S-unit attacks

Tanja Lange
(with lots of slides from Daniel J. Bernstein)

Eindhoven University of Technology

24 November 2022
KpqC
Post-quantum cryptography

Cryptography under the assumption that the attacker has a quantum computer.

- 2015: NIST hosts its first workshop on post-quantum cryptography.
- 2016: NIST announces a standardization project for post-quantum systems.
- 2017: Deadline for submissions to the NIST competition.
- 2019: Second round of NIST competition begins.
- 2020: Third round of NIST competition begins.
- 2021–2022 “not later than the end of March”:
Post-quantum cryptography

Cryptography under the assumption that the attacker has a quantum computer.

- 2015: NIST hosts its first workshop on post-quantum cryptography.
- 2016: NIST announces a standardization project for post-quantum systems.
- 2017: Deadline for submissions to the NIST competition.
- 2019: Second round of NIST competition begins.
- 2020: Third round of NIST competition begins.
- 2021-2022 “not later than the end of March”: 05 Jul NIST announces first selections.
- 2022 \(\rightarrow\infty\) NIST studies further systems.
- 2023/2024?: NIST issues post-quantum standards.
Major categories of public-key post-quantum systems


- **Hash-based** signatures: very solid security and small public keys. Require only a secure hash function (hard to find second preimages).

- **Isogeny-based** encryption: new kid on the block, promising short keys and ciphertexts and non-interactive key exchange. Security relies on hardness of finding isogenies between elliptic curves over finite fields.

- **Lattice-based** encryption and signatures: possibility for balanced sizes. Security relies on hardness of finding short vectors in some (typically special) lattice.

- **Multivariate-quadratic** signatures: short signatures and large public keys. Security relies on hardness of solving systems of multivariate equations over finite fields.

Warning: These are categories of mathematical problems; individual systems may be totally insecure if the problem is not used correctly.

We have a good algorithmic abstraction of what a quantum computer can do, but new systems need more analysis. Any extra structure offers more attack surface.
NIST’s 5 July announcement

The winners:

- Kyber, a KEM based on structured lattices
- Dilithium, a signature scheme based on structured lattices
- Falcon, a signature scheme based on structured lattices
- SPHINCS+, a signature scheme based on hash functions

This is an odd choice, given that KEMs are most urgently needed to ensure long-term confidentiality.

Schemes advancing to round 4, so maybe more winners later:

- BIKE, a KEM based on codes
- Classic McEliece, a KEM based on codes
- HQC, a KEM based on codes
- SIKE, a KEM based on isogenies (now really badly broken, <1 month after NIST’s announcement)

Tanja Lange S-unit attacks
NIST’s 5 July announcement

The winners:

- Kyber, a KEM based on structured lattices
- Dilithium, a signature scheme based on structured lattices
- Falcon, a signature scheme based on structured lattices
- SPHINCS+, a signature scheme based on hash functions

This is an odd choice, given that KEMs are most urgently needed to ensure long-term confidentiality.

Schemes advancing to round 4, so maybe more winners later:

- BIKE, a KEM based on codes
- Classic McEliece, a KEM based on codes
- HQC, a KEM based on codes
- SIKE, a KEM based on isogenies (now really badly broken, <1 month after NIST's announcement)
NIST’s 5 July announcement

The winners:

• Kyber, a KEM based on structured lattices
• Dilithium, a signature scheme based on structured lattices
• Falcon, a signature scheme based on structured lattices
• SPHINCS+, a signature scheme based on hash functions

This is an odd choice, given that KEMs are most urgently needed to ensure long-term confidentiality.
NIST’s 5 July announcement

The winners:

- Kyber, a KEM based on structured lattices
- Dilithium, a signature scheme based on structured lattices
- Falcon, a signature scheme based on structured lattices
- SPHINCS+, a signature scheme based on hash functions

This is an odd choice, given that KEMs are most urgently needed to ensure long-term confidentiality.

Schemes advancing to round 4, so maybe more winners later:

- BIKE, a KEM based on codes
- Classic McEliece, a KEM based on codes
- HQC, a KEM based on codes
- SIKE, a KEM based on isogenies (now really badly broken, < 1 month after NIST’s announcement)
1998 (ANTS-III) Hoffstein, Pipher, and Silverman introduce NTRU, working in ring $\mathbb{Z}[x]/(x^m - 1)$ (modulo $q$ and modulo 3)
Lattice-based cryptography

1998 (ANTS-III) Hoffstein, Pipher, and Silverman introduce NTRU, working in ring $\mathbb{Z}[x]/(x^m - 1)$ (modulo $q$ and modulo 3)

2010 Lyubashevsky, Peikert, and Regev “introduce” Ring-LWE and prove “very strong hardness guarantees”

**Assume** “worst-case problems on ideal lattices are hard for polynomial-time quantum algorithms”

“the ring-LWE distribution is pseudorandom”

security for a “truly practical lattice-based public-key cryptosystem”

Concrete parameters in cryptosystems are chosen assuming much more than polynomial hardness.
Typical structured lattices

NTRU uses $\mathbb{Z}[x]/(x^m - 1)$ for prime $m$. 
Typical structured lattices

NTRU uses $\mathbb{Z}[x]/(x^m - 1)$ for prime $m$.

The winners all use 2-power cyclotomics:
Define $R = \mathbb{Z}[x]/(x^n + 1)$ for some $n \in \{2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, \ldots \}$. From now on consider this case.

