High-Performance Cryptology on GPUs

Joppe W. Bos

Microsoft Research, Redmond

GTC 2013 – Session S3018

Joint work with the Laboratory for Cryptologic Algorithms, EPFL
Cryptology

Cryptography
- “secure communication in the presence of third parties”

Cryptanalysis
- “obtaining the original meaning of encrypted data without using the corresponding secret material”
Cryptology

Cryptography
“secure communication in the presence of third parties”

Cryptanalysis
“obtaining the original meaning of encrypted data without using the corresponding secret material”

Three main areas
- Public-key cryptography: e.g. RSA, (EC)DSA, (EC)DH
- Symmetric cryptography: e.g. AES
- Cryptographic hash functions: e.g. SHA-256, SHA-512, SHA-3
Motivation

Can we use the parallel compute power of GPUs to

- enhance the performance of cryptographic primitives
  - high-throughput
  - low-latency
- speed-up the security assessment of these cryptographic primitives
Motivation

Can we use the parallel compute power of GPUs to

- enhance the performance of cryptographic primitives
  - high-throughput
  - low-latency
- speed-up the security assessment of these cryptographic primitives

We have done similar experiments before...

High-Throughput Hashing

- cloud computing
- high-end servers
- distributed databases
High-Throughput Hashing

- cloud computing
- high-end servers
- distributed databases

SHA-512: random message ranging from 32KB and 128KB

**CPU: Intel Core i7-3520M 2.9 GHz, 2 cores**

9.37 cycles / byte $\rightarrow$ 295 MB / second / core

---


High-Throughput Hashing

- cloud computing
- high-end servers
- distributed databases

SHA-512: random message ranging from 32KB and 128KB

**CPU: Intel Core i7-3520M 2.9 GHz, 2 cores**

9.37 cycles / byte $\rightarrow$ 295 MB / second / core

**NVIDIA GeForce GTX 590, 1.215 GHz (2$\times$ GF110)**

<table>
<thead>
<tr>
<th>Batch size</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput (MB / second / GF110)</td>
<td>670</td>
<td>1100</td>
<td>1650</td>
<td>2100</td>
</tr>
<tr>
<td>Speedup</td>
<td>2.27</td>
<td>3.73</td>
<td>5.59</td>
<td>7.12</td>
</tr>
</tbody>
</table>


Let $p > 3$ be a prime, then any $a, b \in \mathbb{F}_p$ such that $4a^3 + 27b^2 \neq 0$ define an elliptic curve $E_{a,b}$ over $\mathbb{F}_p$. The zero point $\mathbf{0}$, together with the set of points $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ which satisfy the short affine Weierstrass equation

$$y^2 = x^3 + ax + b,$$

form an abelian group $E_{a,b}(\mathbb{F}_p)$. 

Standards (NIST)
- ECDSA as standardized in FIPS 186-3: Digital Signature Standard (DSS)
- 128-bit security level corresponds to 256-bit ECC keys
- 3072-bit RSA keys

ECC is an order of magnitude faster [NSA] for 128-bit security
Public-Key Cryptosystems based on elliptic curves

Elliptic Curves over prime fields – Definition

Let $p > 3$ be a prime, then any $a, b \in \mathbb{F}_p$ such that $4a^3 + 27b^2 \neq 0$ define an elliptic curve $E_{a,b}$ over $\mathbb{F}_p$. The zero point $\mathbf{0}$, together with the set of points $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ which satisfy the short affine Weierstrass equation

$$y^2 = x^3 + ax + b,$$

form an abelian group $E_{a,b}(\mathbb{F}_p)$.

Standards (NIST)

ECDSA as standardized in FIPS 186-3: Digital Signature Standard (DSS)

- 128-bit security level corresponds to 256-bit ECC keys
- 3072-bit RSA keys

ECC is an order of magnitude faster [NSA] for 128-bit security
Cryptanalysis

The Certicom ECC Challenge

“to increase industry understanding and appreciation for the difficulty of the elliptic curve discrete logarithm problem”

ECC2K-130 challenge is over $E(\mathbb{F}_{2^{131}})$

Cryptanalysis

The Certicom ECC Challenge

“to increase industry understanding and appreciation for the difficulty of the elliptic curve discrete logarithm problem”

ECC2K-130 challenge is over $E(\mathbb{F}_{2^{131}})$

Cost to solve the ECC2K-130 on different platforms

- FPGA (XC3S5000, 111 MHz): $\approx 610$ year
- GTX 295: $\approx 1070$ year
- PlayStation 3: $\approx 2650$ year
- Core-2 Q6850 (4 cores): $\approx 3040$ year

The Certicom ECC Challenge

“to increase industry understanding and appreciation for the difficulty of the elliptic curve discrete logarithm problem”

