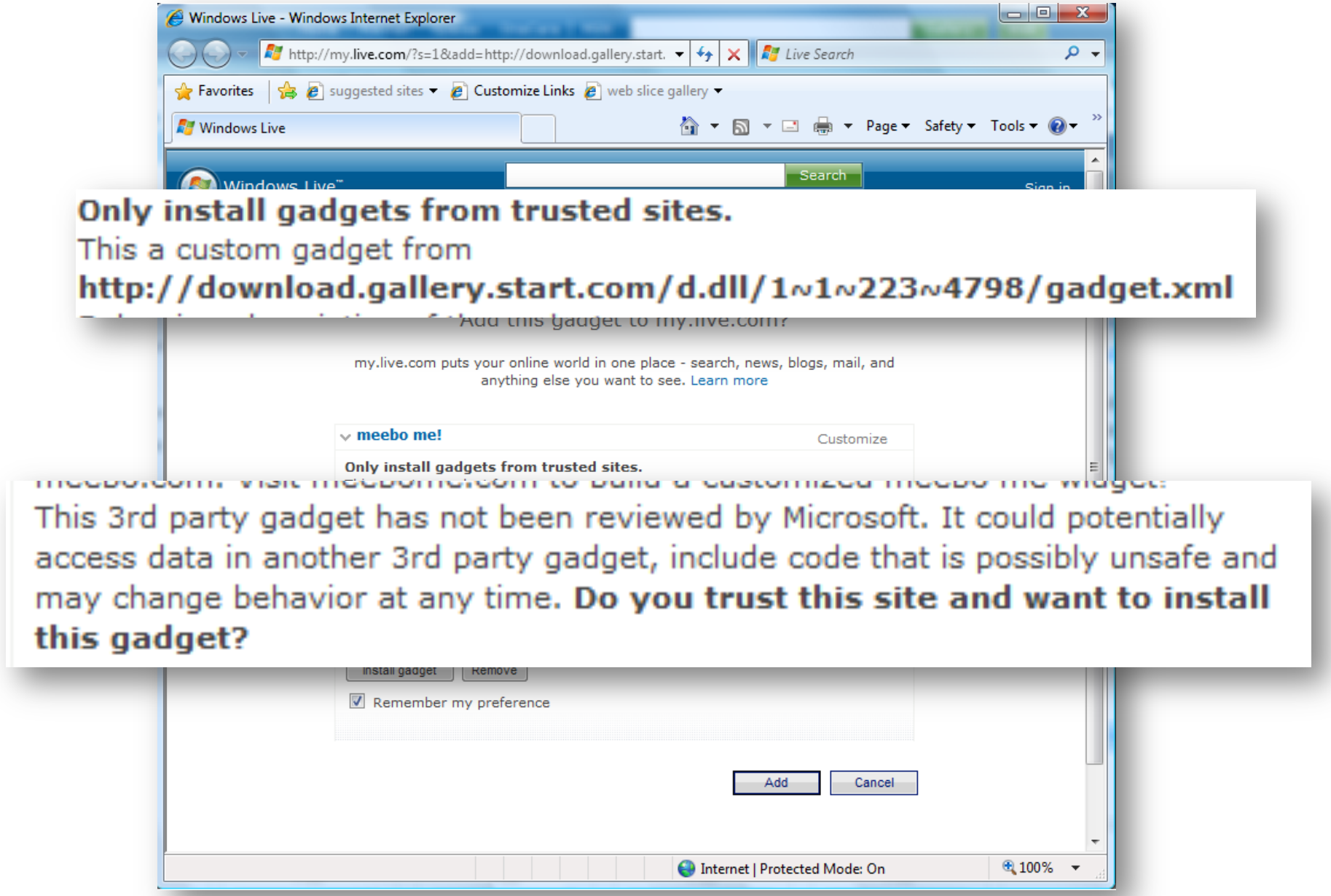
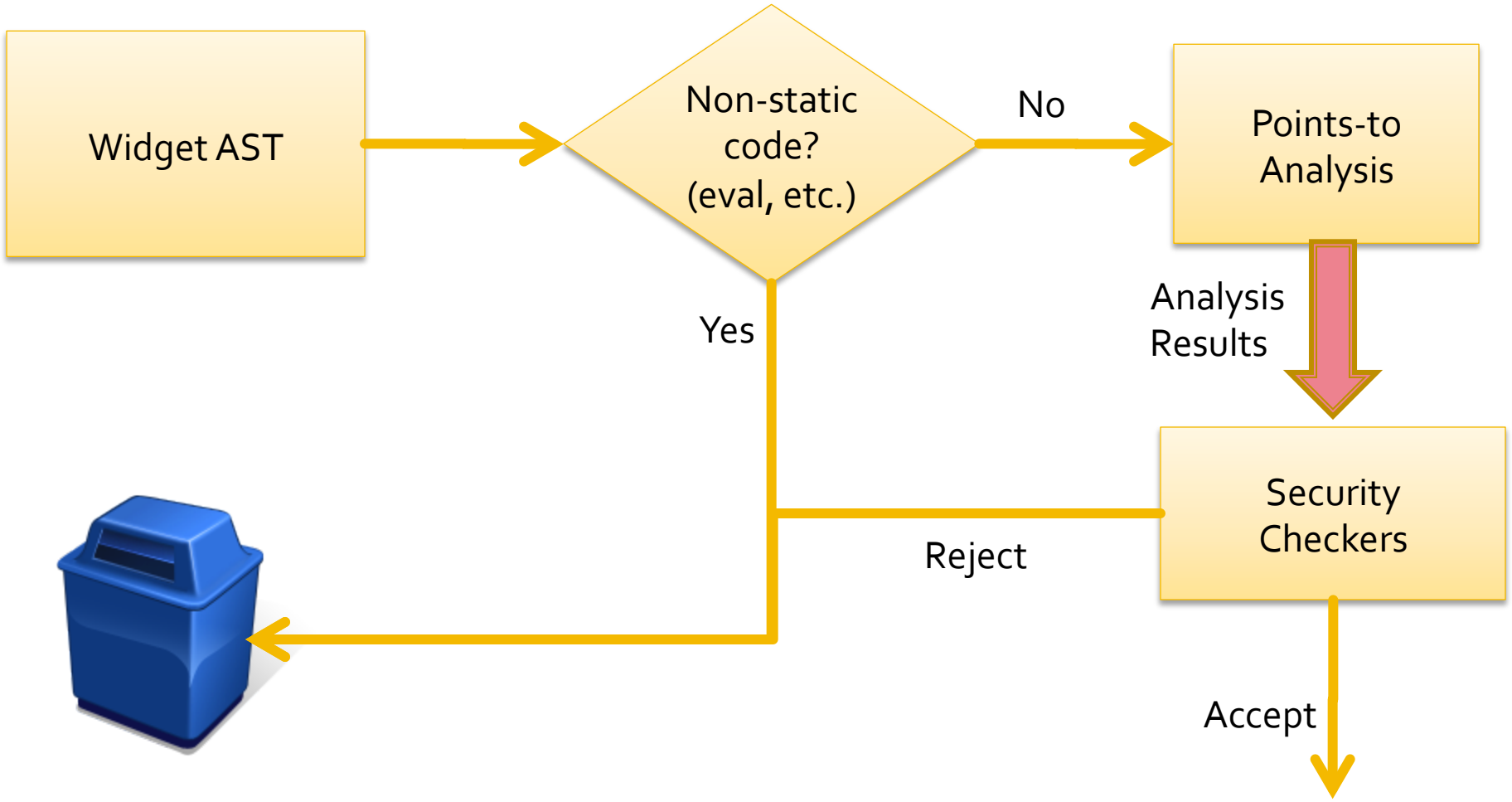
Gatekeeper:
Mostly Static Isolation of JavaScript Code

