Next-Generation Cryptographic Services

I'm going to tell you, and I won't kill you

Session 212

Jon Callas
Security Privateer

Introduction

- Learn about exciting changes to Apple's cryptographic architecture
- First major reworking of crypto in a decade

What You Will Learn

- Changes to existing APIs
- New Transform API
- How to use Apple's Transform library
- How to create your own custom Transforms

CDSA

Common Data Security Architecture

CDSA Is Deprecated

- It is a creature of its time
- That time is the late 1990s
- It is a thinly used, Open Group standard
  - All of the costs, few of the benefits
- Only a Mac OS API, not iOS

Deprecated Does Not Mean Canceled

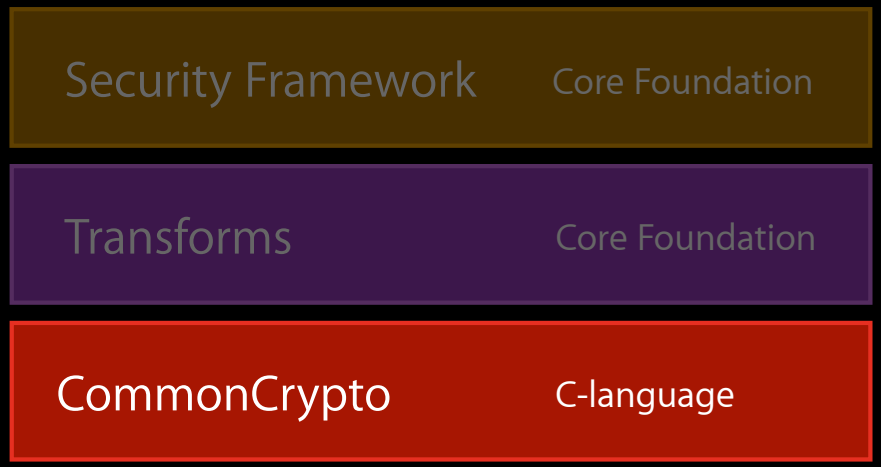
- CDSA is still available
- Start migrating away now
- Some parts of Security Framework have layering issues with CDSA
  - Those parts are deprecated, too

Requirements for a Replacement

Design for the decades ahead

- Less code
- Faster code
- Concurrent code
- Flexible programming
  - Crypto includes ciphers, compression, XML, networking, REST, LDAP, databases…

New Crypto Architecture



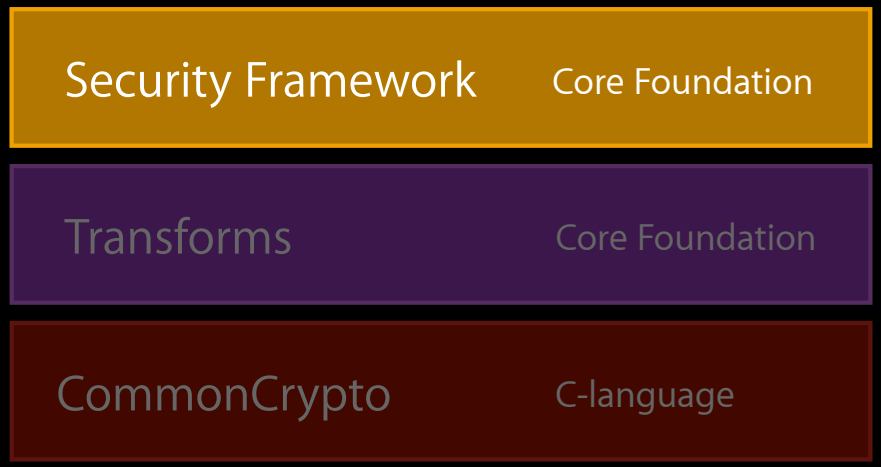
- Low-level (pointer, length)
- Basic core algorithms
- New — Only on Mac OS — PS 140
- Traditional crypto toolkit programming
- Documentation in man pages
  - man CC_crypto

