NTRU Prime

Daniel J. Bernstein, Chitchanok Chuengsatiansup,
Tanja Lange, and Christine van Vredendaal

31 January 2018

NTRU History

- Introduced by Hoffstein–Pipher–Silverman in 1998.
- Security related to lattice problems; pre-version cryptanalyzed with LLL by Coppersmith and Shamir.
- System parameters (p, q), p prime, integer q, $\gcd(3, q) = 1$.
- All computations done in ring $R = \mathbf{Z}[x]/(x^p - 1)$.

NTRU History

- Introduced by Hoffstein–Pipher–Silverman in 1998.
- Security related to lattice problems; pre-version cryptanalyzed with LLL by Coppersmith and Shamir.
- System parameters (p, q), p prime, integer q, $\gcd(3, q) = 1$.
- All computations done in ring $R = \mathbf{Z}[x]/(x^p - 1)$.
- Private key: $f, g \in R$ sparse with coefficients in $\{-1, 0, 1\}$. Additional requirement: f must be invertible in R modulo q.
- Public key $h = 3g/f \bmod q$.
- Can see this as lattice with basis matrix

  $$B = \begin{pmatrix} q\,I_p & 0 \\ H & I_p \end{pmatrix},$$

  where H corresponds to multiplication by $h/3$ modulo $x^p - 1$.
- (g, f) is a short vector in the lattice as result of

  $$(k, f)B = (kq + f \cdot h/3, f) = (g, f)$$

  for some polynomial k (from $fh/3 = g - kq$).

Original NTRU

- System parameters (p, q), p prime, integer q, $\gcd(p, q) = 1$.
- All computations done in ring $R = \mathbf{Z}[x]/(x^p - 1)$, some use additional reduction modulo q, ring denoted by R_q.

Original NTRU

- System parameters (p, q), p prime, integer q, $\gcd(p, q) = 1$.
- All computations done in ring $R = \mathbf{Z}[x]/(x^p - 1)$, some use additional reduction modulo q, ring denoted by R_q.
- Private key: $f, g \in R$ with coefficients in $\{-1, 0, 1\}$, almost all coefficients are zero (small fixed number are nonzero).
  Additional requirement: f must be invertible in R modulo q and modulo 3.
- Public key $h = 3g/f \bmod q$.

Original NTRU

- System parameters (p, q), p prime, integer q, $\gcd(p, q) = 1$.
- All computations done in ring $R = \mathbf{Z}[x]/(x^p - 1)$, some use additional reduction modulo q, ring denoted by R_q.
- Private key: $f, g \in R$ with coefficients in $\{-1, 0, 1\}$, almost all coefficients are zero (small fixed number are nonzero).
  Additional requirement: f must be invertible in R modulo q and modulo 3.
- Public key $h = 3g/f \bmod q$.
- Encryption of message $m \in R$, coefficients in $\{-1, 0, 1\}$:
  Pick random, sparse $r \in R$, same sample space as f; compute:

$$c = r \cdot h + m \bmod q.$$

- Decryption of $c \in R_q$: Compute

$$a = f \cdot c = f(rh + m) \equiv f(3rg/f + m) \equiv 3rg + fm \bmod q,$$

move all coefficients to $[-q/2, q/2]$. If everything is small enough then a equals $3rg + fm$ in R and $m = a/f \bmod 3$.

Why we don't stick with original NTRU.

Reason 1: Decryption failures

- Decryption of $c \in R_q$: Compute

$$a = f \cdot c = f(rh + m) \equiv f(3rg/f + m) \equiv 3rg + fm \bmod q,$$

move all coefficients to $[-q/2, q/2]$. **If everything is small enough** then a equals $3rg + fm$ in $\mathcal{R}$ and $m = a/f \bmod 3$.

Reason 1: Decryption failures

- Decryption of $c \in R_q$: Compute

$$a = f \cdot c = f(rh + m) \equiv f(3rg/f + m) \equiv 3rg + fm \bmod q,$$

move all coefficients to $[-q/2, q/2]$. **If everything is small enough** then a equals $3rg + fm$ in $\mathcal{R}$ and $m = a/f \bmod 3$.