Salvatore Guarnieri, Ben Livshits


Windows Live Personalized Experience - Windows Internet Explorer
http://my.live.com/
Live Search
Favorites
suggested sites
Customize Links
web slice gallery
Windows Live Personalized Experience
Page
Safety
Tools
Windows Live™
Home
Hotmail
Spaces
OneCare
MSN
Search
Sam
Sign out
Add content
Start over | Options | Search only
Home
Add a page
Share
gadgetly sms
Customize
sign in
username
demo
password
sign in
remember me
auto login
sign up »
Spaces Maps
Customize
Spacesmaps is loading...
Username:
(Click here to get one)
Map style:
Original
Save
meebo me!
Customize
Paste your meebo me widget embed code below!
Save Your Widget!
Hangman
Customize
Category:
New Game
Search
Customize
Windows Live Spaces™
Web
Search
powered by Live Search
Please select a color and style
Black
Monotype, large
Save
Horoscopes
Customize
Aries
March 21 - April 19
Seduction through words is your real power, dear Aries. You, whose sign gives you a great gift for anything involving words, are going to be able to attract even more admirers around you today! With the current planetary energy at play, you will surely get first prize in any contest of words. If you've been thinking about writing a love letter, or placing a personal ad, by all means do so now!
Powered by Astrocenter
ColorPicker
Customize
"LabPixies Calculator
Customize
0
Internet | Protected Mode: On
100%


