SGX Enabled Functional Encryption and Applications

Ben Fisch
Stanford University
Functional Encryption (FE)

1. **Public Key (PK)**: The public key is used for encryption and verification.
   - **E(PK, m)**: Encryption using the public key.

2. **Key Manager**:
   - **setup -> pk, msk**: Setup phase that generates the public key (pk) and master secret key (msk).
   - **keygen(msk, f) -> skf**: Key generation phase that generates a secret key for a function (skf).
   - **encrypt(pk, m) -> ct**: Encryption phase that encrypts a message (m) using the public key (pk) to produce a ciphertext (ct).
   - **decrypt(skf, ct) -> f(m)**: Decryption phase that decrypts the ciphertext (ct) using the secret key (skf) for a function (f) to produce the result (f(m)).

3. **Secret Keys (SKf1, SKf2, SKf3)**: Secret keys corresponding to different functions (f1, f2, f3).
   - **f1(m)**: First function applied to the message (m).
   - **f2(m)**: Second function applied to the message (m).
   - **f3(m)**: Third function applied to the message (m).
Multi-Input FE

Public:
- PK

Key Manager:
- MSK

Server:
- SK_f

E(PK, m1)
E(PK, m2)
E(PK, m3)

f(m1, m2, m3)
Intel SGX

Enclave Application

Remote Attestation

Source: ISCA 2015 tutorial slides for Intel SGX
SGX FE Design

Key Manager Platform

- Key Manager Enclave
- PK
- SK
- VK
- SK'

Decryption Platform

- Decryption Enclave
- SK

Authentic Channel

KME Tasks:
- Setup Public Key Cryptosystem
- Send SK to Decryption Enclave(s)
- Setup Signature Scheme
- Approve and sign functions

“Skf”

\(<f>, \text{sig}\)

\E(pk, m)

\(sk_f = (<f>, \text{sig})\)

\(<f>\)

\f(m)\)
Implementation Challenges

• How to represent the function f:
  ✴ Cannot move code into enclave after EINIT
  ✴ Implement interpreter in enclave?
  ✴ Separate enclave runs f and gives attestation?

• Memory access patterns leaked

• Timing attacks
Genomics Application

Genome Database

GWAS studies

Study 3

Study 2

Study 1

Researchers/Doctors

SKf3

SKf2

SKf1
Smart Devices Application

Devices collect data

Encrypt data independently

Apps in cloud learn restricted functions
Better Buildings with BLE Sensors and Gateways

Brad Campbell – bradjc@umich.edu

Distributed applications running on gateways

Energy attribution and analysis

Real-time monitoring
Michigan Architecture for Sensing Systems

- Temperature
- Pressure
- Humidity
- Light
- Acceleration
- Motion
- Power
- Voltage
- Watt-Hours
- Power Factor

Reference to interface description

Data translation script

{ "device": "BLEES", "id": "c098e5300069", "pressure_pascals": 98423.5, "humidity_percent": 41.33, "temperature_celcius": 23.62, "light_lux": 49, }

Translation sandbox

Human readable data stream

MQTT Translation sandbox
The Toastboard

“Professional software development environments usually provide a debugger, which helps programmers to locate and fix faults ... Unfortunately, physical computing does not have analogous support tools and thus it was sometimes difficult for participants in our study to identify what the problem was. “

Daniel Drew
Prof. Bjoern Hartmann
1. LED BAR INDICATORS
2. SCAN BUTTON
3. QUANTITATIVE DATA
4. COMPONENT TESTERS
5. TRANSIENT ANALYSIS
void setup()
{
    stream.setLabelsForAllDimensions({"x", "y", "z"});
    useStream(stream);

    calibrator.setCalibrateFunction(processAcceleroMeterData);
    calibrator.addCalibrateProcess("Resting",
        "Rest accelerometer on flat surface.", restingDataCollected);
    useCalibrator(calibrator);

    pipeline.setClassifier(DTW(false, true, threshold));
    pipeline.addPostProcessingModule(ClassLabelTimeoutFilter(timeout));
    usePipeline(pipeline);

    registerTuneable(threshold, 0.1, 3.0,
        "Similarity",
        "How similar a live gesture needs to be to a training sample.""
        "The lower the number, the more similar it needs to be.");
    registerTuneable(timeout, 1, 1000,
        "Timeout",
        "How long (in milliseconds) to wait after recognizing a "
        "gesture before recognizing another one.");

    useOStream(oStream);
}
Machine Learning for Makers
David Mellis, Ben Zhang, Audrey Leung, Bjoern Hartmann