- Let

$$L(d,t) = \{F \in \mathcal{R} | F \text{ has } d \text{ coefficients equal to } 1 \text{ and } t \text{ coefficients equal to } -1, \text{all others } 0\}.$$

- Then $f \in L(d_f, d_f - 1)$, $r \in L(d_r, d_r)$, and $g \in L(d_g, d_g)$ with $d_r < d_g$.
- Then $3rg + fm$ has coefficients of size at most

$$3 \cdot 2d_r + 2d_f - 1$$

Reason 1: Decryption failures

- Decryption of $c \in R_q$: Compute

$$a = f \cdot c = f(rh + m) \equiv f(3rg/f + m) \equiv 3rg + fm \bmod q,$$

move all coefficients to $[-q/2, q/2]$. **If everything is small enough** then a equals $3rg + fm$ in $\mathcal{R}$ and $m = a/f \bmod 3$.

- Let

$$L(d,t) = \{F \in \mathcal{R} | F \text{ has } d \text{ coefficients equal to } 1 \text{ and } t \text{ coefficients equal to } -1, \text{all others } 0\}.$$

- Then $f \in L(d_f, d_f - 1)$, $r \in L(d_r, d_r)$, and $g \in L(d_g, d_g)$ with $d_r < d_g$.
- Then $3rg + fm$ has coefficients of size at most

$$3 \cdot 2d_r + 2d_f - 1$$

which is larger than $q/2$ for typical parameters. Such large coefficients are highly unlikely – but annoying for applications and guarantees.

Reason 1: Decryption failures

- Decryption of $c \in R_q$: Compute

$$a = f \cdot c = f(rh + m) \equiv f(3rg/f + m) \equiv 3rg + fm \bmod q,$$

  move all coefficients to $[-q/2, q/2]$. **If everything is small enough** then a equals $3rg + fm$ in $\mathcal{R}$ and $m = a/f \bmod 3$.
- Let

$$L(d,t) = \{F \in \mathcal{R} | F \text{ has } d \text{ coefficients equal to } 1 \text{ and } t \text{ coefficients equal to } -1, \text{all others } 0\}.$$

- Then $f \in L(d_f, d_f - 1)$, $r \in L(d_r, d_r)$, and $g \in L(d_g, d_g)$ with $d_r < d_g$.
- Then $3rg + fm$ has coefficients of size at most

$$3 \cdot 2d_r + 2d_f - 1$$

  which is larger than $q/2$ for typical parameters. Such large coefficients are highly unlikely – but annoying for applications and guarantees.
- Security decreases with large q; reduction is important.

Reason 2: Evaluation-at-1 attack

- Ciphertext equals $c = rh + m$ and $r \in L(d_r, d_r)$, so $r(1) = 0$ and $g \in L(d_g, d_g)$, so $h(1) = g(1)/f(1) = 0$.
- This implies
$$c(1) = r(1)h(1) + m(1) = m(1)$$
which gives information about m, in particular if $|m(1)|$ is large.

Reason 2: Evaluation-at-1 attack

- Ciphertext equals $c = rh + m$ and $r \in L(d_r, d_r)$, so $r(1) = 0$ and $g \in L(d_g, d_g)$, so $h(1) = g(1)/f(1) = 0$.
- This implies

$$c(1) = r(1)h(1) + m(1) = m(1)$$

  which gives information about m, in particular if $|m(1)|$ is large.
- For other choices of r and h, such as $L(d_r, d_r - 1)$ or such, one knows $r(1)$ and h is public, so evaluation at 1 leaks $m(1)$.

Reason 2: Evaluation-at-1 attack

- Ciphertext equals $c = rh + m$ and $r \in L(d_r, d_r)$, so $r(1) = 0$ and $g \in L(d_g, d_g)$, so $h(1) = g(1)/f(1) = 0$.
- This implies
$$c(1) = r(1)h(1) + m(1) = m(1)$$
which gives information about m, in particular if $|m(1)|$ is large.
- For other choices of r and h, such as $L(d_r, d_r - 1)$ or such, one knows $r(1)$ and h is public, so evaluation at 1 leaks $m(1)$.
- Original NTRU rejects extreme messages – this is dealt with by randomizing m via a padding (not mentioned so far).
- Could also replace $x^p - 1$ by $\Phi_p = (x^p - 1)/(x - 1)$ to avoid attack.

Reason 3: Mappings to subrings

- Consider $R_q = (\mathbf{Z}/q)[x]/(x^p - 1)$.
- Can possibly get more information on m from homomorphism $\psi : R_q \to T$, for some ring T.
- Typical choice in original NTRU: $q = 2048$ leads to natural ring maps from $(\mathbf{Z}/2048)[x]/(x^p - 1)$ to
  - $(\mathbf{Z}/2)[x]/(x^p - 1)$,
  - $(\mathbf{Z}/4)[x]/(x^p - 1)$,
  - $(\mathbf{Z}/8)[x]/(x^p - 1)$, etc.