Ideal-SVP
Given a nonzero ideal $I \subseteq R$, find a “short” nonzero element $g \in I$.

Ideal $I$ is given by basis $v_1, v_2, \ldots, v_n \in R$ such that $I = \mathbb{Z}v_1 + \mathbb{Z}v_2 + \cdots + \mathbb{Z}v_n$. 

Tanja Lange
S-unit attacks
Typical structured lattices

NTRU uses $\mathbb{Z}[x]/(x^m - 1)$ for prime $m$.

The winners all use 2-power cyclotomics:
Define $R = \mathbb{Z}[x]/(x^n + 1)$ for some $n \in \{2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, \ldots\}$. From now on consider this case.

Ideal-SVP
Given a nonzero ideal $I \subseteq R$, find a “short” nonzero element $g \in I$.

Ideal $I$ is given by basis $v_1, v_2, \ldots, v_n \in R$ such that $I = \mathbb{Z}v_1 + \mathbb{Z}v_2 + \cdots + \mathbb{Z}v_n$.

E.g. for $n = 4$

$v_1 = x^3 + 817$  \quad $\rightarrow$  \quad $g = 2v_1 + 3v_2 - 5v_3 - 2v_4$
$v_2 = x^2 + 540$  \quad \text{this needs work}  \quad \quad \quad \quad \quad = 2x^3 + 3x^2 - 5x + 1$
$v_3 = x + 247$
$v_4 = 1009$
Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
817 & 0 & 0 & 1 \\
540 & 0 & 1 & 0 \\
247 & 1 & 0 & 0 \\
1009 & 0 & 0 & 0 \\
\end{array}
\]

Last row matches the $g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1$ from above (up to sign).

But this doesn’t reach “short” when $n$ is large.
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{align*}
817 & \quad 0 & \quad 0 & \quad 1 \\
540 & \quad 0 & \quad 1 & \quad 0 \\
247 & \quad 1 & \quad 0 & \quad 0 \\
192 & \quad 0 & \quad 0 & \quad -1
\end{align*}
\]

Last row matches the 
\[g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1 \] from above (up to sign).

But this doesn't reach "short" when \(n\) is large.
Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>277</td>
<td>0</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>540</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>247</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>192</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>

Last row matches the $g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x + 1$ from above (up to sign).

But this doesn’t reach “short” when $n$ is large.
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>277</td>
<td>0</td>
<td>-1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>263</td>
<td>0</td>
<td>2</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>247</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>192</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
14 & 0 & -3 & 2 \\
263 & 0 & 2 & -1 \\
247 & 1 & 0 & 0 \\
192 & 0 & 0 & -1 \\
\end{array}
\]
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
14 & 0 & -3 & 2 \\
16 & -1 & 2 & -1 \\
247 & 1 & 0 & 0 \\
192 & 0 & 0 & -1 \\
\end{array}
\]

Last row matches the \( g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 - 1 \) from above (up to sign).

But this doesn’t reach “short” when \( n \) is large.
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>0</td>
<td>-3</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
</tr>
<tr>
<td>55</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>192</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{align*}
14 & \quad 0 & \quad -3 & \quad 2 \\
16 & \quad -1 & \quad 2 & \quad -1 \\
55 & \quad 1 & \quad 0 & \quad 1 \\
137 & \quad -1 & \quad 0 & \quad -2 \\
\end{align*}
\]
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>0</td>
<td>-3</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
</tr>
<tr>
<td>55</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>82</td>
<td>-2</td>
<td>0</td>
<td>-3</td>
</tr>
</tbody>
</table>

Last row matches the $g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1$ from above (up to sign).

But this doesn’t reach “short” when $n$ is large.
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
14 & 0 & -3 & 2 \\
16 & -1 & 2 & -1 \\
55 & 1 & 0 & 1 \\
27 & -3 & 0 & -4 \\
\end{array}
\]

Last row matches the \( g = 2 v_1 + 3 v_2 - 5 v_3 - 2 v_4 = 2 x_3 + 3 x_2 - 5 x_1 + 1 \) from above (up to sign).

But this doesn’t reach “short” when \( n \) is large.
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
14 & 0 & -3 & 2 \\
16 & -1 & 2 & -1 \\
28 & 4 & 0 & 5 \\
27 & -3 & 0 & -4 \\
\end{array}
\]

Last row matches the \( g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1 \) from above (up to sign).

But this doesn’t reach “short” when \( n \) is large.
**Naive lattice-basis reduction:** Reduce largest row
by subtracting closest multiple of another row.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>0</td>
<td>−3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>−1</td>
<td>2</td>
<td>−1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>0</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>−3</td>
<td>0</td>
<td>−4</td>
<td></td>
</tr>
</tbody>
</table>
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>0</td>
<td>-3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>0</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>-2</td>
<td>-2</td>
<td>-3</td>
<td></td>
</tr>
</tbody>
</table>

Last row matches the $g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1$ from above (up to sign).

