

Reverse Engineering Objective-C

Jeff Hui

**The process of understanding a
system through analysis of its
structure, function, and operation**

Wikipedia

It's not Debugging

It's Reverse Engineering!

**The process of understanding a
system without its original
source**

Why?



Learn



Disable

Why?



Learn



Disable



Extend

Archeology

Knowledge Intense

Optimizations

Encryption

Security

Operating Systems

Language Runtimes

Specifications

Assembly

Compilers

Hardware

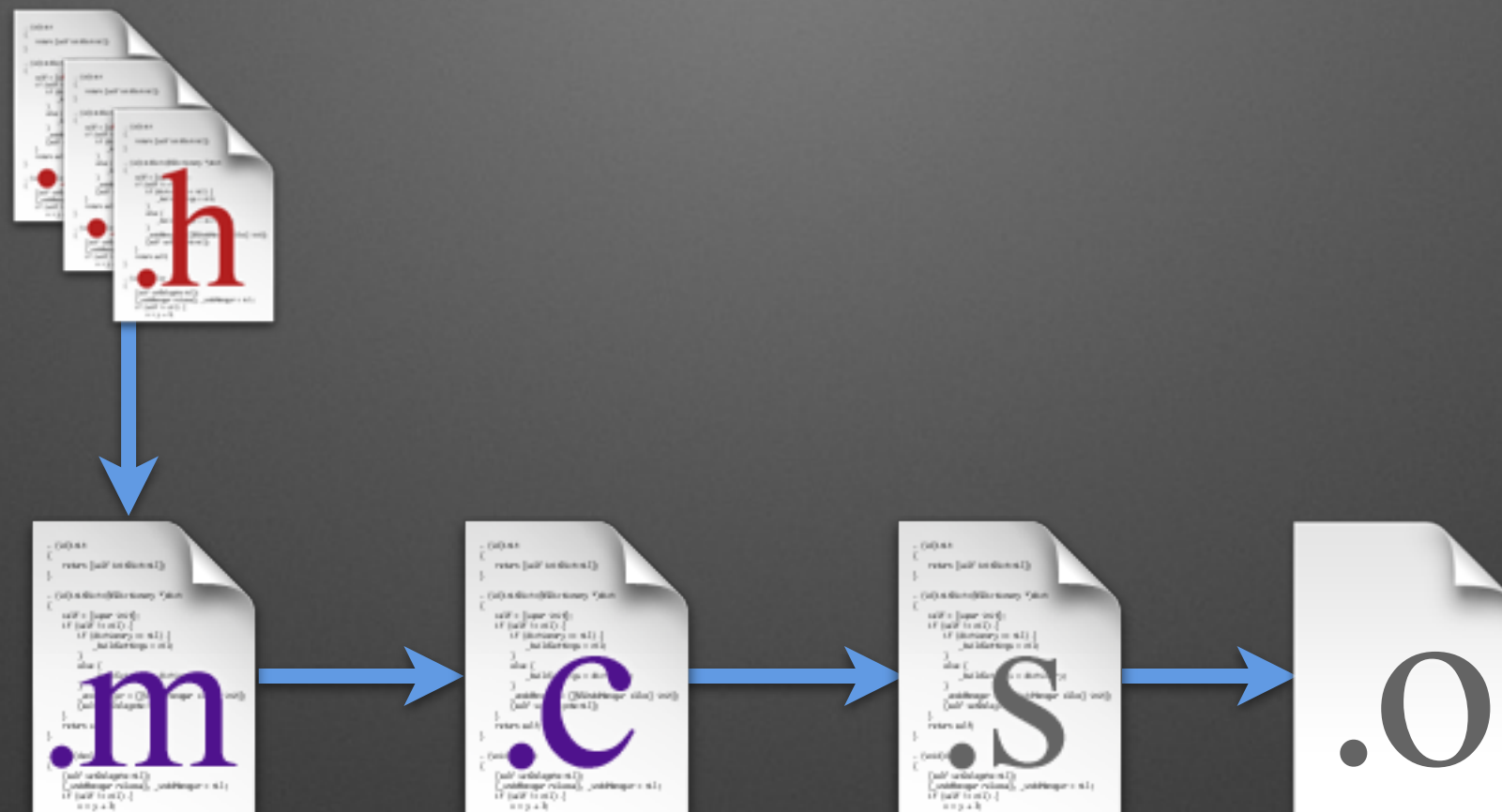
Code Injection

Language Matters

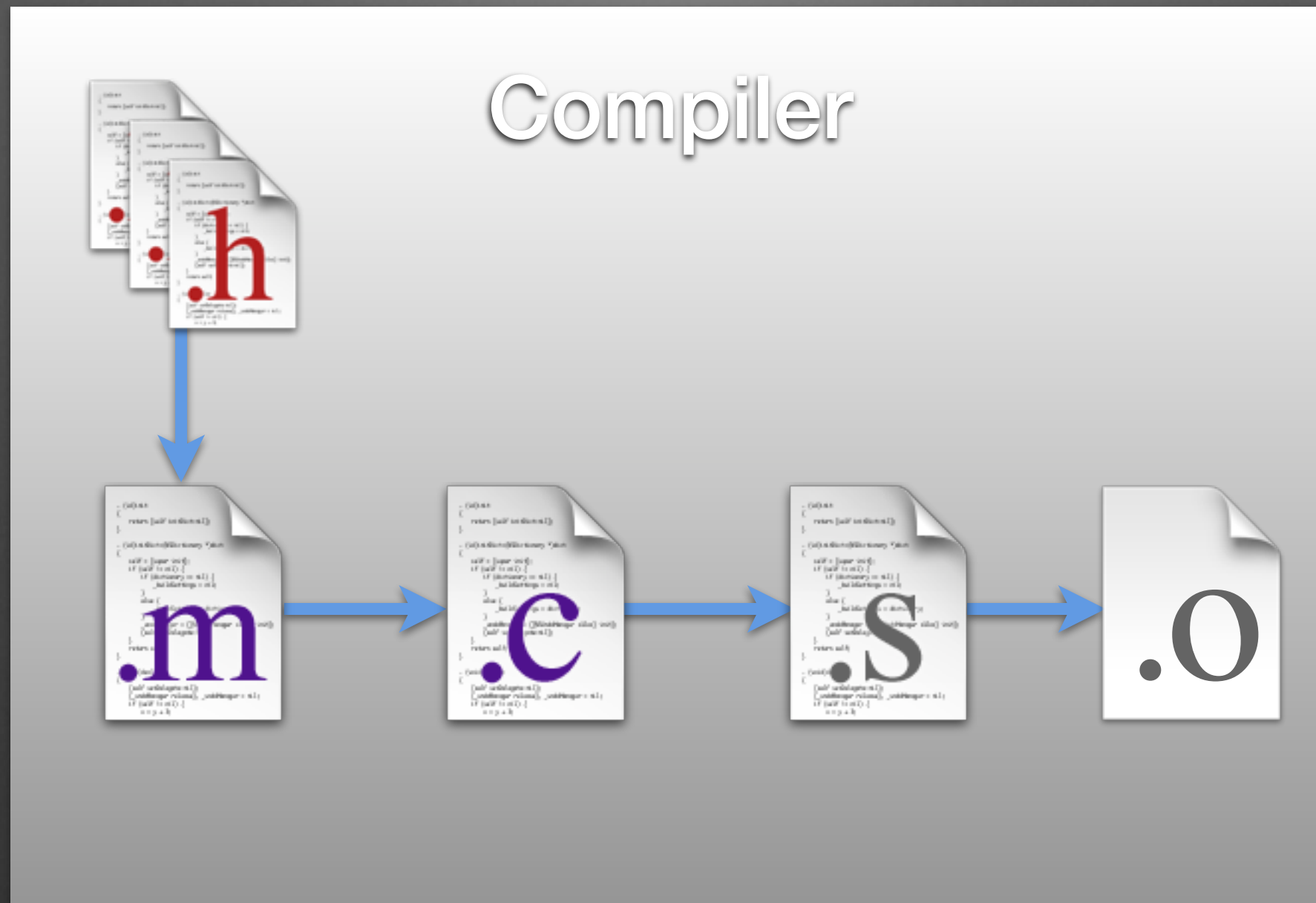
JAVASCRIPT	Source Code	REPL	Dynamic Dispatch
PYTHON, RUBY, JAVA, ETC.	Bytecode	Reflective	Dynamic Dispatch
C	Static Dispatch	Exploitable	No Reflection
OBJECTIVE-C	Dynamic Dispatch	C-compatible	Partially-Reflective

