

Secrets  
&  
Lies



Public domain, courtesy NARA

English  
Letter

English  
word

Navajo  
Word

C

Cat

MOASI

D

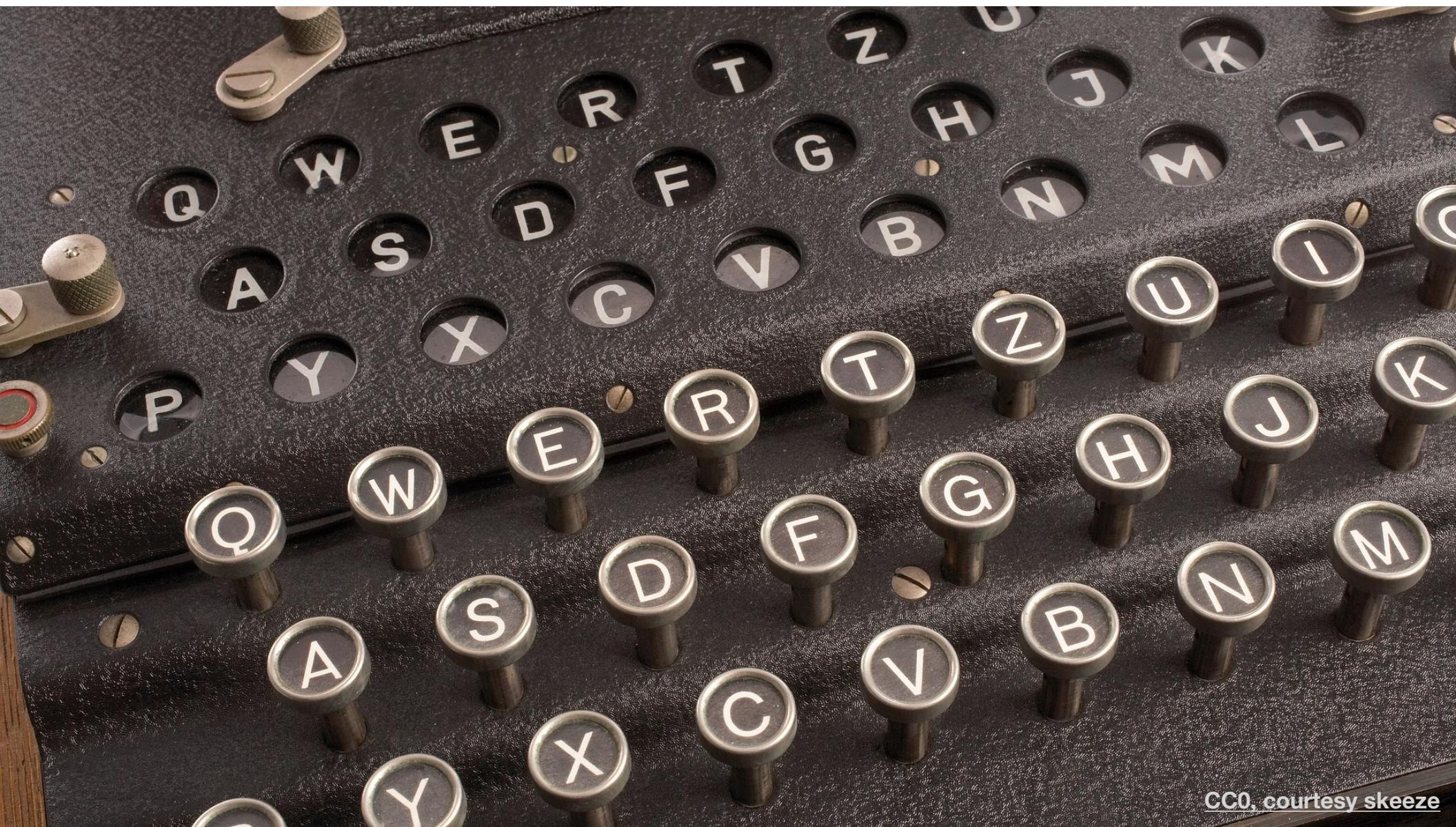
Dog

LHA-CHA-  
EH

E

EIK

DZEH



CC0, courtesy skeeze

Security  
Through  
Obscurity

JOURNAL  
DES  
SCIENCES MILITAIRES.

---

*Janvier 1883.*

---

LA CRYPTOGRAPHIE MILITAIRE.

