# Poster: CFIXX Object Type Integrity for C++

Nathan Burow, Purdue University
Derrick McKee, Purdue University
Scott A. Carr, Purdue University
Mathias Payer, Purdue University

## Abstract

C++ relies on object type information for dynamic dispatch and casting. The association of type information to an object is implemented via the virtual table pointer, which is stored in the object itself. As C++ has neither memory nor type safety, adversaries may therefore overwrite an object's type. If the corrupted type is used for dynamic dispatch, the attacker has hijacked the application's control flow. This vulnerability is widespread and commonly exploited. Firefox, Chrome, and other major C++ applications are network facing, commonly attacked, and make significant use of dynamic dispatch. Control-Flow Integrity (CFI) is the state of the art policy for efficient mitigation of control-flow hijacking attacks. CFI mechanisms determine *statically* (i.e., at compile time) the set of functions that are valid at a given call site, based on C++ semantics. We propose an orthogonal policy, *Object Type Integrity* (OTI), that dynamically tracks object types. Consequently, instead of allowing a set of targets for each dynamic dispatch on an object, only the single, correct target for the object's type is allowed.

To show the efficacy of OTI, we present CFIXX, which enforces OTI. CFIXX enforces OTI by *dynamically* tracking the type of each object and enforcing its integrity against arbitrary writes. CFIXX has minimal overhead on CPU bound applications such as SPEC CPU2006 — 4.98%. On key applications like Chromium, CFIXX has negligible overhead on JavaScript benchmarks: 2.03% on Octane, 1.99% on Kraken, and 2.80% on JetStream. We show that CFIXX can be deployed in conjunction with CFI, providing a significant security improvement.

## 1 Reference

This work will appear at NDSS 2018, we are also submitting a poster about the project. Full reference:

Burow, Nathan, Derrick McKee, Scott A. Carr, and Mathias Payer. "CFIXX: Object Type Integrity for C++". In Proceedings of the 2018 Network and Distributed System Security Symposium (2018)

## 2 Conference and DOI

Network and Distributed Systems Security (NDSS) Symposium 2018
18-21 February 2018, San Diego, CA, USA
ISBN 1-1891562-49-5
http://dx.doi.org/10.14722/ndss.2018.23279
www.ndss-symposium.org

# CFIXX: Object Type Integrity for C++

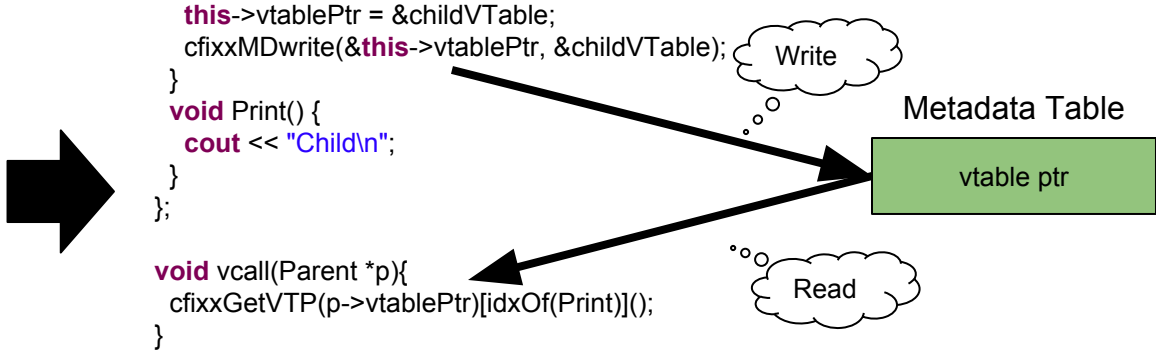
Nathan Burow, Derrick McKee, Scott A. Carr, Mathias Payer
**Purdue University**

## Compiler Transformation

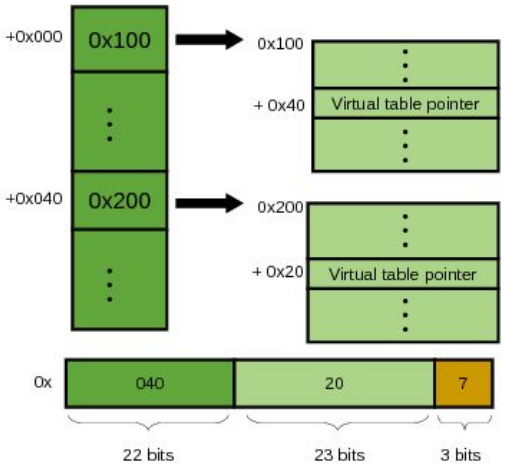
- Prevent all control-flow hijacks, e.g., COOP that subvert C++ dynamic dispatch
- Object Type Integrity (OTI) -- all polymorphic objects have the correct dynamic type
- OTI enforced by integrity protecting virtual table pointers
- 2% Overhead on JS benchmarks with Chromium

```
#include <iostream>

class Child : public Parent {
  Child(){
    this->vtablePtr = &childVTable;
  }
  void Print() {
    cout << "Child\n";
  }
};

void vcall(Parent *p){
  p->vtablePtr[idxOf(Print)]();
}
int main(int argc, void **argv){
  Parent *p = new Parent();
  Child *c = new Child();
  vcall(p);
  vcall(c);
}
```

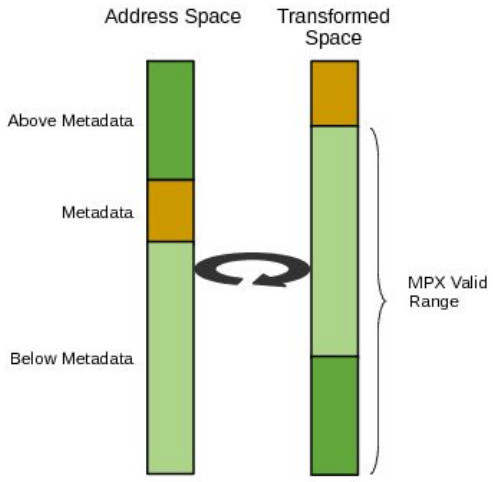
```
#include <iostream>

class Child : public Parent {
  Child(){
    this->vtablePtr = &childVTable;
    cfixxMDwrite(&this->vtablePtr, &childVTable);
  }
  void Print() {
    cout << "Child\n";
  }
};

void vcall(Parent *p){
  cfixxGetVTP(p->vtablePtr)[idxOf(Print)]();
}

int main(int argc, void **argv){
  Parent *p = new Parent();
  Child *c = new Child();
  vcall(p);
  vcall(c);
}
```




## Metadata Design




## MPX Protection




hexhive

Burow, Nathan, Derrick McKee, Scott A. Carr, and Mathias Payer. ”CFIXX: Object Type Integrity for C++”. NDSS (2018)