Objective-C

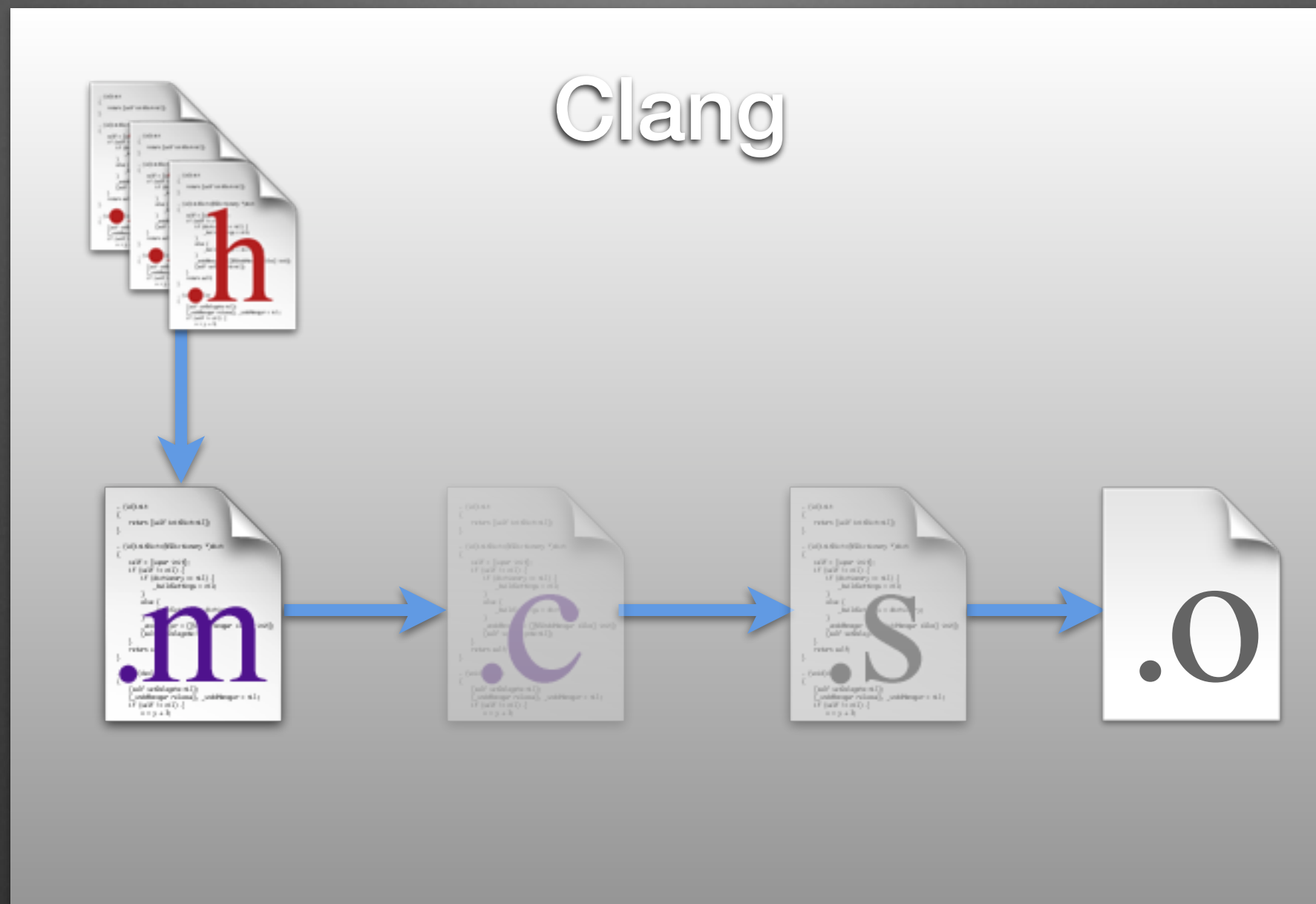
Building an App



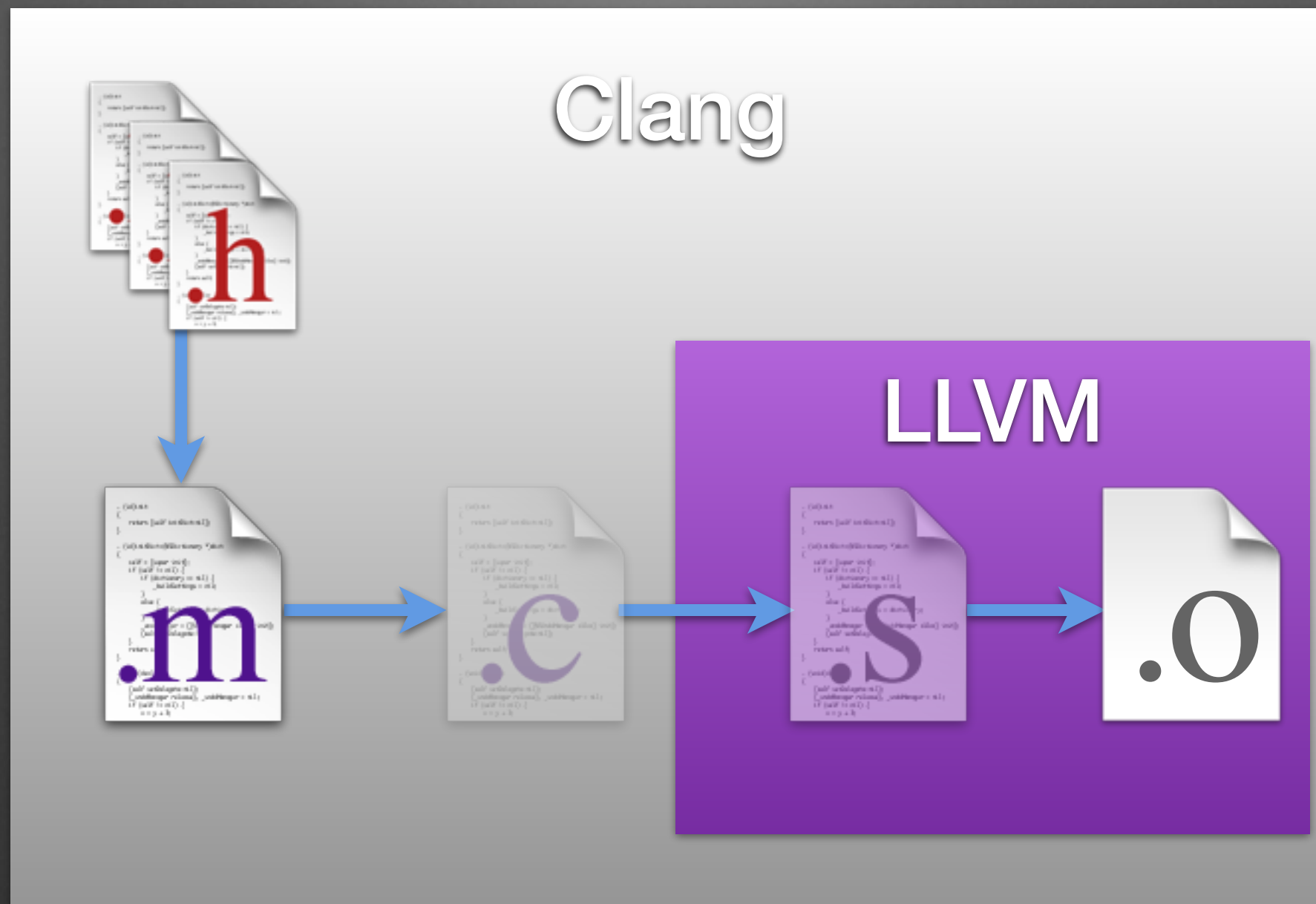
Building an App



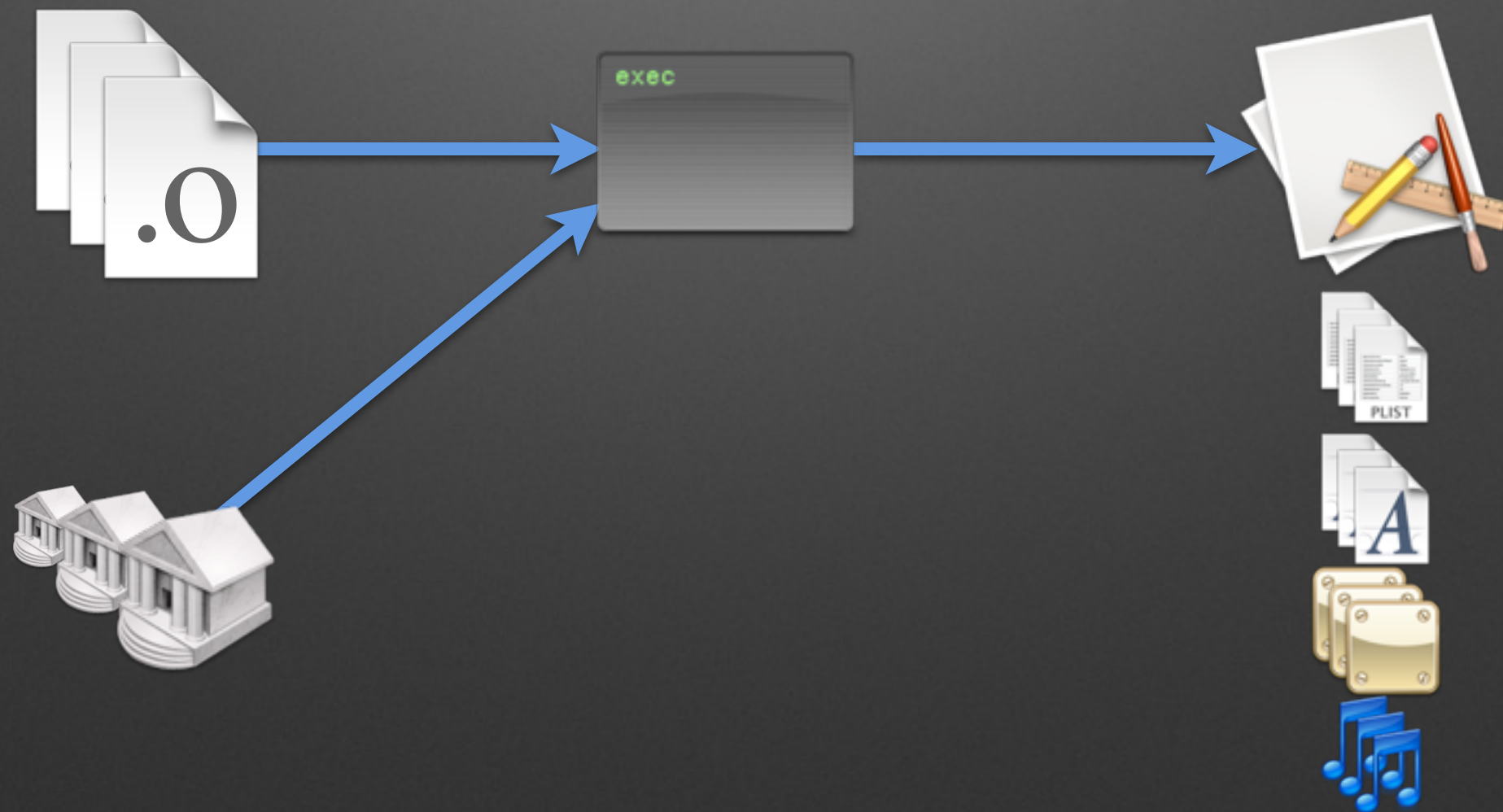
Building an App



Building an App

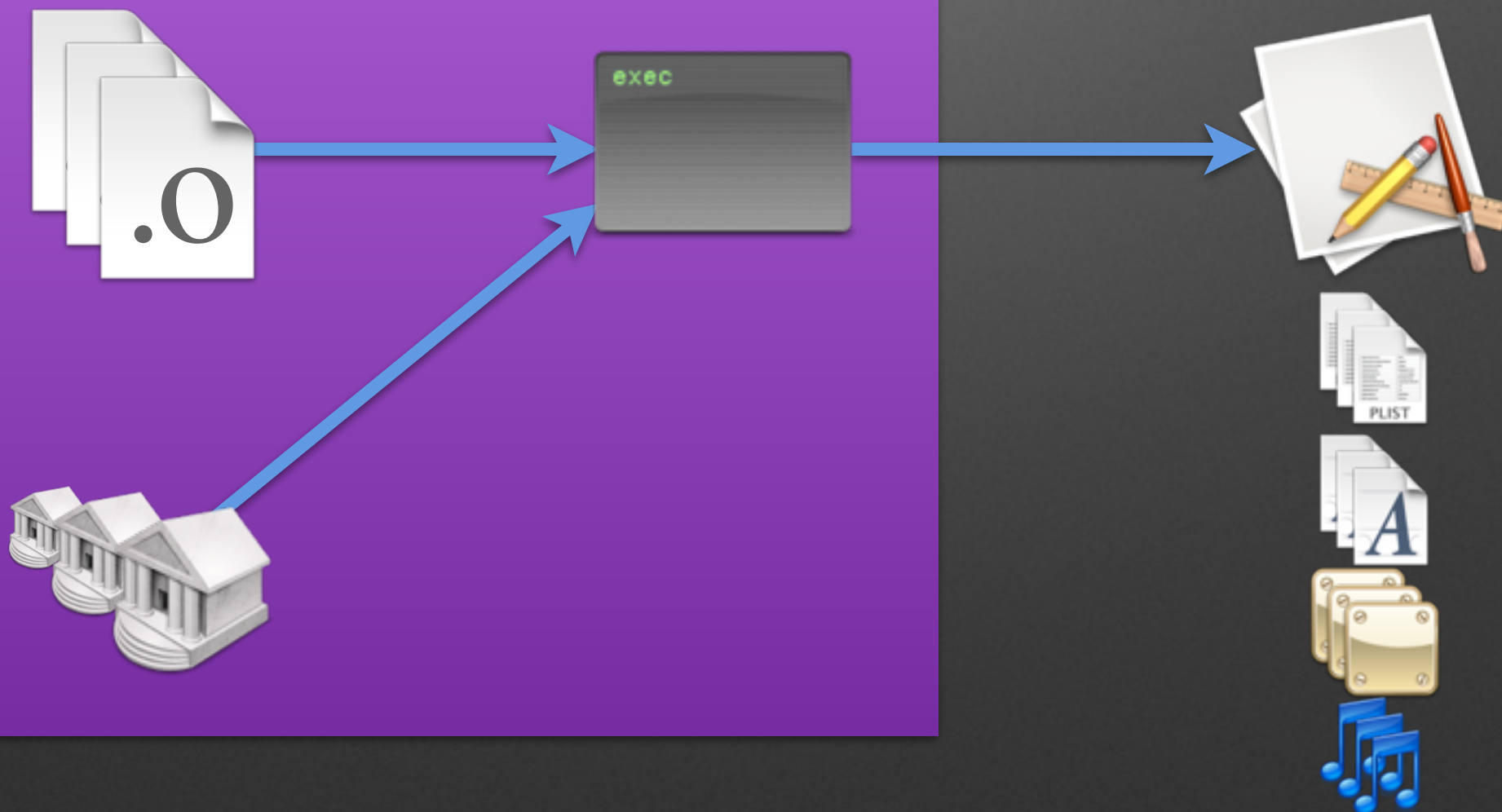


Building an App

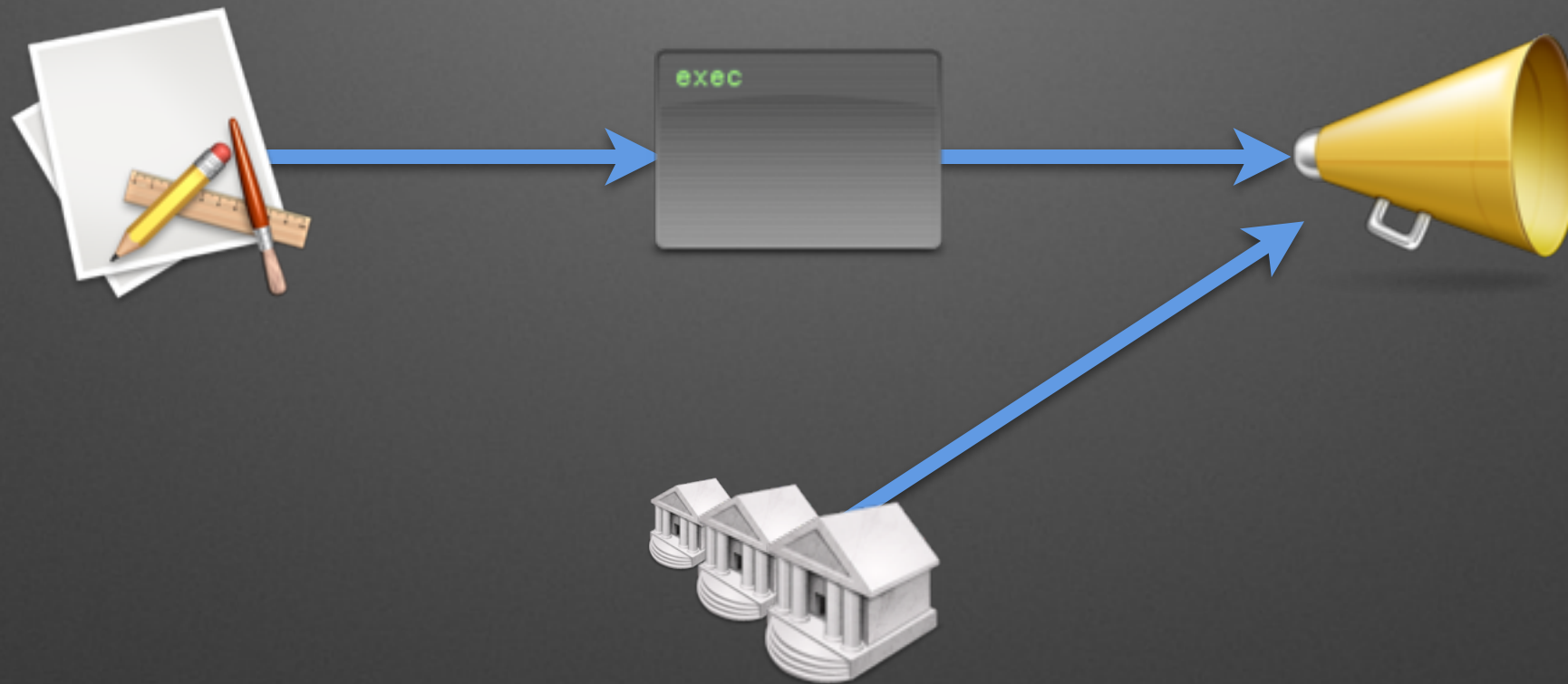


Building an App

Static Linker



Running an App

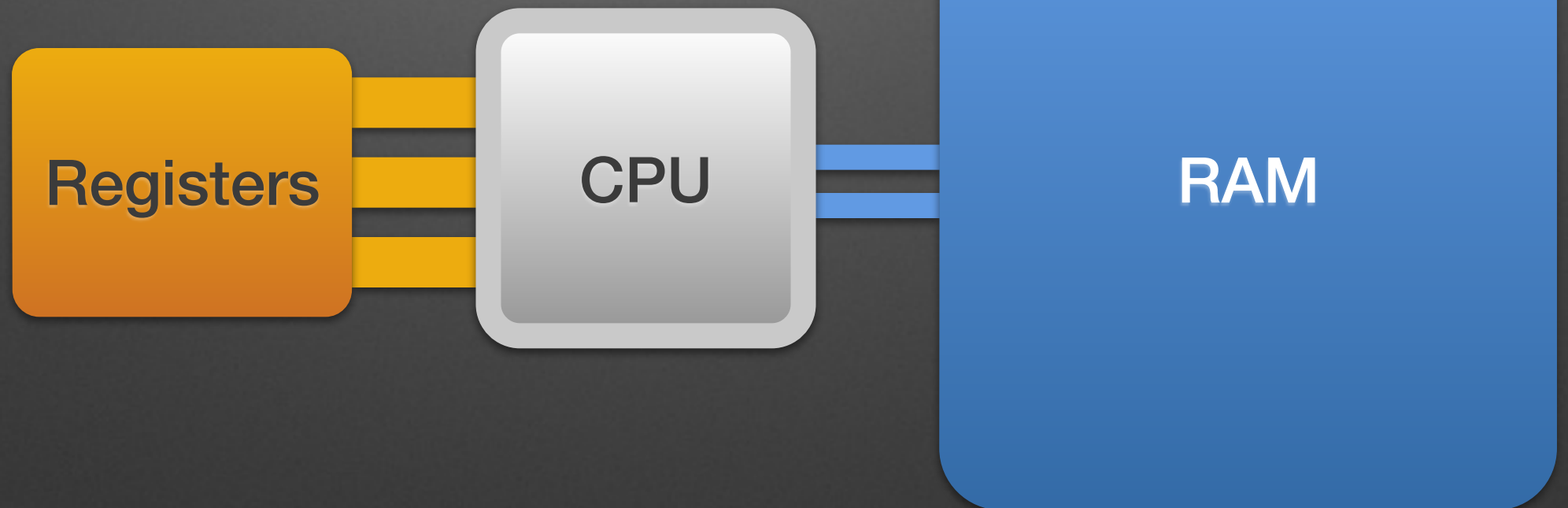


Dynamic Linking

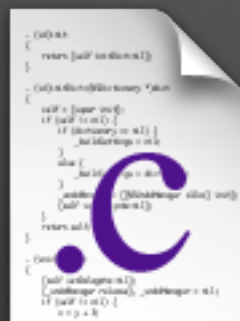
Running an App



Running an App



C to Assembly



