Maintaining cryptographic library for 11 languages

@vixentael

Maintaining cryptographic library for ~~11~~ 12 languages

@vixentael

@vixentael

head of customer solutions,
security software engineer

OSS maintainer: Themis, Acra

focused on applied crypto
and building e2ee protocols

COSSACK
LABS

Data security solutions

We provide hard-to-misuse cryptographic tools to help companies to protect the data that is sensitive for their business.

cossacklabs.com/products

@vixentael

database searchable encryption

cossacklabs.com/acra/

e2ee data collaboration

cossacklabs.com/hermes/

zero knowledge authentication

github.com/cossacklabs/themis/wiki/Secure-Comparator-cryptosystem

cossacklabs.com/whitepapers/

@vixentael

IN CRYPTO

WE TRUST

USABLE

Plan for today

1. Boring Crypto & inspiration
2. The Scale: core & language wrappers
3. Easy to use VS hard to misuse
4. Security & testing
5. Community & cases

…imagine simple use case

“Let’s protect stored data”

Things to decide on:

PLATFORMS | BACKUPS

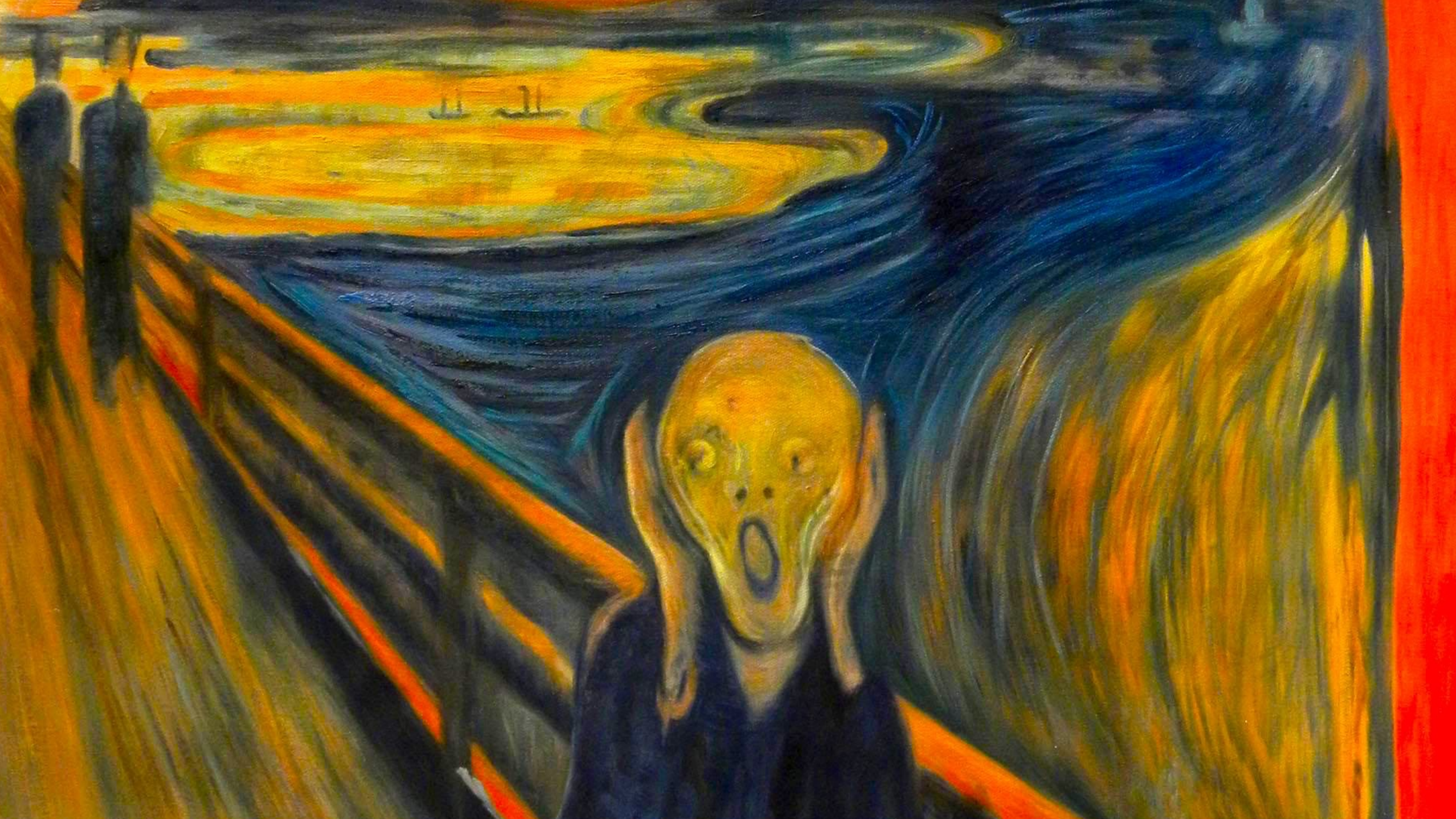
KEY ROTATION | KEY EXCHANGE | KEY REVOCATION

