CryptOpt

Verified Compilation with Randomized Program Search for Cryptographic Primitives

Joel Kuepper, Andres Erbsen, Jason Gross, Owen Conoly, Chuyue Sun, Samuel Tian, David Wu, Adam Chlipala, Chitchanok Chuangsaitiansup, Daniel Genkin, Markus Wagner, Yuval Yarom
Challenges
Challenges

Cryptographic code must be **efficient** and **secure**.
Challenges

Cryptographic code must be efficient and secure.

Traditional approach:

**Hand-optimize the core and “Be Super Careful”.**
Challenges

Cryptographic code must be efficient and secure.
Traditional approach:
Hand-optimize the core and “Be Super Careful”.
Bad:
1. Labor-intensive work done by domain experts → Expensive $$$.
Challenges

Cryptographic code must be efficient and secure.

Traditional approach:
Hand-optimize the core and “Be Super Careful”.

Bad:
1. Labor-intensive work done by domain experts → Expensive $$$.
2. Error-prone
Challenges

Cryptographic code must be efficient and secure.

Traditional approach:
Hand-optimize the core and “Be Super Careful”.

Bad:
1. Labor-intensive work done by domain experts → Expensive $$$.
2. Error-prone

*The fact that these bugs existed in the first place shows that the traditional development methodology (i.e. “being super careful”) has failed.*

- Thomas Pornin, Sep. 2019 (in PQC Mailing List on a Falcon implementation)
Cryptographic Code
Cryptographic Code

![Graph showing the trade-off between Performance and Ease of Reusability in cryptographic code](image-url)
Cryptographic Code

- Performance vs. Ease of Reusability
- Generic C

CryptOpt: Verified Compilation with Randomized Program Search for Cryptographic Primitives
Joel Kuepper
RWC'23
Cryptographic Code

Performance

Ease of Reusability

Parameter-Specific C

Generic C

State of the Art

Method

Summary

Fiat Cryptography

CryptOpt

Joel Kuepper

RWC'23
Cryptographic Code

- CPU-Specific
- Parameter-Specific C
- Generic C
Cryptographic Code

- Hand-Written Assembly
- CPU-Specific
- Parameter-Specific C
- Generic C

Performance vs. Ease of Reusability

State of the Art

Method

Fiat Cryptography

Summary

CryptOpt: Verified Compilation with Randomized Program Search for Cryptographic Primitives

Joel Kuepper

RWC'23
**State of the Art**

**Method**

Fiat Cryptography

**Summary**

Cryptographic Code

- **Performance**
  - Hand-Written Assembly
  - CPU-Specific
  - Parameter-Specific C
  - Generic C

- **Ease of Reusability**

CryptOpt: Verified Compilation with Randomized Program Search for Cryptographic Primitives

Joel Kuepper

RWC'23
State of the Art

Method

Fiat Cryptography

Summary

Cryptographic Code

- **Hand-Written Assembly**
- **CPU-Specific**
- **Parameter-Specific C**
- **Generic C**

Performance vs. Ease of Reusability

CryptOpt: Verified Compilation with Randomized Program Search for Cryptographic Primitives

Joel Kuepper

RWC'23
Cryptographic Code

- Hand-Written Assembly
- CryptOpt
- CPU-Specific
- Parameter-Specific C
- Generic C

State of the Art

Method

Fiat Cryptography

Summary
Idea

Observations:

1. Compilers are general-purpose.
Observations:

1. Compilers are general-purpose.
2. Cryptographic code is “special”.
Idea

Observations:

1. Compilers are general-purpose.
2. Cryptographic code is “special simpler”.
Observations:
1. Compilers are general-purpose.
2. Cryptographic code is “special simpler”.

Idea:
1. Compiling to search for a fast implementation.
Idea

Observations:

1. Compilers are general-purpose.
2. Cryptographic code is “special simpler”.

   Idea:

   1. Compiling to search for a fast implementation.
2. Prove it correct.
Search for Fast Implementation
Search for Fast Implementation

Write Code
Search for Fast Implementation

- Write Code
- Modify Code
Search for Fast Implementation

Write Code

Modify Code

Faster?
Search for Fast Implementation

Write Code

Modify Code

Faster?

