

C++ Club Meeting Notes

Gleb Dolgich

2018-11-22

San Diego trip reports

- ▶ Ben Craig
- ▶ Corentin Jabot
 - ▶ Reddit
- ▶ René Rivera
- ▶ CppCast with Ashley Hedberg

CppCon 2018 Trip Reports

- ▶ JeanHeyd Meneide

Better template support and error detection in C++ Modules with MSVC 2017 version 15.9

[VC Blog](#)

C++17

- ▶ [Viva64: C++17 Refresher](#)
- ▶ [VCBlog: Standard Library Algorithms: Changes and Additions in C++17](#)

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (1/9)

Video

The slide features a grid of colored boxes representing the impact of various Spectre variants on different systems. The columns are labeled: App, OS, SMT Only?, SW Fix?, and HW Fix?. The rows are categorized by variant: Spectre v1, Spectre v2, and Spectre + CPU Bug. The grid includes specific variants like v1, v1.1, v1.2, ret2spec, v2, SpectreRSB, Spectre v4, v3, Lazy FPU, and L1TF. The colors indicate the status of each column for each row.

		Impacts Code In				
		App	OS	SMT Only?	SW Fix?	HW Fix?
Spectre v1	v1	yes ☺	yes ☺	no ☹	slow	no ☹
	v1.1	yes ☺	yes ☺	no ☹	slow	no ☹
	v1.2	yes ☺	yes ☺	no ☹	slow	no ☹
	ret2spec	yes ☺	yes ☺	no ☹	slow	no ☹
Spectre v2	v2	yes ☺	yes ☺	yes ☺	yes ☺	yes ☺
	SpectreRSB	yes ☺	yes ☺	yes ☺	slow	yes ☺
	Spectre v4	yes ☺	yes ☺	no ☹	no ☹	???
Spectre + CPU Bug	v3	no ☹	yes ☺	no ☹	yes ☺	yes ☺
	Lazy FPU	no ☹	yes ☺	no ☹	yes ☺	no
	L1TF	no ☹	yes ☺	yes ☺	no ☹	yes ☺

46

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

CHANDLER CARRUTH

Spectre
Secrets, Side-Channels,
Sandboxes, and Security

CppCon.org

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (2/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Retpolines

- Developed by Google
- Requires recompiling source
- Mitigates Spectre v2 (and SpectreRSB in restricted cases)
- May be faster than STIBP for current CPUs.
- Likely slower and less strong than STIBP on future CPUs

<http://bit.ly/2R50i8J-retpoline>, <http://bit.ly/2NOxtPJ-retpoline-intel>

63



CHANDLER CARRUTH

Spectre
Secrets, Side-Channels,
Sandboxes, and Security

CppCon.org

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (3/9)

The screenshot shows a video player interface. At the top, it displays "No Service" with a signal icon, the time "09:41", and the CppCon 2018 logo with battery level "94%". Below this, the video title is "Spectre: Secrets, Side-Channels, Sandboxes, and Security" followed by "CppCon - 20181002 - CppCon". The video duration is listed as "56:46" and "47:18".

The main video frame shows Chandler Carruth speaking on stage. He is wearing a light blue button-down shirt and dark trousers. To his right are three circular icons: a plus sign, a minus sign, and "1.2x". Below him is a nameplate that reads "CHANDLER CARRUTH".

The slide content on the left side of the video frame includes the text "Rethpolines overhead: under 3%" and "(PGO+ThinLTO)". There are three small circular icons on the left edge of the slide: a lock, a gear, and a downward arrow. The slide number "75" is in the bottom right corner.

At the bottom of the video player are standard control buttons: a lock icon, back, forward, and play/pause buttons. The CppCon logo is also present at the bottom right.