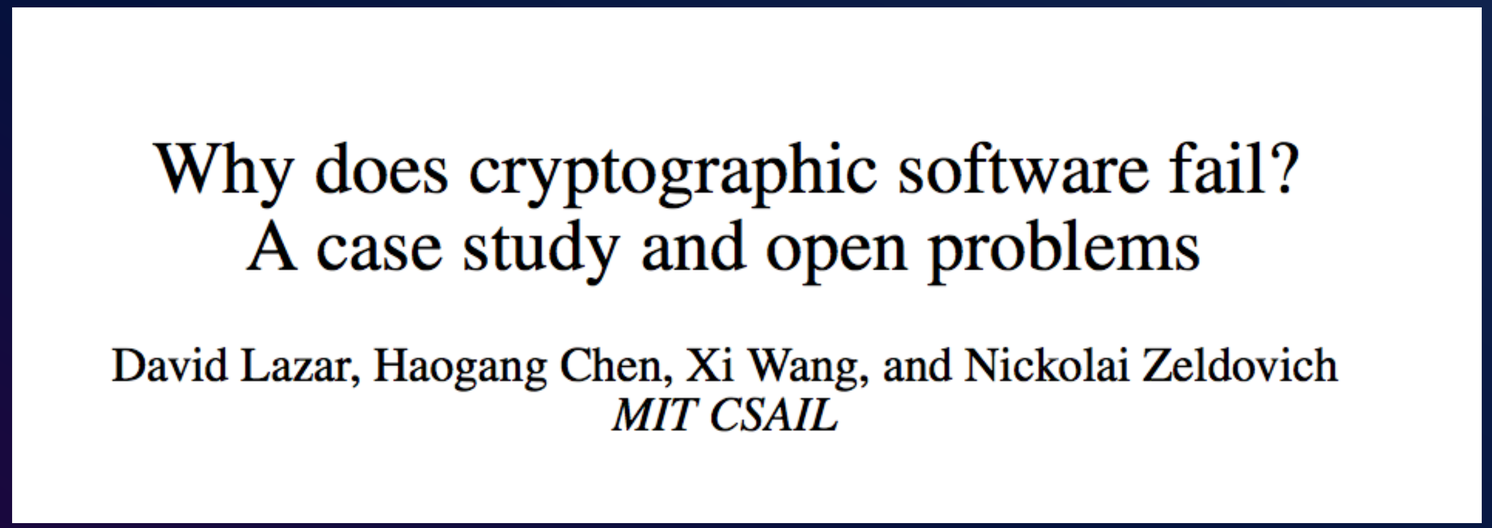
KEY DERIVATION | KEY STORAGE | KMS

MODE | PADDING | IV

DATA SCOPE | CIPHER | KEY LENGTH

@vixentael




Why does cryptographic software fail?
A case study and open problems

David Lazar, Haogang Chen, Xi Wang, and Nickolai Zeldovich
MIT CSAIL

269 CVEs
from 2011-2014

17% bugs inside crypto libs

83% misuses of crypto libs
by individual apps

https://pdos.csail.mit.edu/papers/cryptobugs:apsys14.pdf

@vixentael

Blowfish

AES-SIV

Rabbit

Salsa20

AES

Kuznyechik

SEED

CFB

RSA

DSS

ECDSA

DES

Camelia

SEAL

CBC

DSA

3DES

SHARK

RC4

CTR

ChaCha20

@vixentael

Boring crypto

— crypto that simply works, solidly resists attacks, never needs any upgrades

Daniel J. Bernstein

https://cr.yp.to/talks/2015.10.05/slides-djb-20151005-a4.pdf

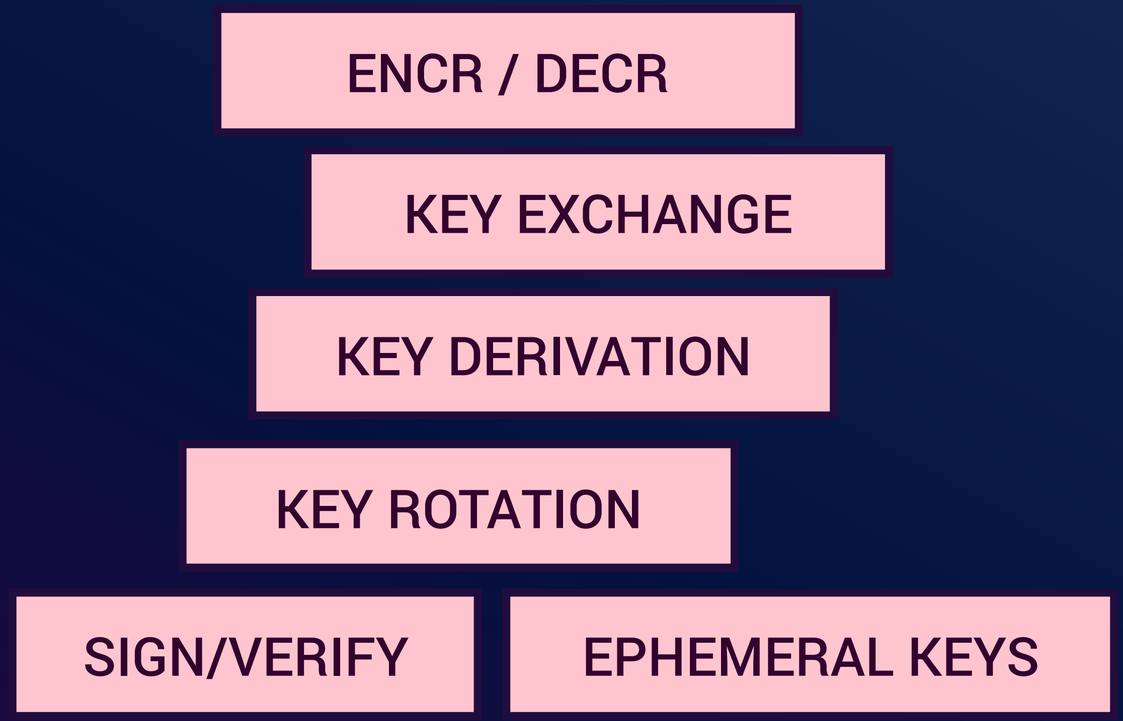
@vixentael

Solve use-cases

I want to **store** data securely
I want to **send** data securely
I want to **verify** data integrity

Solve use-cases

store data securely
send data securely
verify data integrity



@vixentael

Themis

Themis provides strong, usable cryptography for busy people

---

release v0.12.1 | circleci passing | bitrise ✓ | platform Android | iOS | macOS | Linux | Java | WASM | coverage 90% | go report A+

General purpose cryptographic library for storage and messaging for iOS (Swift, Obj-C), Android, desktop Java, C++, Node.js, Python, Ruby, PHP, Go, Rust, WASM.

Perfect fit for multi-platform apps. Hides cryptographic details. Made by cryptographers for developers 🧡

github.com/cossacklabs/themis

@vixentael

Themis: cryptosystems

store encrypted

encrypt for someone

encrypt session communication

authenticate

github.com/cossacklabs/themis

@vixentael

Themis: cryptosystems

store encrypted

encrypt for someone

encrypt session communication

authenticate

SecureCell

AES GCM /
AES CTR

built in KDF

github.com/cossacklabs/themis

@vixentael

Themis: cryptosystems

store encrypted

encrypt for someone

encrypt session communication

authenticate

SecureCell

SecureMessage

AES GCM / AES CTR

ECC + ECDSA / RSA + PSS + PKCS#7

built in KDF

built in key gen

github.com/cossacklabs/themis

@vixentael

Themis: cryptosystems

store encrypted	encrypt for someone	encrypt session communication	authenticate
SecureCell	SecureMessage	SecureSession	
AES GCM / AES CTR	ECC + ECDSA / RSA + PSS + PKCS#7	ECDH + ECC + AES	
built in KDF	built in key gen	ephemeral keys	

github.com/cossacklabs/themis

@vixentael

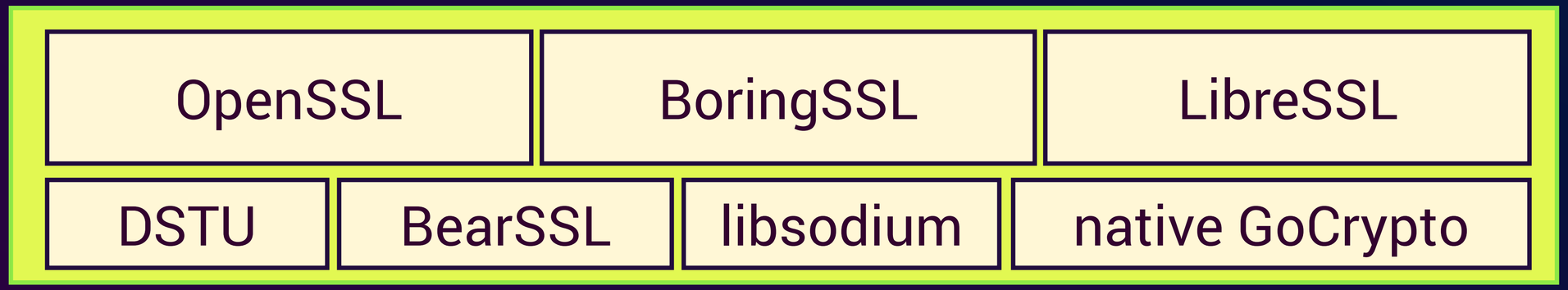
Themis: cryptosystems

store encrypted	encrypt for someone	encrypt session communication	authenticate
SecureCell	SecureMessage	SecureSession	SecureComparator
AES GCM / AES CTR	ECC + ECDSA / RSA + PSS + PKCS#7	ECDH + ECC + AES	OTR SMP + ECC
built in KDF	built in key gen	ephemeral keys	ZKP