New CommonCrypto Architecture

- Starts from Snow Leopard's CommonCrypto
- Recoded internals
  - Both Mac OS, iOS
    - NIST algorithm certificates for iOS
- Automatic selection of optimized implementation

- Raw algorithms, minimal protocol

Security Framework



- Based on Core Foundation data types
- All APIs that used CDSA types are deprecated
- New APIs replace the deprecated ones
  - CDSA data types migrated or removed

Transforms

Security Framework — Core Foundation

Transforms — Core Foundation

CommonCrypto — C-language

- Built on Grand Central Dispatch
  - GCD is work-oriented with queues
  - Transforms are dataflow-oriented
- Model is data pipeline, not threads or queues
  - Act like a generic function in many helpful ways
- General-purpose concurrency mechanism

Gotchas

- Transforms are only in Lion, not iOS
- CommonCrypto is in both Lion and iOS

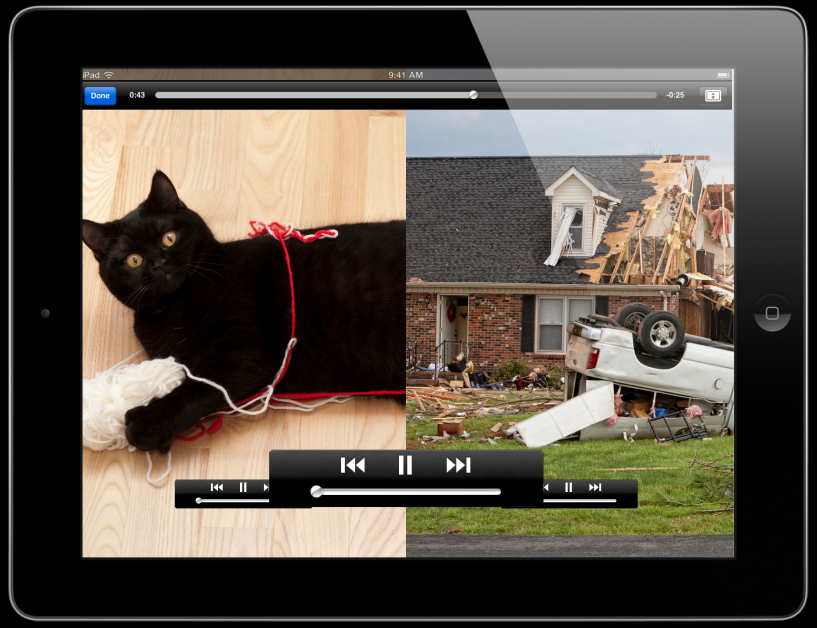
- Not (yet) a full CDSA-replacement
  - We are working on the missing pieces
    - Certificate handling details

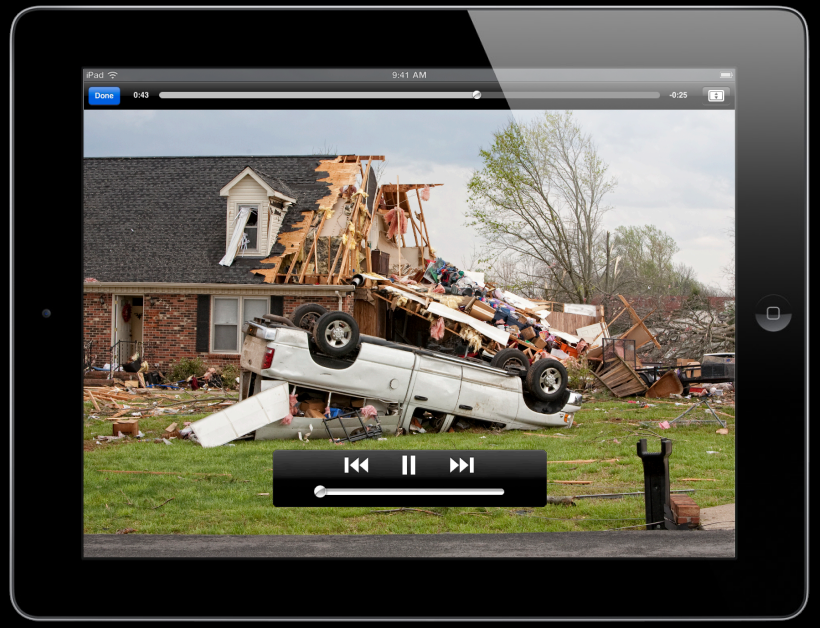
Tell us what you need at bugreporter.apple.com

What Are Transforms?