# LA CRYPTOGRAPHIE MILITAIRE. (1883)

- The system must be practically, if not mathematically, indecipherable;
- It should not require secrecy, and it should not be a problem if it falls into enemy hands;
- It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will;
- It must be applicable to telegraph communications;
- It must be portable, and should not require several persons to handle or operate;
- Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

Security through  
obscurity is  
no

Security



[Public domain, courtesy Wikimedia](#)

**“Rogues are very keen in their profession, and know already much more than we can teach them.”**

*-Alfred Charles Hobbs*



lockpick deadbolt



About 15,500 results



1:29

## How to Pick a Front Door Lock Deadbolt EASY

knifeandbladereviews • 357K views • 5 years ago

Easiest way to pick a deadbolt lock on a front door.



0:46

## Lock picking a deadbolt fast!

charlieaf92 • 99K views • 7 years ago

A close up look at picking a brand name (kwikset) deadbolt lock. NEW VIDEO POSTED: Step by step instructions to pick your first ...



## How to Pick a Lock With Hairpins

Nightowl101 • 10M views • 4 years ago

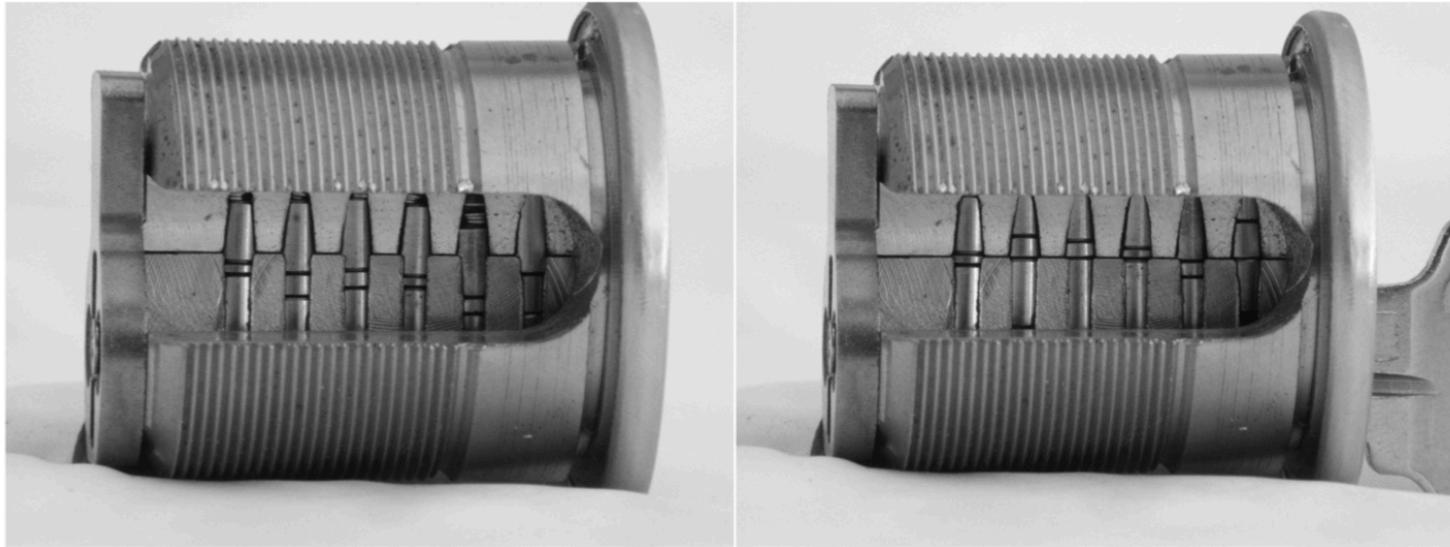


Figure 4: A master keyed pin tumbler lock. *Left:* Each of the six pin stacks has two cuts. *Right:* With the correct *change key* inserted, one of the cuts on each pin stack is aligned at the shear line. Observe that the other cut is sometimes above and sometimes below the shear line.

positions, although the positions will vary (rotate) from lock to lock. Both these schemes can implement a directed graph with several levels of master keys: “sub-master” keys that open a subset of locks in the system and “grand master” keys that open more<sup>1</sup>. The highest-level master key, which opens all locks in a multi-level system, is sometimes called the *Top Master Key (TMK)*.

Master keying has long been understood to reduce security in several important ways. First, of course, the master key represents a very valuable target; compromise of the master key compromises the entire

**“Rogues are very keen in their profession, and know already much more than we can teach them.”**

*-Alfred Charles Hobbs*

Security through  
obscurity is  
no

Security

Security through  
obscurity is  
~~no~~ a thin layer of  
Security

The  
Badlands of  
Security

Obscurity ???