github.com/cossacklabs/themis

@vixentael

Themis



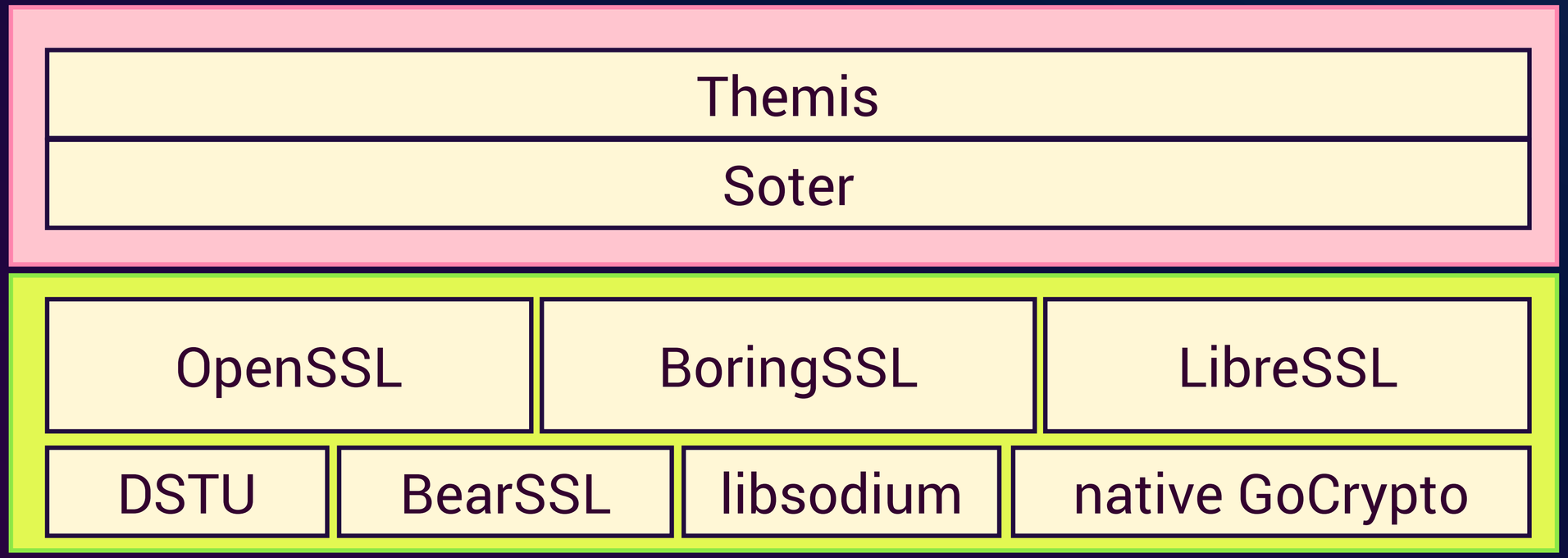
crypto-backends

stable

experimental

@vixentael

Themis



Themis Core

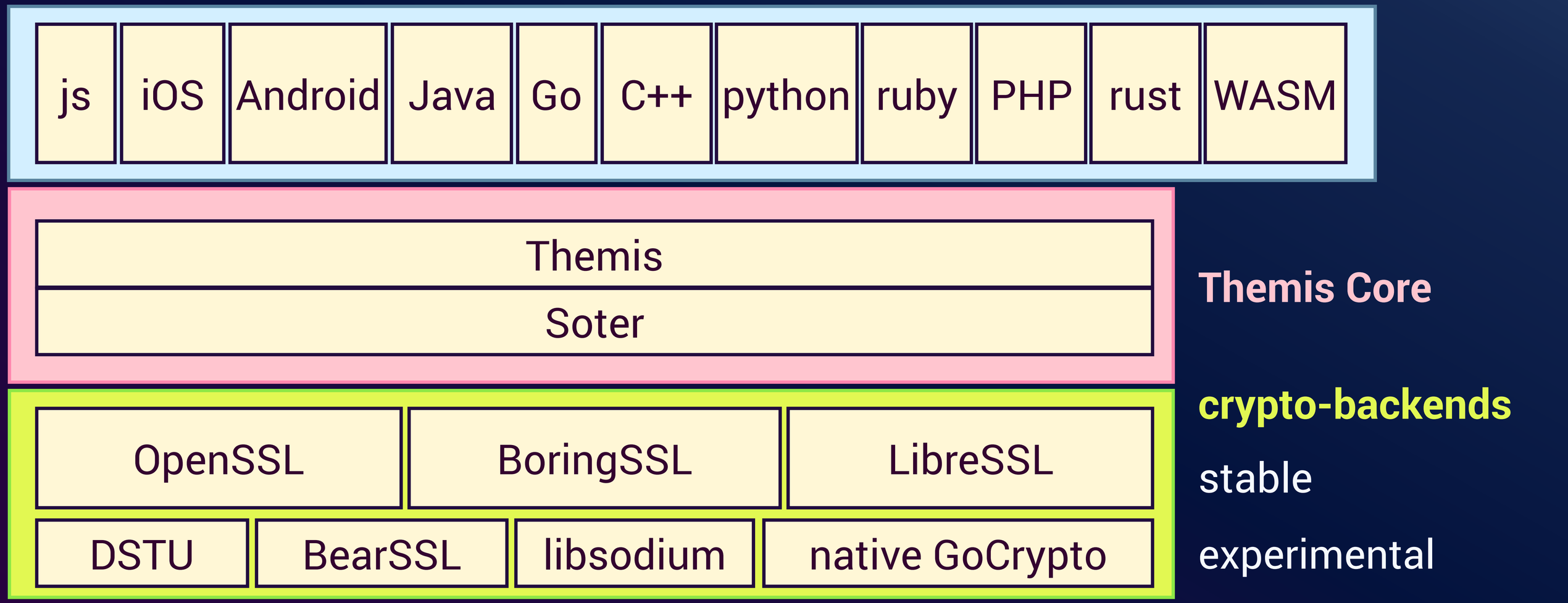
crypto-backends

stable

experimental

@vixentael

Themis



@vixentael

Themis Core (server & desktop OS)

- Ubuntu
- Debian
- CentOS / RHEL
- macOS
- Windows

@vixentael

Themis Core (server & desktop OS)

Ubuntu 18.04 x64	Ubuntu 16.04 x64	Ubuntu 16.04 x32	
Debian 9 x32	Debian 9 x64	Debian 8 x64	Debian 8 x32
CentOS 7 x64			
macOS 10.13	macOS 10.14	macOS 10.15	
Windows			

@vixentael

Themis OSs

Ubuntu 18.04 x64	Ubuntu 16.04 x64	Ubuntu 16.04 x32	Debian 9 x32
Debian 9 x64	Debian 8 x64	Debian 8 x32	CentOS 7 x64
macOS 10.13	macOS 10.14	macOS 10.15	Windows
iOS 9	iOS 10	iOS 11	iOS 12
iOS 13	Android SDK 16 - 29		

@vixentael

Cryptographic tools should be

easy to use or hard to misuse?

@vixentael

Easy to use

```
make

make test

sudo make install
```

@vixentael

Easy to use

noone cares about your lib if they can’t install it using their fav package manager

@vixentael

Themis Core: install

Ubuntu	`apt-get install libthemis-dev`
Debian	`apt-get install libthemis-dev`
CentOS / RHEL	`yum install libthemis-devel`
macOS	`brew install libthemis`
Windows	`make nsis_installer`

@vixentael

Package managers

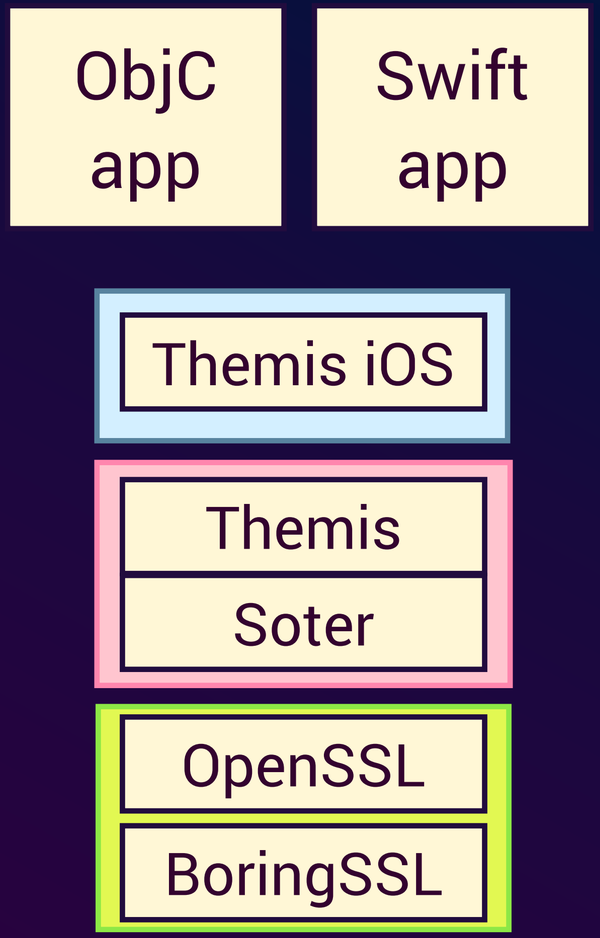
```
npm install jsthemis

pip install pythemis

pip3 install pythemis

npm install wasm-themis

gem install rbthemis
```

```
[dependencies]
themis = "0.12"

pod themis
pod install

github “cossacklabs/themis"
carthage update
```

```
maven { url "https://dl.bintray.com/cossacklabs/maven/" }

go get github.com/cossacklabs/themis/gothemis/...
```

@vixentael

iOS specifics



ObjC <> Swift interoperability

ObjC

C lang

@vixentael

iOS specifics

ObjC <> Swift interoperability

ObjC

C lang

package managers

- CocoaPods
- Carthage
- ~~SPM~~
- manually

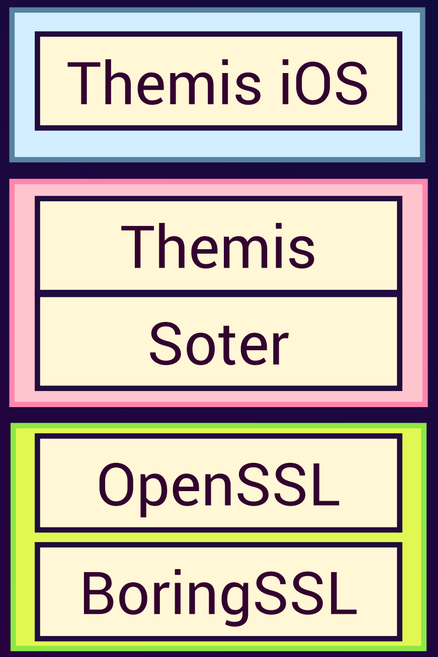
@vixentael

iOS specifics

ObjC app

Swift app

ObjC <> Swift interoperability



ObjC

C lang

Xcode/Swift update

iOS update

device update

package manager update

Open/BoringSSL update

package managers

- CocoaPods
- Carthage
- ~~SPM~~
- manually

@vixentael

iOS specifics

Header not found when adding Themis as dependency in a new development cocoapod O-iOS

#411 by popaaaandrei was closed

Themis Carthage: fix linking issue with OpenSSL 1.0.2.18 ✕ O-iOS

#483 by vixentael was merged on Jun 25

Themis 0.10.2 doesn't compile on iOS device due to bitcode O-iOS bug

#406 by popaaaandrei was closed on Mar 4

Themis iOS and B… O-iOS question

#336 by CodeTeamLab

Update iOS wrapper header structure to compile Swift framework ● O-iOS

#415 by vixentael was merged on Mar 6 • Approved

Can't archive Xcode project with themis installed O-iOS

#538 by chrisleversuch was closed on Sep 26 2 of 4

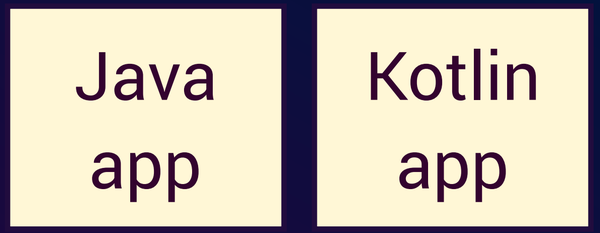
Cannot install Themis via Cocoapods O-iOS installation

#481 by drewpitchford was closed on Jun 18 0 of 2

@vixentael


THIS IS FINE.