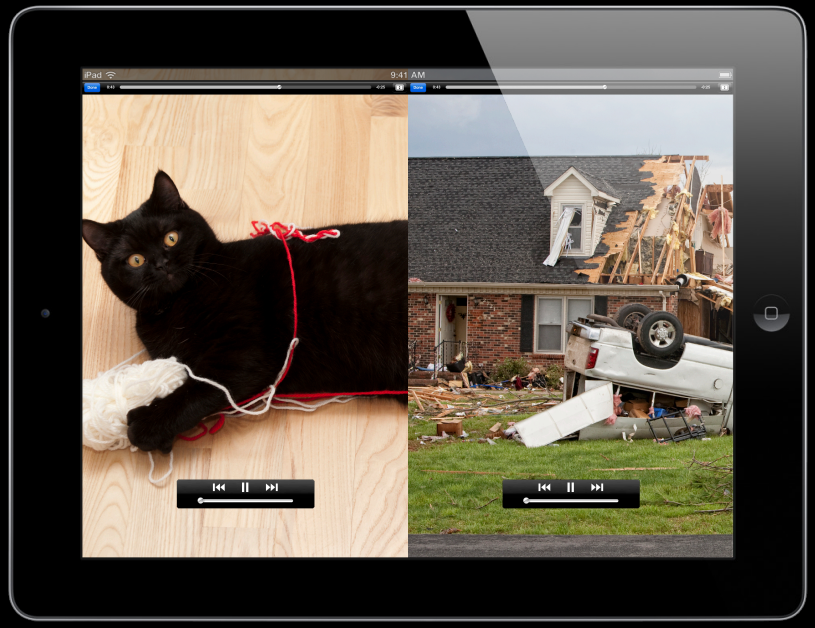
- Processing anything with a sequence of data changes
- Each small change is chained together
- Each change works in parallel
- Not just for crypto
  - Process RSS feeds into a mashup
  - Sequence of code optimizers

Transforms





Transforms



Analogies

Like UNIX Pipelines

- Oriented toward data chunks, not bytes
- Routines rather than processes
- Multiple connections are trivial

Like AudioUnits

- Concurrent
- Not isochronous nor real time

Like Quartz Composer

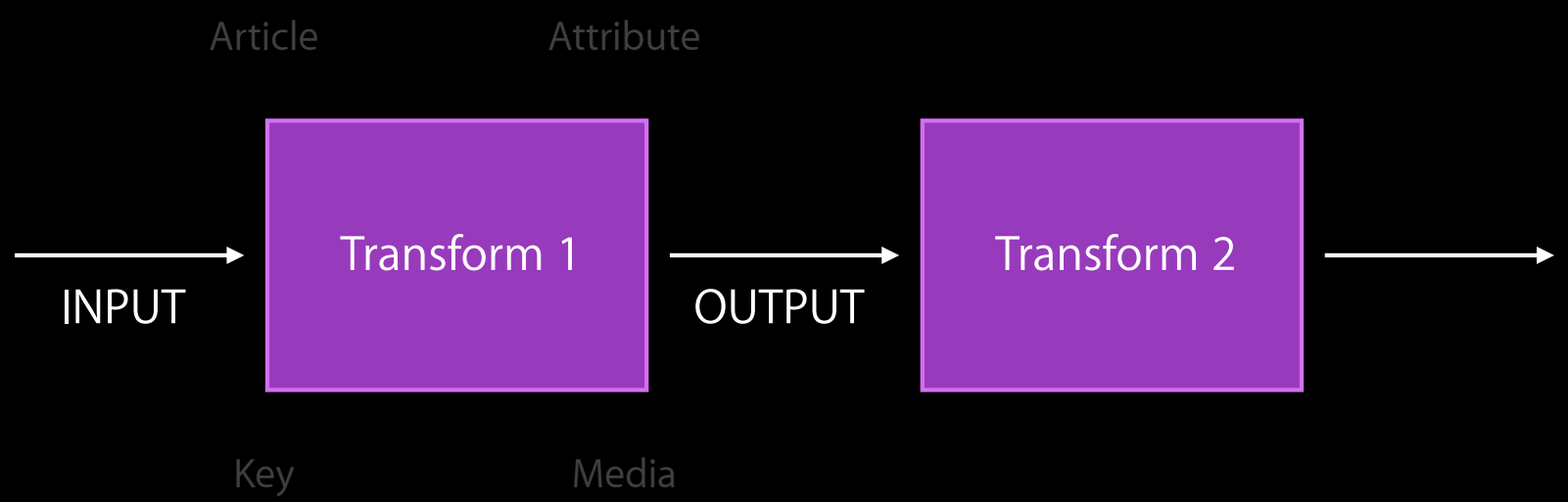
- General data processing, not just images
- Not a visual programming model

