Breaking chrome

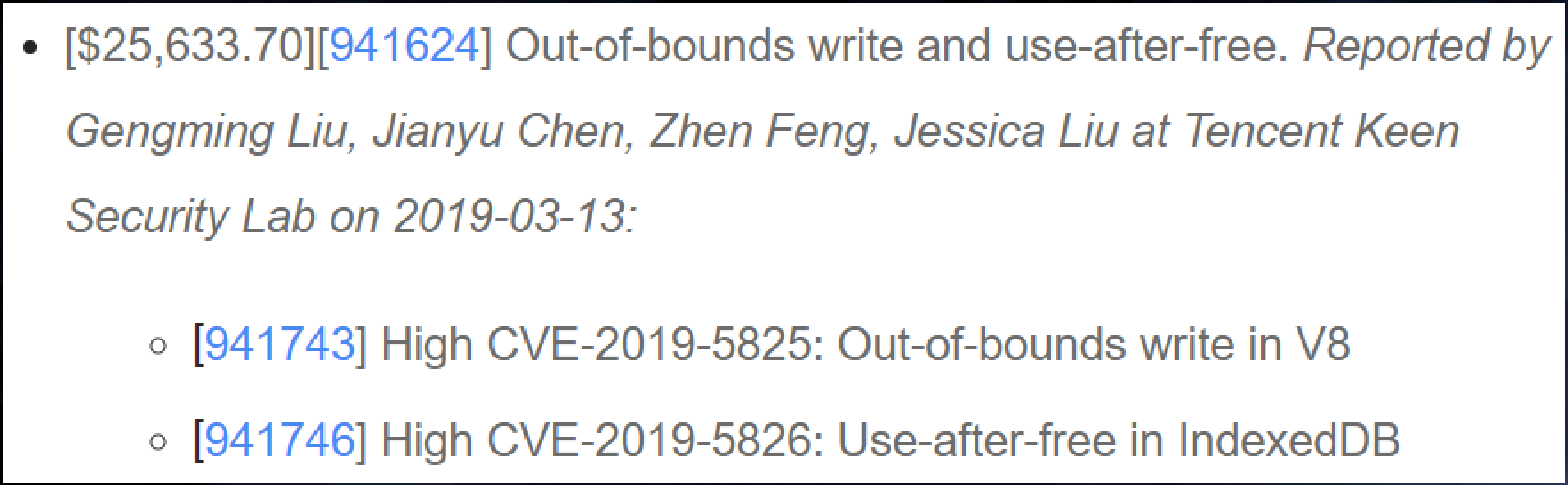
Jeremy Fetiveau
@__x86
beerump 2019

doar-e.github.io

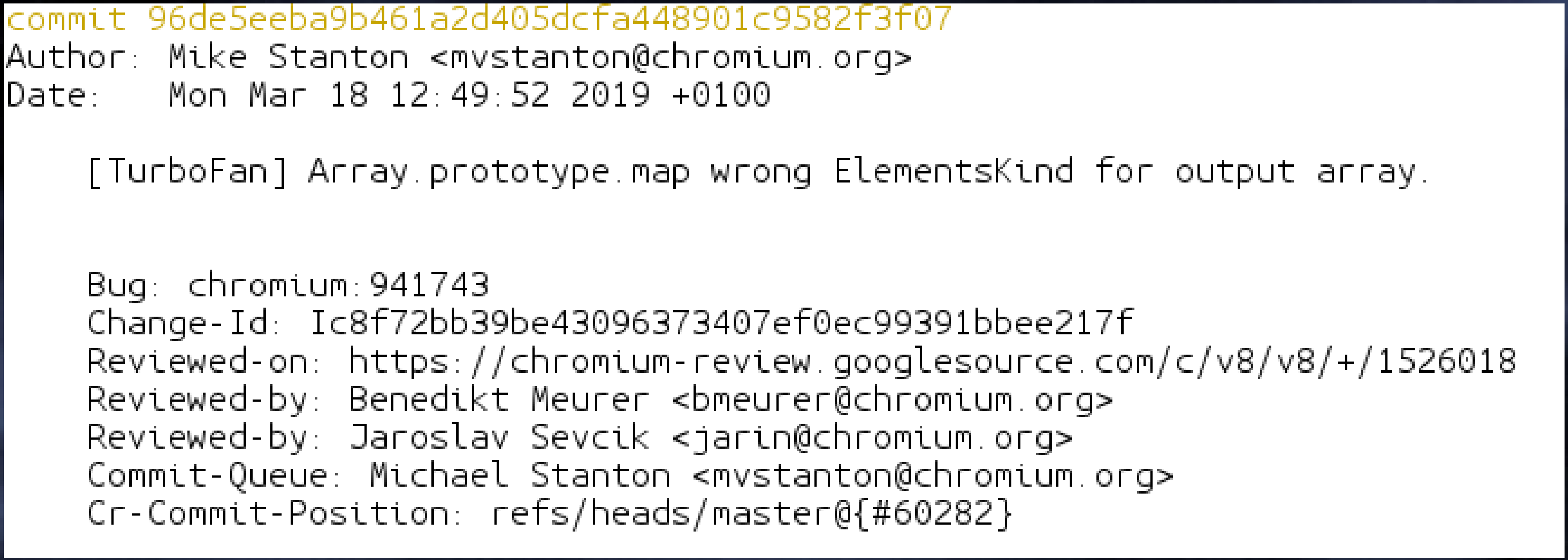
Long story short

- Full chain : RCE+SBX
  - Exfiltration de données
  - Contrôle total de la cible
- SBX
  - kernel, IPC
- RCE
  - Moteur de rendu, moteur javascript

Sample full chain



- [$25,633.70][941624] Out-of-bounds write and use-after-free. *Reported by Gengming Liu, Jianyu Chen, Zhen Feng, Jessica Liu at Tencent Keen Security Lab on 2019-03-13:*
    - [941743] High CVE-2019-5825: Out-of-bounds write in V8
    - [941746] High CVE-2019-5826: Use-after-free in IndexedDB