Android specifics



Java <> Kotlin
interoperability

Java <> C

C lang

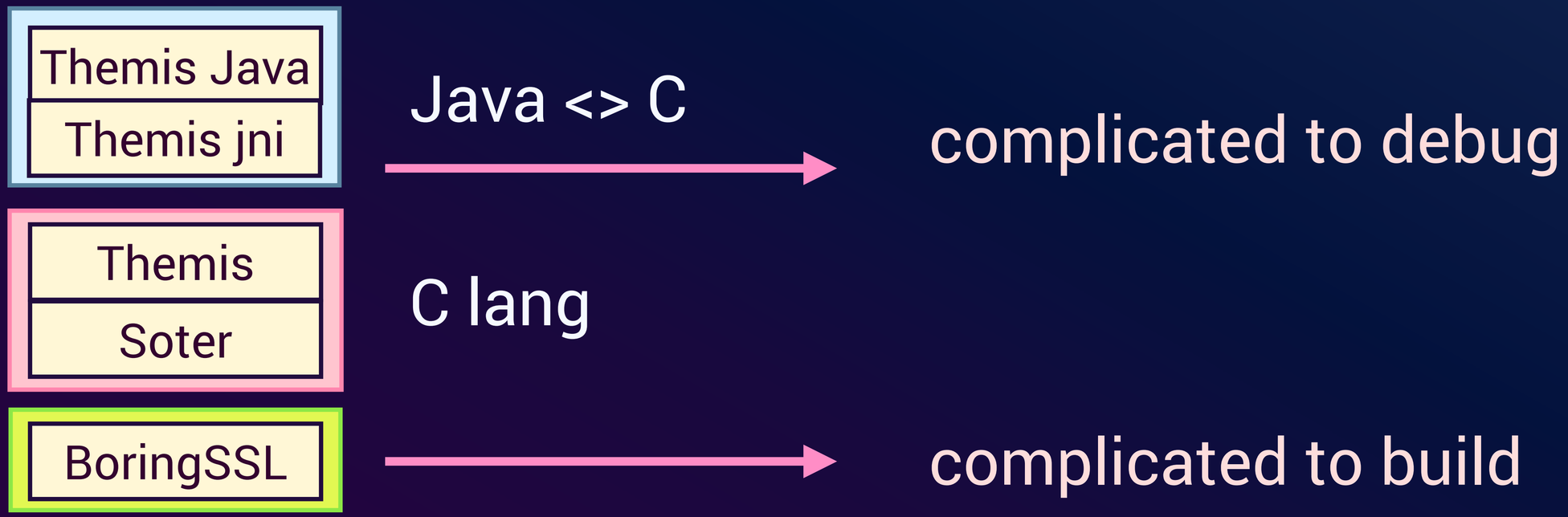
@vixentael

Android specifics

Java app

Kotlin app

Java <> Kotlin interoperability



@vixentael

BoringSSL

```
CMake Error at /home/user/android-sdk/ndk-bundle/build/cmake/
android.toolchain.cmake:169 (message): GCC is no longer supported.

See https://android.googlesource.com/platform/ndk/+/master/docs/
ClangMigration.md.
```

github.com/cossacklabs/themis/pull/352

@vixentael

BoringSSL

```
CMake Error at /home/user/android-sdk/ndk-bundle/build/cmake/
android.toolchain.cmake:169 (message): GCC is no longer supported.

See https://android.googlesource.com/platform/ndk/+/master/docs/
ClangMigration.md.
```

```
cmake {
    arguments "-DCMAKE_TOOLCHAIN_FILE=" + android.ndkDirectory +
              "-DANDROID_NATIVE_API_LEVEL=21",
              "-DANDROID_TOOLCHAIN=gcc",
              "-DCMAKE_BUILD_TYPE=Release",
              "-GNinja"
}
```

gcc -> clang

github.com/cossacklabs/themis/pull/352

@vixentael

BoringSSL

why so slow 🤔

```
$ time ./gradlew --no-daemon --no-parallel --max-workers=2 assembleDebug

real    34m46.028s
user    29m9.580s
sys     6m39.280s
```

github.com/cossacklabs/themis/pull/447

@vixentael

BoringSSL

```
@$(MAKE) -C $(BIN_PATH)/boringssl
@$(MAKE) -C $(BIN_PATH)/boringssl crypto decrepit
```

don’t build examples

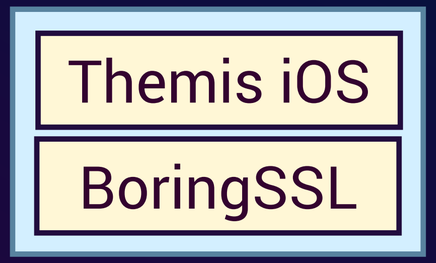
```
$ time ./gradlew --no-daemon --no-parallel --max-workers=2 assembleDebug

real    34m46.028s
user    29m9.580s
sys     6m39.280s
```

```
real    12m46.028s
user    9m1.880s
sys     2m52.020s
```

github.com/cossacklabs/themis/pull/447

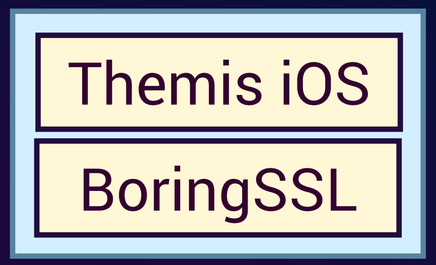
@vixentael

Multi-platform is hard



BoringSSL is used in iOS libs
by Google (Firebase)

Multi-platform is hard

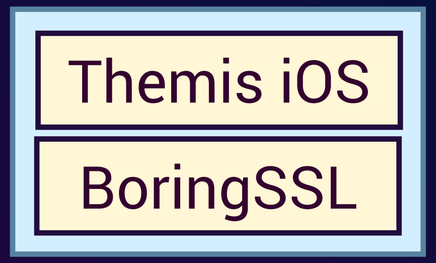


BoringSSL is used in iOS libs
by Google (Firebase)

no AES XTS in BoringSSL iOS CocoaPod

@vixentael

Multi-platform is hard



BoringSSL is used in iOS libs
by Google (Firebase)

↓

no AES XTS in BoringSSL iOS CocoaPod

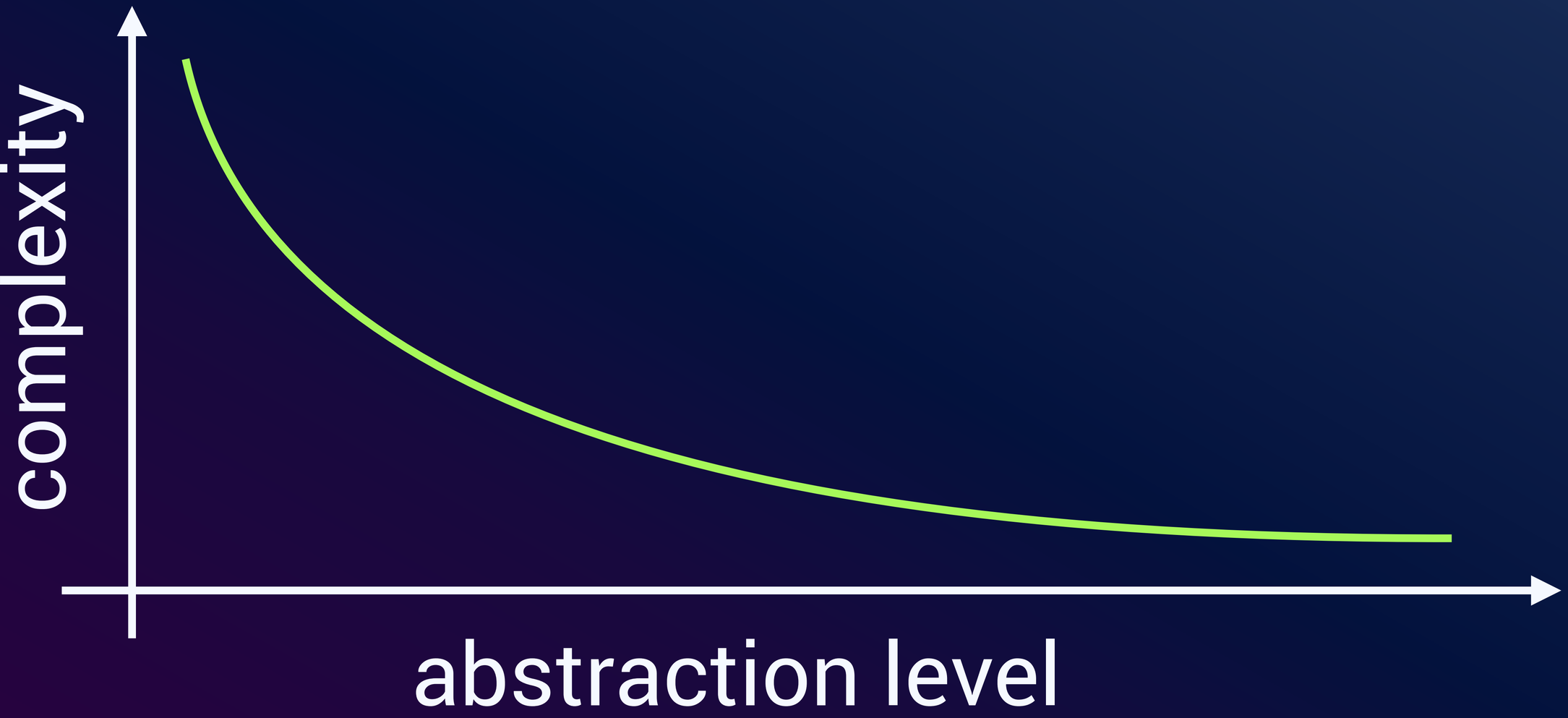
↓

```
#define SOTER_BORINGSSL_DISABLE_XTS
```