What Are Transforms?

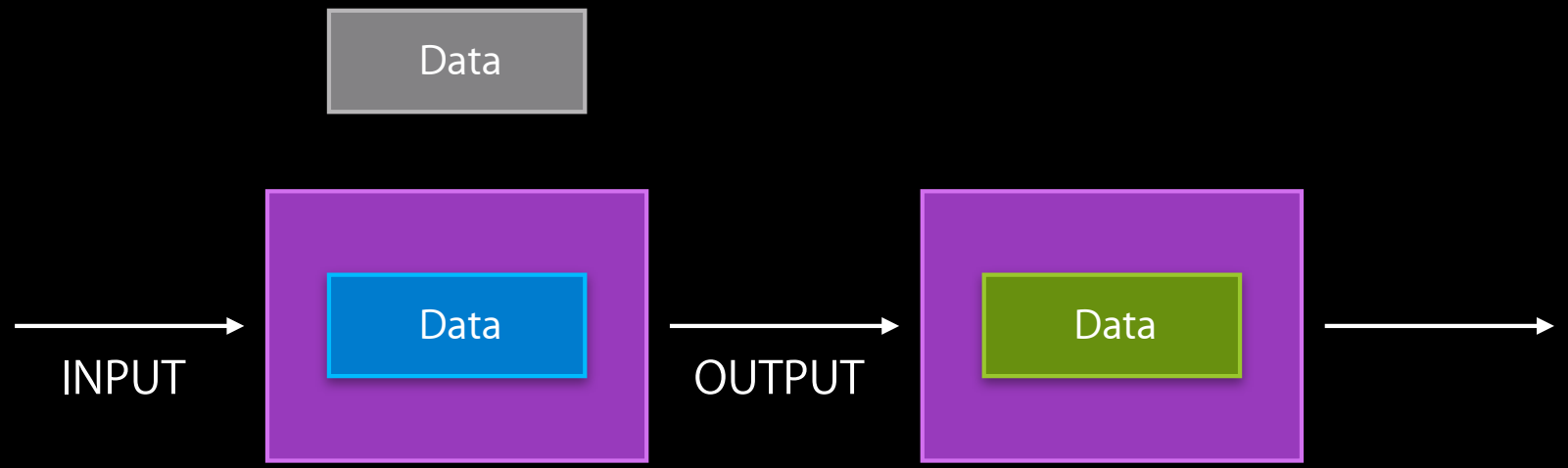
- Data-driven interface to GCD
- CoreFoundation-level design
  - Provides needed data structures
  - Toll-Free Bridging to Cocoa
- Not a kernel interface

Example

Layout


Article
Attribute
Transform 1
Transform 2
INPUT
OUTPUT
Key
Media

Operation



Transform Machinery

Features

- Externalizable with re-internalizing
  - Transform sets can be saved and restored
- Data interface through Core Foundation types
  - Core Foundation is toll-free bridged to NS equivalents in Cocoa
  - All Transforms easily used in Objective C or C
- Routines (blocks), not processes

