I'm writing on behalf of the NTRU Prime team, as a public response to NIST's request for a summary of expected changes in round 3.

NTRU Prime is a small lattice system. Subject to this constraint, our primary objective is to eliminate unnecessary complications in security review. We correctly predicted that such complications would lead to security failures in NISTPQC lattice submissions. We evaluated a variety of trapdoor functions from this perspective before submission, again during round 1, and again during round 2.

On this basis we have once again decided against decryption failures; modules; errors; and all other changes that we have considered to our family of trapdoor functions. We therefore plan to submit the same family of trapdoor functions in round 3. NTRU Prime will therefore have an unchanged family of trapdoor functions throughout round 1, round 2, and round 3.

Our CCA conversion includes various hashing safeguards, some already in round 1 and some added in round 2. These safeguards cost 32 bytes in ciphertext size and a considerable fraction of our CPU time. However, even with these safeguards, NTRU Prime often outperforms other small lattice KEMs, as the following references show:

https://cr.yp.to/papers.html#paretoviz
https://bench.cr.yp.to/results-kem.html#amd64-hiphop
https://github.com/mupq/pqm4/blob/master/benchmarks.md

More importantly, the costs of our hashing safeguards are negligible in applications. We plan to submit the same CCA conversion in round 3.

NTRU Prime will thus be fully compatible between round 2 and round 3, when users choose the same parameters.

Regarding parameter selection, we are concerned that pre-quantum Core-SVP levels 2^100, 2^106, and 2^111, proposed for category 1 for Dilithium, NTRU, and Kyber respectively, will turn out to be inadequate against generic lattice attacks. We will not add dimensions below our 653 (pre-quantum Core-SVP 2^129). We recommend our original dimension 761 (pre-quantum Core-SVP 2^153) for an extra security margin.

We have seen various requests for larger dimensions, even larger than our dimension 857 (pre-quantum Core-SVP 2^175). To accommodate these requests and prevent any accusations of a lack of flexibility, we plan to add some larger dimensions as a supplement to our current dimensions.

We have also considered adding intermediate parameter sets to further illustrate our flexibility, showing that NTRU Prime offers even larger advantages in Core-SVP under various size limits compared to, e.g., Kyber. The call for proposals explicitly allowed multiple parameter sets per category. However, NIST has now made an announcement that seems to discourage "too many parameter sets".
The rest of this message is regarding the problems caused by NIST's unstable definitions of security categories. These are problems for the NISTPQC process broadly, not just for NTRU Prime.

The call for proposals specified AES-128 key search as a "floor" for category 1 in "all metrics that NIST deems to be potentially relevant to practical security". The call for proposals similarly specified floors for other categories. However, this is not a clear and stable category definition unless the metrics are clear and stable.

As an illustration of how much impact metrics have, there is a 40-year literature studying metrics for realistic large-scale two-dimensional models of computation. Standard theorems---see, e.g.,


---imply that these metrics assign 50% higher asymptotic exponents to large-integer multiplication, large-array sorting, etc. than a "gates" metric does. (Analogous three-dimensional metrics studied in, e.g.,

https://link.springer.com/article/10.1007/BF01744565

are for machines that appear far more difficult to build than quantum computers, and still have 33% higher exponents than a "gates" metric.) Any attack that has large-scale sorting as a bottleneck is affected by this, whereas AES-128 key search is not.

The call for proposals highlighted "classical gates" and "quantum gates" (with limited depth) as metrics. However, NIST is not requiring lattice submissions to meet the "classical gates" floor. (See examples below.)

NIST also has not defined a replacement metric for submissions to use. All lattice submissions have Core-SVP evaluations, but AES-128 does not. Core-SVP is not a metric for the cost of computation: it is a mechanism for claiming security levels (in an undefined metric) specifically for lattices. Ray Perlner's message dated 9 Jun 2020 15:39:09 +0000 stated "we feel that the CoreSVP metric does indicate which lattice schemes are being more and less aggressive in setting their parameters", but the mapping from Core-SVP evaluations to categories remains undefined.