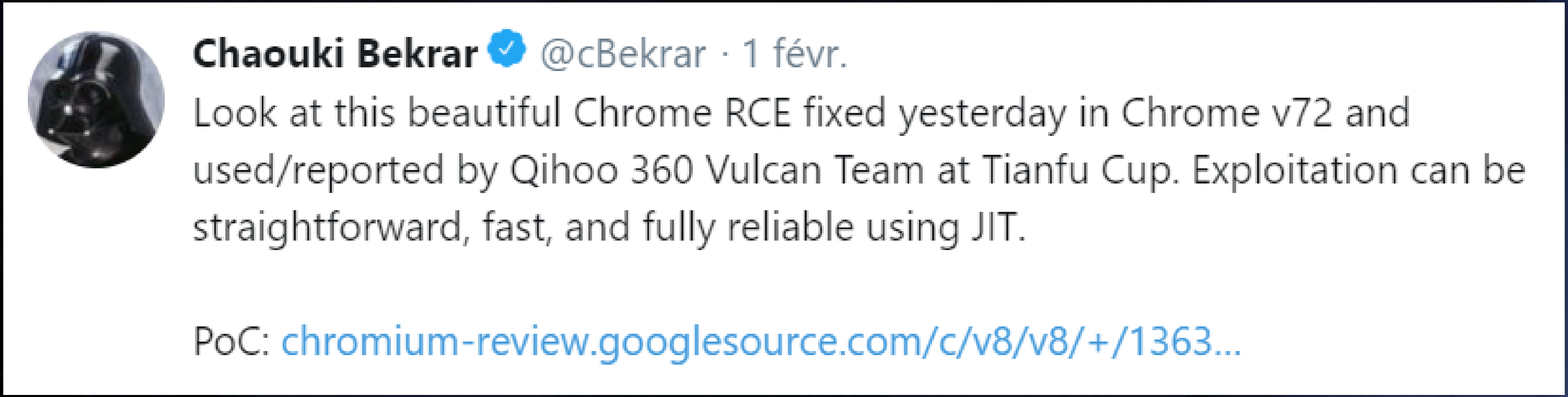


```
commit 96de5eeba9b461a2d405dcfa448901c9582f3f07
Author: Mike Stanton <mvstanton@chromium.org>
Date:   Mon Mar 18 12:49:52 2019 +0100

    [TurboFan] Array.prototype.map wrong ElementsKind for output array.


    Bug: chromium:941743
    Change-Id: Ic8f72bb39be43096373407ef0ec99391bbee217f
    Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/1526018
    Reviewed-by: Benedikt Meurer <bmeurer@chromium.org>
    Reviewed-by: Jaroslav Sevcik <jarin@chromium.org>
    Commit-Queue: Michael Stanton <mvstanton@chromium.org>
    Cr-Commit-Position: refs/heads/master@{#60282}
```