More Features

- Attributes + Connections make data flow between transforms
- Data to a connected attribute is like a pipe
- Automatic I/O flow between all transforms

Attributes and Flow Control

- Each attribute has an input queue
  - Transforms stall when sending to an attribute with a full queue
- A transform can pushback a value onto an attribute
  - Single pushback value per attribute
- Attributes are asynchronous, serialized in a transform

Special Attributes

INPUT, OUTPUT, LABEL, ABORT, DEBUG

- INPUT and OUTPUT are conventional data flow
- LABEL is the transform's name
- Setting ABORT signals an error
- DEBUG logs information

Attribute Data

- Most strictly, they are all CFTypeRefs
- Most commonly, they are CFData
  - A transform library can make its own restrictions that make sense
- An attribute can have multiple destinations

Let Us Solve a Problem

Send binary data on the web

- The Internet likes seven-bit-clean data
- Typically this means binary becomes base-64

Code Example

Encode some data with base-64

```
CFDataRef dataToEncode;
CFErrorRef error = NULL;

SecTransformRef encodingRef = SecEncodeTransformCreate(
                                    kSecBase64Encoding,&error);

SecTransformSetAttribute(encodingRef,
                         kSecTransformInputAttributeName,
                          dataToEncode, &error);

CFDataRef resultData = SecTransformExecute(encodingRef, &error);
```

A More Complex Example…

Encrypt and base-64

```
SecKeyRef key;
CFDataRef dataToEncryptAndEncode;
SecTransformRef encryptionRef = SecEncryptTransformCreate(keyRef, &error);
SecTransformRef encodingRef = SecEncodeTransformCreate(kSecBase64Encoding,
                                                       &error);
SecGroupTransformRef group = SecTransformCreateGroupTransform();

SecTransformConnectTransforms(
                  encryptionRef, kSecTransformOutputAttributeName,
                  encodingRef, kSecTransformInputAttributeName, group,
                   &error);

SecTransformSetAttribute(encryptionRef, kSecTransformInputAttributeName,
                  dataToEncryptAndEncode,  &error);

SecTransformExecuteAsync(group, ^(data, error, isFinal){} );
```

Lion's Transform Library

- Basic cryptography
  - Encrypt/decrypt, sign/verify, hashes, key wrapping, string-to-key
- Encoding
  - Base-32, base-64, Zlib compression
- Miscellaneous
  - File reading

Custom Transforms

Sample Code in Headers

Complete example Custom Transform

- In the header for `<Security/SecCustomTransform.h>`
- Creates a custom transform for the Caesar cipher

Operation and Lifecycle

- Transforms run via a series of lifecycle events
- Each event executes a block
- Each event can be overridden with a new block
- No overrides = null transform
  - Creating a new transform starts with the null transform and tailors it

Advanced Override Use

- Overrides are not just once
- You can re-override with new handlers
- Useful for selecting implementations, etc.
- Very high-level operations can be very fast

Attribute Set Notification

- This event can happen for
  - All attributes
  - A named attribute
- A specific handler occludes the generic