Windows Live - Windows Internet Explorer
http://my.live.com/?s=1&add=http://download.gallery.start.
Live Search
Favorites
suggested sites
Customize Links
web slice gallery
Windows Live
Page
Safety
Tools
Search
Only install gadgets from trusted sites.
This a custom gadget from
http://download.gallery.start.com/d.dll/1~1~223~4798/gadget.xml
my.live.com puts your online world in one place - search, news, blogs, mail, and anything else you want to see. Learn more
meebo me!
Customize
Only install gadgets from trusted sites.
This 3rd party gadget has not been reviewed by Microsoft. It could potentially access data in another 3rd party gadget, include code that is possibly unsafe and may change behavior at any time. Do you trust this site and want to install this gadget?
Remember my preference
Add
Cancel
Internet | Protected Mode: On
100%

Gatekeeper System



Quick Example

```
function foo(code) {
  document.write(code);

  var x = document;
  x.write(code);
}
```

1

2

iframes

- Pros
  - Easy
  - Provides complete isolation
- Cons
  - Only provides isolation
  - Limits where and how content can appear and be placed

JavaScript Protection

Tool	Method	Runtime Overhead	Expressiveness
WebSandbox Caja	Full dynamic rewrite	High	High (almost all of JavaScript)
FBJS	Static check + access methods	Moderate	Moderate
ADSafe	Static check	Very Low	Very low
Gatekeeper	**Points-to Analysis + dynamic checks**	**Low**	**Moderate**

Gatekeeper Goals

- Sweet spot
  - Low runtime overhead
  - High expressiveness
- Reason about JavaScript programs
  - Sound analysis
  - Applications
    - Security
    - Reliability
    - etc.

Enemies of Static Analysis

Construct	Live [2714]	Sidebar [4501]	Google [1171]
Non-const index	176	1736	192
`innerHTML`	2,053	1,535	288
`eval`	10	353	55
`setTimeout`	49	824	65
`setInterval`	16	377	13
`arguments`	6	175	3
`Function`	4	142	21
`with`	2	422	2
`document.write(url);`	1	102	108

Dynamic Checks

- Not currently implemented
- Non-constant array index
  - var a; …; this[a]();
  - Ensure index doesn't equal something dangerous
- innerHTML writes
  - Parse for valid HTML and possibly subset of JavaScript