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (4/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

```
# %bb.1:  
    movq    (%rdi), %rax  
    leaq    8(%rsp), %rsi  
    leaq    24(%rsp), %rdx  
    callq   *48(%rax)  
    movq    24(%rsp), %rdi  
    testq   %rdi, %rdi  
    je     .LBB0_3  
  
# %bb.1:  
    movq    (%rdi), %rax  
    movq    48(%rax), %r11  
    leaq    8(%rsp), %rsi  
    leaq    24(%rsp), %rdx  
    callq   __llvm_retpoline_r11  
    movq    24(%rsp), %rdi  
    testq   %rdi, %rdi  
    je     .LBB0_3
```



CHANDLER CARRUTH

Spectre
Secrets, Side-Channels,
Sandboxes, and Security

65

CppCon.org

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (5/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

```
_@_llvm_retpoline_r11:          # @_llvm_retpoline_r11
# %bb.0:
    callq  .LBB1_2
.LBB1_1:                      # Block address taken
                                # => This Inner Loop Header: Depth=1
    pause
    lfence
    jmp   .LBB1_1
    .p2align 4, 0x90
.LBB1_2:                      # Block address taken
    movq  %r11, (%rsp)
    retq
```

74



CHANDLER CARRUTH

Spectre
Secrets, Side-Channels,
Sandboxes, and Security

CppCon.org

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (6/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Speculative Load Hardening (SLH)

- Automatic hardening against v1
- Developed by Google
- Changes compiler generated code to make v1 attacks impossible
- Very complex: one of the most complicated low-level transforms in LLVM
- Easy to deploy: clang++ -mspeculative-load-hardening ...

<http://bit.ly/2NPcgVY-slh>

77

CHANDLER CARRUTH

Spectre

Secrets, Side-Channels,
Sandboxes, and Security

CppCon.org



CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (7/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

SLH overhead is HUGE: 30% - 40% CPU

90



CHANDLER CARRUTH

**Spectre
Secrets, Side-Channels,
Sandboxes, and Security**

CppCon.org

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (8/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Isolate secret data from risky code

- Sandbox untrusted code (or code handling untrusted input) from data w/ OS/HW barrier (process boundary for example)
- Effective against essentially all known vulnerabilities
- Only realistic mitigation for v4 (SSB)
- Also protects against non-spectre side channel attacks like Heartbleed
- Every browser moving to this model via site- (or origin-) isolation

<http://bit.ly/2zEivmN-chromium-post-spectre>

91



CHANDLER CARRUTH

Spectre
Secrets, Side-Channels,
Sandboxes, and Security

CppCon.org

CppCon 2018: Chandler Carruth “Spectre: Secrets, Side-Channels, Sandboxes, and Security” (9/9)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Conclusion

- Spectre: misspeculation + side channel -> leak secrets
- New, active area for research -> ongoing influx of vulnerabilities
- Have a threat model, because we can't afford to mitigate everything
- Tailor mitigations to each application's risk and performance
- Convince CPU vendors to make these problems go away

97



CHANDLER CARRUTH

Spectre
Secrets, Side-Channels,
Sandboxes, and Security

CppCon.org

Meltdown, Spectre etc.

- ▶ Matt Klein
- ▶ Graham Sutherland via Frank Denneman

Hardware Effects

This repository demonstrates various hardware effects that can degrade application performance in surprising ways and that may be very hard to explain without knowledge of the low-level CPU and OS architecture. For each effect I try to create a proof of concept program that is as small as possible so that it can be understood easily.

- ▶ [GitHub – Reddit](#)
 - ▶ Demonstrates: bandwidth saturation, branch misprediction, branch target misprediction, cache aliasing, cache/memory hierarchy bandwidth, data dependencies, denormal floating point numbers, false sharing, hardware prefetching, memory-bound program, non-temporal stores, software prefetching, write combining.

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (1/13)

Video

The image is a composite of several video frames. At the top right, the CPPCON 2018 logo is displayed with the text "THE C++ CONFERENCE • BELLEVUE, WASHINGTON". Below the logo, on the left, is a grid of six video frames showing the "Scientific and Technical Academy Awards" ceremony. The frames show various award presentations on stage with people in formal attire. On the right side of the composite, there is a larger frame showing a man, identified as MARK ELENDT, speaking at a podium. He is wearing a dark polo shirt. The text "MARK ELENDT" is overlaid on the bottom right of this frame. At the bottom right of the composite, there is a box containing the text "Patterns and Techniques Used in the Houdini 3D Graphics Application". The SideFX logo is visible in the bottom right corner of the award ceremony grid. The bottom right corner of the entire composite also contains the text "CppCon.org".

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

INTRODUCTION

Scientific and Technical Academy Awards

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

SideFX

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (2/13)

Historical Roots

RETROSPECTIVE

MAGI

ABEL

III

DIGITAL EFFECTS

DIGITAL PROD.