NIST IR 8309's handling of categories is not consistent across lattice submissions. Consider the following examples from three different submissions:

(P80) Category 3 for pre-quantum Core-SVP $2^{153}$; 153 is 80% of 192.

(P78) Category 1 for pre-quantum Core-SVP $2^{100}$; 100 is 78% of 128.

(P71) Category 3 for pre-quantum Core-SVP $2^{136}$; 136 is 71% of 192.

P78 is the lowest of these three examples in Core-SVP, and P71 is the lowest in Core-SVP as a percentage of the AES key size for the category.

However, NIST's wording was strikingly more negative for P80 than for P78 and P71:

(P80) "quite aggressive compared to most of the other submissions targeting the same security categories"; need to study "whether they actually meet their claimed security categories";
(P78) "lowest CoreSVP security strength parameter set of any of the lattice schemes still in the process"; need more study on "understanding the concrete security";

(P71) "lower CoreSVP complexity than many of the other schemes targeting the same security strength categories"; need to "understand exactly ... bit security strengths".

Notice, e.g., that NIST asks whether P80 "actually" meets its "claimed" security category, while NIST does not ask the same question regarding P78 or P71.

If NIST were applying the "classical gates" metric then none of P80, P78, and P71 would be able to confidently claim these categories. For example, the uncertainties in Core-SVP seem very unlikely to turn Core-SVP $2^{100}$ into $2^{143}$ "classical gates". Most of the remaining lattice submissions (at least Dilithium, NTRU, Kyber, and NTRU Prime; perhaps also SABER after the announcement that SABER's security levels were miscalculated) would have to adjust their category assignments. Even worse, some of these submissions (at least Dilithium, NTRU, and Kyber) would have to remove some previously proposed parameters, which seems contrary to the idea of being ready for standardization.

All of these submissions argue, with varying levels of detail and references, that the "classical gates" metric underestimates the actual cost of known attacks. NIST seems receptive to the idea of using a more realistic metric, but has taken four years to post its "preliminary thoughts" on the realism of several different metrics. It is not clear what metrics NIST will end up defining, and it is not clear how long NIST will take to settle on the definitions. What is clear is that NIST has not applied the categories consistently, as illustrated by NIST IR 8309 assigning P80 more negative wording than P78 and P71.

The different wording regarding P80, P78, and P71 appears to have translated into different action, and this seems particularly important for NIST's handling of NTRU Prime. As context, NIST IR 8309 describes finalists in general as the most promising to fit the majority of use cases and most likely to be ready for standardization soon after the end of the third round.

We have shown that NTRU Prime fits practically all use cases. As far as we can tell, beyond general concerns about the safety of lattice-based cryptography and about the safety of all small lattice proposals, NTRU Prime is ready for standardization now with our existing parameter sets. The only negative comments that NIST IR 8309 made regarding NTRU Prime were regarding parameter sets. Specifically, NIST seemed to criticize

* NTRU Prime's assignment of pre-quantum Core-SVP $2^{153}$ to Category 3 (this is exactly P80 above),

* NTRU Prime's assignment of post-quantum Core-SVP $2^{159}$ (pre-quantum Core-SVP $2^{175}$) to Category 4 (this has a larger security margin than P80),

* NTRU Prime's assignment of post-quantum Core-SVP $2^{117}$ (pre-quantum Core-SVP $2^{129}$) to Category 2 (this has a larger security margin than P80), and
NTRU Prime’s "narrower range of CoreSVP values" (our understanding now is that this wasn't a negative comment but merely a request for larger parameters going forward).

Meanwhile various lattice submissions with objectively more dangerous parameter selections were given less critical wording by NIST and were selected as finalists. We see no explanation for why NIST treated P78 and P71 in those submissions more gently than P80 in NTRU Prime.