Process Data

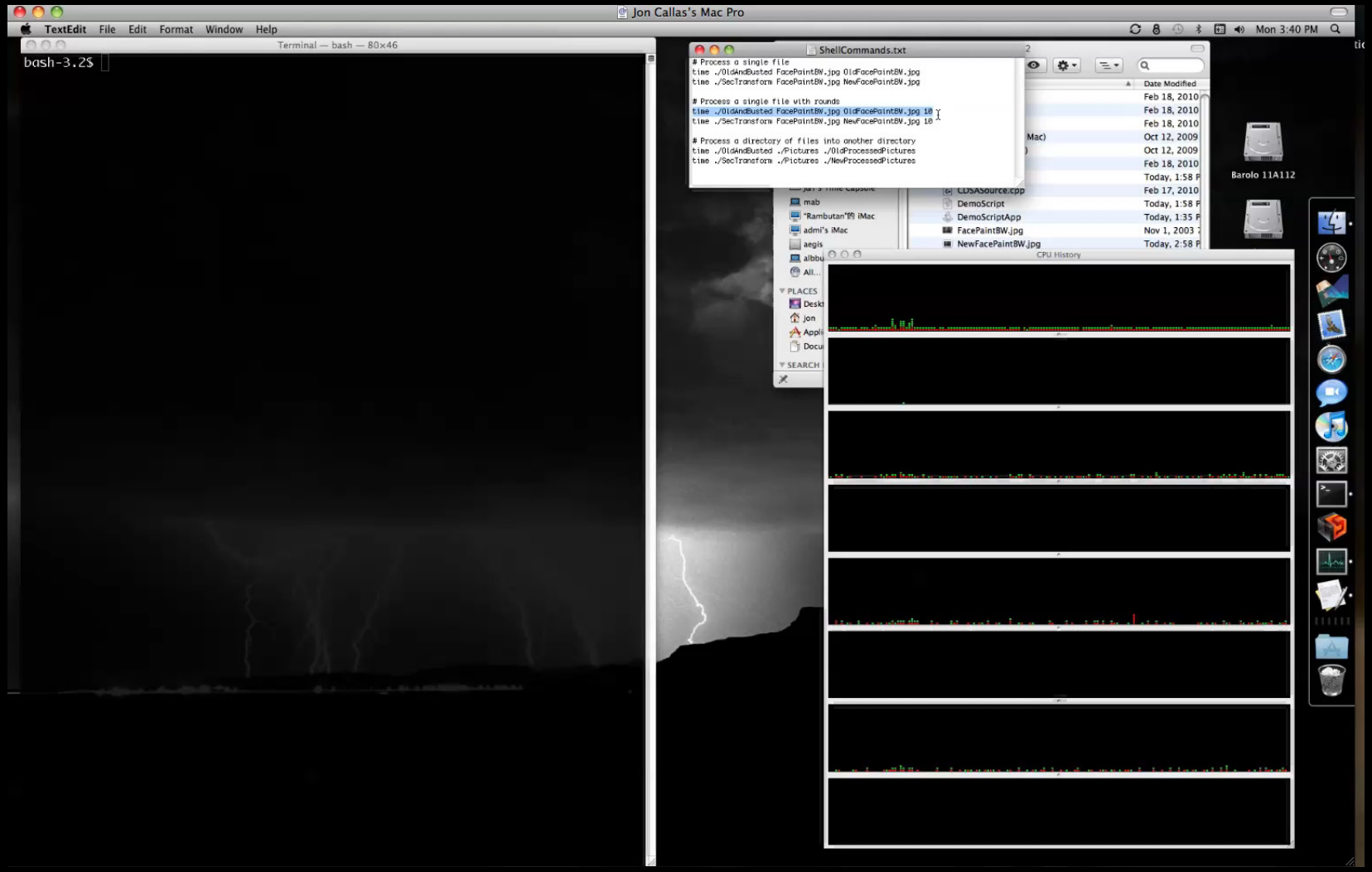
- A shortcut for handling the process between INPUT and OUTPUT
- Process Data is called with the datum that was sent to INPUT
- Its return value is sent to OUTPUT and to the next transform's INPUT
- NULL is sent when there is no more data
  - Return NULL when you want to close things down