@vixentael

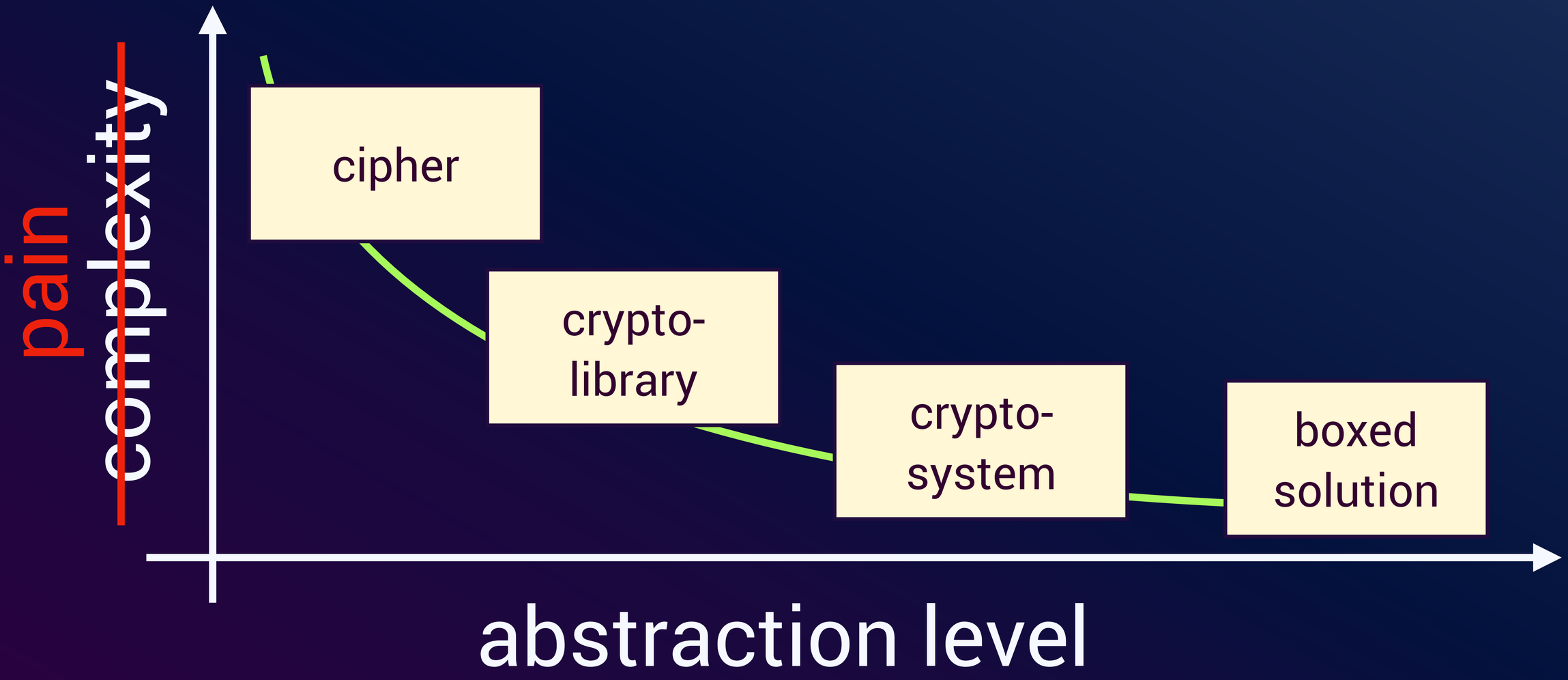
Hard to misuse

encryption integration



@vixentael

encryption integration



@vixentael

CommonCrypto AES

```
status = CCCrypt(operation,
                 CCAlgorithm(kCCAlgorithmAES128),
                 CCOptions(kCCOptionPKCS7Padding),
                 keyBytes,
                 key.count,
                 ivBytes,
                 encryptedBytes,
                 input.count,
                 &outBytes,
                 outBytes.count,
                 &outLength)
```

@vixentael

Easy to make mistakes

```
do {
    let aes = try AES(key: "keykeykeykeykeyk", iv: "drowssapdrowssap") // aes128
    let ciphertext = try aes.encrypt(Array("Nullam quis risus eget urna mollis ornare vel eu leo.".utf8))
} catch { }
```

@vixentael

Easy to make mistakes

should use KDF(key)

should be random

```
do {
    let aes = try AES(key: "keykeykeykeykeyk", iv: "drowssapdrowssap") // aes128
    let ciphertext = try aes.encrypt(Array("Nullam quis risus eget urna mollis ornare vel eu leo.".utf8))
} catch { }
```

uses AES CBC, not AES GCM

padding? salt?

@vixentael

Themis: hard to make mistakes

```
do {
    let cellSeal = TSCellSeal(key: UUID().uuidString.data(using: .utf8)!)!
    let encryptedMessage = try cellSeal.wrap("message".data(using: .utf8)!,
                                             context: nil)
} catch {}
```

github.com/cossacklabs/themis

@vixentael

Themis: hard to make mistakes

built-in KDF to make keys stronger

```
do {
    let cellSeal = TSCellSeal(key: UUID().uuidString.data(using: .utf8)!)!
    let encryptedMessage = try cellSeal.wrap("message".data(using: .utf8)!,
                                             context: nil)
} catch {}
```

hides cryptographic details: salt, IV, KDF, padding

uses AES-256-GCM

github.com/cossacklabs/themis

@vixentael

One API to rule them all!

@vixentael

```
import themis

do {
    let cellSeal = TSCellSeal(key: UUID().uuidString.data(using: .utf8)!)
    let encryptedMessage = try cellSeal.wrap("message".data(using: .utf8)!,
                                             context: nil)
} catch {}
```

@vixentael

```rust
use themis::secure_cell::SecureCell;

let cell = SecureCell::with_key(key)?.seal();
let encrypted = cell.encrypt(b"source data")?;
```

@vixentael

```
from pythemis.scell import SCellContextImprint

scell = SCellContextImprint(key)
encrypted_message = scell.encrypt(b'message')
```

```rust
use themis::secure_cell::SecureCell;

let cell = SecureCell::with_key(key)?.seal();
let encrypted = cell.encrypt(b"source data")?;
```

```ruby
require 'rbthemis'

scell_seal = Themis::Scell.new(key, Themis::Scell::SEAL_MODE)
encrypted_message = scell_seal.encrypt(message, context)
```

```javascript
var themis = require('jsthemis')

var scell = new themis.SecureCellSeal(key)
var encrypted_message = scell.encrypt('message', context)
```

```java
import com.cossacklabs.themis.*;

SecureCell cell = new SecureCell(key, SecureCell.MODE_SEAL);
SecureCellData cellData = cell.protect(context, data);
```

```cpp
#include <themispp/secure_cell.hpp>

themispp::secure_cell_seal_t sm(key);
try {
    std::vector<uint8_t> encrypted_message = sm.encrypt(mes
} catch (const themispp::exception_t& e) {
    e.what();
}
```

```go
import "github.com/cossacklabs/themis/gothemis/cell"

secureCell := cell.New(secretKey, cell.ModeSeal)
protectedData, _, err := secureCell.Protect(data, context)
```

@vixentael

Themis API incapsulates

✅ 1. Cipher suit, mode, padding, IV, key, memory.

✅ 2. KDF, key management.

✅ 3. Authentication, ephemeral keys.

@vixentael

Testing

@vixentael

Testing

```
cloc src

328 text files.
SUM: 35737
```

@vixentael

Testing

```
cloc src

328 text files.
SUM: 35737
```

```
cloc tests

3508 text files.
SUM: 341312
```

@vixentael

Testing

unit tests

per each language, crypto: NIST-specified for PRNG & AES

medium.com/@cossacklabs/automated-security-testing-56ee1253c1fd

@vixentael

Testing

fuzzing

unit tests

AFL

per each language, crypto: NIST-specified for PRNG & AES

medium.com/@cossacklabs/automated-security-testing-56ee1253c1fd

@vixentael

Testing

memory, sanitizers, SATS

fuzzing

unit tests

clang, Valgrind, Splint, Cppcheck

AFL

per each language, crypto: NIST-specified for PRNG & AES

medium.com/@cossacklabs/automated-security-testing-56ee1253c1fd

@vixentael

Testing

integration tests	per OS, per language
memory, sanitizers, SATS	clang, Valgrind, Splint, Cppcheck
fuzzing	AFL
unit tests	per each language, crypto: NIST-specified for PRNG & AES

medium.com/@cossacklabs/automated-security-testing-56ee1253c1fd

@vixentael

Testing

backwards compatibility tests	between versions
integration tests	per OS, per language
memory, sanitizers, SATS	clang, Valgrind, Splint, Cppcheck
fuzzing	AFL
unit tests	per each language, crypto: NIST-specified for PRNG & AES

medium.com/@cossacklabs/automated-security-testing-56ee1253c1fd

@vixentael

Testing

OWASP Source Code Analysis Tools

owasp.org/index.php/Source_Code_Analysis_Tools

Clouseau – git repo inspector for keys & creds.