Reason 3: Mappings to subrings

- Consider $R_q = (\mathbf{Z}/q)[x]/(x^p - 1)$.
- Can possibly get more information on m from homomorphism $\psi : R_q \to T$, for some ring T.
- Typical choice in original NTRU: $q = 2048$ leads to natural ring maps from $(\mathbf{Z}/2048)[x]/(x^p - 1)$ to
  - $(\mathbf{Z}/2)[x]/(x^p - 1)$,
  - $(\mathbf{Z}/4)[x]/(x^p - 1)$,
  - $(\mathbf{Z}/8)[x]/(x^p - 1)$, etc.
- Unclear whether these can be exploited to get information on m.
- Maybe, complicated. [Silverman-Smart-Vercauteren '04]
- If you pick bad rings, then yes. [Eisenträger-Hallgren-Lauter '14, Elias-Lauter-Ozman-Stange '15, Chen-Lauter-Stange '16, Castryck-Iliashenko-Vercauteren '16]

Reasons 4 and 5

- Rings of original NTRU also have
  - a large proper subfield (used in attack by [Bauch-Bernstein-Lange-de Valence-van Vredendaal '17], attack by [Albrecht-Bai-Ducas '16], and attack in Bernstein's 2014 blogpost).
  - many easily computable automorphisms (usable to find a fundamental basis of short units which is used in [Campbell-Groves-Shepherd '14] and subsequently [Cramer-Ducas-Peikert-Regev '15]).

Reasons 4 and 5

- Rings of original NTRU also have
  - a large proper subfield (used in attack by [Bauch-Bernstein-Lange-de Valence-van Vredendaal '17], attack by [Albrecht-Bai-Ducas '16], and attack in Bernstein's 2014 blogpost).
  - many easily computable automorphisms (usable to find a fundamental basis of short units which is used in [Campbell-Groves-Shepherd '14] and subsequently [Cramer-Ducas-Peikert-Regev '15]).
- Whether paranoia, or valid panic; what can we do about it?

NTRU Prime ring

- Differences from original NTRU:
  prime degree, large Galois group, inert modulus.

NTRU Prime ring

- Differences from original NTRU:
  prime degree, large Galois group, inert modulus.
- Choose monic irreducible polynomial $P \in \mathbf{Z}[x]$.
- Choose prime q such that P is irreducible modulo q; this means that q is inert in $\mathcal{R} = \mathbf{Z}[x]/P$ and $(\mathbf{Z}/q)[x]/P$ is a field.

NTRU Prime ring

- Differences from original NTRU:
  prime degree, large Galois group, inert modulus.
- Choose monic irreducible polynomial $P \in \mathbf{Z}[x]$.
- Choose prime q such that P is irreducible modulo q; this means that q is inert in $\mathcal{R} = \mathbf{Z}[x]/P$ and $(\mathbf{Z}/q)[x]/P$ is a field.
- Further choose P of prime degree p with large Galois group.
- Specifically, set $P = x^p - x - 1$.
  This has Galois group S_p of size $p!$.
- NTRU Prime works over the NTRU Prime *field*

$$\mathcal{R}/q = (\mathbf{Z}/q)[x]/(x^p - x - 1).$$

NTRU Prime: added defenses

Prime degree, large Galois group, inert modulus.

NTRU Prime: added defenses

Prime degree, large Galois group, inert modulus.

- ➜ Only subfields of $\mathbf{Q}[x]/P$ are itself and $\mathbf{Q}$. Avoids structures used by, e.g., multiquad attack.
- ➜ Large Galois group means no easy to compute automorphisms. Roots of P live in degree-$p!$ extension. Avoids structures used by Campbell–Groves–Shepherd attack (obtaining short unit basis). No hopping between units, so no easy way to extend from some small unit to a fundamental system of short units.
- ➜ No ring homomorphism to smaller nonzero rings. Avoids structures used by Chen–Lauter–Stange attack.

NTRU Prime: added defenses

Prime degree, large Galois group, inert modulus.