Obfu - what?

sskret



crypto



$b=0 : k \leq K, f \in E(k, \cdot)$

$\phi = 1 : f \in \text{Perms}[X]$

$b$   
↓



$x_1, x_2, \dots, x_g$

$f(x_1), f(x_2), \dots, f(x_g)$



$|b' - b| \approx 0 \Rightarrow [E \text{ is a PRP}]$

$b' \in \{0, 1\}$

A CRYPTO NERD'S  
IMAGINATION:

HIS LAPTOP'S ENCRYPTED.  
LET'S BUILD A MILLION-DOLLAR  
CLUSTER TO CRACK IT.

NO GOOD! IT'S  
4096-BIT RSA!

BLAST! OUR  
EVIL PLAN  
IS FOILED!



WHAT WOULD  
ACTUALLY HAPPEN:

HIS LAPTOP'S ENCRYPTED.  
DRUG HIM AND HIT HIM WITH  
THIS \$5 WRENCH UNTIL  
HE TELLS US THE PASSWORD.

GOT IT.



Raising  
the  
Bar

# Racing the Bear



fighting  
the  
dots

```
$ strings -n 8 HackMe
UIApplicationLaunchOptionsKey
ViewController
AppDelegate
_TtC6HackMe14ViewController
v24@0:8@16
@32@0:8@16@24
@24@0:8@16
T@"WKWebView",N,W,VwebView
https://o0rq0uwei.h.execute-api.us-east-1.amazonaws.com/secrets/secrets?token=
Ha!U'llNever(Find)Me
TYYLWKMUAPZHLYXJMPUW
TlqI7CIrzHp4/pY1wpRErEytls0IKeS4BzD9/3quqQ=
_TtC6HackMe11AppDelegate
B32@0:8@16@24
...
```

```
$ strings -n 8 HackMe
UIApplicationLaunchOptionsKey
ViewController
AppDelegate
_TtC6HackMe14ViewController
v24@0:8@16
@32@0:8@16@24
@24@0:8@16
T@"WKWebView",N,W,VwebView
https://o0rq0uwei.h.execute-api.us-east-1.amazonaws.com/secrets/secrets?token=Ha!U'llNever\(Find\)Me
TYYLWKMUAPZHLYXJMPUW
TlqI7CIrzHp4/pY1wpRErEytls0IKeS4BzD9/3quqQ=
_TtC6HackMe11AppDelegate
B32@0:8@16@24
...
```

HackMe.hop

Labels Proc. Str ☆ ⓘ

hackme

Tag Scope

Address	Type	Name
0x100004d94	P	-[HackMe.ViewController webView]
0x100004db4	P	-[HackMe.ViewController setWebView:]
0x100005040	P	-[HackMe.ViewController send:]
0x1000051c8	P	-[HackMe.ViewController initWithNibName:bundle:]
0x10000541c	P	-[HackMe.ViewController initWithCoder:]
0x100005510	P	-[HackMe.ViewController .cxx_destruct]
0x100005d80	P	-[HackMe.AppDelegate window]
0x100005db0	P	-[HackMe.AppDelegate setWindow:]
0x100005de4	P	-[HackMe.AppDelegate application:didFinishLaunching:]
0x100005ec0	P	-[HackMe.AppDelegate applicationWillTerminate:]
0x100005f24	P	-[HackMe.AppDelegate init]
0x100005fe4	P	-[HackMe.AppDelegate .cxx_destruct]

```

loc_100005038:
000100005038    brk      #0x1
00010000503c    brk      #0x1
-[_TtC6HackMe14ViewController send:]:
000100005040    sub      sp, sp, #0x40
000100005044    stp      x20, x19, [sp, #0x20]
000100005048    stp      x29, x30, [sp, #0x30]
00010000504c    add      x29, sp, #0x30
000100005050    mov      x19, x2
000100005054    mov      x20, x0
000100005058    cbz      x19, loc_100005070

00010000505c    mov      x0, x19
000100005060    bl       imp_stubs_swift_getObject
000100005064    str      x0, [sp, #0x80 + var_68]
000100005068    str      x19, [sp, #0x80 + var_80]
00010000506c    b        loc_1000050a4

loc_100005070:
000100005070    nop
000100005074    ldr      x0, =0x0
000100005078    cbnz   x0, loc_10000509c
00010000507c    nop

```

483 labels

Address 0x100005040, Segment \_\_TEXT, -[\_TtC6HackMe14ViewController send:] + 0, Section \_\_text, file offset 0x5040 - Alt+Double Click to follow link in a new pane