github.com/cfpb/clouseau

Fuzzing resources

github.com/secfigo/Awesome-Fuzzing

@vixentael

Testing

integrated with Github
unit tests, memory

iOS, Android, macOS
examples and tests

everything o.o

@vixentael

Security

zeroing, minimization, memory,
constant time checks, etc

cossacklabs.com/blog/macros-in-crypto-c-code.html

github.com/veorq/cryptocoding

@vixentael

One readme is not enough

@vixentael

1. Language-specific docs

Building Themis wrappers (11)

- Android wrapper installation
- Java wrapper installation
- iOS/macOS wrapper installation
- C++ wrapper installation
- WebAssembly wrapper installation
- Node.js wrapper installation
- PHP wrapper installation
- Go wrapper installation
- Python wrapper installation
- Ruby wrapper installation
- Rust wrapper installation

Tutorials & How Tos

About the tutorials

- Java and Android HowTo
- Swift HowTo
- Objective-C HowTo
- Python HowTo
- PHP HowTo
- Ruby HowTo
- CPP HowTo
- Go HowTo
- NodeJS HowTo
- Rust HowTo
- WebAssembly HowTo

@vixentael

1. Language-specific docs

Building Themis wrappers (11)

- Android wrapper installation
- Java wrapper installation
- iOS/macOS wrapper installation
- C++ wrapper installation
- WebAssembly wrapper installation
- Node.js wrapper installation
- PHP wrapper installation
- Go wrapper installation
- Python wrapper installation
- Ruby wrapper installation
- Rust wrapper installation

Tutorials & How Tos

About the tutorials

- Java and Android HowTo
- Swift HowTo
- Objective-C HowTo
- Python HowTo
- PHP HowTo
- Ruby HowTo
- CPP HowTo
- Go HowTo
- NodeJS HowTo
- Rust HowTo
- WebAssembly HowTo

“give me code!”

“too much to read”

@vixentael

2. “Safe to copypaste” code snippets

```
session = SSession(b'user_id2', client_private_key, CustomSimpleTransport())
# this call is only made by the client
encrypted_message = session.connect_request()

# send connect request to the peer
response_bytes = user_communication_send_method(encrypted_message)
message = session.unwrap(response_bytes)

# establish the session
while not session.is_established():
    response_bytes = user_communication_send_method(message)
    message = session.unwrap(response_bytes)
```

@vixentael

2. “Safe to copypaste” code snippets

```
session = SSession(b'user_id2', client_private_key, CustomSimpleTransport())
# this call is only made by the client
encrypted_message = session.connect_request()

# send connect request to the peer
response_bytes = user_communication_send_method(encrypted_message)
message = session.unwrap(response_bytes)

# establish the session
while not session.is_established():
    response_bytes = user_communication_send_method(message)
    message = session.unwrap(response_bytes)
```

“how to use it in the app?”

@vixentael

3. Example applications

```
cloc docs/examples

15427 text files
SUM: 1015922
```

@vixentael

3. Example applications

```
cloc docs/examples

15427 text files
SUM: 1015922
```

```
cloc src

328 text files.
SUM: 35737
```

```
cloc tests

3508 text files.
SUM: 341312
```

@vixentael

3. Example applications

```
cloc docs/examples

15427 text files
SUM: 1015922
```

“but I am building unique app!”

4. Tutorials and use case specific apps

During the development stage, we frequently do Proof-of-Concept projects to test different assumptions. They serve as interesting demos (examples) of what Themis is capable of:

Demo	Description	Repo	Blog post
0fc	Anonymous web chat * Python * webthemis (C++ + HTML/JS)	repo	blog post
Sesto	Secure storage * Python * webthemis (C++ + HTML/JS)	repo	blog post
Swift Alps demo	Secure communication (iOS app with Python server based on Secure Session) * Swift * Python	repo	slides
Zero-Knowledge Architectures workshop	iOS app for storing and sharing encrypted notes stored in Firebase database * Swift	repo	

@vixentael

4. Tutorials and use case specific apps

During the development stage, we frequently do Proof-of-Concept projects to test different assumptions. They serve as interesting demos (examples) of what Themis is capable of:

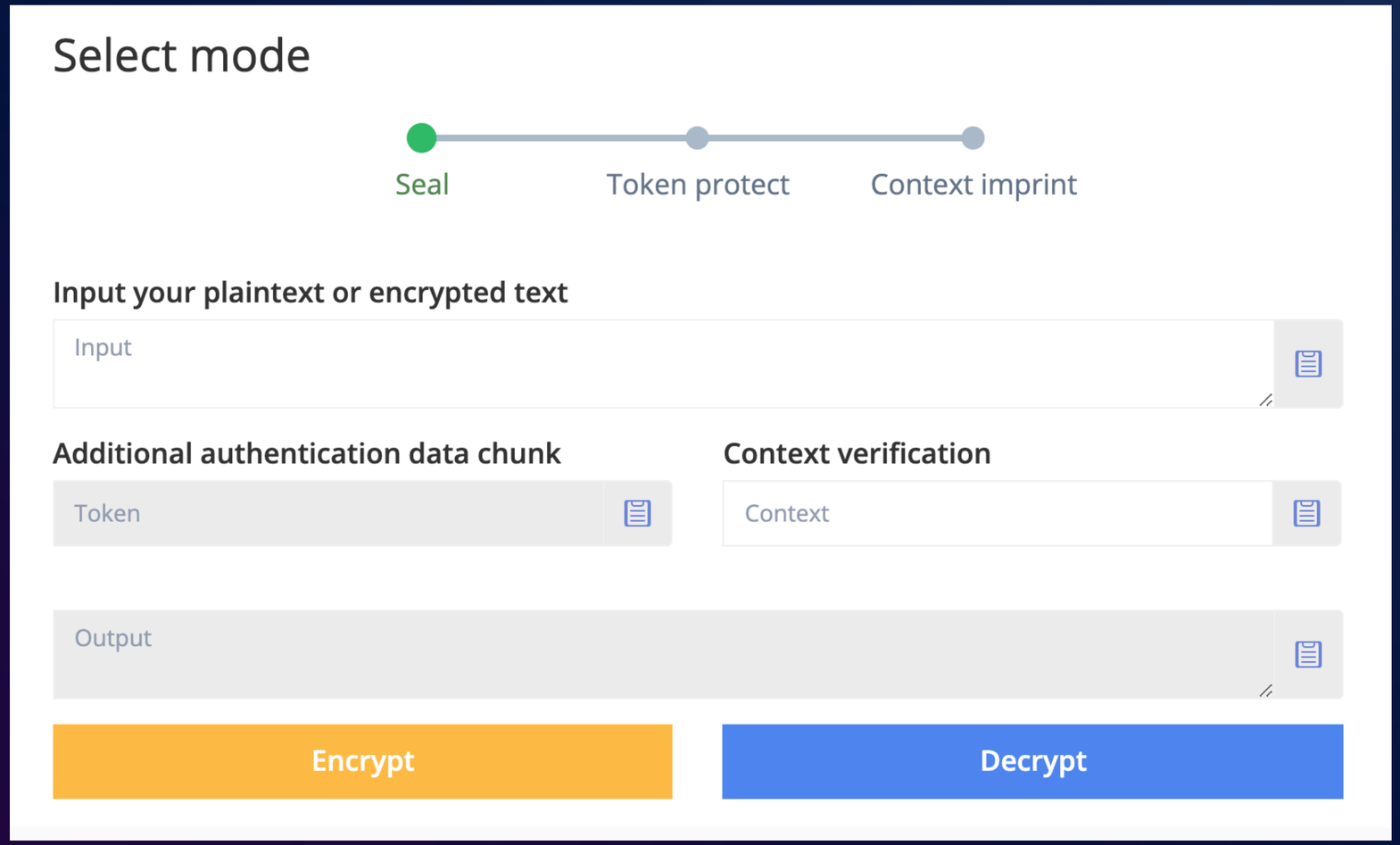
Demo	Description	Repo	Blog post
0fc	Anonymous web chat * Python * webthemis (C++ + HTML/JS)	repo	blog post
Sesto	Secure storage * Python * webthemis (C++ + HTML/JS)	repo	blog post
Swift Alps demo	Secure communication (iOS app with Python server based on Secure Session) * Swift * Python	repo	slides
Zero-Knowledge Architectures workshop	iOS app for storing and sharing encrypted notes stored in Firebase database * Swift	repo	

“your app works, but my app doesn’t”