ECC2K-130 challenge is over $E(\mathbb{F}_{2^{131}})$

Cost to solve the ECC2K-130 on different platforms

- FPGA (XC3S5000, 111 MHz): $\approx 610$ year
- GTX 295: $\approx 1070$ year
- PlayStation 3: $\approx 2650$ year
- Core-2 Q6850 (4 cores): $\approx 3040$ year

256-bit keys are roughly $10^{19}$ times as difficult to break

High-throughput is often not the most important factor
Low-latency is often much more valuable
High-throughput is often not the most important factor
Low-latency is often much more valuable

**Option 1** Parallel $F_p$ arithmetic ($p$ prime)

- Try and implement a multi-core version of modular multiplication using a residue number system
- One of the few techniques to speed-up RSA on many-core platforms
Cryptography, NIST-p224, 112-bit security

High-throughput is often not the most important factor
Low-latency is often much more valuable

**Option 1 Parallel $\mathbb{F}_p$ arithmetic ($p$ prime)**

- Try and implement a multi-core version of modular multiplication using a residue number system
- One of the few techniques to speed-up RSA on many-core platforms

**Option 2 Parallel EC-arithmetic**

Idea: for ECC we have more freedom

- Compute the $\mathbb{F}_p$ arithmetic per thread for throughput
- Compute the EC-arithmetic in parallel

Use the Montgomery-ladder.
Low-latency for ECC

Cost per bit for scalar multiplication using $E(F_{p^{224}})$:

<table>
<thead>
<tr>
<th>Approach</th>
<th>#mul in $F_{p^{224}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>State-of-the-art</td>
<td>$\approx 8 - 10$</td>
</tr>
</tbody>
</table>

Low-latency for ECC

\[
(P + Q, 2Q) = (\tilde{P}, \tilde{Q}) = ((\tilde{P}_x, \tilde{P}_z), (\tilde{Q}_x, \tilde{Q}_z)) = \\
\begin{aligned}
\tilde{P}_x &= 2(P_x Q_z + Q_x P_z)(P_x Q_x + aP_z Q_z) \\
&\quad + 4bP_z^2 Q_z^2 - G_x(P_x Q_z - Q_x P_z)^2 \\
\tilde{P}_z &= (P_x Q_z - Q_x P_z)^2 \\
\tilde{Q}_x &= (Q_x^2 - aQ_z^2)^2 - 8bQ_x Q_z^3 \\
\tilde{Q}_z &= 4(Q_x Q_z(Q_x^2 + aQ_z^2) + bQ_z^4)
\end{aligned}
\]

Cost per bit for scalar multiplication using \( E(F_{p224}) \):

<table>
<thead>
<tr>
<th>Approach</th>
<th>#mul in ( F_{p224} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>State-of-the-art</td>
<td>( \approx 8 - 10 )</td>
</tr>
<tr>
<td>Montgomery ladder</td>
<td>18</td>
</tr>
</tbody>
</table>

Low-latency for ECC

\[(P + Q, 2Q) = (\tilde{P}, \tilde{Q}) = ((\tilde{P}_x, \tilde{P}_z), (\tilde{Q}_x, \tilde{Q}_z)) = \]
\[
\begin{align*}
\tilde{P}_x &= 2(P_x Q_z + Q_x P_z)(P_x Q_x + a P_z Q_z) \\
&\quad + 4 b P_z^2 Q_z^2 - G_x (P_x Q_z - Q_x P_z)^2 \\
\tilde{P}_z &= (P_x Q_z - Q_x P_z)^2 \\
\tilde{Q}_x &= (Q_x^2 - a Q_z^2)^2 - 8 b Q_x Q_z^3 \\
\tilde{Q}_z &= 4(Q_x Q_z (Q_x^2 + a Q_z^2) + b Q_z^4)
\end{align*}
\]

Cost per bit for scalar multiplication using \(E(\mathbb{F}_{p^{224}})\):

<table>
<thead>
<tr>
<th>Approach</th>
<th>#mul in (\mathbb{F}_{p^{224}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>State-of-the-art</td>
<td>(\approx 8 - 10)</td>
</tr>
<tr>
<td>Montgomery ladder</td>
<td>18</td>
</tr>
<tr>
<td>GPU, using 7 threads</td>
<td>3</td>
</tr>
</tbody>
</table>

- **Advantage**: latency is reduced by a factor 3
- **Disadvantage**: Use 7 threads, per warp 4 threads are idle