Subsets of JavaScript

EcmaScript 262

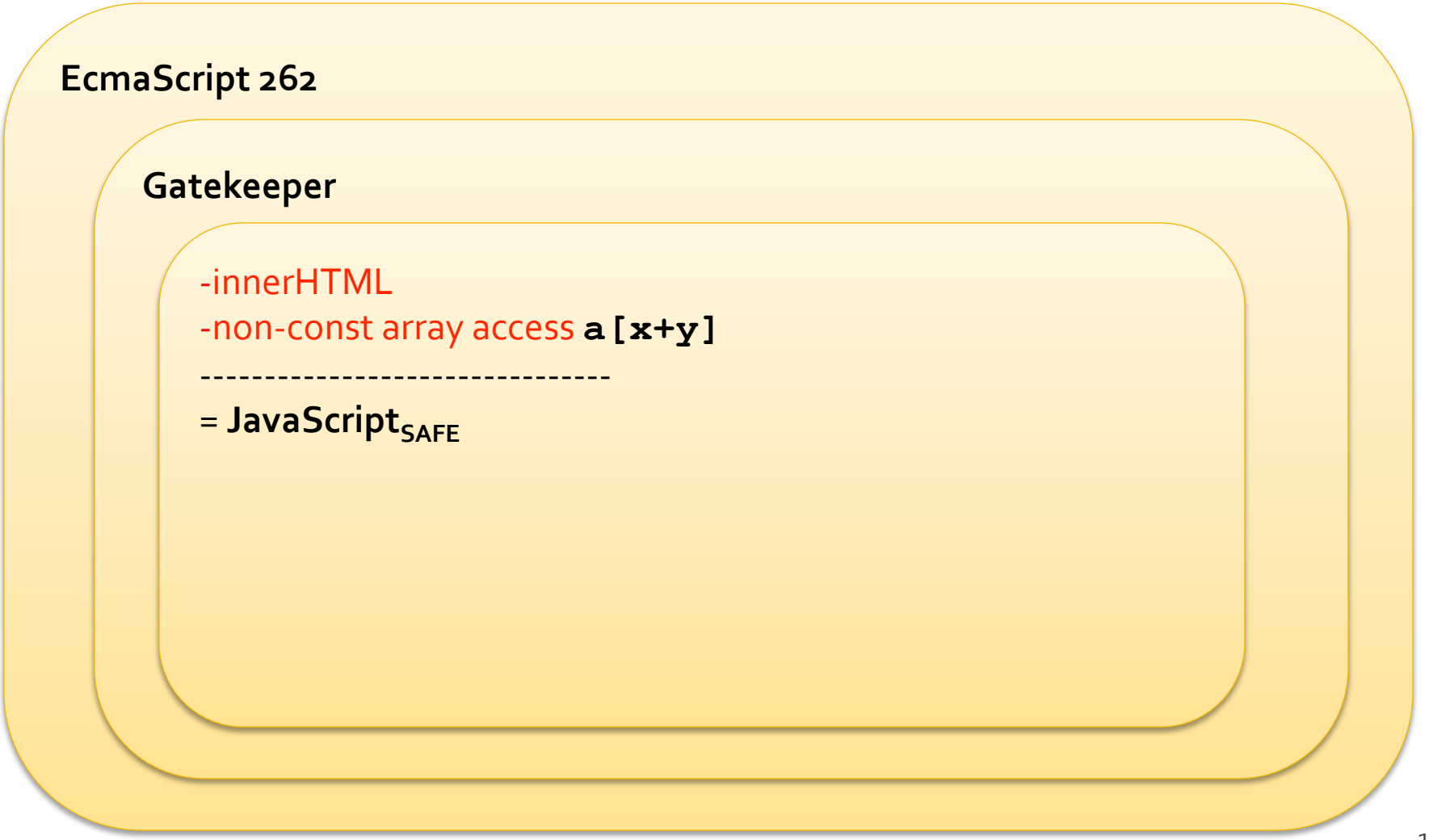
Subsets of JavaScript

EcmaScript 262

- `eval`
- `setTimeout`
- `setInterval`
- `Function`
- `with`
- `arguments` array

= Gatekeeper

Subsets of JavaScript



Subsets of JavaScript

EcmaScript 262

Gatekeeper

$\text{JavaScript}_{\text{SAFE}}$

s ::=

```
// assignments
v1=v2
v = bot
return v
// calls
v = new vo(v1,…,vn)
v=vo(vthis,v1,…,vn)
// heap
v1=v2.f
v1.f=v2
// declarations
v=function(v1,…,vn){s}
```