```
int add(int a, int b) {  
    return a + b;  
}
```


_add:

push ebp

mov ebp, esp

sub esp, 0x8

mov eax, dword [ss:ebp-0x8+arg_4]

mov ecx, dword [ss:ebp-0x8+arg_0]

mov dword [ss:ebp-0x8+var_4], ecx

mov dword [ss:ebp-0x8+var_0], eax

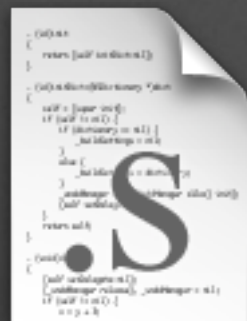
mov eax, dword [ss:ebp-0x8+var_4]

add eax, dword [ss:ebp-0x8+var_0]

add esp, 0x8

pop ebp

ret

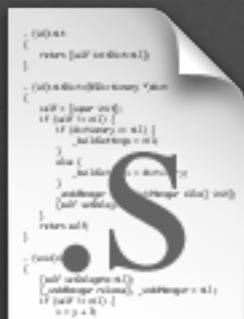


`_add:`

```
push ebp
mov ebp, esp
sub esp, 0x8
```

Prologue

```
mov eax, dword [ss:ebp-0x8+arg_4]
mov ecx, dword [ss:ebp-0x8+arg_0]
mov dword [ss:ebp-0x8+var_4], ecx
mov dword [ss:ebp-0x8+var_0], eax
mov eax, dword [ss:ebp-0x8+var_4]
add eax, dword [ss:ebp-0x8+var_0]
add esp, 0x8
pop ebp
ret
```



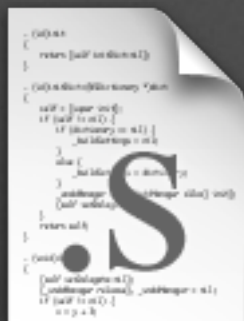
`_add:`

```
push ebp
mov ebp, esp
sub esp, 0x8
```

Prologue

```
mov eax, dword [ss:ebp-0x8+arg_4]
mov ecx, dword [ss:ebp-0x8+arg_0]
mov dword [ss:ebp-0x8+var_4], ecx
mov dword [ss:ebp-0x8+var_0], eax
mov eax, dword [ss:ebp-0x8+var_4]