HackMe.hop

Labels Proc. Str ⌂

Search

Tag Scope

```

_TtC6HackMe14ViewController
@16@0:8
v24@0:8@16
@32@0:8@16@24
@24@0:8@16
@?
webView
T@["WKWebView"],N,W,VwebView
Ha!U!INever(Find)Me
TYYLWKMUAPZHLXJMPUW
Tlql7ClrzHp4/pY1wpRERytIqs0IKeS4BzD9/3quQ=
https://o0rq0uweiH.execute-api.us-east-1.amazonaws.com/secrets/secrets...
_TtC6HackMe11AppDelegate
B32@0:8@16@24
window
T@["UIWindow"],N,&Vwindow
UIApplicationDelegate
B48@0:8@16@24@32@40
B40@0:8@16@24@32
v40@0:8@16q24d32
v32@0:8@16q24
v56@0:8@16(CGRect={CGPoint=dd}{CGSize=dd})24
v32@0:8@16@24
v48@0:8@16@24@32@?40
v56@0:8@16@24@32@40@?24
162 strings

```

loc\_100005038:

000100005038	brk	#0x1
00010000503c	brk	#0x1

- [\_TtC6HackMe14ViewController send]:

000100005040	sub	sp, sp, #0x40
000100005044	stp	x20, x19, [sp, #0x20]
000100005048	stp	x29, x30, [sp, #0x30]
00010000504c	add	x29, sp, #0x30
000100005050	mov	x19, x2
000100005054	mov	x20, x0
000100005058	cbz	x19, loc_100005070

---

00010000505c	mov	x0, x19
000100005060	bl	imp_stubs_swift_getObject
000100005064	str	x0, [sp, #0x80 + var_68]
000100005068	str	x19, [sp, #0x80 + var_80]
00010000506c	b	loc_1000050a4

---

loc\_100005070:

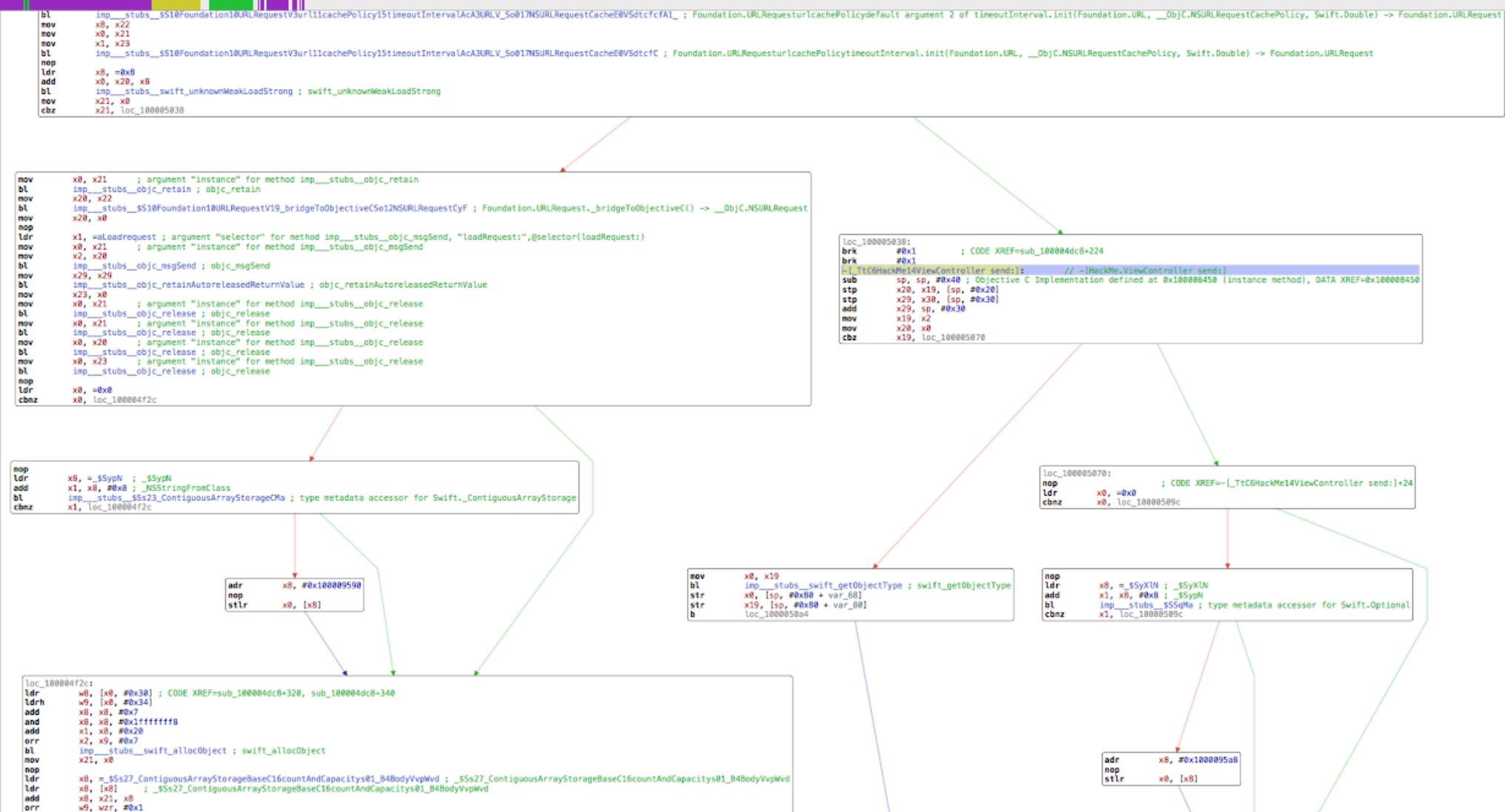
000100005070	nop	
000100005074	ldr	x0, =0x0
000100005078	cbnz	x0, loc_10000509c