But this doesn’t reach “short” when $n$ is large.
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>0</td>
<td>−3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>−1</td>
<td>5</td>
<td>−3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>0</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>−2</td>
<td>−2</td>
<td>−3</td>
<td></td>
</tr>
</tbody>
</table>
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
3 & 2 & -1 & 5 \\
2 & -1 & 5 & -3 \\
1 & 7 & 0 & 9 \\
11 & -2 & -2 & -3 \\
\end{array}
\]
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
3 & 2 & -1 & 5 \\
2 & -1 & 5 & -3 \\
1 & 7 & 0 & 9 \\
9 & -1 & -7 & 0
\end{array}
\]

Last row matches the 
\[ g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1 \text{ from above (up to sign).} \]

But this doesn’t reach “short” when \( n \) is large.
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{align*}
3 & \quad 2 & -1 & \quad 5 \\
2 & \quad -1 & \quad 5 & \quad -3 \\
-2 & \quad 5 & \quad 1 & \quad 4 \\
9 & \quad -1 & -7 & \quad 0
\end{align*}
\]

... Last row matches the
\[
g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1 \text{ from above (up to sign).}
\]

But this doesn’t reach “short” when \( n \) is large.

Tanja Lange
S-unit attacks
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>-1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>5</td>
<td>-3</td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-3</td>
<td>-6</td>
<td>-5</td>
<td></td>
</tr>
</tbody>
</table>
Doesn’t look so hard . . .

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

$$\begin{pmatrix} 3 & 2 & -1 & 5 \\ 2 & -1 & 5 & -3 \\ -2 & 5 & 1 & 4 \\ 4 & 2 & -5 & -1 \end{pmatrix}$$

Last row matches the $g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_3 + 1$ from above (up to sign).

But this doesn’t reach “short” when $n$ is large.

Tanja Lange S-unit attacks
Doesn’t look so hard …

Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
3 & 2 & -1 & 5 \\
2 & -1 & 5 & -3 \\
-5 & 3 & 2 & -1 \\
4 & 2 & -5 & -1 \\
\end{array}
\]
Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{ccccc}
3 & 2 & -1 & 5 \\
2 & -1 & 5 & -3 \\
-5 & 3 & 2 & -1 \\
-1 & 5 & -3 & -2 \\
\end{array}
\]

Last row matches the \( g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x_3 + 3x_2 - 5x_1 + 1 \) from above (up to sign).

But this doesn’t reach “short” when \( n \) is large.
Naive lattice-basis reduction: Reduce largest row by subtracting closest multiple of another row.

\[
\begin{array}{cccc}
3 & 2 & -1 & 5 \\
2 & -1 & 5 & -3 \\
-5 & 3 & 2 & -1 \\
-1 & 5 & -3 & -2
\end{array}
\]

Last row matches the \( g = 2v_1 + 3v_2 - 5v_3 - 2v_4 = 2x^3 + 3x^2 - 5x + 1 \) from above (up to sign).

But this doesn’t reach “short” when \( n \) is large.
Lower bound on shortest nonzero element

Let $K = \mathbb{Q}(\zeta_{2n})$ and let $\iota_1, \iota_3, \ldots, \iota_{n-1}, \iota_{-1}, \ldots, \iota_{-(n-1)}$ be the embeddings of $K$ into $\mathbb{C}$. For $z \in \mathbb{C}$ let $|z| = \sqrt{z \cdot \overline{z}}$.

Minkowski embedding:
Apply $\{\iota_1, \ldots, \iota_{n-1}, \iota_{-1}, \ldots, \iota_{-(n-1)}\}$ to the nonzero ideal $I \subseteq R = \mathbb{Z}[x]/(x^n + 1)$. Obtain an $n$-dim lattice of covolume $\sqrt{n^n} \cdot \#(R/I)$.

E.g., for $n = 4$ as above $1009 \mapsto (1009, 1009, 1009, 1009)$;
$x + 247 \mapsto (\zeta_8^1 + 247, \zeta_8^3 + 247, \zeta_8^{-3} + 247, \zeta_8^{-1} + 247)$;
$x^2 + 540 \mapsto (\zeta_8^2 + 540, \zeta_8^6 + 540, \zeta_8^{-6} + 540, \zeta_8^{-2} + 540)$;
$x^3 + 817 \mapsto (\zeta_8^3 + 817, \zeta_8^9 + 817, \zeta_8^{-9} + 817, \zeta_8^{-3} + 817)$;
$I \mapsto 4$-dim lattice of covolume $4^{4/2} \cdot 1009 \approx 11.27^4$;
Lower bound on shortest nonzero element

Let $K = \mathbb{Q}(\zeta_{2n})$ and let $\iota_1, \iota_3, \ldots, \iota_{n-1}, \iota_{-1}, \ldots, \iota_{-(n-1)}$ be the embeddings of $K$ into $\mathbb{C}$. For $z \in \mathbb{C}$ let $|z| = \sqrt{z \cdot \bar{z}}$.

Minkowski embedding:
Apply $\{\iota_1, \ldots, \iota_{n-1}, \iota_{-1}, \ldots, \iota_{-(n-1)}\}$ to the nonzero ideal $I \subseteq R = \mathbb{Z}[x]/(x^n + 1)$. Obtain an $n$-dim lattice of covolume $\sqrt{n^n \cdot \#(R/I)}$.