add eax, dword [ss:ebp-0x8+var_0]
```

```
add esp, 0x8
pop ebp
ret
```



`_add:`

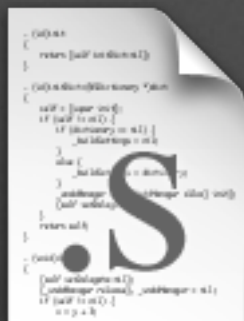
```
push ebp
mov ebp, esp
sub esp, 0x8
```

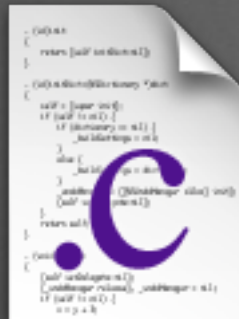
Prologue

```
mov eax, dword [ss:ebp-0x8+arg_4]
mov ecx, dword [ss:ebp-0x8+arg_0]
mov dword [ss:ebp-0x8+var_4], ecx
mov dword [ss:ebp-0x8+var_0], eax
mov eax, dword [ss:ebp-0x8+var_4]
add eax, dword [ss:ebp-0x8+var_0]
```

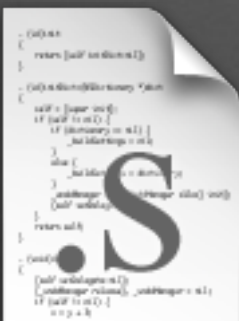
```
add esp, 0x8
pop ebp
ret
```

Epilogue

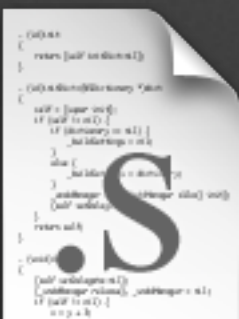




```
add(1, 2);
```



```
mov dword[ss:esp], 0x1
mov dword[ss:esp+0x4], 0x2
call _add
```



```
mov rbx, 0x1
mov rbi, 0x2
call _add
```


Calling Conventions (x86_64)

- rdi arg1
- rsi arg2
- rdx arg3
- rcx arg4
- r8 arg5
- r9 arg6

Compiling Objective-C

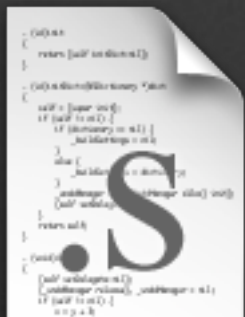


@implementation Calculator

- (int)add:(int)a and:(int)b { return a + b; }
@end



int methImpl_Calculator_add_and_
 Person *self, SEL _cmd, int a, int b
) { return a + b; }