---

00010000507c

Address 0x100005040, Segment \_\_TEXT, -[\_TtC6HackMe14ViewController send:] + 0, Section \_\_text, file offset 0x5040 - Alt+Double Click to follow link in a new pane

## HackMe.hop



Address 0x100005040, Segment \_TEXT, -[\_TtC6HackMe14ViewController send:] + 0, Section \_text, file offset 0x5040 - Alt+Mouse, or trackpad gesture to zoom - Double-tap to focus on a block

```
r1 = r23;
r0 = Foundation.URLRequesturlcachePolicytimeoutInterval.init();
r0 = r20 + *objc_ivar_offset_TtC6HackMe14ViewController_webView;
r0 = swift_unknownWeakLoadStrong();
r21 = r0;
if (r21 != 0x0) {
    r0 = [r21 retain];
    r20 = r22;
    r20 = Foundation.URLRequest._bridgeToObjectiveC();
    r0 = [r21 loadRequest:r20];
    r29 = r29;
    r23 = [r0 retain];
    r0 = [r21 release];
    r0 = [r21 release];
    r0 = [r20 release];
    r0 = [r23 release];
    r0 = *qword_100009590;
    if (r0 == 0x0) {
        r1 = *type metadata for Any + 0x8;
        r0 = type metadata accessor for Swift._ContiguousArrayStorage();
        if (r1 == 0x0) {
            r8 = 0x100009590;
            asm { stlr      x0, [x8] };
    }
}
```

If it's so secret,  
Stop writing it  
down

```
let key = "0rlsRU5vjPvHUad70ysqX67k4DN50/sNYRD5zqnCLbg="

echo 0rlsRU5vjPvHUad70ysqX67k4DN50/sNYRD5zqnCLbg= | base64 -D | xxd -i

let key = Data([
    0xd2, 0xb9, 0x6c, 0x45, 0x4e, 0x6f, 0x8c, 0xfb, 0xc7, 0x51, 0xa7, 0x7b,
    0xd3, 0x2b, 0x2a, 0x5f, 0xae, 0xe4, 0xe0, 0x33, 0x79, 0xd3, 0xfb, 0xd,
    0x61, 0x10, 0xf9, 0xce, 0xa9, 0xc2, 0x2d, 0xb8
])
```

```
$ otool -d HackMe
```

HackMe:

Contents of (\_\_DATA, \_data) section

0000000100009538	00000000	00000000	00000000	00000000
0000000100009548	00000000	00000000	00000000	00000000
0000000100009558	00000000	00000000	00000000	00000000
0000000100009568	00000020	00000000	00000040	00000000
0000000100009578	456cb9d2	fb8c6f4e	7ba751c7	5f2a2bd3
0000000100009588	33e0e4ae	0dfbd379	cef91061	b82dc2a9
0000000100009598	00000000	00000000	00000000	00000000
00000001000095a8	00000000	00000000	00000001	00000000
00000001000095b8	00006540	00000001	00000000	00000000
00000001000095c8	00000000	00000000	00000000	00000000
00000001000095d8	00000000	00000000	00000000	00000000
00000001000095e8	00000000	00000000	00000000	00000000
00000001000095f8	00000000	00000000	00008408	00000001
0000000100009608	00000008	00000000	00000000	00000000
0000000100009618	00000000	00000000	00000000	00000000
0000000100009628	00000000	00000000	00000000	00000000
0000000100009638	00000000	00000000	00000000	00000000
0000000100009648	00000000	00000000	00000000	00000000

```
let englishKey = "Ha!U'llNever(Find)Me!I'mSoSekret"
```

```
head -c32 /dev/random | xxsd -i
```

```
let aesKey = Data([
    0x4e, 0x5a, 0x88, 0xec, 0x22, 0x2b, 0xcc, 0x7a, 0x78, 0xfe, 0x96, 0x35,
    0xc2, 0x94, 0x44, 0xac, 0x4c, 0xad, 0x96, 0xab, 0x34, 0x20, 0xa7, 0x92,
    0xe0, 0x1c, 0xc3, 0xf7, 0xfd, 0xea, 0xba, 0xa4
])
```

Insert

Dr. Evil GIF

Here

you know the one...