Yes
Search for Fast Implementation

- Write Code
- Modify Code
- Faster?
  - Yes
  - No
- Undo
Search for Fast Implementation

- Write Code
- Modify Code
- Faster?
  - Yes
  - No
- Undo

“Random Local Search”
Example: \((X + Y) \cdot Z + Z^2\)
Example: \[(X + Y) \cdot Z + Z^2\]
Example: \((X + Y) \cdot Z + Z^2\)
Example: \((X + Y) \cdot Z + Z^2\)
Example: \((X + Y) \cdot Z + Z^2\)
Example: \((X + Y) \cdot Z + Z^2\)
Example: $(X + Y) \cdot Z + Z^2$
Example: \((X + Y) \cdot Z + Z^2\)

Write Code → Modify Code → Faster? → Yes → Faster? → Yes → Undo

Yes → Faster? → No → Modify Code
Example: \((X + Y) \cdot Z + Z^2\)
Example: \((X + Y) \cdot Z + Z^2\)

```
X
  \rightarrow ADD
    \rightarrow adcx
    \rightarrow Mul
      \rightarrow mulx
      \rightarrow Mul
        \rightarrow mulx
        \rightarrow ADD
          \rightarrow add

Y
  \rightarrow ADD
    \rightarrow adcx
    \rightarrow Mul
      \rightarrow mulx
      \rightarrow Mul
        \rightarrow mulx
        \rightarrow ADD
          \rightarrow add

Z
  \rightarrow Mul
    \rightarrow Mul
      \rightarrow Mul
        \rightarrow Mul
          \rightarrow Mul
            \rightarrow Mul
              \rightarrow Mul
                \rightarrow Mul
                  \rightarrow Mul
                    \rightarrow Mul
                      \rightarrow Mul
                        \rightarrow Mul
                          \rightarrow Mul
                            \rightarrow Mul
                              \rightarrow Mul
                                \rightarrow Mul
                                  \rightarrow Mul
                                    \rightarrow Mul
                                      \rightarrow Mul
                                        \rightarrow Mul
                                          \rightarrow Mul
                                            \rightarrow Mul
                                              \rightarrow Mul
                                                \rightarrow Mul
                                                  \rightarrow Mul
                                                    \rightarrow Mul
                                                      \rightarrow Mul
                                                        \rightarrow Mul
                                                          \rightarrow Mul
                                                            \rightarrow Mul
                                                              \rightarrow Mul
                                                                \rightarrow Mul
                                                                  \rightarrow Mul
                                                                    \rightarrow Mul
                                                                      \rightarrow Mul
                                                                        \rightarrow Mul
                                                                             \rightarrow Mul
                                                                                 \rightarrow Mul
                                                                                     \rightarrow Mul
                                                                                       \rightarrow Mul
                                                                                         \rightarrow Mul
                                                                                           \rightarrow Mul
                                                                                             \rightarrow Mul
                                                                                               \rightarrow Mul
                                                                                                 \rightarrow Mul
                                                                                                   \rightarrow Mul
                                                                                                     \rightarrow Mul
                                                                                                       \rightarrow Mul
                                                                                                         \rightarrow Mul
                                                                                                          \rightarrow Mul
                                                                                                            \rightarrow Mul
                                                                                                             \rightarrow Mul
                                                                                                              \rightarrow Mul
                                                                                                               \rightarrow Mul
                                                                                                                \rightarrow Mul
                                                                                                                  \rightarrow Mul
                                                                                                                    \rightarrow Mul
                                                                                                                     \rightarrow Mul
                                                                                                                      \rightarrow Mul
                                                                                                                        \rightarrow Mul
                                                                                                                         \rightarrow Mul
                                                                                                                          \rightarrow Mul
                                                                                                                           \rightarrow Mul
                                                                                                                            \rightarrow Mul
                                                                                                                             \rightarrow Mul
                                                                                                                               \rightarrow Mul
                                                                                                                                \rightarrow Mul
                                                                                                                                   \rightarrow Mul
                                                                                                                                    \rightarrow Mul
                                                                                                                                     \rightarrow Mul
                                                                                                                                      \rightarrow Mul
                                                                                                                                       \rightarrow Mul
                                                                                                                                        \rightarrow Mul
                                                                                                                                             \rightarrow Mul
                                                                                                                                                \rightarrow Mul
                                                                                                                                                  \rightarrow Mul
                                                                                                                                                    \rightarrow Mul
                                                                                                                                                       \rightarrow Mul
                                                                                                                                                    \rightarrow Mul
                                                                                                                                                       \rightarrow Mul
                                                                                                                                                    \rightarrow Mul
                                                                                                                                                 \rightarrow Mul
                                                                                                                                                 \rightarrow Mul
                                                                                                                                                \rightarrow Mul
                                                                 mktime
```
Example: \((X + Y) \cdot Z + Z^2\)

```
  mov rax, [X]
  clc
  adcx rax, [Y]
  mov rdx, [Z]
  mulx r8, r9, rax
  mulx r10, r11, [Z]
  add r11, r9
  mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
clc
adcx rax, [Y]
mov rdx, [Z]
mulx r8, r9, rax
mulx r10, r11, [Z]
add r11, r9
mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
clc
adc x rax, [Y]

mov rdx, [Z]
mul x r8, r9, rax
mul x r10, r11, [Z]
add r11, r9
mov [out], r11
```
State of the Art