An application limited to 1024 bytes for keys and plaintexts reaches Core-SVP $2^{129}$ with NTRU Prime's proposed parameters and nothing better than $2^{111}$ with Kyber's proposed parameters. $2^{129}$ is higher security relative to category 2 than $2^{111}$ relative to category 1, and obviously higher security on an absolute scale. NIST's report did not acknowledge this security advantage of NTRU Prime.

We are concerned that the lack of clear, stable, consistently applied category definitions will be used in the continuation of NISTPQC to again make NTRU Prime's parameter choices artificially sound worse than more dangerous parameter choices in other submissions. If we try to reduce this risk by downgrading (e.g.) $2^{129}$ to category 1, while Kyber is allowed to remain in category 1 with just $2^{111}$, then NTRU Prime will be unfairly punished in performance comparisons.

We request that NIST issue clear and stable definitions of the metrics used to define NIST's security categories. At this point in the NISTPQC process, clarity and stability are more important than the exact level of realism. Beyond the floor for the categories, one can reasonably argue that users should take higher Core-SVP levels for all lattice submissions in light of continued advances in lattice attacks; but NIST should handle this in a way that is fair to all submissions. As soon as the evaluation criteria are made clear, we will be happy to adjust our category assignments accordingly.

---Dan
This message has three questions for NIST.

In https://www.youtube.com/watch?v=CBGX1OMzN1o&t=37m55s a few days ago, NIST stated "We're we're still uh have some some questions about NTRU Prime" but didn't elaborate. What are NIST's questions about NTRU Prime?

The late notification and lack of information are problematic. NIST has asked for round-3 tweaks by 1 October, which is just a few days from now. Did I miss some NIST publication listing NIST's questions?

I see only one part of NIST IR 8309 that can be understood as a question about NTRU Prime: namely, "whether they actually meet their claimed security categories will need to be determined" regarding the parameter choices. Meanwhile NIST did not ask the same question regarding other submissions that have objectively more dangerous parameter selections.

The NTRU Prime team email dated 21 Sep 2020 11:49:53 +0200 gave examples of this phenomenon. Procedurally, seeing the issue raised by NIST was a prerequisite for responding to it (and the difficulty of responding was exacerbated by the lack of clarity regarding NIST's "categories").

This section of the talk appeared to be presenting NIST's rationale for selecting lattice finalists and lattice alternates. It would thus seem that the existence of NIST's "questions" regarding NTRU Prime played a role in this. Why didn't NIST IR 8309 say this and provide the list of questions?

---Dan (speaking for myself)
Dear Dan,

Thanks for your last two official comments on NTRUprime. We'll try to address your questions.

As noted, in my talk at PQCrypto I mentioned that NIST had some questions regarding NTRUprime. In the chat which followed the session, we were asked the same question you asked, i.e. what questions do we have about NTRUprime? We answered that they mainly deal with algebraic cryptanalysis of lattice KEMS which exploit the structure of the ring that they choose, as well as what we wrote in our write-up of NTRUprime in NISTIR 8309. In the chat we also had a few more comments. You were active on the chat, so we assumed you saw our answers. I wasn't trying to bring up anything last-minute. Indeed, some of this has been discussed via email between NIST and NTRUprime back in August. We will expand a bit more below.

While your comment (from Sept 21) focuses on the parametrization of NTRUprime, this was not a major reason for assigning NTRUprime as an alternate rather than a finalist. In particular, we felt that given known facts regarding lattice cryptanalysis, the cyclotomic structures used by the lattice finalists are more researched, and should by default, be considered better understood. In our view, the submitters of NTRUprime have voiced concerns about cyclotomic lattices, but have not proposed any concrete attack against them or any strong argument that NTRUprime's ring choice would rule out a similar attack. We are open to the possibility that there is such an attack, as we are open to the possibility that NTRUprime's ring choice is susceptible to an attack from which cyclotomics are immune. But, without a strong argument that such an attack is ruled out by NTRUprime's ring choice (this seems to us difficult to provide without seeing the hypothetical attack first,) any attack on cyclotomics would undermine our confidence in structured lattices overall. As such, NTRUprime's most probable path to standardization involves two major research results.