OMNIBUS

PDI

CANSTON CSURI

POLYGON PICTURES

SideFX

1980 1981 1982 1983 1984

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (3/13)

The slide features a collage of images. On the left is a photograph of the British new wave band Duran Duran from their 1982 album 'Rio'. The band members are posed against a red background. On the right is a photograph of Mark Elendt, a man with short grey hair, wearing a black polo shirt with the SideFX logo, standing on a stage and gesturing with his hands. The top right corner of the slide displays the 'cppcon | 2018' logo with the subtitle 'THE C++ CONFERENCE • BELLEVUE, WASHINGTON'. The bottom right corner contains the text 'CppCon.org'.

1982

INTRODUCTION

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (4/13)

The slide deck for Mark Elendt's talk at CPPCon 2018 features a dark blue header with the conference logo and text. Below the header, there are four main sections: a collage of images from the 'Early Days' of Houdini, a retrospective photo of a man working at a desk, a video of Mark Elendt speaking, and a summary of the talk's content.

CPPCon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Early Days

RETROSPECTIVE

MARK ELENDT

**Patterns and Techniques
Used in the Houdini
3D Graphics Application**

CppCon.org

SideFX

Early Days

RETROSPECTIVE

MARK ELENDT

**Patterns and Techniques
Used in the Houdini
3D Graphics Application**

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (5/13)

Retrospective

RETROSPECTIVE

PRISMS

SideFX

Greg Hermanovic

Kim Davidson

1985 1986 1987 1988 1989

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

THE C++ CONFERENCE • BELLEVUE, WASHINGTON

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (6/13)

1997

RETROSPECTIVE

SideFX

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (7/13)

The slide is titled "Houdini Origins" and features a timeline from 1990 to 1999. It shows two screenshots of the software interface: one from 1990 labeled "PRISMS" and one from 1999 labeled "RETROSPECTIVE". A red arrow points from the 1990 screenshot to a red box labeled "Draft STL". A blue arrow points from the 1990 screenshot to a blue box labeled "Prisims (C++)". A red arrow points from the 1999 screenshot to a red box labeled "C++ Standard Released First version of STL". An orange arrow points from the 1999 screenshot to an orange box labeled "Houdini (C++)". The SideFX logo is visible at the bottom right. The CppCon 2018 logo is in the top right corner.

1990

1992

1994

1996

1998

1999

PRISMS

Draft STL

Prisims (C++)

C++ Standard Released
First version of STL

Houdini (C++)

RETROSPECTIVE

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (8/13)

VFX Reference Platform

SOFTWARE ECOSYSTEM

2015

- gcc v4.8.2
- boost 1.5.x
- Qt 4.8.x
- Python 2.7.x

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (9/13)

The image shows a presentation slide from CppCon 2018. At the top right, the logo "cppcon | 2018" is displayed above the text "THE C++ CONFERENCE • BELLEVUE, WASHINGTON". On the left, there is a sidebar with the "VFX Reference Platform" logo and the text "SOFTWARE ECOSYSTEM". The main content area features a dark background with white text. On the left, the title "VFX Reference Platform" is displayed. In the center, the year "2016" is shown above a list of dependencies and standards:

- gcc v4.8.2 -> unchanged
- boost 1.5.x -> boost1.5.8
- Qt 4.8.x -> Qt 5.6.1
- Python 2.7.x -> Python 2.7.5
- **C++11**

On the right side of the slide, there is a video frame showing Mark Elendt speaking. Below the video frame, his name "MARK ELENDT" is displayed. At the bottom right of the slide, the text "SideFX" is visible, and at the very bottom right, the URL "CppCon.org" is shown.