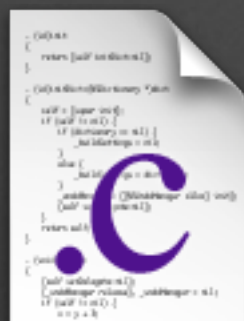


methImpl_Calculator_add_and_

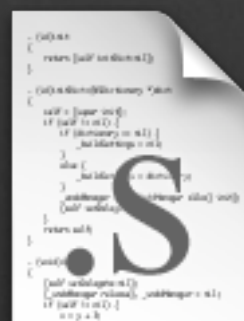
Compiling Objective-C



[obj call];



objc_msgSend(
obj, sel_registerName("call")
);



mov rdi, esp
mov rsi, qword [ds:objc_sel_call]
call qword [ds:imp___got___objc_msgSend]

Calling Conventions (x86_64)

• rdi	self
• rsi	_cmd
• rdx	arg1
• rcx	arg2
• r8	arg3
• r9	arg4
• rbp	(stack register)

Let's Reverse Engineer



Hopper



Hopper

methImpl_Person_daysUntilBirthday

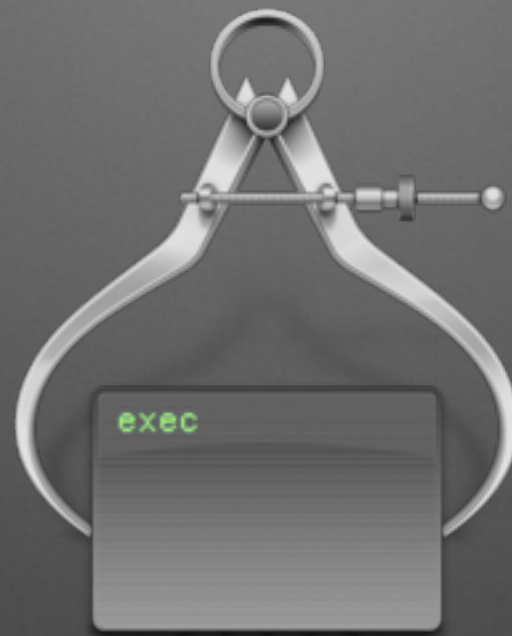


class-dump



class-dump

All objc objects are ids



LLDB


```
jeff@terminator ~/workspace> █
```

```
I
```



```
jeff@terminator ~/workspace> lldb  
(lldb) attach Xcode  
Process 34751 stopped  
Executable module set to "/Applications/Xcode.app/Contents/MacOS/Xcode".  
Architecture set to: x86_64-apple-macosx.  
(lldb) █
```

}

S/Xcode".

Architecture set to: x86_64-apple-macosx.

(lldb) break set --name '-[IDEWorkspace _setupExecutionEnvironment]'

Breakpoint 1: where = IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment], address = 0x0000000102e6bc0c

(lldb) con

Process 34751 resuming

Process 34751 stopped

* thread #1: tid = 0x85fe3, 0x0000000102e6bc0c IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment], queue = 'com.apple.main-thread, stop reason = breakpoint 1.1

frame #0: 0x0000000102e6bc0c IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment]

IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment]:

-> 0x102e6bc0c: pushq %rbp
0x102e6bc0d: movq %rsp, %rbp
0x102e6bc10: pushq %r15
0x102e6bc12: pushq %r14

(lldb) █


```
frame #10: 0x00007fff8d433b14 CoreFoundation`__CFRunLoopRun
+ 1636
frame #11: 0x00007fff8d433275 CoreFoundation`CFRunLoopRunSp
ecific + 309
frame #12: 0x00007fff86139f0d HIToolbox`RunCurrentEventLoop
InMode + 226
frame #13: 0x00007fff86139cb7 HIToolbox`ReceiveNextEventCom
mon + 479
frame #14: 0x00007fff86139abc HIToolbox`_BlockUntilNextEven
tMatchingListInModeWithFilter + 65
frame #15: 0x00007fff848da28e AppKit`_DPSNextEvent + 1434
frame #16: 0x00007fff848d98db AppKit`-[NSApplication nextEv
entMatchingMask:untilDate:inMode:dequeue:] + 122
frame #17: 0x00007fff848cd9cc AppKit`-[NSApplication run] +
553
frame #18: 0x00007fff848b8803 AppKit`NSApplicationMain + 94
0
frame #19: 0x00007fff8d3c25fd libdyld.dylib`start + 1
frame #20: 0x00007fff8d3c25fd libdyld.dylib`start + 1
(lldb) █
```



```
0x102e6bc4b: callq 0x1031370e0 ; symbol stub
for: objc_retainAutoreleasedReturnValue
0x102e6bc50: movq %rax, %r12
0x102e6bc53: movq 5016414(%rip), %rsi ; "initWithWo
rkspaceArena:"
0x102e6bc5a: movq %r15, %rdi
0x102e6bc5d: movq %r12, %rdx
0x102e6bc60: callq *%r13
0x102e6bc63: movq %rax, %r15
0x102e6bc66: movq 5016403(%rip), %rsi ; "setExecuti
onEnvironment:"
0x102e6bc6d: movq %r14, %rdi
0x102e6bc70: movq %r15, %rdx
0x102e6bc73: callq *%r13
0x102e6bc76: movq 4348363(%rip), %rbx ; (void *)0x0
0007fff8a2880d0: objc_release
0x102e6bc7d: movq %r15, %rdi
0x102e6bc80: movq %rbx, %rax
0x102e6bc83: callq *%rax
0x102e6bc85: movq %r12, %rdi
```



```
0x102e6bc9e:  nop
0x102e6bc9f:  nop
(lldb) break set --addr 0x102e6bc73
Breakpoint 2: where = IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment] + 103, address = 0x0000000102e6bc73
(lldb) con
Process 34751 resuming
Process 34751 stopped
* thread #1: tid = 0x85fe3, 0x0000000102e6bc73 IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment] + 103, queue = 'com.apple.main-thread, stop reason = breakpoint 2.1
    frame #0: 0x0000000102e6bc73 IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment] + 103
IDEFoundation`-[IDEWorkspace _setupExecutionEnvironment] + 103:
-> 0x102e6bc73:  callq  *%r13
    0x102e6bc76:  movq    4348363(%rip), %rbx                ; (void *)0x0007fff8a2880d0: objc_release
    0x102e6bc7d:  movq    %r15, %rdi
    0x102e6bc80:  movq    %rbx, %rax
(lldb) █
```



```
-> 0x102e6bc73:  callq  *%r13
    0x102e6bc76:  movq   4348363(%rip), %rbx           ; (void *)0x0
0007fff8a2880d0:  objc_release
    0x102e6bc7d:  movq   %r15, %rdi
    0x102e6bc80:  movq   %rbx, %rax
```