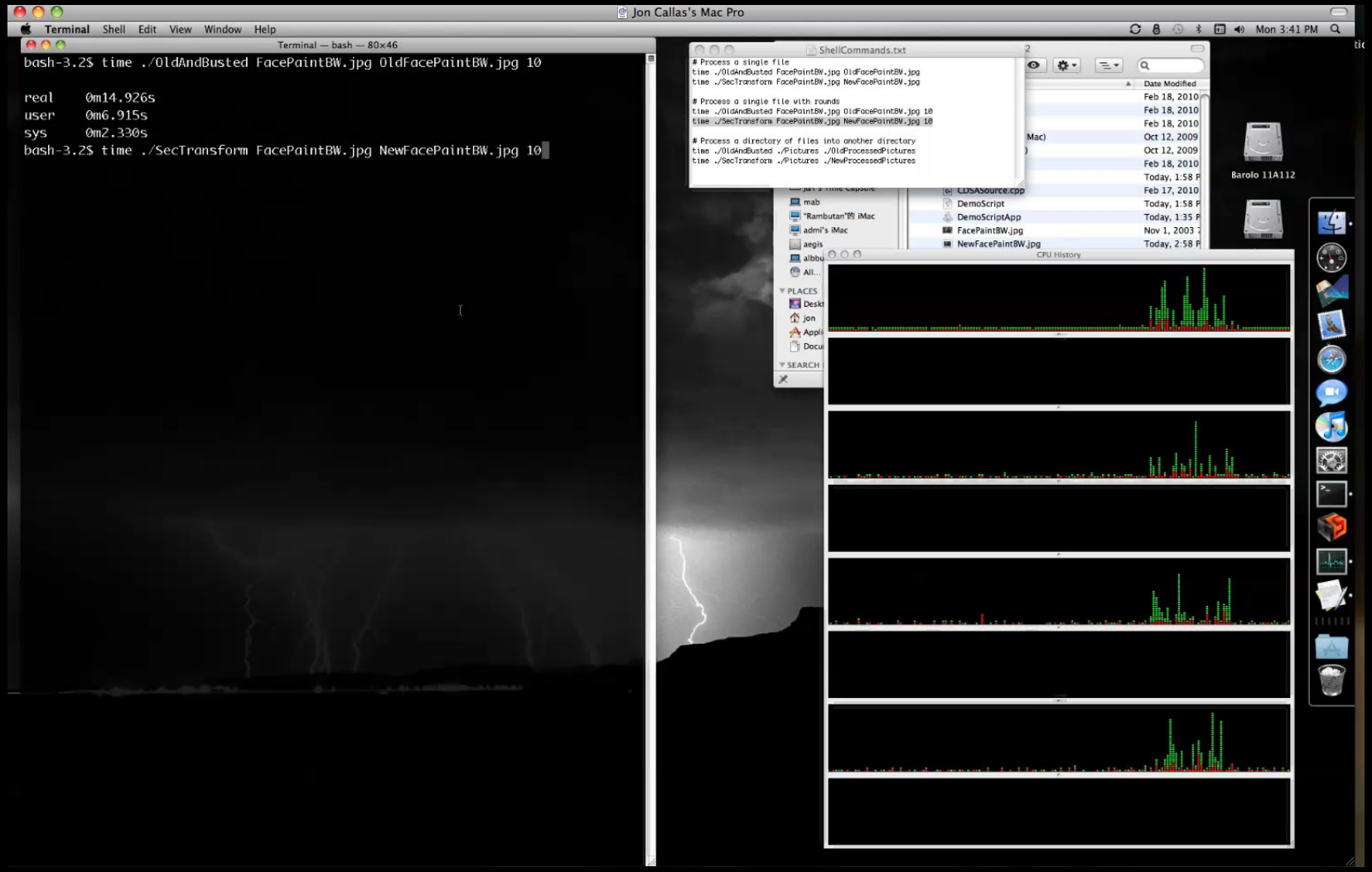
Lifecycle Functions

- Execute Starting
  - Called just before the INPUT stream starts
- Finalize
  - Called just before release