E.g., for $n = 4$ as above $1009 \mapsto (1009, 1009, 1009, 1009)$;
$x + 247 \mapsto (\zeta_8^1 + 247, \zeta_8^3 + 247, \zeta_8^{-3} + 247, \zeta_8^{-1} + 247)$;
$x^2 + 540 \mapsto (\zeta_8^2 + 540, \zeta_8^6 + 540, \zeta_8^{-6} + 540, \zeta_8^{-2} + 540)$;
$x^3 + 817 \mapsto (\zeta_8^3 + 817, \zeta_8^9 + 817, \zeta_8^{-9} + 817, \zeta_8^{-3} + 817)$;
$I \mapsto 4$-dim lattice of covolume $4^{4/2} \cdot 1009 \approx 11.27^4$;

Use this to bound length of $g \in I - \{0\}$ with $\prod_{\iota} |\iota(g)| = \#(R/g) \geq \#(R/I)$ so
$\|g\|_2 = \sqrt{\sum_{\iota} |\iota(g)|^2} \geq \sqrt{n(\prod_{\iota} |\iota(g)|)^{1/n}} \geq \sqrt{n \#(R/I)^{1/n}} = (\text{covol } I)^{1/n}$.

In our example $g = 2x^3 + 3x^2 - 5x + 1 \mapsto (2\zeta_8^3 + 3\zeta_8^2 - 5\zeta_8 + 1, 2\zeta_8^9 + 3\zeta_8^6 - 5\zeta_8^3 + 1, 2\zeta_8^{-3} + 3\zeta_8^{-2} - 5\zeta_8^{-1} + 1)$
$\|g\|_2 = \sqrt{4\sqrt{2^2 + 3^2 + 5^2 + 1}} \approx 12.49 > 11.27.$
Upper bound on shortest nonzero element

1889 Minkowski “geometry of numbers” implies

$$||g||_2 \leq 2(n/2)!^{1/n} \pi^{-1/2} (\text{covol } l)^{1/n}$$

for some $g \in I - \{0\}$, i.e., some nonzero $g \in I$ has

$$\eta = \frac{||g||_2}{(\text{covol } l)^{1/n}} \leq 2(n/2)!^{1/n} \pi^{-1/2},$$

where $\eta$ is called the “Hermite factor”.

E.g. $n = 4$: $\eta \leq 1.35$. $n = 512$: $\eta \leq 11.03$. Have $2(n/2)!^{1/n} \pi^{-1/2} \approx \sqrt{2n/e\pi}$ for large $n$.

This shows that very short elements exist.

But can we find them?
Performance of known algorithms

Algorithm input: nonzero ideal $I \subseteq R = \mathbb{Z}[x]/(x^n + 1)$.
Output: nonzero $g = g_0 + \cdots + g_{n-1}x^{n-1} \in I$ with $(g_0^2 + \cdots + g_{n-1}^2)^{1/2} = \eta \cdot (\#(R/I))^{1/n}$.

Algorithms using only additive structure of $I$:
- LLL (fast): $\eta^{1/n} \approx 1.022$.
- BKZ-80 (not hard): $\eta^{1/n} \approx 1.010$.
- BKZ-160 (public attack): $\eta^{1/n} \approx 1.007$.
- BKZ-300 (large-scale attack): $\eta^{1/n} \approx 1.005$.

BKZ-$\beta$ repeatedly computes a shortest basis in a lattice of dimension $\beta$.
Quality and cost increase with $\beta$.

These algorithms work for arbitrary lattices.
**Can we do better using ideal structure?**
Notation for infinite places of $K = \mathbb{Q}[x]/(x^n + 1)$

Define $\zeta_m = \exp(2\pi i / m) \in \mathbb{C}$ for nonzero $m \in \mathbb{Z}$.

For any $c \in 1 + 2\mathbb{Z}$ have $(\zeta_{2n}^c)^n + 1 = 0$ so there is a unique ring morphism $\iota_c : K \to \mathbb{C}$ taking $x$ to $\zeta_{2n}^c$.

All roots of $x^n + 1$ in $\mathbb{C}$: $\zeta_{2n}^1, \ldots, \zeta_{2n}^{n-1}, \zeta_{2n}^{-(n-1)}, \ldots, \zeta_{2n}^{-1}$.

All $\iota : K \to \mathbb{C}$: $\iota_1, \ldots, \iota_{n-1}, \iota_{-(n-1)}, \ldots, \iota_{-1}$.

Define $|g|_c = |\iota_c(g)|^2 = \iota_c(g)\iota_{-c}(g)$.

The maps $g \mapsto |g|_c$ are the infinite places of $K$.

All infinite places: $g \mapsto |g|_1, g \mapsto |g|_3, \ldots, g \mapsto |g|_{n-1}$.

Same as: $g \mapsto |g|_{-1}, g \mapsto |g|_{-3}, \ldots, g \mapsto |g|_{-n-1}$.

$$\sum_{c \in \{1, 3, \ldots, n-1\}} |g_0 + \cdots + g_{n-1}x^{n-1}|_c = \frac{n}{2} (g_0^2 + \cdots + g_{n-1}^2).$$
Notation for finite places of $K = \mathbb{Q}[x]/(x^n + 1)$

Nonzero ideals of $R$ factor into prime ideals.

For each nonzero prime ideal $P$ of $R$, define

$$|g|_P = #(R/P)^{-\text{ord}_P g}.$$

“Norm of $P$” is $#(R/P)$.
The maps $g \mapsto |g|_P$ are the \textbf{finite places} of $K$.

For each prime number $p$:
Factor $x^n + 1$ in $\mathbb{F}_p[x]$ to see the prime ideals of $R$ containing $p$.

E.g. $p = 2$: Prime ideal $2R + (x + 1)R = (x + 1)R$.