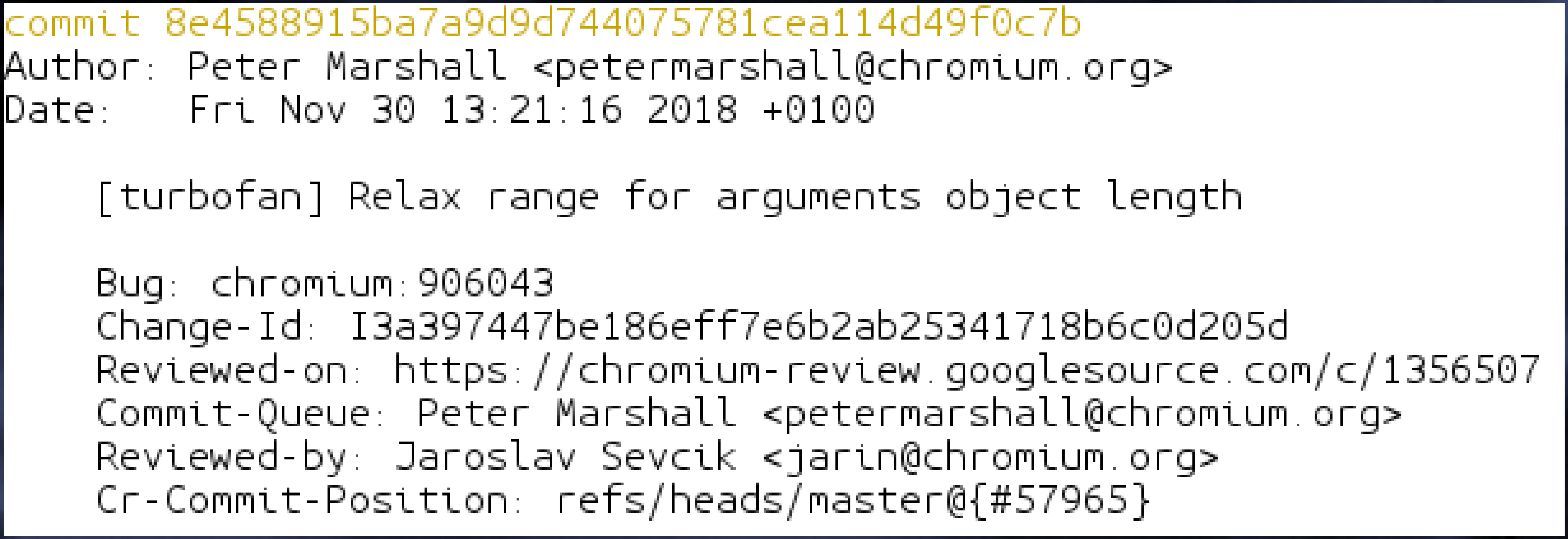
Chrome Tianfu Cup



Chaouki Bekrar @cBekrar · 1 févr.

Look at this beautiful Chrome RCE fixed yesterday in Chrome v72 and used/reported by Qihoo 360 Vulcan Team at Tianfu Cup. Exploitation can be straightforward, fast, and fully reliable using JIT.

PoC: chromium-review.googlesource.com/c/v8/v8/+/1363...