Iterative Refinement of Training Data

Calibration Across Sensor Models
```
calibrator.setCalibrateFunction(processAccelerometerData);
calibrator.addCalibrateProcess("Resting", "Rest accelerometer on flat surface, w/ z-axis vertical.", restingDataCollected);
calibrator.addCalibrateProcess("Shaking", "Shake accelerometer vigorously in all directions.", shakingDataCollected);
useCalibrator(calibrator);
```

Tuning of Parameters
```
registerRunnable(timeout, 1, 3000,
"Timeout": "How long (in milliseconds) to wait after recognizing a "
"gesture before recognizing another one.");
```

Feedback on Signal Quality
```
if (stddev[0] / range > 0.05 ||
stddev[1] / range > 0.05 ||
stddev[2] / range > 0.05)
return CalibrateResult(CalibrateResult::WARNING,
"Accelerometer seemed to be moving; consider recollecting the "
"calibration sample.");
```

VARIABILITY 0.40
How different from the training data a new gesture can be and still be considered the same gesture. The higher the number, the more similar it can be.

TIMEOUT 500.00
How long (in milliseconds) to wait after recognizing a gesture before recognizing another one.
Drill Sergeant:
Supporting Physical Construction Projects through an Ecosystem of Augmented Tools

Eldon Schoop & Michelle Nguyen, Mitchell Karchemsky, Daniel Lim, Valkyrie Savage, Bjoern Hartmann & Sean Follmer


```json
...
  "tool": "drill",
  "task": "drillToDepth",
  "dynamic_params": ["depth"],
  "depth": "screw_depth"
}, ...
```

instructions

object

paper templates
Private Data Collection in an Internet of Things
Henry Corrigan-Gibbs and Dan Boneh

• How can we collect privacy-sensitive data in a privacy-preserving way?
• Can we efficiently collect interesting aggregate statistics without collecting any individual user’s data?
• What are the theoretical limits on the efficiency of these techniques?
Motivation

- California drought
- Collaboration with earth sciences and residential housing departments to reduce water usage across campus
- Gain an understanding of the patterns driving water consumption
Tethys: Energy Harvesting Networked Water Flow Flow Sensor
Auditing IoT Communications with TLS-RaR

Judson Wilson, Henry Corrigan-Gibbs, Riad S. Wahby, Keith Winstein, Philip Levis, Dan Boneh

Stanford University
How can we audit TLS communication between our IoT devices and the cloud?

Secure Devices:
- No man-in-the-middle
- Signed firmware

*Nest used for illustrative purposes only.*
Goals

1) Preserve End-to-End Integrity
2) Fewest changes possible to TLS
3) Avoid changes to server side
Solution: TLS-RaR

A standard TLS connection...

Begin TCP Connection

Enter TLS Session

Handshake

AES-GCM

Encrypted Session

Time
Solution: TLS-RaR
Use standard TLS features to **rotate** keys,
Solution: TLS-RaR

Use standard TLS features to **Rotate** keys, and then securely **Release** the previous keys to auditing devices.
CESEL

- Flexible hardware architecture for accelerating cryptography

- Fast Hash Function
- Finite Field Coprocessor
- R-LWE Coprocessor
- HW RNG and HW Counter

Micro-controller with extended instructions:
- S-Box
- Polynomial Multiplication
- Vector Arithmetic
Problem

Internet of Things applications are complex distributed systems that include embedded devices, Internet gateways, and backend cloud services.

Their software often uses three or more programming languages, operating systems, and processor architectures for devices with dramatically different resources. This heterogeneity makes applications error-prone, laborious to develop and notoriously insecure.
How can we make the development process of IoT applications simpler and more secure?