VFX Reference Platform

SOFTWARE ECOSYSTEM

2016

- gcc v4.8.2 -> unchanged
- boost 1.5.x -> boost1.5.8
- Qt 4.8.x -> Qt 5.6.1
- Python 2.7.x -> Python 2.7.5
- **C++11**

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (10/13)

The image shows a presentation slide from CppCon 2018. In the top right corner, the text "cppcon | 2018" and "THE C++ CONFERENCE • BELLEVUE, WASHINGTON" is displayed. On the left side, there is a dark grey rectangular area containing the text "VFX Reference Platform" and "SOFTWARE ECOSYSTEM". In the center, the year "2018" is at the top, followed by a bulleted list of software updates: "gcc v4.8.2 -> gcc v6.3.1", "boost 1.5.8 -> boost1.6.1", "Qt 5.6.1 -> unchanged", "Python 2.7.5 -> unchanged", and "C++11 -> C++14". At the bottom right of this central area is the SideFX logo. On the right side of the slide, there is a video frame showing a man, identified as "MARK ELENDT" in a white box, speaking and holding a remote. Below the video frame, the title "Patterns and Techniques Used in the Houdini 3D Graphics Application" is displayed. At the bottom right of the slide, the website "CppCon.org" is shown.

VFX Reference Platform

SOFTWARE ECOSYSTEM

2018

- gcc v4.8.2 -> gcc v6.3.1
- boost 1.5.8 -> boost1.6.1
- Qt 5.6.1 -> unchanged
- Python 2.7.5 -> unchanged
- C++11 -> C++14

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (11/13)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Array Class CUSTOM CONTAINERS

```
template <typename T>
class UT_Array
{
    UT_Array() {}

    void growCapacity(size_t size) {
        if (_array)
            _array = (T *)realloc(_array, size*sizeof(T));
        else
            _array = (T *)malloc(size*sizeof(T));
        _size = size;
    }
}
```

std::trivially_relocatable

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (12/13)

The slide is titled "Data Representation" and "HOUDINI GEOMETRY". It features a central diagram of a green triangle with blue vertices and a thick black outline. To the left is C++ code defining points and triangles. To the right is C++ code defining points, velocity, temperature, and faces.

Left Side (Data Representation):

```
Points = {  
    float3 P[4];  
    float3 Velocity[4];  
    float Temperature[4];  
};  
  
Triangles {  
    vector<point> Vertices[2];  
    string Shader[2];  
};
```

Right Side (Houdini Geometry):

```
Points[] = {  
    P = {{0,0,0},{2,0,0},  
          {1,1,0},{-.2,-9,0}},  
    Velocity = {{1,0,0},{1,0,0},  
                {1,0,0},{1,0,0}},  
    Temperature = {20,20,  
                  20,20},  
}  
  
Faces[] = {  
    Vertices = {{0,2,3},{0,1,2}},  
    Shader = {"green","green"},  
}
```

The slide is part of the "SideFX" series and includes the "cppcon | 2018" logo at the top right. On the right side, there is a video frame showing Mark Elendt speaking, with his name "MARK ELENDT" overlaid. Below the video frame is a summary text: "Patterns and Techniques Used in the Houdini 3D Graphics Application". At the bottom right is the CppCon.org logo.

CppCon 2018: Mark Elendt “Patterns and Techniques Used in the Houdini 3D Graphics Application” (13/13)

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

Data Representation HOUDINI GEOMETRY

```
template <typename POD_T>
class UT_PageArray {
    class PageData {
        POD_T * _data;
    };
    PageData *_pages;

    POD_T &operator[](size_t i) {
        size_t page, offset;
        splitIndex(i, page, offset);
        return _pages[page][offset];
    }

    inline void splitIndex(size_t i, size_t &page, size_t &offset)
    {
        page = i >> PAGE_BITS;
        offset = i & PAGE_MASK;
    }
};
```

SideFX

MARK ELENDT

Patterns and Techniques
Used in the Houdini
3D Graphics Application

CppCon.org

Who is STL? I mean the person, not the library

Reddit

Hey. I'm Stephan T. Lavavej ("Steh-fin Lah-wah-wade"), and I've worked on MSVC's STL since 2007. I've also worked on several Standard proposals that were accepted (notably the transparent operator functors). I filmed a bunch of videos for MS's Channel 9 years ago, introducing various Core Language and Standard Library topics, and I've given talks at C++Now (formerly BoostCon) and CppCon which have been recorded.

Real world problems with #pragma once?