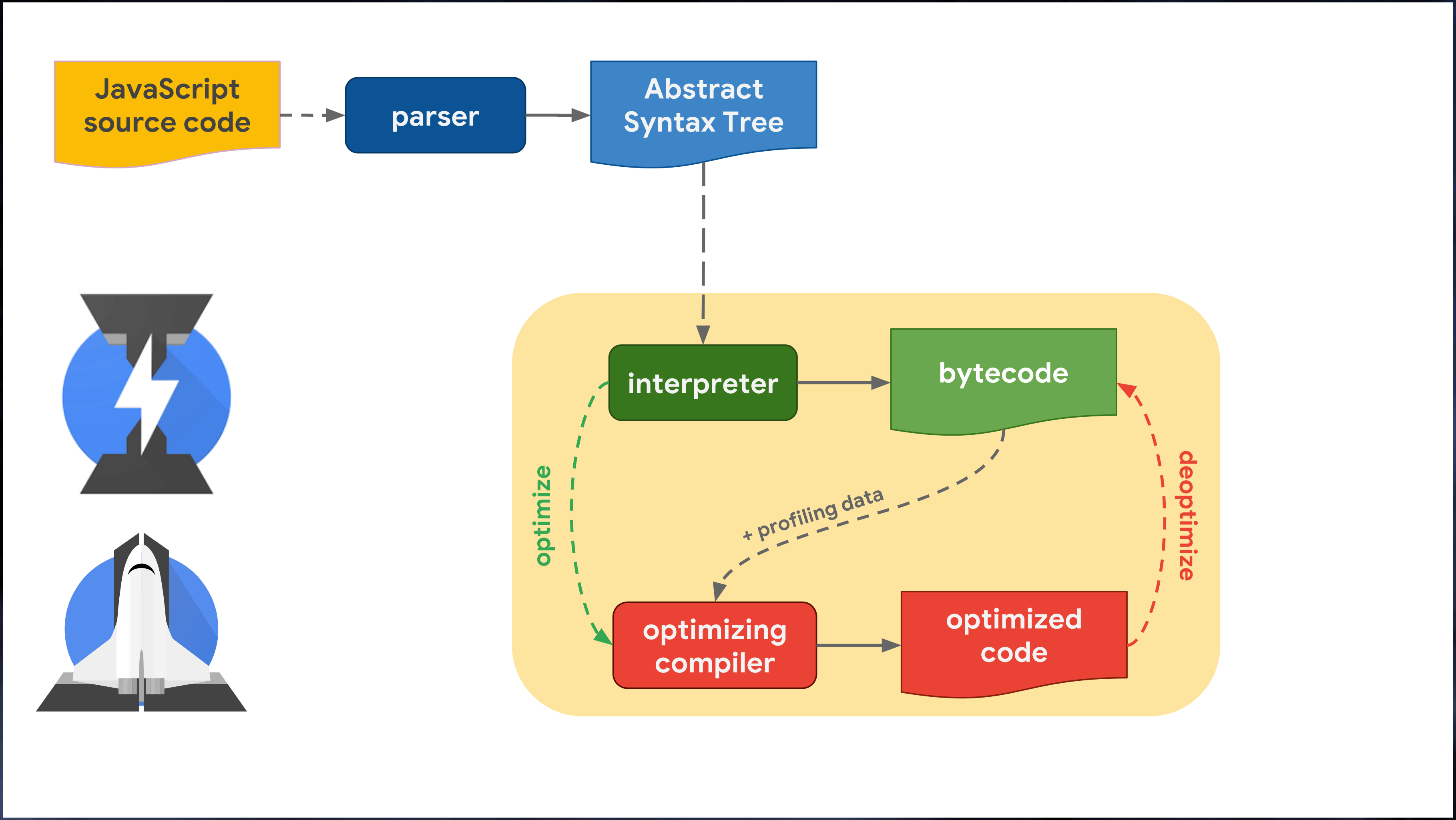
```
commit 8e4588915ba7a9d9d744075781cea114d49f0c7b
Author: Peter Marshall <petermarshall@chromium.org>
Date:   Fri Nov 30 13:21:16 2018 +0100

    [turbofan] Relax range for arguments object length

    Bug: chromium:906043
    Change-Id: I3a397447be186eff7e6b2ab25341718b6c0d205d
    Reviewed-on: https://chromium-review.googlesource.com/c/1356507
    Commit-Queue: Peter Marshall <petermarshall@chromium.org>
    Reviewed-by: Jaroslav Sevcik <jarin@chromium.org>
    Cr-Commit-Position: refs/heads/master@{#57965}
```