E.g. “unramified degree-1 primes”:
$p \in 1 + 2n\mathbb{Z} \Rightarrow$ exactly $n$ $n$th roots $r_1, \ldots, r_n$ of $-1$ in $\mathbb{F}_p$.
$x^n + 1 = (x - r_1)(x - r_2)\ldots(x - r_n)$ in $\mathbb{F}_p[x]$.
Prime ideals $pR + (x - r_1)R, \ldots, pR + (x - r_n)R$. 

Tanja Lange
S-unit attacks
Notation for places $g \mapsto \left|g\right|_v$ for, e.g., $n = 4$, $R = \mathbb{Z}[x]/(x^4 + 1)$

$$g = g_0 + g_1x + g_2x^2 + g_3x^3, \quad \zeta_8 = \exp(2\pi i/8):$$

$\nu_{-1}(g) = g_0 + g_1\zeta_8^{-1} + g_2\zeta_8^{-2} + g_3\zeta_8^{-3};$

$\nu_1(g) = g_0 + g_1\zeta_8 + g_2\zeta_8^2 + g_3\zeta_8^3; \quad \left|g\right|_1 = \left|\nu_1(g)\right|^2.$

$\nu_{-3}(g) = g_0 + g_1\zeta_8^{-3} + g_2\zeta_8^{-6} + g_3\zeta_8^{-9};$

$\nu_3(g) = g_0 + g_1\zeta_8^3 + g_2\zeta_8^6 + g_3\zeta_8^9; \quad \left|g\right|_3 = \left|\nu_3(g)\right|^2.$

$P_{17,2} = 17R + (x - 2)R:$

$P_{17,8} = 17R + (x - 8)R:$

$P_{17,-8} = 17R + (x + 8)R:$

$P_{17,-2} = 17R + (x + 2)R:$

$P_{41,3} = 41R + (x - 3)R:$

etc.

$\left|g\right|_{17,2} = 17^{-\text{ord}_{P_{17,2}} g}.$

$\left|g\right|_{17,8} = 17^{-\text{ord}_{P_{17,8}} g}.$

$\left|g\right|_{17,-8} = 17^{-\text{ord}_{P_{17,-8}} g}.$

$\left|g\right|_{17,-2} = 17^{-\text{ord}_{P_{17,-2}} g}.$

$\left|g\right|_{41,3} = 41^{-\text{ord}_{P_{41,3}} g}.$
$S$-units of $K = \mathbb{Q}[x]/(x^n + 1)$

Assume $\infty \subseteq S \subseteq \{\text{places of } K\}$.
Useful special case: $S$ has all primes $\leq y$ for some $y$.
[Warning: Often people rename $S - \infty$ as $S$.]

$g \in K^\times$ is an $S$-unit $\iff gR = \prod_{P \in S} P^{e_P}$ for some $e_P$
$\iff |g|_v = 1$ for all $v \in \{\text{places of } K\} - S$
$\iff$ the vector $v \mapsto \log |g|_v$ is 0 outside $S$.

**$S$-unit lattice:** set of such vectors $v \mapsto \log |g|_v$.

E.g. Temporarily allowing $n = 1$, $K = \mathbb{Q}$:
$\{\infty, 2, 3\}$-units in $\mathbb{Q}$ = $\pm 2\mathbb{Z}3\mathbb{Z}$. ("3-smooth".)
Lattice: $(\log 2, -\log 2, 0)\mathbb{Z} + (\log 3, 0, -\log 3)\mathbb{Z}$. 
Special case: unit attacks

0. Define $S = \infty$. $\{\text{\(\infty\)-units of } K\} = \{\text{units of } R\} = R^*$. 
1. Input a nonzero ideal $I$ of $R$. 
2. Find a generator of $I$: some $g$ with $gR = I$. 
3. Find a unit $u$ “close to $g$”. 
4. Output $g/u$.

This assumes $R^*$ is known and $I$ is principal.

Quality of the output:
How small is $g/u$ compared to $I$?
Most cryptosystems require approx SVP to be hard.

**History:** 2014 Bernstein: this is “reasonably well known among computational algebraic number theorists” and is a threat to lattice-based cryptography. 
2014 Campbell–Groves–Shepherd: exploit cyclotomic units to break a lattice-based system from 2009 Gentry. Assume finding $g$ with quantum algorithm. 
**S-unit attacks**

0. Choose a finite set $S$ of places including $\infty$.
1. Input a nonzero ideal $I$ of $R$.
2. Find an $S$-generator of $I$: some $g$ with $gR = I\prod_{P \in S} P^{e_P}$.
3. Find an $S$-unit $u$ “close to $g/I$”. This is an $S$-unit-lattice close-vector problem.
4. Output $g/u$.

Step 2 has a poly-time quantum algorithm from 2016 Biasse–Song, building on unit-group algorithm from 2014 Eisenträger–Hallgren–Kitaev–Song. Also has non-quantum algorithms running in subexponential time, assuming standard heuristics; for analysis and speedups see 2014 Biasse–Fieker.

Critical for Step 3 speed: constructing short vectors in the $S$-unit lattice.

**History:** 2015 Bernstein: apply unit attacks to close principal multiple of $I$.
2016 Bernstein: $S$-unit attacks.
“Cyclotomic units” in $R = \mathbb{Z}[x]/(x^n + 1)$

$\pm 1, \pm x, \pm x^2, \ldots, \pm x^{n-1} = \mp 1/x$ are units.
“Cyclotomic units” in $R = \mathbb{Z}[x]/(x^n + 1)$

$\pm 1, \pm x, \pm x^2, \ldots, \pm x^{n-1} = \mp 1/x$ are units.