Letters + Numbers + Symbols

$\times 96$

~~$96^{32}$~~   
 $256^{32} \approx 100 \text{ trillion}$

TYYLWKMUAPZHLYXJMPUW

# swiftmask

```
head -c 20 /dev/random | xxd -i

let mask = Data([
    0x77, 0xa6, 0x7f, 0x09, 0x1f, 0x74, 0x2d, 0xf3, 0xb6, 0x88, 0xb6, 0xff,
    0x05, 0xc3, 0x9f, 0x3c, 0x02, 0x3a, 0xb1, 0xac
])

print(zip(api, mask).map(^))

let masked = Data(
    [35, 255, 38, 69, 72, 63, 96, 166, 247, 216,
     236, 183, 73, 154, 199, 118, 79, 106, 228, 251]
)

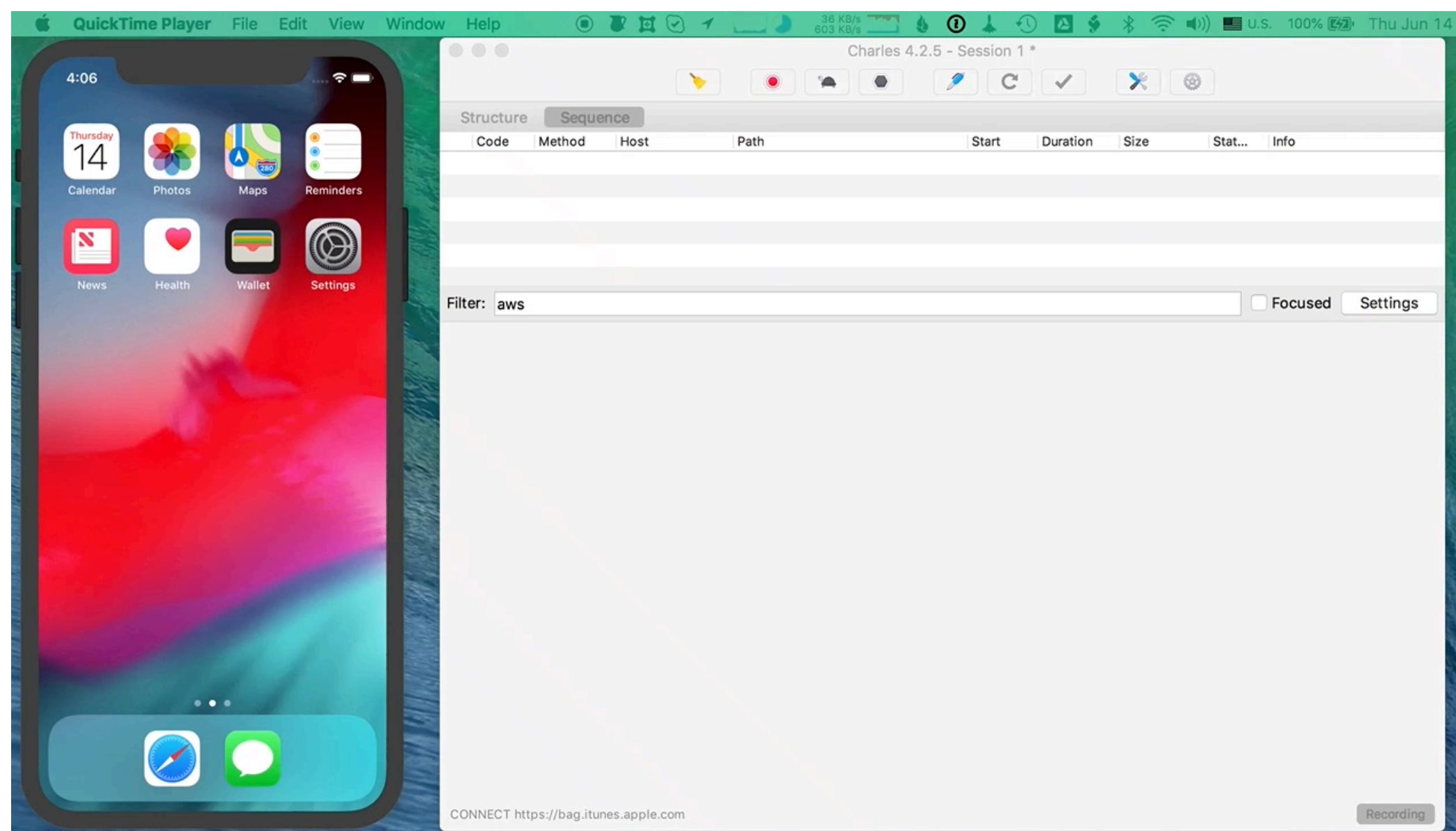
let demasked = String(bytes: zip(masked, mask).map(^),
                      encoding: .utf8)!
```