@vixentael

5. Codeless simulators

verify own configuration

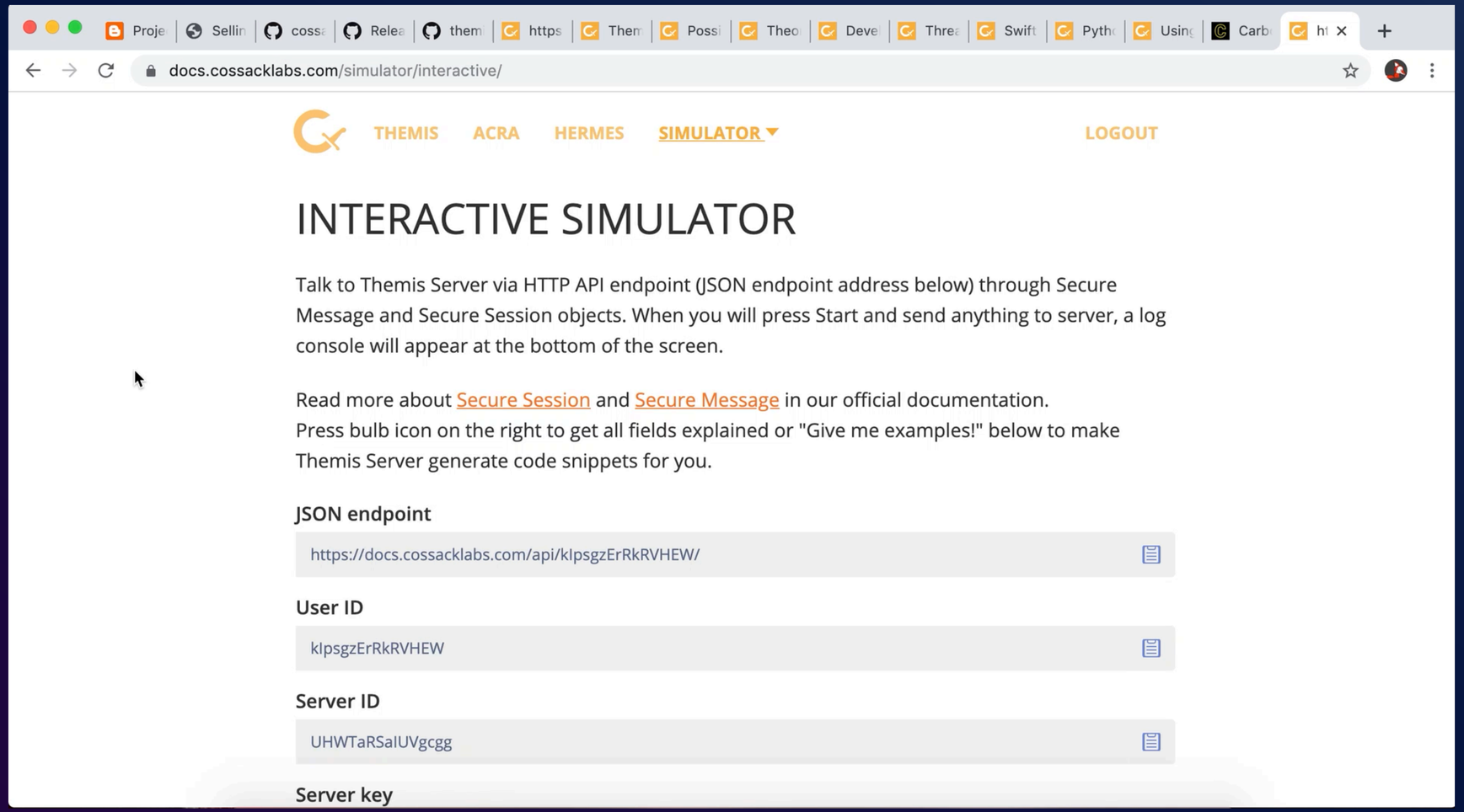


docs.cossacklabs.com/simulator/interactive/

@vixentael

5. Codeless simulators

debug the whole flow



@vixentael

One readme is not enough

1. Language-specific docs
2. “Safe to copypaste” code snippets
3. Example applications
4. Tutorials and use case specific apps
5. Codeless simulators

@vixentael

Community

Are you responsible for how users use your software?

@vixentael

Community

answering Q

- **Secure session vs SRP** question
  #529 by OlexandrStepanov was closed on Sep 23
- **[question] bindings for C# and .net** question
  #520 by dodikk was closed on Aug 15 1 of 1
- **Presence of 32bit native libraries without 64bit support** O-Android 🤖 question
  #479 by vanceanderson41 was closed on Jul 17 2 of 4
- **iOS - Lost key stored in Keychain when user change Device** question
  #342 by CodeTeamLabs was closed on Jan 12
- **Objective C - Store Public Key in Firebase Database and use AFNetworking SecurityPolicy** question
  #338 by CodeTeamLabs was closed on Jul 16
- **Themis iOS and BoringSSL: Objective-C Implementation** O-iOS 📱 question
  #336 by CodeTeamLabs was closed on Nov 12, 2018
- **Can I use secp256k1 keys and/or ECDH secret to encrypt using themis?** core help wanted ❤️ question
  #322 by FrankC01 was closed on Jul 16

@vixentael

Community

ATTACKS ON CRYPTOSYSTEMS

Table of Contents

POSSIBLE ATTACKS ON CRYPTOSYSTEMS IN THEMIS

nown attacks on the cryptoalgorhitms

THREAD SAFETY

Table of Contents

Projects that use Themis

Themis is widely-used for both non-commercial and commercial projects. We've mentioned some of the public projects in a blog post in December 2018, and a more complete list is as follows (in alphabetical order):

Credits and honourable mentions

People

Significant contributions to Themis encryption library have been made by Ignat Korchagin (secumod) and Andrey Mnazakanov (mnaza).

A special thank you goes to the awesome contributors that help us make Themis better:

@vixentael

Community

Submitting apps to the App Store

If you're using Themis as your means of encryption within your iOS/macOS app that you're submitting to the App Store, your encryption falls under the "open source" exception (although if your app is not open source/distributed free of charge, we strongly recommend that you seek legal advice).

Themis is a free cryptographic library that builds on the existing, community-tested cryptographic instruments (OpenSSL, LibreSSL, BoringSSL, depending on the target platform). It is open source and Apache 2-licensed, with its full source code publicly available online on GitHub.

This means that you should indicate that you're using encryption and only submit annual self-classification reports (use this handy table to self-check). Read more about Apple regulations on cryptography and check Apple docs.

@vixentael

Where Themis is used?

mobile apps

web-first apps

other libraries

Cossack Labs software

e-commerce

fintech

chats

cossacklabs.com/blog/themis-contributors-and-projects-2018.html

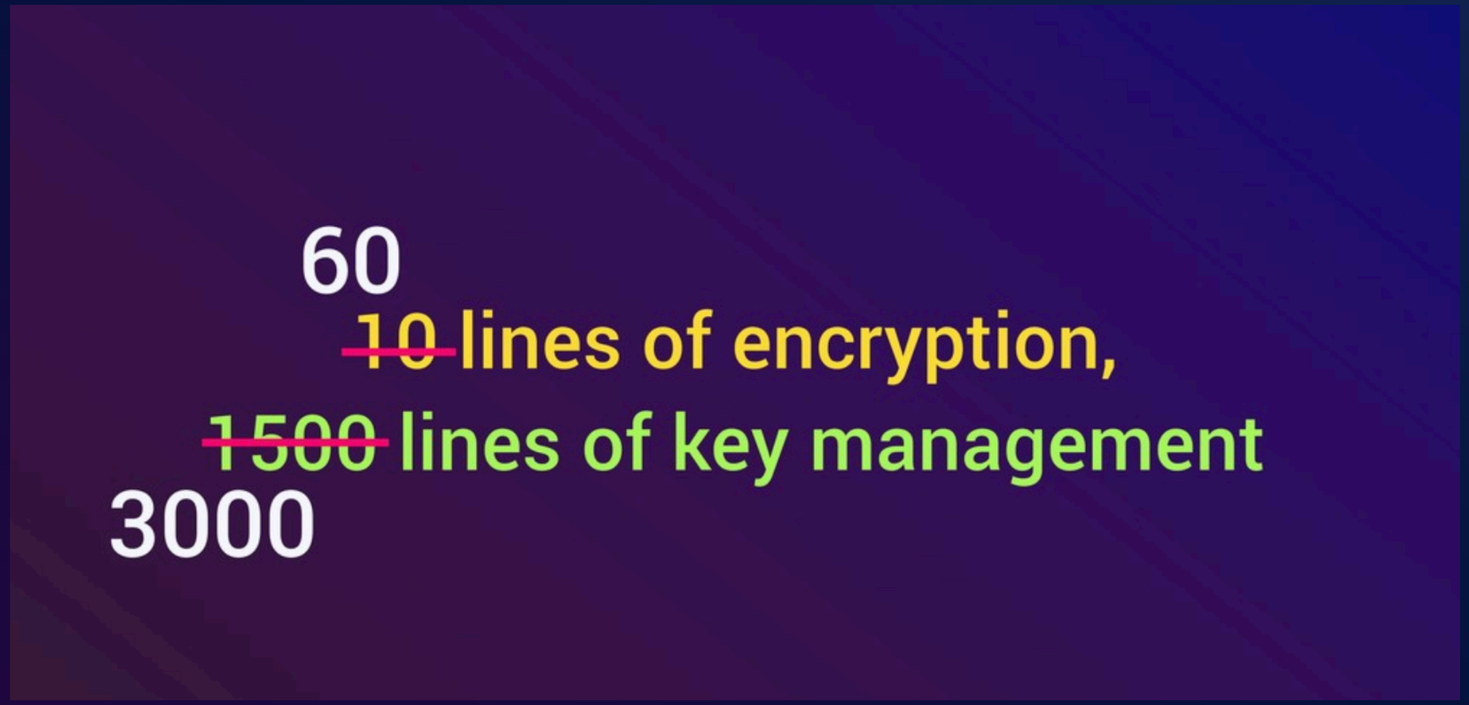
@vixentael

<HospitalApp>: cryptocore 130 lines

```
// ================================================================================
// AES
// ================================================================================

/// generates random AES256 key
func ___generateAES256Key(count: Int = 32) throws -> Data {

    var keyData = Data(count: count)
    let status = keyData.withUnsafeMutableBytes {
        SecRandomCopyBytes(kSecRandomDefault, count, $0.baseAddress!)
    }

    guard status == errSecSuccess else {
        throw CryptoStatus(status: status)
    }

    return keyData
}

/// encrypt bytes with simmetric key
func ___encrypt(data: Data, withSimmetricKey key: Data) throws -> Data {
    // init Secure Cell in seal mode
    guard let cellSeal = TSCellSeal(key: key) else {
        throw CryptoError.encryptSeal
    }
```