$(1 - x^3)/(1 - x) = 1 + x + x^2 \in R.$

This is a unit since $(1 - x)/(1 - x^3) =$
“Cyclotomic units” in $R = \mathbb{Z}[x]/(x^n + 1)$

$\pm 1, \pm x, \pm x^2, \ldots, \pm x^{n-1} = \mp 1/x$ are units.

$$(1 - x^3)/(1 - x) = 1 + x + x^2 \in R.$$  
This is a unit since $(1 - x)/(1 - x^3) = (1 - x^{2^{n-1}+1})/(1 - x^3) \in R$.

For $c \in 1 + 2\mathbb{Z}$: $R$ has automorphism $\sigma_c : x \mapsto x^c$.

$\sigma_c(1 + x + x^2) = 1 + x^c + x^{2c}$ is a unit.

Useful to symmetrize: define $u_c = 1 + x^c + x^{-c}$. 
“Cyclotomic units” in $R = \mathbb{Z}[x]/(x^n + 1)$

$\pm 1, \pm x, \pm x^2, \ldots, \pm x^{n-1} = \mp 1/x$ are units.

$(1 - x^3)/(1 - x) = 1 + x + x^2 \in R$.
This is a unit since $(1 - x)/(1 - x^3) = (1 - x^{2n^2+1})/(1 - x^3) \in R$.

For $c \in 1 + 2\mathbb{Z}$: $R$ has automorphism $\sigma_c : x \mapsto x^c$.
$\sigma_c(1 + x + x^2) = 1 + x^c + x^{2c}$ is a unit.
Useful to symmetrize: define $u_c = 1 + x^c + x^{-c}$.

$x^\mathbb{Z} \prod_c u_c^{\mathbb{Z}}$ has finite index in $R^*$. Index is called $h^+$.
Assume $h^+ = 1$. Proven, assuming GRH, for $n \in \{2, 4, 8, \ldots, 256\}$; see 2014 Miller.
Heuristics say true for all powers of 2; see 2004 Buhler–Pomerance–Robertson, 2015 Miller.
Unit lattice for $n = 8$

$|u_1|_1 = |1 + \zeta_{16} + \zeta_{16}^{-1}|^2 \approx \exp 2.093.$
$|u_1|_3 = |1 + \zeta_{16}^3 + \zeta_{16}^{-3}|^2 \approx \exp 1.137.$
$|u_1|_5 = |1 + \zeta_{16}^5 + \zeta_{16}^{-5}|^2 \approx \exp -2.899.$
$|u_1|_7 = |1 + \zeta_{16}^7 + \zeta_{16}^{-7}|^2 \approx \exp -0.330.$

Define

$\text{Log}_\infty f = (\log |f|_1, \log |f|_3, \log |f|_5, \log |f|_7).$

$\text{Log}_\infty u_1 \approx (2.093, 1.137, -2.899, -0.330).$
$\text{Log}_\infty u_3 \approx (1.137, -0.330, 2.093, -2.899).$
$\text{Log}_\infty u_5 \approx (-2.899, 2.093, -0.330, 1.137).$
$\text{Log}_\infty u_7 \approx (-0.330, -2.899, 1.137, 2.093).$

$\text{Log}_\infty \mathbb{R}^* \text{ is lattice of dim } n/2 - 1 = 3 \text{ in hyperplane}$

$\{ (l_1, l_3, l_5, l_7) \in \mathbb{R}^4 : l_1 + l_3 + l_5 + l_7 = 0 \}.$

Short lattice basis: $\text{Log}_\infty u_1, \text{Log}_\infty u_3, \text{Log}_\infty u_5.$
Reducing modulo units

Assume $I$ is principal.
Start with generator $g = g_0 + g_1x + \cdots + g_{n-1}x^{n-1}$ of $I$.
Compute $\text{Log}_\infty g = (\log |g|_1, \log |g|_3, \ldots, \log |g|_{n-1})$.

Replacing $g$ with $gu$ replaces $|g|_c$ with $|g|_c|u|_c$.
Easy to track $||g||_2^2 = \sum_c |g|_c = (n/2)(g_0^2 + \cdots + g_{n-1}^2)$.
Reducing modulo units

Assume $I$ is principal.

Start with generator $g = g_0 + g_1x + \cdots + g_{n-1}x^{n-1}$ of $I$.

Compute $\Log_{\infty} g = (\log |g|_1, \log |g|_3, \ldots, \log |g|_{n-1})$.

Replacing $g$ with $gu$ replaces $|g|_c$ with $|g|_c|u|_c$.

Easy to track $\|g\|_2^2 = \sum_c |g|_c = (n/2)(g_0^2 + \cdots + g_{n-1}^2)$.

Try to reduce $\Log_{\infty} g$ modulo unit lattice:

Adjust $\Log_{\infty} g$ by subtracting vectors from $\Log_{\infty}(R^*)$.

Actually, precompute some combinations of basis vectors and subtract closest vector within this set; repeat several times; keep smallest $g_0^2 + \cdots + g_{n-1}^2$.

Note that unit hyperplane is orthogonal to norm:

$\#(R/I) = \#(R/g) = \prod_c |g|_c = \exp \sum_c \log |g|_c$. 