Reddit

Pointer-to-member-functions can be tricky

- ▶ Post
- ▶ Snippet
- ▶ Raymond Chen: Pointers to member functions are very strange animals

Prepare thy Pitchforks: A De-facto Standard Project Layout

- ▶ Early Reddit post
- ▶ Later Reddit post
- ▶ Blog post
- ▶ GitHub repo

Reimplementing NumPy in C++

- ▶ [NumCpp](#)
- ▶ [xtensor](#)

Other linear algebra libraries

- ▶ [Blaze](#)
- ▶ [Eigen](#)
 - ▶ [the official repo](#)
 - ▶ [docs](#)

Visual C++ Team Blog - std::any: How, when, and why

Post

When you need to store an object of an arbitrary type, pull std::any out of your toolbox. Be aware that there are probably more appropriate tools available when you do know something about the type to be stored.

C++ Best Practices, by Jason Turner

[GitHub](#)

Library: SQLite ORM

SQLite ORM light header only library for modern C++

- ▶ [Code](#) (BSD-2-Clause)
 - ▶ C++14
 - ▶ No raw string queries
 - ▶ CRUD support; pure select query support
 - ▶ Custom types binding support
 - ▶ Supports: BLOB - maps to `std::vector<char>` or a custom type;
`FOREIGN KEY`; composite keys; `JOIN`; transactions; `ORDER BY` and `LIMIT`,
`OFFSET`, `GROUP BY / DISTINCT`, `INDEX`, `COLLATE`
 - ▶ Migration functionality
 - ▶ The only dependency - **libsqllite3**
 - ▶ No undefined behaviour - if something goes wrong lib throws an exception
 - ▶ In-memory database support - provide `:memory:` or empty filename

Library: Inja - a template engine for modern C++

► Code

- Licence: MIT
- Header-only
- Uses NLochmann's [JSON library](#)
- [Conan wrapper](#)

```
1 json data;
2 data["name"] = "world";
3 inja::render("Hello {{ name }}!", data); // Returns "Hello world!"
```

Library: C++ REST SDK (formerly Casablanca) by Microsoft

- ▶ **Code**

- ▶ Licence: MIT
- ▶ C++11
- ▶ Supports Windows, Linux, macOS, iOS, Android

The C++ REST SDK is a Microsoft project for cloud-based client-server communication in native code using a modern asynchronous C++ API design. This project aims to help C++ developers connect to and interact with services.

Library: Caffe2 - A New Lightweight, Modular, and Scalable Deep Learning Framework

- ▶ [Website](#)
- ▶ [Code](#)
 - ▶ Licence: Apache-2.0

Tool: Superluminal profiler for Windows

Website

- ▶ Combines sampling and instrumentation
- ▶ Visualizes thread communication flow
- ▶ Kernel-level callstacks
- ▶ Dynamic filtering of areas of interest
- ▶ High frequency sampling (8 KHz)
- ▶ Timeline view, call graph, source view
- ▶ 7-day free trial, then EUR 99/149/289

Conan, vcpkg or build2?

Reddit

- ▶ Pragmatic choice: vcpkg or Conan (they work today and are complete enough)
- ▶ Pragmatic no-brainer choice: vcpkg (it's the simplest and it have more packages ready)
- ▶ Pragmatic but need finer control choice: Conan (it gives more options)
- ▶ (Very) Long term choice: Build2 (shows great promises because it uses a coherent model...)
- ▶ Ideal choice (from the future): help SG15 (the group reflecting on tools vs C++) define interfaces for build systems and dependency managers so that your choice is not impacted by your dependencies choices.

Improving C++ Builds with Split DWARF

Article

```
1 $ g++ -c -g -fPIC -fPIC main.cpp -o main.o  
2 $ g++ main.o -o app
```

Having some fun with higher-order functions

- ▶ Article by Barry Revzin
- ▶ Boost.HOF

Compile-time raytracer by Tristan Brindle

- ▶ [Code](#)
- ▶ [Reddit](#)

Iterators: What Must Be Done?

- ▶ Article

Quote

Andrey Mokhov (@andreymokhov) via Twitter:

*Inside every large program there is a small build system
struggling to get out.*

Twitter



chrisk5000
@chrisk5000

Almost made fun of an electrician today like "is it an electrician's guild rule that you have to perform 5 minutes of ritual complaining about What Was The Previous Electrician Thinking before you're allowed to fix anything?" but then I remembered I'm a software engineer

8,883 Likes 2,464 Retweets

15 Nov 2018 at 17:25 via Twitter Web Client