Fiat Cryptography

Summary

Example: \((X + Y) \cdot Z + Z^2\)

- Write Code
- Modify Code
- Faster?
  - Yes
  - No
  - Undo

```
mov rax, [X]
clc
adcx rax, [Y]
mov rdx, [Z]
mulx r8, r9, rax
mulx r10, r11, [Z]
add r11, r9
mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
clc
adcx rax, [Y]

mov rdx, [Z]
mulx r8, r9, rax

mulx r10, r11, [Z]

add r11, r9
mov [out], r11
```
Example: $(X + Y) \cdot Z + Z^2$

```plaintext
mov rax, [X]
clc
adcx rax, [Y]
mov rdx, [Z]
mulx r8, r9, rax
mulx r10, r11, [Z]
add r11, r9
mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
adcx 
clc
adcx rax, [Y]

mov rdx, [Z]

mulx r10, r11, [Z]

mulx r8, r9, rax

add r11, r9
mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

- Write Code
- Modify Code
- Faster? Yes
- Faster? No
- Undo

```
mov rax, [X]
clc
adcx rax, [Y]
mov rdx, [Z]
mulx r8, r9, rax
mulx r10, r11, [Z]
mux r8, r9, rax
add r11, r9
mov [out], r11
```
**Example:** $(X + Y) \cdot Z + Z^2$

```
write code
modify code
```

```
reorder
template change
```

```
Faster?
Yes
No
undo
```

```
X
Y
Z
```

```
mov rax, [X]
clc
adcx rax, [Y]
```

```
mov rdx, [Z]
mulx r10, r11, [Z]
```

```
mulx r8, r9, rax
```

```
add r11, r9
mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
clc
adcx rax, [Y]
mov rdx, [Z]
mulx r8, r9, rax
mov rdx, [Z]
mulx r10, r11, [Z]
add r11, r9
mov [out], r11
```
Example: \((X + Y) \cdot Z + Z^2\)

\[
\begin{align*}
\text{mov } & \text{rax, [X]} \\
\text{clc} & \\
\text{adcx } & \text{rax, [Y]} \\
\text{mov } & \text{rdx, [Z]} \\
\text{mulx } & \text{r8, r9, rax} \\
\text{mulx } & \text{r10, r11, [Z]} \\
\text{add } & \text{r11, r9} \\
\text{mov } & \text{[out], r11}
\end{align*}
\]
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
adc
dax, [Y]

mov rdx, [Z]
mulx r8, r9, rax
mulx r10, r11, [Z]
add r11, r9
mov [out], r11
```

```
X
Y
Z
```

Write Code

Modify Code

Faster?

Yes

No

Undo

Faster?
Example: \((X + Y) \cdot Z + Z^2\)

```
mov rax, [X]
add rax, [Y]
mov rdx, [Z]
mulx r8, r9, rax
mulx r10, r11, [Z]
add r11, r9
mov [out], r11
```
Performance

![Graph showing speedup vs. number of modifications](image-url)

**Method**
- Fiat Cryptography

**Summary**
- State of the Art
Fiat Cryptography

Fiat Cryptography
Fiat Cryptography

Functional Program → Fiat Cryptography
Fiat Cryptography

Functional Program → Fiat Cryptography

Write Code

Modify Code

Faster?

Yes

No

Undo
Fiat Cryptography


Write Code

Modify Code

Faster?

Yes

No

Undo
Fiat Cryptography


- Write Code
- Modify Code

Faster? △

- Yes
- No

Undo

Assembly
### Performance: Field Arithmetic

**Geometric Mean (4x AMD, 6x Intel)**

<table>
<thead>
<tr>
<th>Curve</th>
<th>Multiply</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Clang</td>
<td>GCC</td>
<td>Clang</td>
</tr>
<tr>
<td>Curve25519</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-224</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-256</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-384</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIKEp434</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Curve448</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-521</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poly1305</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>secp256k1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Performance: Field Arithmetic

### Geometric Mean (4x AMD, 6x Intel)

<table>
<thead>
<tr>
<th>Curve</th>
<th>Multiply Clang</th>
<th>Multiply GCC</th>
<th>Square Clang</th>
<th>Square GCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Curve25519</td>
<td>1.19</td>
<td>1.14</td>
<td>1.14</td>
<td>1.18</td>
</tr>
<tr>
<td>P-224</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-256</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-384</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIKEp434</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Curve448</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-521</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poly1305</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>secp256k1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Performance: Field Arithmetic