V8, moteur JavaScript de chrome

Pipeline v8



https://mathiasbynens.be/notes/shapes-ics

TurboFan

Une optimisation spéculative

```
for (let i = 0; i < 0x10000; ++i)
```

move( )

Déoptimisation!

move( )

Phases d’optimisations

- Création d’un graphe *sea of nodes*
- *Reduction* des nodes
- OperationTyper associe un type à un Operateur
  - Add(x,y) => résultat dans [m,n]
- Strength reduction de n * 2
  - N << 1
- Context specialization de JSLoadProperty(‘0’)
  - LoadField + LoadElement + CheckBounds + IfTrue + Merge + Phi + ...

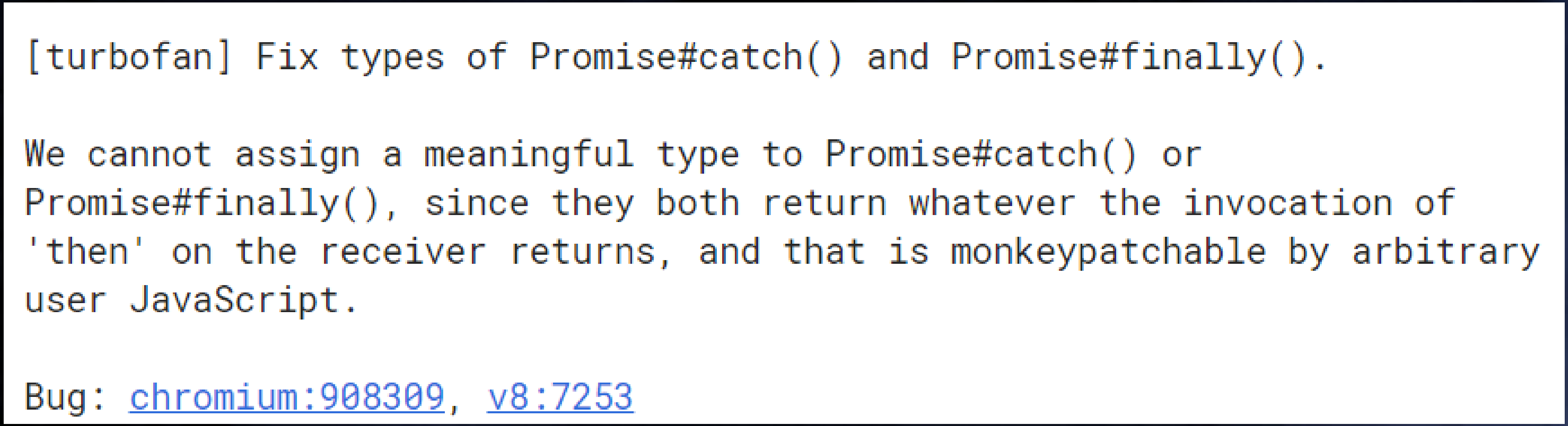
Quel intérêt?