- Only subfields of $\mathbf{Q}[x]/P$ are itself and $\mathbf{Q}$. Avoids structures used by, e.g., multiquad attack.
- Large Galois group means no easy to compute automorphisms. Roots of P live in degree-$p!$ extension. Avoids structures used by Campbell–Groves–Shepherd attack (obtaining short unit basis). No hopping between units, so no easy way to extend from some small unit to a fundamental system of short units.
- No ring homomorphism to smaller nonzero rings. Avoids structures used by Chen–Lauter–Stange attack.

Irreducibility also avoids the evaluation-at-1 attack which simplifies padding.

Streamlined NTRU Prime: private and public key

- System parameters (p, q, t), p, q prime, $q \geq 32t + 1$.
- Pick g small in $\mathcal{R}$

  $$g = g_0 + \cdots + g_{p-1}x^{p-1} \text{ with } g_i \in \{-1, 0, 1\}$$

  No weight restriction on g, only size restriction on coefficients; g required to be invertible in $\mathcal{R}/3$.
- Pick t-small $f \in \mathcal{R}$

  $$f = f_0 + \cdots + f_{p-1}x^{p-1} \text{ with } f_i \in \{-1, 0, 1\} \text{ and } \sum |f_i| = 2t$$

  Since $\mathcal{R}/q$ is a field, f is invertible.
- Compute public key $h = g/(3f)$ in $\mathcal{R}/q$.
- Private key is f and $1/g \in \mathcal{R}/3$.

Streamlined NTRU Prime: private and public key

- System parameters (p, q, t), p, q prime, $q \geq 32t + 1$.
- Pick g small in $\mathcal{R}$

$$g = g_0 + \cdots + g_{p-1}x^{p-1} \text{ with } g_i \in \{-1, 0, 1\}$$

  No weight restriction on g, only size restriction on coefficients; g required to be invertible in $\mathcal{R}/3$.
- Pick t-small $f \in \mathcal{R}$

$$f = f_0 + \cdots + f_{p-1}x^{p-1} \text{ with } f_i \in \{-1, 0, 1\} \text{ and } \sum |f_i| = 2t$$

  Since $\mathcal{R}/q$ is a field, f is invertible.
- Compute public key $h = g/(3f)$ in $\mathcal{R}/q$.
- Private key is f and $1/g \in \mathcal{R}/3$.
- Difference from original NTRU: more key options, 3 in denominator.

Streamlined NTRU Prime: KEM/DEM

- Streamlined NTRU Prime is a Key Encapsulation Mechanism (KEM).
- Combine with Data Encapsulation Mechanism (DEM) to send messages.

Streamlined NTRU Prime: KEM/DEM

- Streamlined NTRU Prime is a Key Encapsulation Mechanism (KEM).
- Combine with Data Encapsulation Mechanism (DEM) to send messages.

KEM:

- Alice looks up Bob's public key h.
- Picks t-small $r \in \mathcal{R}$ (i.e., $r_i \in \{-1, 0, 1\}, \sum |r_i| = 2t$).
- Computes hr in $\mathcal{R}/q$, lifts coefficients to $\mathbf{Z} \cap [-(q-1)/2, (q-1)/2]$.

Streamlined NTRU Prime: KEM/DEM

- Streamlined NTRU Prime is a Key Encapsulation Mechanism (KEM).
- Combine with Data Encapsulation Mechanism (DEM) to send messages.

KEM:

- Alice looks up Bob's public key h.
- Picks t-small $r \in \mathcal{R}$ (i.e., $r_i \in \{-1, 0, 1\}, \sum |r_i| = 2t$).
- Computes hr in $\mathcal{R}/q$, lifts coefficients to $\mathbf{Z} \cap [-(q-1)/2, (q-1)/2]$.
- Rounds each coefficient to the nearest multiple of 3 to get c.
- Computes $\mathrm{hash}(r) = (C|K)$.
- Sends $(C|c)$, uses session key K for DEM.

Rounding hr saves bandwidth and adds same entropy as adding ternary m.