Tanja Lange S-unit attacks 19
Experiments for small $n$

Geometric average of $\eta^{1/n}$ over 100000 experiments:

<table>
<thead>
<tr>
<th>$n$</th>
<th>Model</th>
<th>Attack</th>
<th>Tweak</th>
<th>Shortest</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.01516</td>
<td>1.01518</td>
<td>1.01518</td>
<td>1.01518</td>
</tr>
<tr>
<td>8</td>
<td>1.01968</td>
<td>1.01972</td>
<td>1.01696</td>
<td>1.01696</td>
</tr>
<tr>
<td>16</td>
<td>1.01861</td>
<td>1.01860</td>
<td>1.01628</td>
<td>1.01627</td>
</tr>
</tbody>
</table>

“Shortest”: Take $I$, find a shortest nonzero vector $g$,
output $\eta = (g_0^2 + \cdots + g_{n-1}^2)^{1/2}/\#(R/I)^{1/n}$.

[Assuming BKZ-$n$ software produces shortest nonzero vector.]

“Attack”: Same $I$, find a generator, reduce mod unit lattice $\to g$,
output $(g_0^2 + \cdots + g_{n-1}^2)^{1/2}/\#(R/I)^{1/n}$.

“Model”: Take a hyperplane point, reduce mod unit lattice $\to \log_\infty g$,
output $(g_0^2 + \cdots + g_{n-1}^2)^{1/2}$.

“Tweak”: Multiply by $x + 1$, reduce, repeat for $I, (x + 1)I, (x + 1)^2I, (x + 1)^3I, (x + 1)^4I, \ldots$.
Often $(x + 1)^e g$ is closer to unit lattice than $g$.
(This is including a finite place of norm 2 in $S$.)
Nice $S$-units for cyclotomics (as in this talk)

Can use Gauss sums and Jacobi sums.
For details and more credits see 2021 talk given by Bernstein at SIAM-AG.

For each prime number $p \in 1 + 2n\mathbb{Z}$, and each group morphism $\chi : \mathbb{F}_p^* \to \zeta_{2n}^\mathbb{Z}$, define

$$\text{Gauss}_{S_p}(\chi) = \sum_{a \in \mathbb{F}_p^*} \chi(a)\zeta^a_p.$$ 

Then $\text{Gauss}_{S_p}(\chi)$ is an $S$-unit for $S = \infty \cup p$.

E.g. $n = 16$, $\zeta_{2n} = \zeta_{32}$, $p = 97 \in 1 + 2n\mathbb{Z}$:
There is a morphism $\chi : \mathbb{F}_97^* \to \zeta_{32}^\mathbb{Z}$ with $\chi(5) = \zeta_{32}$.

$\text{Gauss}_{S_p}(\chi) = \zeta_{32}^0\zeta_{97}^1 + \zeta_{32}^1\zeta_{97}^5 + \zeta_{32}^2\zeta_{97}^{25} + \cdots$.

$\text{Gauss}_{S_p}(\chi^2) = \zeta_{32}^0\zeta_{97}^1 + \zeta_{32}^2\zeta_{97}^5 + \zeta_{32}^4\zeta_{97}^{25} + \cdots$.

Stickelberger and augmented Stickelberger lattices used in 2019 Ducas–Plançon–Wesolowski are exponent vectors in factorizations of (some) ratios of Gauss sums.
Traditional method to find $S$-units: filtering

Take random small element $u \in R$: e.g. $u = x^{31} - x^{41} + x^{59} + x^{26} - x^{53}$.

1. Does $(R/u)$ factor into primes $\leq y$?
   Needs fast computation of norms and factorization.
   Lots of algorithmic speedups.

2. Is $u$ an $S$-unit for $S = \infty \cup \{ P : #(R/P) \leq y \}$?

Small primes $\Rightarrow$ fast non-quantum factorization.
[Helpful speedups: almost always $(R/P) \in 1 + 2n\mathbb{Z}$. Batch factorization.]

Standard heuristics $\Rightarrow y^{2+o(1)}$ choices of $u$ include $y^{1+o(1)}$ $S$-units, spanning all $S$-units, for
- appropriate $n^{1/2+o(1)}$ choice for $\log y$,
- appropriate $n^{1/2+o(1)}$ choice for $\sum_i u_i^2$.

Total time $\exp(n^{1/2+o(1)})$.

Can tricks from NFS on extensions be applied to reach $1/3 + o(1)$?
Automorphisms and subrings

Apply each $\sigma_c$ to quickly amplify each $u$ found into, typically, $n$ independent $S$-units.

What if $u$ is invariant under (say) two $\sigma_c$?
Automorphisms and subrings

Apply each $\sigma_c$ to quickly amplify each $u$ found into, typically, $n$ independent $S$-units.

What if $u$ is invariant under (say) two $\sigma_c$? Great!
Start with $u$ from proper subrings. Makes $\#(R/u)$ much more likely to factor into small primes.

Examples of useful subrings of $R = \mathbb{Z}[x]/(x^n + 1)$:

- $\mathbb{Z}[x^2]/(x^n + 1) = \{u \in R : \sigma_{n+1}(u) = u\}$.
- $R^+ = \{u \in R : \sigma_{-1}(u) = u\}$.

Also use subrings to speed up $\#(R/u)$ computation: see https://s-unit.attacks.cr.yp.to/norms.html.