If you know the answer or want to hear an idea, come and talk with me!
IoT generates a huge amount of times series data
Time series can be hard to interpret - *how to prioritize human attention?*

**Challenge:** reduce noise while preserving interesting features
ASAP: Automatic Smoothing Parameter Selection for Time Series Visualization

(Kexin Rong, Peter Bailis)

Motivation

- Prioritize user attention

Key Insight

- Smooth as much as possible without losing “interesting” features

Approach

- Optimize signal smoothing parameters on the fly
Embedded Device Generation

Embedded Device Generation

seeed
Open Hardware Facilitator
Embedded Device Generation

Application Logic
(Used as Specification)

Traditional Embedded Development

Hardware & Firmware
(Satisfies Specification)

"Real World"

Formal Domain

Analysis

Component Library

Synthesis

Seeded Component

Formalized Schematic
MIC DROP: Massive IoT Computations via Dimensionality Reduction Optimization

Sahaana Suri
Internet of Things
Millions of heterogeneous sensor readings per second
Gain real-time insight into system behavior via automated tools and machine learning systems.
Automatically select dimensionality reduction technique that preserves information of interest!

Quickly determine how to uncover data’s intrinsic dimensionality subject to latency constraints!

Gain real-time insight into system behavior via automated tools and machine learning systems!

Millions of heterogeneous sensor readings per second!
Learn quickly and bound error by training on bootstrap-based samples

Train multiple models at once via memoization

Seed techniques by using results from previously tested techniques

Gain real-time insight into system behavior via automated tools and machine learning systems

Millions of heterogeneous sensor readings per second
Automated Arbitrarily Complete Full-Loopback Driver Verification

Sergio Benitez, Alejandro Russo, David Mazieres

Problem:

91% of critical CVEs caused by drivers leading to serious vulnerabilities

NetUSB flaw leaves 'millions' of routers, IoT devices vulnerable to hacking

The flaw can be exploited to conduct denial-of-service attacks or remote hijacking.

Observation:

Devices modeled by state machines

Our Approach:

Guarantee device is programmed according to state machine at compile time via programming language properties
Automated Arbitrarily Complete Full-Loopback Driver Verification

Sergio Benitez, Alejandro Russo, David Mazieres

Completed:

✓ Formalized and proved type system.
✓ Formalized and proved correctness properties.
✓ Developed (small) OS kernel and drivers.
✓ Found bugs in manuals and hardware!

Observation:

Device specification/hardware still susceptible to bugs.
State “actions” cannot be statically checked.

Approach:

Generate (arbitrarily) complete correctness tests using SM model.
Run tests against real hardware (full-loopback).
When developers write code, they make **mistakes**.
When developers write code, they make **mistakes**.

When developers design hardware, they make **mistakes**.
When developers write code, they make mistakes.

When developers design hardware, they make mistakes.

And when they write code for custom hardware, bugs come from the HW and SW, but also from the interface.
High-level languages and generators can help.

We're automatically building complete HW/SW stacks for image processing algorithms.

Come ask questions!
What we're doing...

Poster Title: Building a Component Library from Datasheets

### MAXIMUM RATING

<table>
<thead>
<tr>
<th>Rating</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collector - Emitter Voltage</td>
<td>$V_{CEO}$</td>
<td>65</td>
<td>Vdc</td>
</tr>
<tr>
<td>Collector - Base Voltage</td>
<td>$V_{CBO}$</td>
<td>80</td>
<td>Vdc</td>
</tr>
<tr>
<td>Emitter - Base Voltage</td>
<td>$V_{CEO}$</td>
<td>6.0</td>
<td>Vdc</td>
</tr>
<tr>
<td>Collector Current</td>
<td>$I_C$</td>
<td>100</td>
<td>mA</td>
</tr>
<tr>
<td>Total Device Dissipation @ $T_A = 25°C$</td>
<td>$P_D$</td>
<td>0.25</td>
<td>mW</td>
</tr>
<tr>
<td>Derate above $25°C$</td>
<td></td>
<td></td>
<td>mW/°C</td>
</tr>
<tr>
<td>Total Device Dissipation @ $T_D = 25°C$</td>
<td>$P_D$</td>
<td>1.5</td>
<td>W</td>
</tr>
<tr>
<td>Derate above $25°C$</td>
<td></td>
<td></td>
<td>W/°C</td>
</tr>
</tbody>
</table>

### THERMAL CHARACTERISTICS

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thermal Resistance, Junction-to-Ambient</td>
<td>$R_{JUA}$</td>
<td>200</td>
<td>°C/W</td>
</tr>
<tr>
<td>Thermal Resistance, Junction-to-Case</td>
<td>$R_{JUC}$</td>
<td>83.3</td>
<td>°C/W</td>
</tr>
</tbody>
</table>
Come chat with us if you're interested in...

- Embedded hardware design productivity
- New applications built on a detailed component library
- Extracting data from PDFs and tables
- Giving feedback