Streamlined NTRU Prime: decapsulation

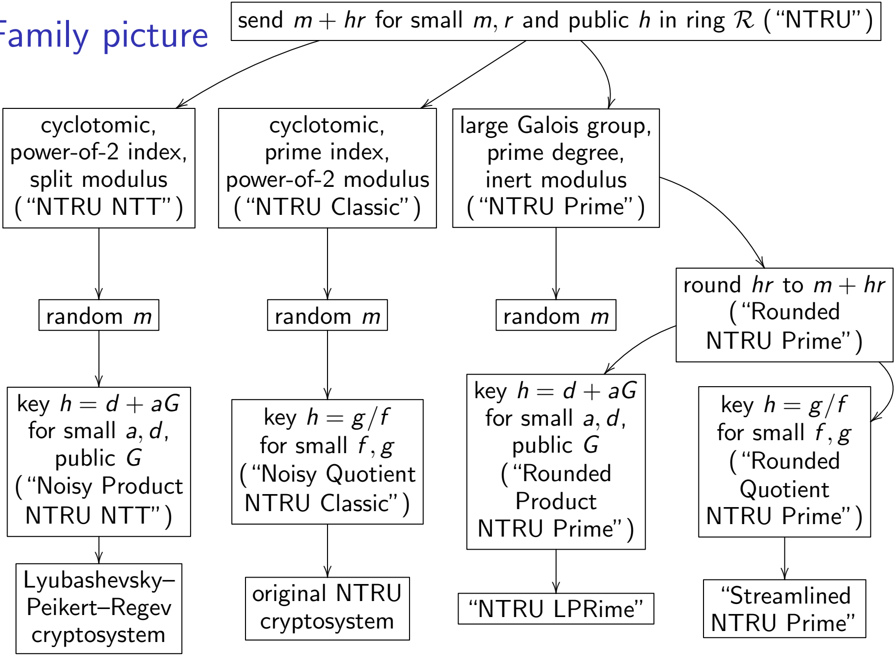
Bob decrypts $(C|c)$:

- Reminder $h = g/(3f)$ in $\mathcal{R}/q$.
- Computes $3fc = 3f(hr + m) = gr + 3fm$ in $\mathcal{R}/q$, lifts coefficients to $\mathbf{Z} \cap [-(q-1)/2, (q-1)/2]$.
- Reduces the coefficients modulo 3 to get $a = gr \in \mathcal{R}/3$.
- Computes $r' = a/g \in \mathcal{R}/3$, lifts r' to $\mathcal{R}$.
- Computes $\mathrm{hash}(r') = (C'|K')$ and c' as rounding of hr'.
- Verifies that $c' = c$ and $C' = C$.

If all checks verify, $K = K'$ is the session key between Alice and Bob and can be used in a data encapsulation mechanism (DEM).

Choosing $q \geq 32t + 1$ means no decryption failures, so $r = r'$ and verification works unless $(C|c)$ was incorrectly generated or tempered with.

Family picture



Streamlined NTRU Prime: Security

- What we know so far:

	Original NTRU	**Common R-LWE**	**Streamlined NTRU Prime**
Polynomial P	$x^p - 1$	$x^p + 1$	$x^p - x - 1$
Degree p	prime	power of 2	prime
Modulus q	2^d	prime	prime
# factors of P in $\mathcal{R}/q$	> 1	p	1
# proper subfields	> 1	many	1
Every m encryptable	✗	✓	✓
No decryption failures	✗	✗	✓

Streamlined NTRU Prime: Security

- What we know so far:

	Original NTRU	**Common R-LWE**	**Streamlined NTRU Prime**
Polynomial P	$x^p - 1$	$x^p + 1$	$x^p - x - 1$
Degree p	prime	power of 2	prime
Modulus q	2^d	prime	prime
$\#$ factors of P in $\mathcal{R}/q$	> 1	p	1
$\#$ proper subfields	> 1	many	1
Every m encryptable	✗	✓	✓
No decryption failures	✗	✗	✓

- Because of the last 2 ✓'s the analysis is simpler than that of original NTRU.

Streamlined NTRU Prime Security: parameters

- We investigated security against the strongest known attacks; meet-in-the-middle (mitm), hybrid attack of BKZ and mitm, algebraic attacks, and sieving.
- Streamlined NTRU Prime 4591^{761} and NTRU LPRime 4591^{761} both use $p = 761$ and $q = 4591$.
- The resulting sizes and Haswell speeds show that reducing the attack surface has very low cost:

Metric	**Streamlined NTRU Prime** 4591^{761}	**NTRU LPRime** 4591^{761}
Public-key size	1218 bytes	1047 bytes
Ciphertext size	1047 bytes	1175 bytes
Encapsulation time	59456 cycles	94508 cycles
Decapsulation time	97684 cycles	128316 cycles
Pre-quantum security	248 bits	225 bits

- Quantum computers will speed up attacks by less than squareroot.