- Bug “logiques”
- Très forte fiabilité
  - >= 99% ;-)

Quel genre de bugs

- Des optimisations incorrectes de TurboFan
- Bug classes différentes
- Pas mal de problèmes de typing
  - Typer
  - Operation Typer
  - Type cache

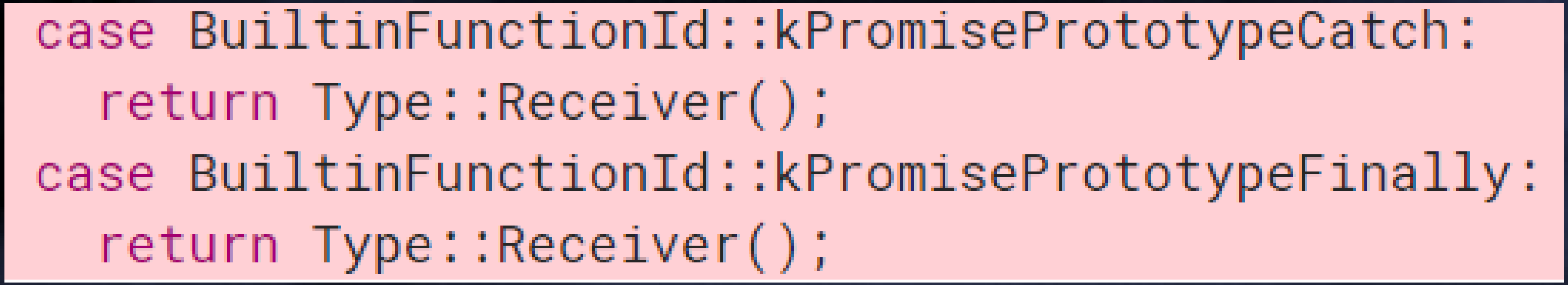
Promise#catch (no CVE)



```
[turbofan] Fix types of Promise#catch() and Promise#finally().

We cannot assign a meaningful type to Promise#catch() or
Promise#finally(), since they both return whatever the invocation of
'then' on the receiver returns, and that is monkeypatchable by arbitrary
user JavaScript.

Bug: chromium:908309, v8:7253
```

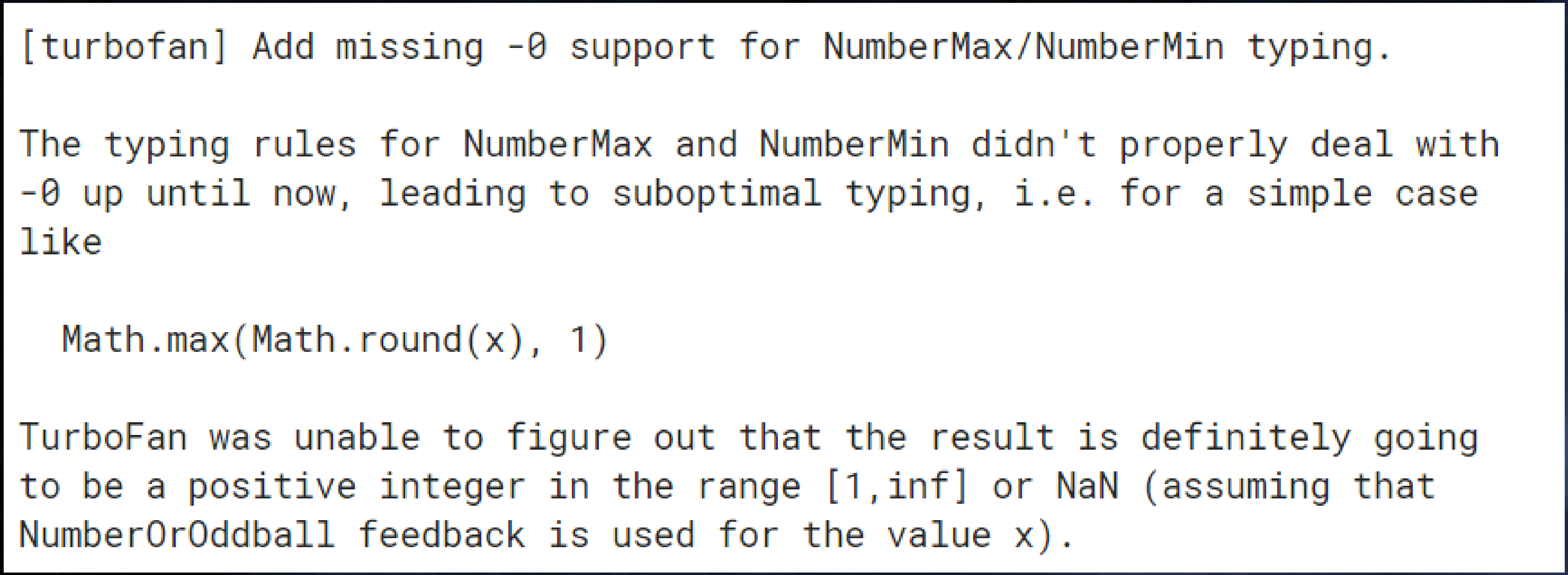


```
case BuiltinFunctionId::kPromisePrototypeCatch:
  return Type::Receiver();
case BuiltinFunctionId::kPromisePrototypeFinally:
  return Type::Receiver();
```