@vixentael

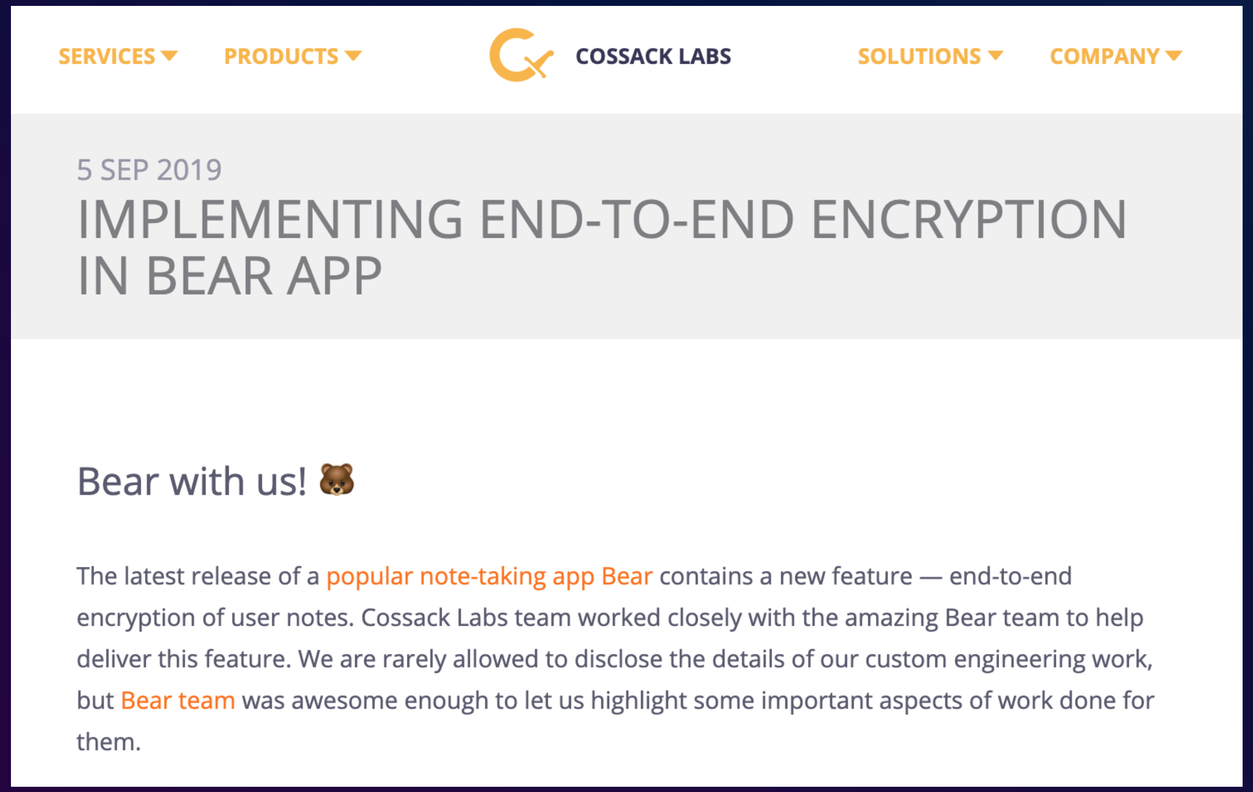
Bear: cryptocore 60 lines



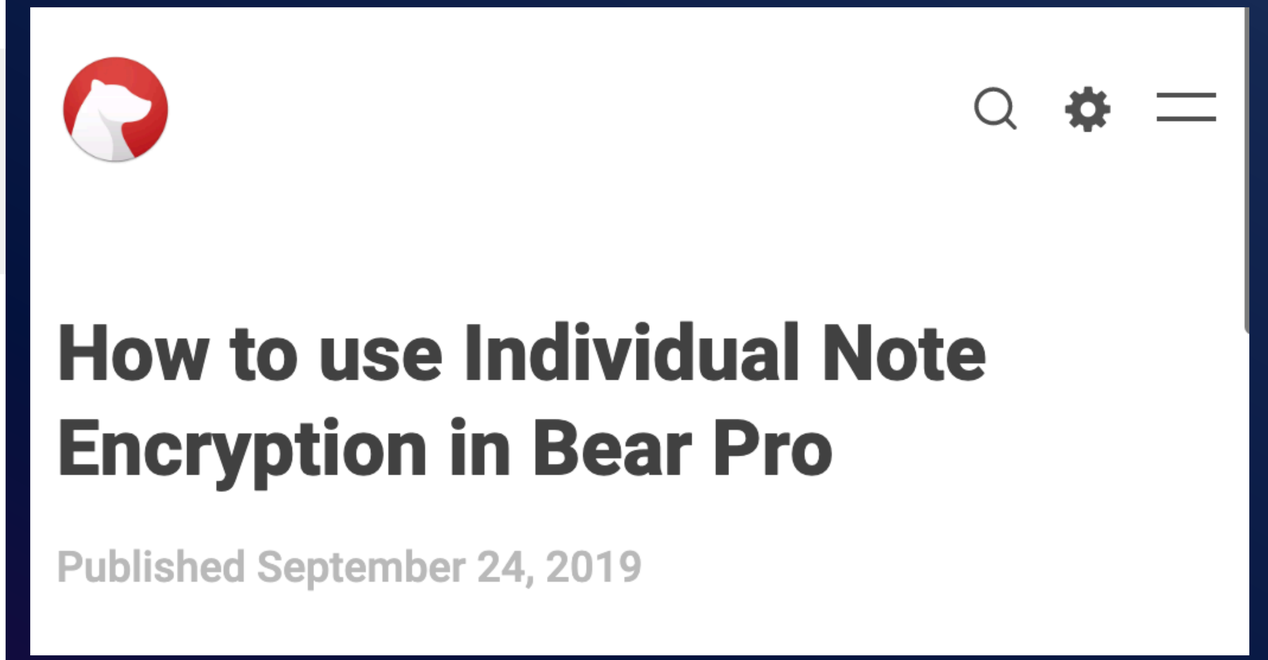
speakerdeck.com/vixentael/10-lines-of-encryption-1500-lines-of-key-management?slide=42

@vixentael

End-to-end encryption in Bear



cossacklabs.com/blog/



blog.bear.app/

@vixentael

notes protection (e2ee)

passphrase KDF

auto-locking timer

obfuscation

hint encryption

encrypted user settings

anti-RE & anti-debugging

zeroing secrets

failed attempts counter

continuous improvements

TLS / certificate pinning

prepare for incidents

cossacklabs.com/blog/end-to-end-encryption-in-bear-app.html

@vixentael

Key points

Good tools allow to focus on product, not on crypto code

@vixentael

1. Encryption lib should be: multi-platform, maintained, secure by default, open sourced, easy to install, hard to misuse, tested.

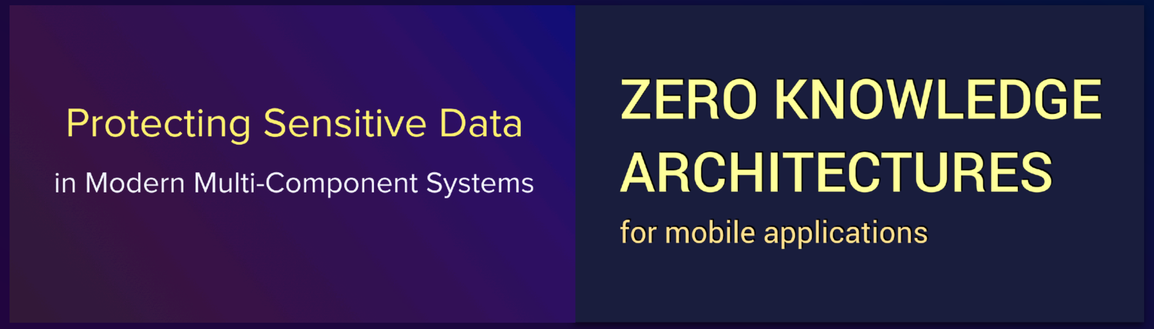
2. Supporting libs is VERY complicated.

3. Better to spend time on features than the crypto code.

@vixentael

@vixentael

COSSACK LABS



github.com/vixentael/my-talks