Bonus slides: why automorphisms matter

Targets and history:

- 2014.10 Campbell–Groves–Shepherd describe an ideal-lattice-based system "Soliloquy"; claim quantum poly-time key recovery.
- 2010 Smart–Vercauteren system is practically identical to Soliloquy.
- 2009 Gentry system (simpler version described at STOC) has the same key-recovery problem.
- 2012 Garg–Gentry–Halevi multilinear maps have the same key-recovery problem (and many other security issues).

Smart–Vercauteren; Soliloquy

- Parameter: $k \geq 1$.
- Define $R = \mathbf{Z}[x]/\Phi_{2^k}$.
- Public key: prime q and $c \in \mathbf{Z}/q$.
- Secret key: short element $g \in R$ with $gR = qR + (x - c)R$; i.e., short generator of the ideal $qR + (x - c)R$.

Smart–Vercauteren; Soliloquy

- Parameter: $k \geq 1$.
- Define $R = \mathbf{Z}[x]/\Phi_{2^k}$.
- Public key: prime q and $c \in \mathbf{Z}/q$.
- Secret key: short element $g \in R$ with $gR = qR + (x - c)R$;
  i.e., short generator of the ideal $qR + (x - c)R$.
- 1993 Cohen textbook "A course in computational algebraic number theory" explains how to find generators.

Smart–Vercauteren; Soliloquy

- Parameter: $k \geq 1$.
- Define $R = \mathbf{Z}[x]/\Phi_{2^k}$.
- Public key: prime q and $c \in \mathbf{Z}/q$.
- Secret key: short element $g \in R$ with $gR = qR + (x - c)R$; i.e., short generator of the ideal $qR + (x - c)R$.
- 1993 Cohen textbook "A course in computational algebraic number theory" explains how to find generators.
- Smart–Vercauteren comment that this would take exponential time.
- But it actually takes subexponential time. Same basic idea as NFS.
- Campbell–Groves–Shepherd claim quantum poly time. Claim disputed by Biasse, not defended by CGS.

Smart–Vercauteren; Soliloquy

- Parameter: $k \geq 1$.
- Define $R = \mathbf{Z}[x]/\Phi_{2^k}$.
- Public key: prime q and $c \in \mathbf{Z}/q$.
- Secret key: short element $g \in R$ with $gR = qR + (x - c)R$; i.e., short generator of the ideal $qR + (x - c)R$.
- 1993 Cohen textbook "A course in computational algebraic number theory" explains how to find generators.
- Smart–Vercauteren comment that this would take exponential time.
- But it actually takes subexponential time. Same basic idea as NFS.
- Campbell–Groves–Shepherd claim quantum poly time. Claim disputed by Biasse, not defended by CGS.
- 2016 Biasse–Song: different algorithm that takes quantum poly time, building on 2014 Eisenträger–Hallgren–Kitaev–Song.

How to get a short generator?

- Have ideal I of R.
- Want short g with $gR = I$; have g' with $g'R = I$.
- Know $g' = ug$ for some unit $u \in R^*$.
- To find u move to log lattice.

$$\operatorname{Log} g' = \operatorname{Log} u + \operatorname{Log} g,$$

  where Log is Dirichlet's log map.
- Dirichlet's unit theorem:
  $\operatorname{Log} R^*$ is a lattice of known dimension.
- Finding $\operatorname{Log} u$ is a closest-vector problem in this lattice.

Quote from Campbell–Groves–Shepherd

"A simple generating set for the cyclotomic units is of course known. The image of $\mathcal{O}^\times$ [here R^*] under the logarithm map forms a lattice. The determinant of this lattice turns out to be much bigger than the typical loglength of a private key α [here g], so it is easy to recover the causally short private key given *any* generator of $\alpha\mathcal{O}$ [here I], *e.g.* via the LLL lattice reduction algorithm."

Automorphisms

- $x \mapsto x^3$, $x \mapsto x^5$, $x \mapsto x^7$, etc. are automorphisms of $R = \mathbf{Z}[x]/\Phi_{2^k}$.
- Easy to see $(1 - x^3)/(1 - x) \in R^*$; for inverse use expansion.
- "Cyclotomic units" are defined as

$$R^* \cap \left\{ \pm x^{e_0} \prod_i (1 - x^i)^{e_i} \right\}.$$

- Weber's conjecture:

  All elements of R^* are cyclotomic units.
- Experiments confirm that SV is quickly broken by LLL using, e.g., 1997 Washington textbook basis for cyclotomic units.
- Shortness of basis is critical; this was not highlighted in CGS analysis.