Widget Corpus

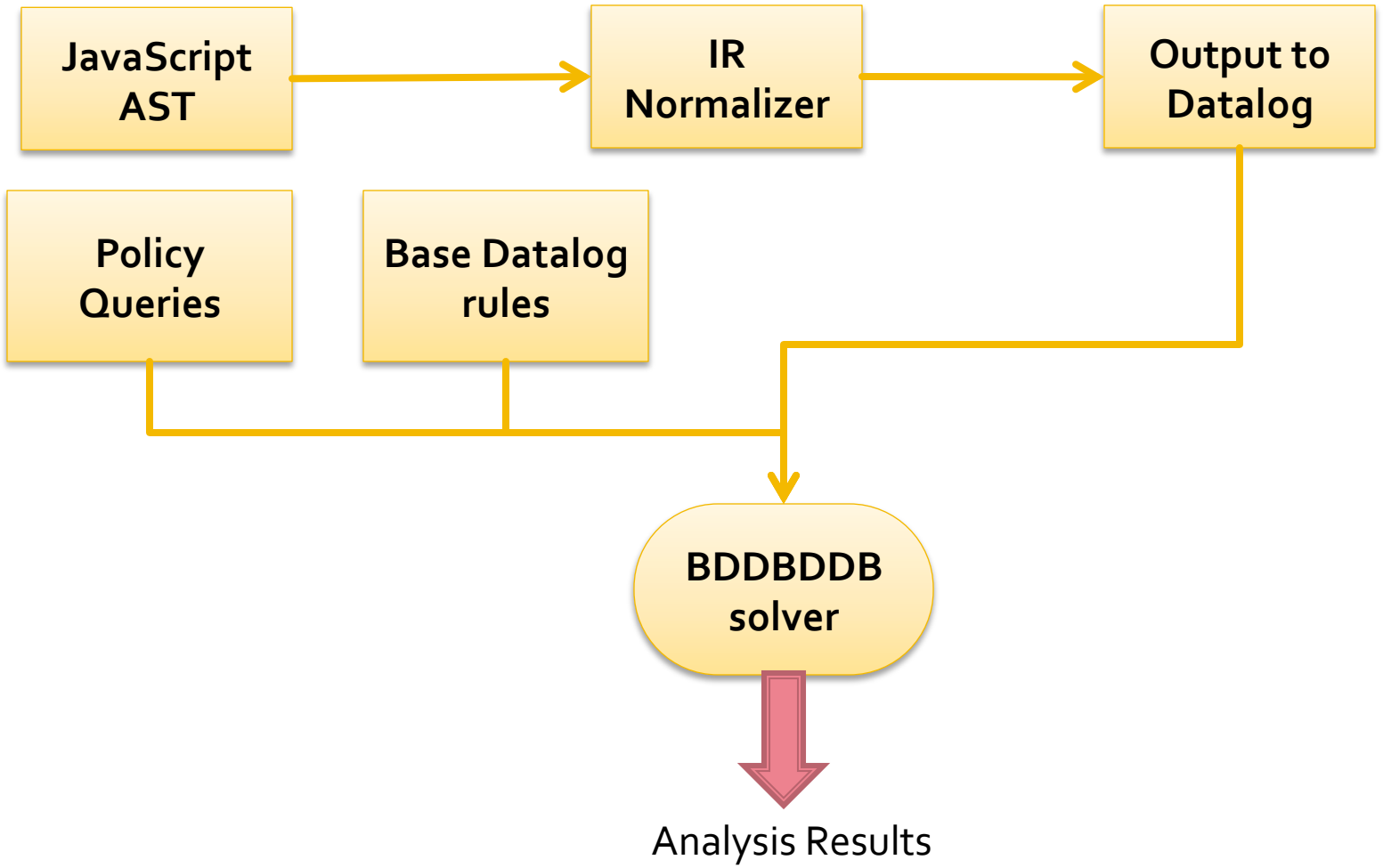
	Total	LOC (avg)	Gatekeeper	$\text{JavaScript}_{\text{SAFE}}$
Live	2,714	105	2,653 (97%)	643 (24%)
Sidebar	4,501	261	2,946 (65%)	1,767 (39%)
Google	1,171	137	962 (82%)	768 (65%)

- Collected by scraping widget galleries

Points-To Analysis

- Overall, similar to Java
- Closure-based language
  - No a-priori call graph
  - Supports higher-order functions
- Prototype based language
  - Uses prototype for inheritance
  - Prototype chains can be manipulated at runtime
  - No a-priori inheritance hierarchy

Closer Look at Analysis



JavaScript AST

- Uses an existing JavaScript parser
- Non-parseable JavaScript is immediately rejected

Normalizer

- Uses constant propagation to turn constant array access into field access
- Reduces complex statements to a series of simple statements
- Outputs datalog facts for simple statements

Processing JavaScript Statements

$v_1 = v_2$	$\textsc{Assign}(v_1, v_2)$.	[Assignment]
$v = \bot$		[BotAssignment]
$v = \texttt{new}\ v_0(v_1, v_2, ..., v_n)$	$\textsc{PtsTo}(v, d_{fresh})$. $\textsc{Prototype}(d_{fresh}, h) :-$ $\quad \textsc{PtsTo}(v_0, m)$, $\quad \textsc{HeapPtsTo}(m, \texttt{"prototype"}, h)$. $\forall z > 0 : \textsc{Actual}(i, z, v_z)$. $\forall z > 0 : \textsc{CallRet}(i, v)$.	[Constructor]
$v = v_0(v_1, v_2, \ldots, v_n)$	$\forall z > 0 : \textsc{Actual}(i, z, v_z)$. $\forall z > 0 : \textsc{CallRet}(i, v)$.	[Call]
$v_1 = v_2.f$	$\textsc{Load}(v_1, v_2, f)$.	[Load]
$v_1.f = v_2$	$\textsc{Store}(v_1, f, v_2)$.	[Store]
$v = \texttt{function}(v_1, ..., v_n)\ \{s\}$	$\textsc{PtsTo}(v, d_{fresh})$. $\textsc{HeapPtsTo}(d_{fresh},$ $\quad \texttt{"prototype"}, p_{fresh})$. $\textsc{FuncDecl}(d_{fresh})$. $\textsc{Prototype}(p_{fresh}, h_{FP})$. $\textsc{Formal}(d_{fresh}, z, v_z)$.	[FunctionDecl]