**Geometric Mean (4x AMD, 6x Intel)**

<table>
<thead>
<tr>
<th>Curve</th>
<th>Multiply</th>
<th>Square</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Clang</td>
<td>GCC</td>
</tr>
<tr>
<td>Curve25519</td>
<td>1.19</td>
<td>1.14</td>
</tr>
<tr>
<td>P-224</td>
<td>1.31</td>
<td>1.87</td>
</tr>
<tr>
<td>P-256</td>
<td>1.27</td>
<td>1.79</td>
</tr>
<tr>
<td>P-384</td>
<td>1.12</td>
<td>1.66</td>
</tr>
<tr>
<td>SIKEp434</td>
<td>1.30</td>
<td>1.70</td>
</tr>
<tr>
<td>Curve448</td>
<td>1.02</td>
<td>0.95</td>
</tr>
<tr>
<td>P-521</td>
<td>1.20</td>
<td>1.06</td>
</tr>
<tr>
<td>Poly1305</td>
<td>1.10</td>
<td>1.15</td>
</tr>
<tr>
<td>secp256k1</td>
<td>1.34</td>
<td>1.73</td>
</tr>
</tbody>
</table>
Performance: Scalar Multiplication
Performance: Scalar Multiplication

Geometric Mean (4x AMD, 6x Intel)
Performance: Scalar Multiplication

Intel 12th Generation

Cycles

<table>
<thead>
<tr>
<th>Method</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>HACL* fe-64</td>
<td>367k</td>
</tr>
<tr>
<td>sandy2x</td>
<td>480k</td>
</tr>
<tr>
<td>amd64-64</td>
<td>452k</td>
</tr>
<tr>
<td>amd64-51</td>
<td>420k</td>
</tr>
<tr>
<td>donna-c64</td>
<td>453k</td>
</tr>
<tr>
<td>OpenSSL</td>
<td>420k</td>
</tr>
<tr>
<td>OpenSSL fe-51</td>
<td>419k</td>
</tr>
<tr>
<td>OpenSSL fe-51</td>
<td>397k</td>
</tr>
<tr>
<td>OpenSSL fe-64</td>
<td>411k</td>
</tr>
<tr>
<td>OpenSSL fe-64</td>
<td>387k</td>
</tr>
</tbody>
</table>
Performance: Scalar Multiplication

Intel 13\textsuperscript{th} Generation

- HACL* fe-64: 366k cycles
- sandy2x: 480k cycles
- amd64-64: 452k cycles
- amd64-51: 421k cycles
- donna-c64: 451k cycles
- OpenSSL fe-51: 420k cycles
- OpenSSL fe-51: 398k cycles
- OpenSSL fe-64: 412k cycles
- OpenSSL fe-64: 392k cycles
Correctness

Fiat Cryptography


Write Code

Modify Code

Faster?

Assembly

Yes → No → Undo

Yes
Correctness

Functional Program → Fiat Cryptography

Faster? → Yes
Modify Code
Fiat IR → Checker
Write Code

Assembly

No → Undo
Correctness


Fiat IR → Checker

Fiat IR → Write Code

Modify Code

Faster? → Yes

No → Undo

Assembly
Correctness


- Yes
- No → Undo

Assembly
Correctness


Fiat IR → Checker

Faster?

Yes → Modify Code → Fiat IR

No → Undo

Assembly

Write Code
Summary

Compilation of cryptographic code = Search

Random Local Search + Runtime

Proven-correct assembly for field arithmetic by Fiat Cryptography now with on-par performance to hand-optimized assembly.

GitHub Project: https://0xade1a1de.github.io/CryptOpt
Compilation of cryptographic code
⇒ Search
Compilation of cryptographic code
⇒ Search
Random Local Search + Runtime
Compilation of cryptographic code

⇒ Search

Random Local Search + Runtime

**Proven-correct** assembly for field arithmetic by Fiat Cryptography now with *on-par performance* to hand-optimized assembly.
Compilation of cryptographic code

⇒ Search

Random Local Search + Runtime

Proven-correct assembly for field arithmetic by Fiat Cryptography now with on-par performance to hand-optimized assembly.
Summary

Compilation of cryptographic code
⇒ Search
Random Local Search + Runtime
Proven-correct assembly for field arithmetic by Fiat Cryptography now with on-par performance to hand-optimized assembly.

GitHub Project

https://0xade1a1de.github.io/CryptOpt
Bet and Run

![Graph showing speedup vs. number of modifications.](image)