EN SPACE = 0  
EM SPACE = 1

<https://github.com/rnapier/SwiftStego>





# Certificate Pinning

<https://talk.objc.io/episodes/S01E57-certificate-pinning>

# A Lot of Trust

## You Expect...

- Verisign
- Network Solutions
- Thawte
- DigiCert
- Digital Signature Trust

## But Also...

- Amazon, Cisco, Apple, ...
- US, Belgium, Germany, Netherlands, Switzerland, Turkey, ...
- Actalis, Atos, Buypass, Certigna, Certinomis, Chambers of Commerce, E-Tugra, Echoworx, Izenpe, QuoVadis, Starfield, ...

<http://support.apple.com/kb/ht5012>

```
extension CertificateValidator: URLSessionDelegate {
    func urlSession(_ session: URLSession, didReceive challenge: URLAuthenticationChallenge,
                    completionHandler: @escaping (URLSession.AuthChallengeDisposition,
                                                 URLCredential?) → Void) {
        let protectionSpace = challenge.protectionSpace

        if (protectionSpace.authenticationMethod == NSURLAuthenticationMethodServerTrust) {

            if let trust = protectionSpace.serverTrust {
                SecTrustSetAnchorCertificates(trust, trustedCertificates as CFArray)
                SecTrustSetAnchorCertificatesOnly(trust, true)

                var result = SecTrustResultType.invalid
                let status = SecTrustEvaluate(trust, &result)
                if status == errSecSuccess {
                    switch result {
                        case .proceed, .unspecified:
                            completionHandler(.useCredential, URLCredential(trust: trust))
                            return

                        default:
                            print("Could not verify certificate: \(result)")
                    }
                }
            }
        }

        // Something failed. Cancel
        completionHandler(.cancelAuthenticationChallenge, nil)
    }
}
```

```
guard let url = Bundle.main.url(forResource: name, withExtension: "cer") else {  
    preconditionFailure("\(name) not found")  
}  
  
try! validator = CertificateValidator(certificateURL: url)  
session = URLSession(configuration: .default, delegate: validator, delegateQueue: nil)
```

**<https://github.com/rnapier/CertificateValidator>**

Secret  
Knock

1. generate Secret
2. ??? (also send 404)
3. Secure!

True

Story

Spend an hour,  
(ok, maybe a few)

Save the day

```
xcrun simctl launch booted net.robnapier.HackMe --wait-for-debugger
```

\$ Spending ?  
? Some \$  
\$ money ?

```
func a(b: URL, c: @escaping (Data?) → Void) {
    URLSession.shared.dataTask(with: b) { (d, e, f) in
        if let g = f {
            print(g)
            c(nil)
        } else if let data = d {
            c(data)
        } else {
            c(nil)
        }
    }.resume()
}
```

```
func o1(_ o7: o2, o3: @escaping (o4) → o5) {  
    o20(o6(o7)) {  
        if let o8 = $2 {  
            o9(o8)  
            o3(nil)  
        } else if let o10 = $0 {  
            o3(o10)  
        } else {  
            o3(nil)  
        }  
    }()  
}
```

```
func o1(_ o7: URL, o3: @escaping (Data?) → Void) {  
    o20(o6(o7: URL) { (Data?, URLResponse, Error?) in  
        if let o8: Error = $2: Error? {  
            o9(o8: Error)  
            o3(nil: Data)  
        } else if let o10: Data = $0: Data? {  
            o3(o10: Data)  
        } else {  
            o3(nil: Data)  
        }  
    }: URLSessionDataTask)()  
}
```

```
func o1(_ o7: URL, o3: @escaping (Data?) → Void) {  
    o20(urlSession.dataTask(with: o7) {  
        if let o8: Error = $2: Error? {  
            o9(o8: Error)  
            o3(nil: Data)  
        } else if let o10: Data = $0: Data? {  
            o3(o10: Data)  
        } else {  
            o3(nil: Data)  
        }  
    }: URLSessionDataTask)()  
}
```