Pointer Analysis Inference Rules

% Basic rules		
$\textsc{PtsTo}(v, h)$	$:-$	$\textsc{Alloc}(v, h)$.
$\textsc{PtsTo}(v, h)$	$:-$	$\textsc{FuncDecl}(v, h)$.
$\textsc{PtsTo}(v_1, h)$	$:-$	$\textsc{PtsTo}(v_2, h), \textsc{Assign}(v_1, v_2)$.
$\textsc{DirectHeapStoresTo}(h_1, f, h_2)$	$:-$	$\textsc{Store}(v_1, f, v_2), \textsc{PtsTo}(v_1, h_1), \textsc{PtsTo}(v_2, h_2)$.
$\textsc{DirectHeapPointsTo}(h_1, f, h_2)$	$:-$	$\textsc{DirectHeapStoresTo}(h_1, f, h_2)$.
$\textsc{PtsTo}(v_2, h_2)$	$:-$	$\textsc{Load}(v_2, v_1, f), \textsc{PtsTo}(v_1, h_1),$ $\textsc{HeapPtsTo}(h_1, f, h_2)$.
$\textsc{HeapPtsTo}(h_1, f, h_2)$	$:-$	$\textsc{DirectHeapPointsTo}(h_1, f, h_2)$.
% Call graph		
$\textsc{Calls}(i, m)$	$:-$	$\textsc{Actual}(i, 0, c), \textsc{PtsTo}(c, m)$.
% Interprocedural assignments		
$\textsc{Assign}(v_1, v_2)$	$:-$	$\textsc{Calls}(i, m), \textsc{Formal}(m, z, v_1),$ $\textsc{Actual}(i, z, v_2), z > 0$.
$\textsc{Assign}(v_2, v_1)$	$:-$	$\textsc{Calls}(i, m), \textsc{MethodRet}(m, v_1),$ $\textsc{CallRet}(i, v_2)$.
% Prototype handling		
$\textsc{HeapPtsTo}(h_1, f, h_2)$	$:-$	$\textsc{Prototype}(h_1, h), \textsc{HeapPtsTo}(h, f, h_2)$.

Query Results

- Thousands of positive matches for policy queries
- No exploits, but many instances of undesirable code
  - Polluting the global namespace when instructed not to
  - Using JavaScript functions when instructed not to (XMLHttpRequest)
  - Modifying built in objects

Alert Message Boxes

AlertCalls(i) :- **PointsTo**("global", h),
HeapPointsTo(h, "alert", h2), **Calls**(i, h2) .

```
if(response.status!=200){
  alert("Billboard Top10:Retrieve List error.");
  return;
}
```

Frozen Violation

```
FrozenViolation(v, h1) :- Store(v, _, _),
                          PointsTo(v, h1), BuiltinObject(h1) .
FrozenViolation(v, h1) :- Store(v, _, _),
                          PointsTo(v, h1), BuiltinObject(h2),
                          Reaches(h2, f, h1) .
```

```
String.prototype.trim = function()
{
    return this.replace(/^\s+|\s+$/g,"");
}
```

Location Changes

```
StoreToLocationObject(h)  :-
  PointsTo("global", h1),
  HeapStoreTo(h1, "location", h) .
...
```

```
StoreInLocationObject(h)  :-
  LocationObject(h1),
  HeapStoreTo(h1, _, h) .
```

```
CallLocationMethod(i)     :-
  LocationObject(h),
  HeapPointsTo(h, "assign", h1),
  Calls(i, h1) .
...
```

Location Changes in the Wild

```
window.location = "http://e-r.se/google-
  calculator/index.htm"
```

Conclusions…

- Sound points-to analysis for a useful subset of the JavaScript language
- (Mostly) Statically analyze large fraction of widgets
- Developed several representative analyses expressed declaratively in Datalog
- Focus on various widget safety properties
- Find 600 violating widgets and 1200 total violations in over 8,300 widgets

Future Work and Questions

- Investigate how this analysis could be applied to more complex JavaScript programs
- Investigate if data could be tracked through the DOM
- Could this be used to help optimize away some complex dynamic checks?
- What other uses could their be for this analysis?

Thank you