TLDR, lors de l’optimisation par TurboFan :

- associe Type::Receiver à un node
- une fois optimisé, code part du principe qu’il verra un Type::Receiver
- pas forcément vrai!

NumberMax/Min (no CVE)



```
[turbofan] Add missing -0 support for NumberMax/NumberMin typing.

The typing rules for NumberMax and NumberMin didn't properly deal with
-0 up until now, leading to suboptimal typing, i.e. for a simple case
like

  Math.max(Math.round(x), 1)

TurboFan was unable to figure out that the result is definitely going
to be a positive integer in the range [1,inf] or NaN (assuming that
NumberOrOddball feedback is used for the value x).
```

TLDR :

integer

vs

integer OU **<u>minuszero</u>**

String.lastIndexOf (no CVE)

turbofan

```
Type::Range(-1.0, String::kMaxLength - 1.0, t->zone());
```

VS

réalité

```
String::kMaxLength
```

Tianfu Cup - kArgumentsLengthType

turbofan

```
Type::Range(0.0, Code::kMaxArguments, zone())
```

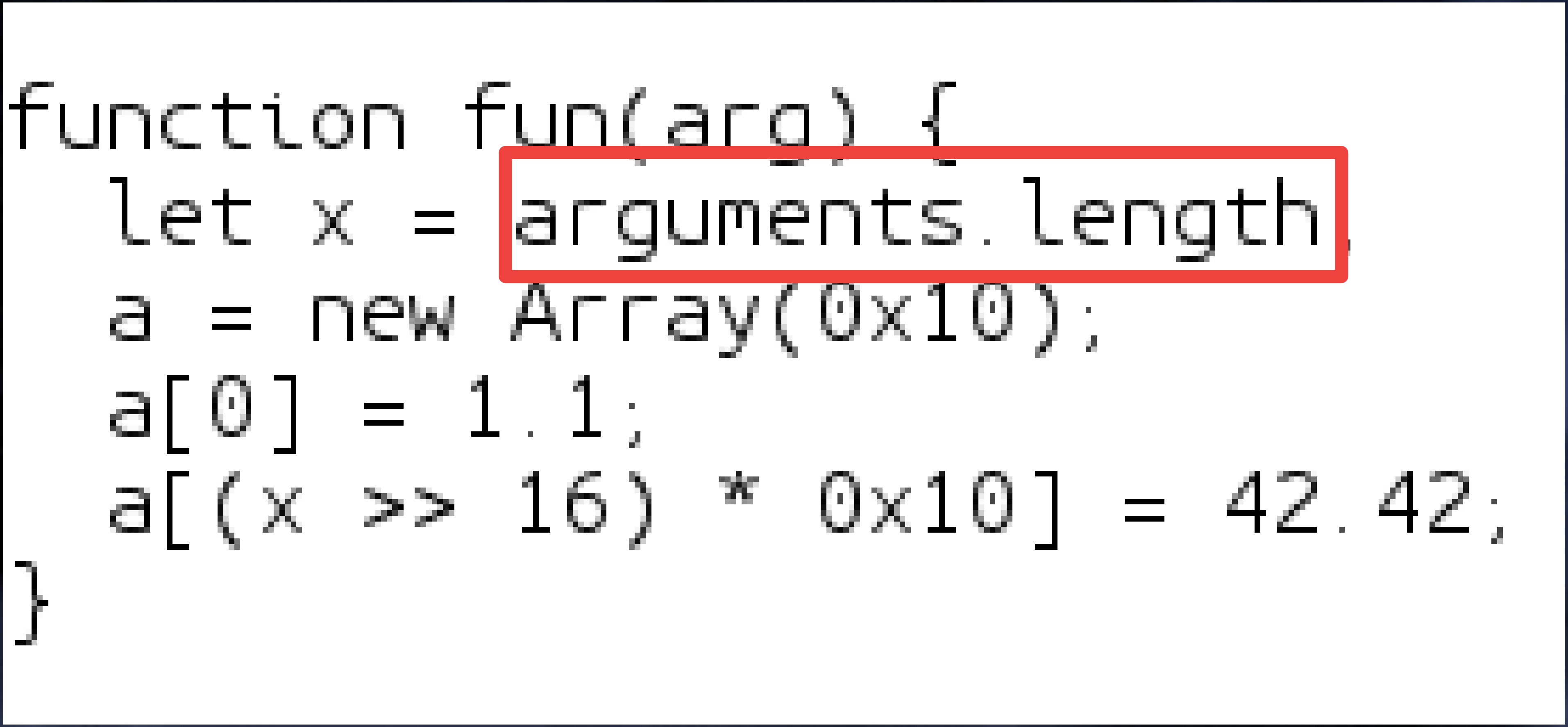
VS

réalité

```
CreateRange(0.0, FixedArray::kMaxLength);
```

```
func(...argumentsArray)
```

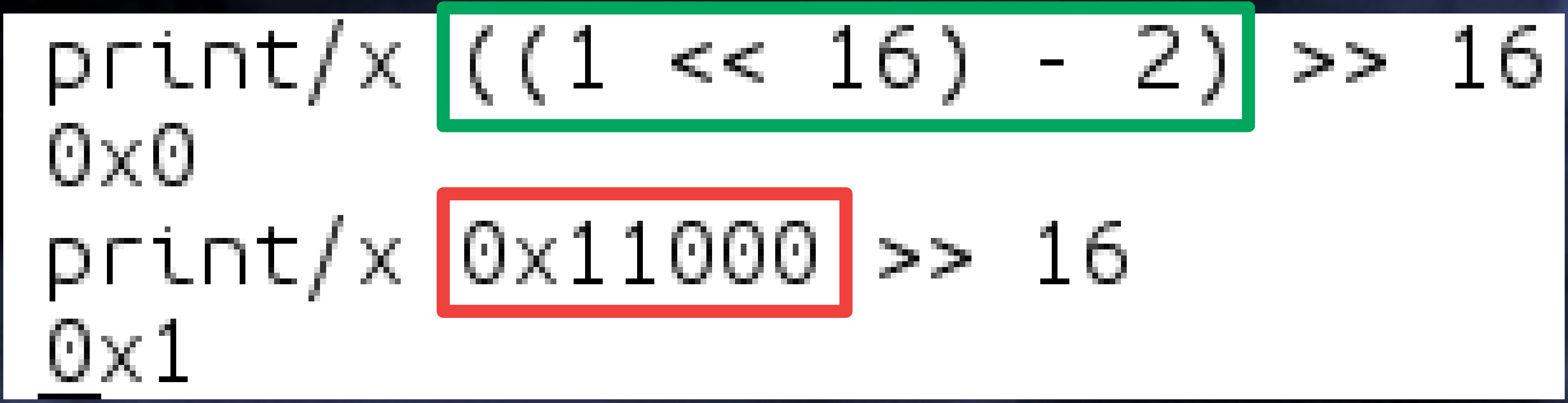
Tianfu Cup - ArgumentsLength



```
function fun(arg) {
  let x = arguments.length;
  a = new Array(0x10);
  a[0] = 1.1;
  a[(x >> 16) * 0x10] = 42.42;
}
```