Overview: Constructing small $S$-units

$$\sigma_c \downarrow \downarrow x + 1 \rightarrow \rightarrow \sqrt{P_1 P_{-1} \text{ gen}}} \uparrow \uparrow \text{random in } R \uparrow \uparrow \text{random in } R^+ \uparrow \uparrow \text{Gauss ratios}$$

Tanja Lange S-unit attacks 24
Conjectured scalability: $\exp(n^{1/2+o(1)})$

Simple algorithm variant, skipping many speedups:

Take traditional $\log y \in n^{1/2+o(1)}$.
Take $S = \infty \cup \{P : \#(R/P) \leq y\}$.
Precompute $\{S\text{-unit } u \in R: \sum_i u_i^2 \leq n^{1/2+o(1)}\}$. 

To randomize, multiply $I$ by some random primes in $S$. Can repeat $yO(1)$ times.

Compute $S$-generator $g$ of $I$ (quantum or classical).
Clear denominators: Multiply by generators of $P_c$. $\Rightarrow$ element of $I$ that $S$-generates $I$.
Replace $g$ with $gu/v$ having log vector closest to $I$; repeat until stable $\Rightarrow$ short element of $I$. 

Heuristics $\Rightarrow \eta \leq n^{1/2+o(1)}$, time $\exp(n^{1/2+o(1)})$.

"Vector within $\varepsilon$ of shortest in subexponential time."

Compare to typical cryptographic assumption: $\eta \leq n^{2+o(1)}$ is hard to reach.
Conjectured scalability: \( \exp(n^{1/2+o(1)}) \)

Simple algorithm variant, skipping many speedups:

Take traditional \( \log y \in n^{1/2+o(1)} \).

Take \( S = \infty \cup \{ P : \#(R/P) \leq y \} \).

Precompute

\[
\{ S\text{-unit } u \in R : \sum_i u_i^2 \leq n^{1/2+o(1)} \}.
\]

To randomize, multiply \( I \) by some random primes in \( S \). Can repeat \( y^{O(1)} \) times.

Compute \( S\)-generator \( g \) of \( I \) (quantum or classical).

Clear denominators: Multiply by generators of \( P_c P_{-c} \) (this assumes \( h^+ = 1 \))

\[ \Rightarrow \text{element of } I \text{ that } S\text{-generates } I. \]

Replace \( g \) with \( g\frac{u}{v} \) having log vector closest to \( I \);
repeat until stable \( \Rightarrow \) short element of \( I \).

Heuristics \( \Rightarrow \eta \leq n^{1/2+o(1)} \), time \( \exp(n^{1/2+o(1)}) \).

“Vector within \( \varepsilon \) of shortest in subexponential time.”

Compare to typical cryptographic assumption: \( \eta \leq n^{2+o(1)} \) is hard to reach.
Non-randomness of $S$-unit lattices

Number of points of a lattice $L$ in a big ball $B \approx \frac{\text{vol } B}{\text{covol } L}$.

For almost all lattices $L$ (1956 Rogers, . . . , 2019 Strömbergsson–Södergren):
If $\text{vol } B = \text{covol } L$ then length of shortest nonzero vector in $L \approx \text{radius of } B$.

2016 Laarhoven: analogous heuristics for effectiveness of reduction via subtracting off short vectors from database. 2019 Pellet-Mary–Hanrot–Stehlé, 2021 Ducas–Pellet-Mary: Apply these heuristics to $S$-unit lattices $\Rightarrow$ very small chance that previous slide works.
Non-randomness of $S$-unit lattices

Number of points of a lattice $L$ in a big ball $B \approx \frac{\text{vol } B}{\text{covol } L}$.

For almost all lattices $L$ (1956 Rogers, . . . , 2019 Strömbergsson–Södergren):
If $\text{vol } B = \text{covol } L$ then length of shortest nonzero vector in $L \approx$ radius of $B$.

2016 Laarhoven: analogous heuristics for effectiveness of reduction via subtracting off short vectors from database. 2019 Pellet-Mary–Hanrot–Stehlé, 2021 Ducas–Pellet-Mary: Apply these heuristics to $S$-unit lattices $\Rightarrow$ very small chance that previous slide works.

But all of these heuristics provably fail for the lattice $\mathbb{Z}^d$.
Are these accurate for $S$-unit lattices?

2021 Bernstein–Lange “Non-randomness of $S$-unit lattices”:
The standard length/reduction heuristics provably fail for $S$-unit lattices for $(1) n = 1$, any $S$; (2) each $n$ as $S$ grows (roughly what the previous slide uses); (3) minimal $S$, any $n$.
See https://s-unit.attacks.cr.yp.to/spherical.html.
Evidence for the conjecture

For traditional $\log y \in n^{1/2+o(1)}$, time budget $\exp(n^{1/2+o(1)})$:
Standard smoothness heuristics $\Rightarrow$ find short $S$-units spanning the $S$-unit lattice, as in 2014 Biasse–Fieker; and find $S$-generator of $I$.

Various quantifications of the behavior of $S$-unit lattices are much closer to $\mathbb{Z}^d$ than to random lattices.
Model reduction as $\mathbb{Z}^d$ reduction $\Rightarrow$ find short $S$-generator of $I$.

Full attack software now available: https://s-unit.attacks.cr.yp.to/filtered.html.
Numerical experiments are consistent with the heuristics.

Ongoing work: attack speedups; more precise $S$-unit models and predictions; more numerical evidence for comparison to the models; other fast $S$-unit constructions, exploiting more cyclotomic structure.