<table>
<thead>
<tr>
<th>Ref</th>
<th>Platform</th>
<th>cores</th>
<th>MHz</th>
<th>Min. L [ms]</th>
<th>Max. T [op/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[7]</td>
<td>8800 GTS (1)</td>
<td>96</td>
<td>1200</td>
<td>305.0</td>
<td>1413</td>
</tr>
<tr>
<td>[7]</td>
<td>8800 GTS (1)</td>
<td>96</td>
<td>1200</td>
<td>30.3</td>
<td>3138</td>
</tr>
<tr>
<td>[7]</td>
<td>GTX 285 (1)</td>
<td>240</td>
<td>1476</td>
<td>24.3</td>
<td>9990</td>
</tr>
<tr>
<td>[7]</td>
<td>GTX 295 (2)</td>
<td>240</td>
<td>1242</td>
<td>10.6</td>
<td>79,198</td>
</tr>
<tr>
<td>New</td>
<td>GTX 480 (1)</td>
<td>480</td>
<td>1401</td>
<td>2.3</td>
<td>237,415</td>
</tr>
<tr>
<td>New</td>
<td>GTX 580 (1)</td>
<td>512</td>
<td>1544</td>
<td>1.9</td>
<td>290,535</td>
</tr>
<tr>
<td>[9]</td>
<td>Intel core-i7 2600K</td>
<td>4</td>
<td>3400</td>
<td>0.09</td>
<td>46,176</td>
</tr>
</tbody>
</table>

### Results

#### GTX 295 (single GT200) vs GTX 285

- Latency reduced by a factor 2.3
- Throughput increased by a factor 7.9

<table>
<thead>
<tr>
<th>Ref</th>
<th>Platform</th>
<th>cores / GPU</th>
<th>MHz</th>
<th>Min. L [ms]</th>
<th>Max. T [op/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[7]</td>
<td>8800 GTS (1)</td>
<td>96</td>
<td>1200</td>
<td>305.0</td>
<td>1413</td>
</tr>
<tr>
<td>[8]</td>
<td>8800 GTS (1)</td>
<td>96</td>
<td>1200</td>
<td>30.3</td>
<td>3138</td>
</tr>
<tr>
<td></td>
<td>GTX 285 (1)</td>
<td>240</td>
<td>1476</td>
<td>24.3</td>
<td>9990</td>
</tr>
<tr>
<td></td>
<td>GTX 295 (2)</td>
<td>240</td>
<td>1242</td>
<td>10.6</td>
<td>79,198</td>
</tr>
<tr>
<td>New</td>
<td>GTX 480 (1)</td>
<td>480</td>
<td>1401</td>
<td>2.3</td>
<td>237,415</td>
</tr>
<tr>
<td></td>
<td>GTX 580 (1)</td>
<td>512</td>
<td>1544</td>
<td>1.9</td>
<td>290,535</td>
</tr>
<tr>
<td>[9]</td>
<td>Intel core-i7 2600K</td>
<td>4</td>
<td>3400</td>
<td>0.09</td>
<td>46,176</td>
</tr>
</tbody>
</table>


## Results

### GTX 580 vs Intel core-i7
- CPU stills wins by a factor 21, 1.9 ms is acceptable in many scenarios
- Throughput increased by a factor 6.3

<table>
<thead>
<tr>
<th>Ref</th>
<th>Platform</th>
<th>cores / GPU</th>
<th>MHz</th>
<th>Min. L [ms]</th>
<th>Max. T [op/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[7]</td>
<td>8800 GTS (1)</td>
<td>96</td>
<td>1200</td>
<td>305.0</td>
<td>1 413</td>
</tr>
<tr>
<td>[8]</td>
<td>8800 GTS (1)</td>
<td>96</td>
<td>1200</td>
<td>30.3</td>
<td>3 138</td>
</tr>
<tr>
<td></td>
<td>GTX 285 (1)</td>
<td>240</td>
<td>1476</td>
<td>24.3</td>
<td>9 990</td>
</tr>
<tr>
<td></td>
<td>GTX 295 (2)</td>
<td>240</td>
<td>1242</td>
<td>10.6</td>
<td>79,198</td>
</tr>
<tr>
<td>New</td>
<td>GTX 480 (1)</td>
<td>480</td>
<td>1401</td>
<td>2.3</td>
<td>237 415</td>
</tr>
<tr>
<td></td>
<td><strong>GTX 580</strong> (1)</td>
<td><strong>512</strong></td>
<td><strong>1544</strong></td>
<td><strong>1.9</strong></td>
<td><strong>290 535</strong></td>
</tr>
<tr>
<td>[9]</td>
<td>Intel core-i7 2600K</td>
<td>4</td>
<td>3400</td>
<td>0.09</td>
<td>46 176</td>
</tr>
</tbody>
</table>

Conclusions

- GPUs are useful as a cryptographic accelerator
- High-throughput is easy, low-latency is a challenge
- Faster (parallel) arithmetic → faster cryptanalysis: security implications

Future work on GPUs

- Optimize integer factoring using GPUs (implications for RSA)
  
  J. W. Bos, T. Kleinjung: ECM at Work. in Asiacrypt 2012

- Study the security of elliptic curve based schemes in more detail

- Rethink arithmetic building blocks: faster cryptography
  - Faster parallel algorithms
  - Minimize thread-communication
  - Minimize memory-per-thread