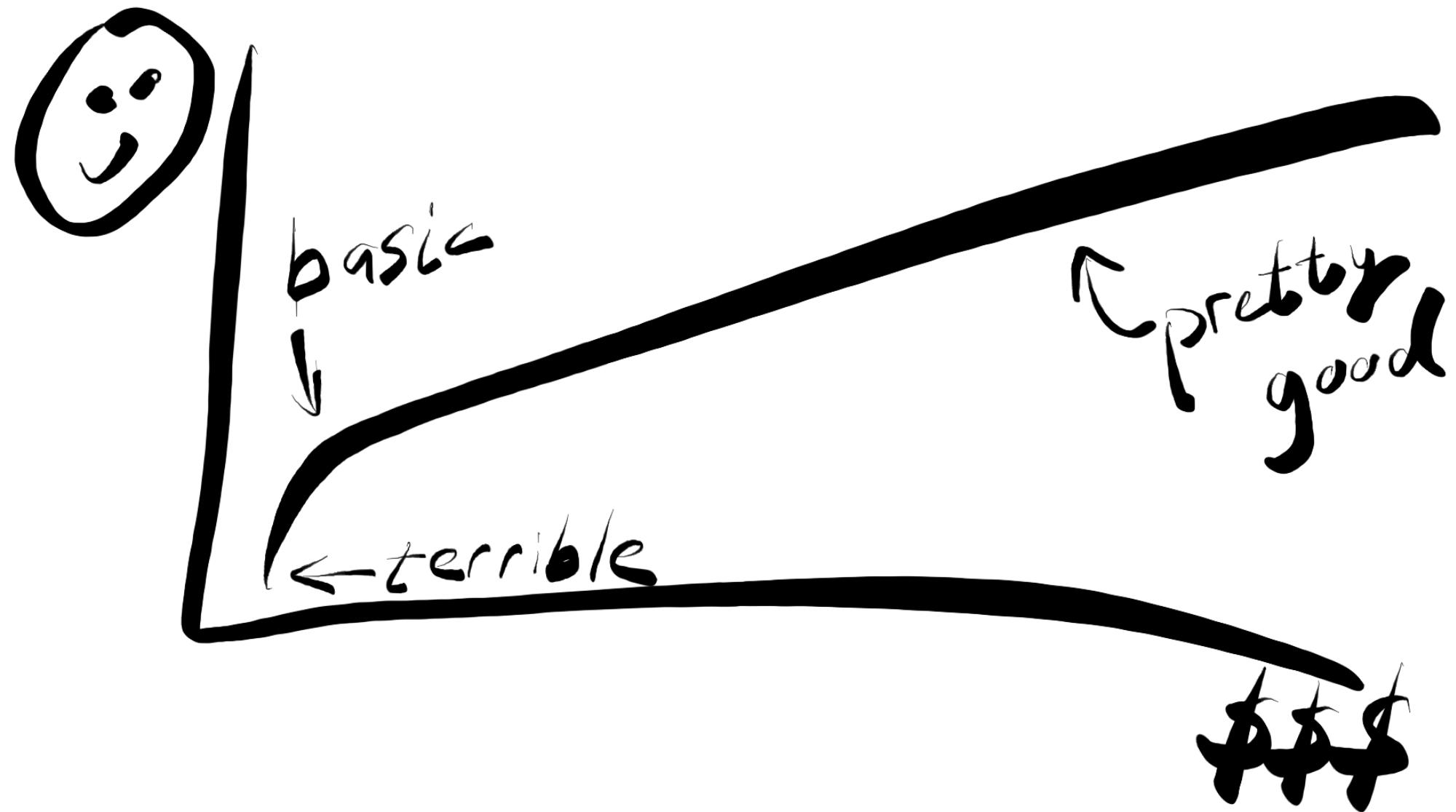
```
func o1(_ o7: URL, o3: @escaping (Data?) → Void) {  
    urlSession.dataTask(with: o7) {  
        if let o8: Error = $2: Error? {  
            o9(o8: Error)  
            o3(nil: Data)  
        } else if let o10: Data = $0: Data? {  
            o3(o10: Data)  
        } else {  
            o3(nil: Data)  
        }  
    }.resume()  
}
```

```
func o1(_ url: URL,  
       completion: @escaping (Data?) → Void) {  
    urlSession.dataTask(with: url) {  
        if let error = $2 {  
            o9(error)  
            completion(nil)  
        } else if let data = $0 {  
            completion(data)  
        } else {  
            completion(nil)  
        }  
    }.resume()  
}
```

Hardening

No Pain, No Gain

But so much pain



1. Obfuscation is for you,  
not users!
2. Focus on the basics  
strings, network
3. Hopper & Charles

Strings bad  
Data less bad

ObjC very bad  
(but I still ❤ ObjC)

Secrets  
&  
Lies

<https://github.com/rnapier/secrets>