

- Only works if there are no *pointers* or *references* to a person

• send information when passed to the

• Delete operation new to provided application

• Delete operation & avoid taking the address

• Doesn't stop references

- Doesn't stop references/pointers when used as a member in another object

• Would require virtual checking/static analysis



```
struct person
{
    //...
    // Wrapped __person functions
    //...

    // Prevents dynamic allocation
    void* operator new(size_t) = delete;
    void* operator new[](size_t) = delete;

    // Prevents taking the address
    person* operator&() = delete;
    const person* operator&() const = delete;
};
```



Copy on Write structs



Copy on Write `structs`

- Only works if there are no *pointers* or *references* to a **person**
- **send** enforces this when passed to a thread
- Delete **operator new** to avoid heap allocations
- Delete **operator&** to avoid taking the address
- Doesn't stop references
- Doesn't stop references/pointers when used as a member in another object
- Would require viral checking/static analysis

```
struct person
{
    //...
    // Wrapped __person functions
    //...

    // Prevents dynamic allocation
    void* operator new(size_t) = delete;
    void* operator new[](size_t) = delete;

    // Prevents taking the address
    person* operator&() = delete;
    const person* operator&() const = delete;
};
```