(lldb) reg read

General Purpose Registers:

```
rax = 0x00007f8b63f60cd0
rbx = 0x0000000000000000
rcx = 0x0000000000000084
rdx = 0x00007f8b63f60cd0
rdi = 0x00007f8b63c2f270
rsi = 0x00000001031a8efa "setExecutionEnvironment:"
rbp = 0x00007fff5d818220
rsp = 0x00007fff5d8181f0
r8  = 0x00000000000001fff
r9  = 0xffff80749eb2e80f
r10 = 0x000000010df86100
r11 = 0x00007fff749db301 (void *)0x6000007fff742668
r12 = 0x00007f8b614d17f0
```


The “Hard” Part

```

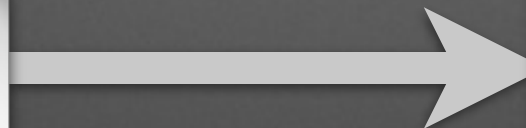
- (id)ns
{
    return [self ns];
}

- (id)nsDictionaryWithName:(NSString *)name
{
    NSString *name = [name lowercaseString];
    if ([name isEqualToString:@""]) {
        return nil;
    }
    NSString *key = [name stringByAppendingString:@"Dictionary"];
    return [self objectForKey:key];
}

- (id)ns
{
    return [self ns];
}

- (id)ns
{
    return [self ns];
}

```



```

- (id)ns
{
    return [self ns];
}

- (id)nsDictionaryWithName:(NSString *)name
{
    NSString *name = [name lowercaseString];
    if ([name isEqualToString:@""]) {
        return nil;
    }
    NSString *key = [name stringByAppendingString:@"Dictionary"];
    return [self objectForKey:key];
}

- (id)ns
{
    return [self ns];
}

- (id)ns
{
    return [self ns];
}

```

Code Injection

Code Injection



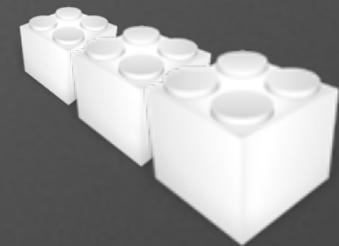
Binary
Patching



Dynamic
Patching



Dynamic
Linking



Plugin

Image Credits

- Skull And Crossbones designed by Anton Outkine from the Noun Project
- Icons from Apple's Terminal, Xcode, SystemInformation, CoreServices
- Icon from Hopper

Questions?

- Objective-C Type Encodings
- MACH-O ABI & DYLB
- mach_inject & mach_override - C-Function Overriding
- Objective-C Blocks ABI
- LLVM Compiler Infrastructure Project
- objc4 - Objective-C 2.0 OSS project
- ISAs - i386, x86_64, arm, arm64
- Darwin - iOS & OSX OSS Kernel

Code Obfuscation

- Ruins Stacktraces
- Breaks Reflection
- Doesn't stop a Debugger
- Only slows down initially

What about FairPlay?

What about FairPlay?

Encryption is only on disk

How do I stop this?

You can't

Make it more difficult

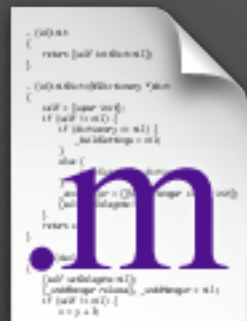
- Bug Debuggers
- Partial Code Load
- Integrity checks
- Remote Server
- Don't DRY the protection code
- Obfuscate protection code

Compiling Objective-C Blocks

Compiling Objective-C

Compiling Objective-C

```
- (int)daysUntilBirthday {  
    int v = 1;  
    return (^int(int i){  
        return i + v;  
    })(2);  
}
```



Compiling Objective-C

```
- (int)daysUntilBirthday {  
    int v = 1;  
    return (^int(int i){  
        return i + v;  
    })(2);  
}
```



```
int __22-[Person daysUntilBirthday]__block_invoke(  
    struct __block_literal_1 *,  
    int i  
) {}
```



Compiling Objective-C

struct __block_literal_1

Class isa

int flags

int reserved

void (*invoke)(__block_literal_1*, ...);

struct __block_descriptor_1
*descriptor

const int v

struct __block_descriptor_1

unsigned long int size

void (*copy)(void*, void*)

void (*release)(void *)

const char *signature

Compiling Objective-C

Objective-C “Object”

struct __block_literal_1

Class isa

int flags

int reserved

void (*invoke)(__block_literal_1*, ...);

struct __block_descriptor_1
*descriptor

const int v

struct __block_descriptor_1

unsigned long int size

void (*copy)(void*, void*)

void (*release)(void *)

const char *signature

Compiling Objective-C

Objective-C “Object”

struct __block_literal_1
Class isa
int flags
int reserved
void (*invoke)(__block_literal_1*, ...);
struct __block_descriptor_1 *descriptor
const int v

struct __block_descriptor_1
unsigned long int size
void (*copy)(void*, void*)
void (*release)(void *)
const char *signature

Block Type Info

Compiling Objective-C

Objective-C “Object”

struct __block_literal_1
Class isa
int flags
int reserved
void (*invoke)(__block_literal_1*, ...);
struct __block_descriptor_1 *descriptor
const int v

struct __block_descriptor_1
unsigned long int size
void (*copy)(void*, void*)
void (*release)(void *)
const char *signature

Block Type Info
Enclosed Variable