Code Size Comparison

Speed Comparison


Jon Callas's Mac Pro
TextEdit File Edit Format Window Help
Mon 3:40 PM
Terminal — bash — 80×46
bash-3.2$
ShellCommands.txt
Process a single file
time ./OldAndBusted FacePaintBW.jpg OldFacePaintBW.jpg
time ./SecTransform FacePaintBW.jpg NewFacePaintBW.jpg
Process a single file with rounds
time ./OldAndBusted FacePaintBW.jpg OldFacePaintBW.jpg 10
time ./SecTransform FacePaintBW.jpg NewFacePaintBW.jpg 10
Process a directory of files into another directory
time ./OldAndBusted ./Pictures ./OldProcessedPictures
time ./SecTransform ./Pictures ./NewProcessedPictures
Date Modified
Feb 18, 2010
Feb 18, 2010
Feb 18, 2010
Oct 12, 2009
Oct 12, 2009
Feb 18, 2010
Today, 1:58 P
Feb 17, 2010
Today, 1:58 P
Today, 1:35 P
Nov 1, 2003
Today, 2:58 P
Barolo 11A112
mab
"Rambutan"(f) iMac
admi's iMac
aegis
PLACES
jon
SEARCH
CDSASource.cpp
DemoScript
DemoScriptApp
FacePaintBW.jpg
NewFacePaintBW.jpg
CPU History


Jon Callas's Mac Pro
Terminal Shell Edit View Window Help
Mon 3:41 PM
Terminal — bash — 80×46
bash-3.2$ time ./OldAndBusted FacePaintBW.jpg OldFacePaintBW.jpg 10
real 0m14.926s
user 0m6.915s
sys 0m2.330s
bash-3.2$ time ./SecTransform FacePaintBW.jpg NewFacePaintBW.jpg 10
ShellCommands.txt
Process a single file
time ./OldAndBusted FacePaintBW.jpg OldFacePaintBW.jpg
time ./SecTransform FacePaintBW.jpg NewFacePaintBW.jpg
Process a single file with rounds
time ./OldAndBusted FacePaintBW.jpg OldFacePaintBW.jpg 10
time ./SecTransform FacePaintBW.jpg NewFacePaintBW.jpg 10
Process a directory of files into another directory
time ./OldAndBusted ./Pictures ./OldProcessedPictures
time ./SecTransform ./Pictures ./NewProcessedPictures
Date Modified
Barolo 11A112
CDSASource.cpp
DemoScript
DemoScriptApp
FacePaintBW.jpg
NewFacePaintBW.jpg
CPU History

Results of the Comparison

Transforms vs. CDSA smackdown

10.1%

Transform vs. CDSA
Lines of Code

7.1×

Transform vs. CDSA
Performance Gain

Summary

- Originally a way to speed up crypto
- Became general concurrent programming
- Builds on the shoulders of GCD and CF
- Order-of-magnitude improvements in code size and speed

OpenSSL

We Are Deprecating OpenSSL Dylibs

Why?

- Not designed for upward compatible binary release
- Lack of stable ABI means a dylib per OpenSSL release
- We need a FIPS 140 platform

Migration Strategy

Why?

- You write apps for Mac OS or iOS
- You like our new technology
  - Automatic algorithm optimizations
  - Concurrency
  - Small code
- You want to inherit our evaluations
- We are taking this path ourselves

Please tell us what you need at bugreporter.apple.com

Migration Strategy

What to do

- CommonCrypto for most basic algorithms
  - Most like OpenSSL
  - Still missing asymmetric, etc.
- Transforms and Security Framework in apps
  - Concurrent, scalable
- Secure Transport
  - Our SSL/TLS library
  - Comes for free in CFNetwork, NSURL*

Please tell us what you need at bugreporter.apple.com

Remaining with OpenSSL

Why?

- You are doing a Unix application
- You use something that uses OpenSSL
- It is the devil you know

Remaining with OpenSSL

What to do

- Get OpenSSL from MacPorts.org
  - Most recent version, 1.0.0d is there
  - Statically link the libraries
  - Include anything else in your bundle
- This is what the OpenSSL team recommends
  - Static linking is the binary model
  - It prevents dylib version skew

Wrap-up

- CDSA and OpenSSL dylibs are being deprecated
  - Help us provide replacements
- CommonCrypto is low-level
- Transforms are CF-level
  - Originally created for cryptography
  - Became general-purpose concurrent programming
  - Builds on the shoulders of GCD and CF
  - Order-of-magnitude improvements in code size and speed

Related Sessions

Security Overview	Nob Hill Tuesday 11:30AM

Labs

Security Lab	Core OS Lab B Friday 11:30AM


WWDC2011