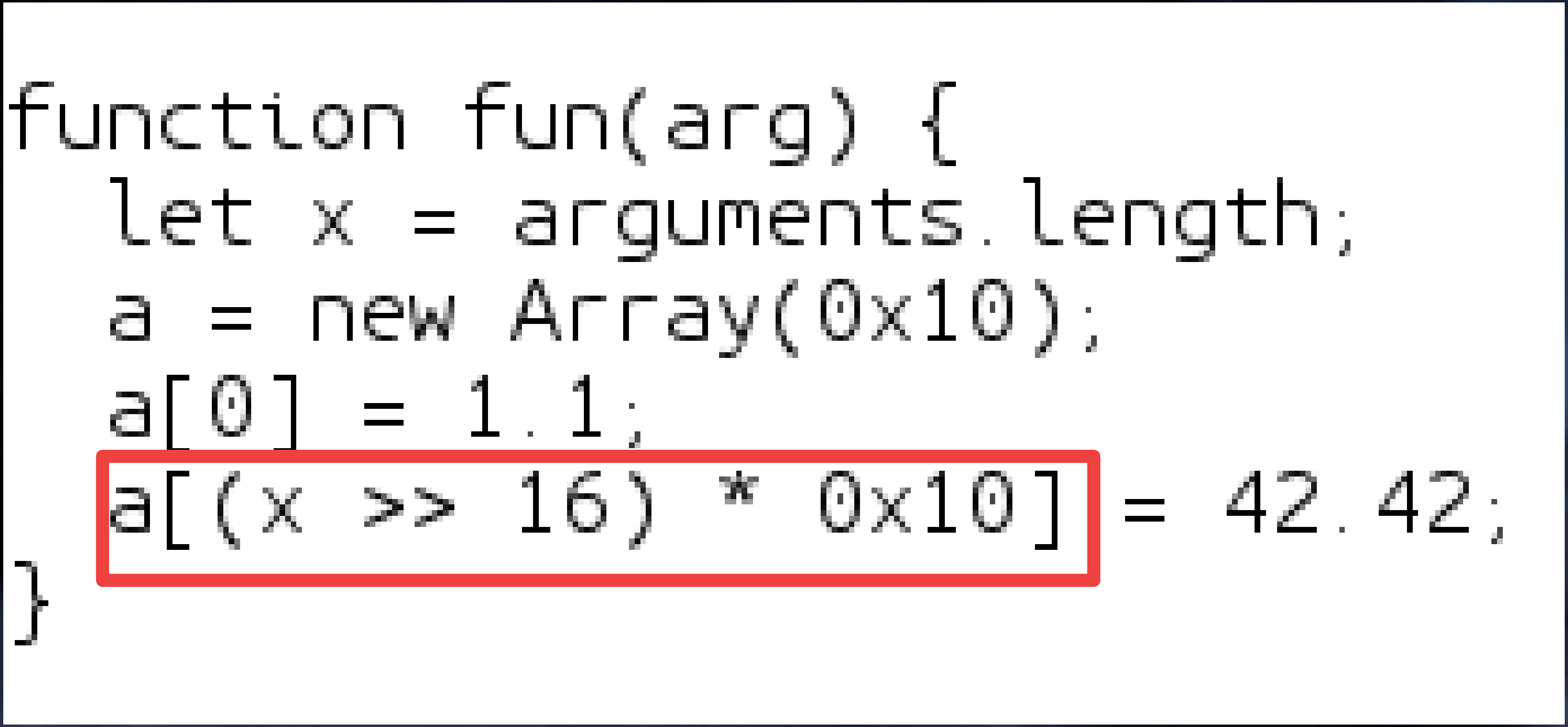
Tianfu Cup - ArgumentsLength

```
args = [];
args.length = 0x11000;
fun(...args)
```



```
print/x ((1 << 16) - 2) >> 16
0x0
print/x 0x11000 >> 16
0x1
```

Tianfu Cup - ArgumentsLength



```
function fun(arg) {
  let x = arguments.length;
  a = new Array(0x10);
  a[0] = 1.1;
  a[(x >> 16) * 0x10] = 42.42;
}
```

Quel intérêt?

- Optimisations incorrectes
- Impactent d’autres optimisations
  - BCE (Bounds Check Elimination)
  - Redundancy Elimination

BCE : Bound Check Elimination

- Désactivé récemment!
- BCE incorrecte == OOB
- Pas de BCE == pas d’OOB?
- Donc bugs de typing inexploitables?

Ma théorie

- Beaucoup de phases d’optimisations
- Beaucoup de réductions différentes
- Beaucoup de comportements à modifier
- Que faire sans la BCE?
  - Influencer la Redundancy Elimination?
  - Plus simple?

Exploitation sans BCE

- Typing incorrect
- Lowering (= “spécialisation”) de CheckBounds
- CheckBounds => CheckedUint32Bounds

Exploitation sans BCE

- Cas particulier!
- CheckedUint32Bounds(index, MaxValue)
- Rajoute un Uint32LessThan + Unreachable
- “if index >= LoadField(length) then abort()”

Exploitation sans BCE

- Propagation du mauvais typing
- UInt32LessThan
  - constant folded
- Reste du code
  - dead code eliminated
- Bound checks == “optimized out”
  - OOB :-)

Diary Of A Reverse Engineer

DOAR·E

doar-e.github.io

Introduction to TurboFan

Circumventing Chrome's hardening of typer bugs

Présentation : Attacking TurboFan

Merci pour votre attention!

doar-e.github.io

shout-out to :

0vercl0k & yrp604

@__x86

@typhooncon