1) An attack undermining NIST's confidence in the choice of algebraic structures used by Kyber, NTRU, and Saber

2) The establishment of a strong theoretical barrier for attacks to be extended to the NTRUprime ring.

As the timeline for this path to standardization is by necessity longer than likely paths to standardization for the other finalists, after much deliberation and debate we classified NTRUprime as an alternate. On rereading the report, I realize we did not make this reasoning as spelled-out as it could have been, and for that we apologize.

It should also be noted that the sheer number of structured lattice KEMs in the process always meant we would have to make hard cuts in choosing finalists and alternates. In contrast, as an alternate, NTRUprime is still very much in the process. The same can not be said for other very strong submissions in the structured lattice KEM category that were eliminated from consideration at the end of the 1st and 2nd round of our process.

The fact that there is a lot of discussion in the report about parameter choices and security levels is not because it has factored heavily into our selections thus far, but rather with the intent that, going forward, all submissions in the 3rd round will have the opportunity, through tweaks, to minimize the chances that we choose not to standardize a scheme that we would have, had they chosen other parameters. The issue that the range of parameters offered by the various lattice submissions is highly variable when assigned to NIST security categories was in fact raised on the pqc-forum late in our selection process by you in late June. This was after we had already settled on our list of finalists and alternates,
but before our 2nd round report was published. We agreed that it could cause problems going forward, so we highlighted it in our report. The wordings we used varied for substantive reasons (e.g. NTRU's choice to highlight two different computational cost models, and the fact that failing to meet category 1 would result in us discarding a parameter set, while failing to meet a higher level could simply mean changing how it is labeled in our standard.) But the wording also varied due to the personal idiosyncrasies of the writing styles the 13 different authors of the NIST report. If you think subtle differences in wording are any indication of how the NIST team as a whole will evaluate the candidates at the end of the 3rd round, 12 to 18 months from now, independent of any tweaks made by the submitters, and any subsequent analysis that may come to light, you are reading too much into the tone here.

Finally, you request further clarification than we have already given regarding how we intend to assign security levels. We feel we have given about as much as we can give without prejudging scientific questions to which we do not know the answer. We have already stated that gate count in the RAM model exceeding that of brute force AES key search or brute force SHA collision search as appropriate will be taken as strong a priori evidence of meeting any of the NIST categories. If you think more aggressive parameters than are justified via the RAM model are appropriate for meeting a given security target, you will need to make a strong argument that convinces us. We've given reasons why previous arguments have not completely convinced us, so you should take that into account. We think there is a real possibility such arguments could convince us -- we just haven't been completely convinced by any yet. Likewise, while variations in wording about how we raise questions about a particular parameter set are likely to be uninformative, we think we were fairly consistent in which lattice parameter sets we did and did not raise concerns about when claiming each security strength category. Parameter sets we didn't raise concerns about are probably ok for their claimed security levels, barring developments in cryptanalysis. If you think we've missed something in this regard, please let us know, because it likely indicates a relevant research result we were not aware of at the time of writing our report.

In any event, we think the public statements we've made on the forum and in our report are sufficient for any submission team working in good faith to determine what parameter sets will be uncontroversial, controversial and unacceptable for the claimed security levels given the current state of knowledge. Keep in mind that extra care is warranted for the lowest (and perhaps highest) security levels offered by a submission. For example, while a controversial assignment of category 1 runs the risk of the parameter set in question not being standardized, a controversial assignment of category 3 probably just runs the risk of the parameter set in question being downgraded to category 2, or at worst, category 1. Stakeholders who care about category 3 can then take our assignment into account when deciding which NIST standardized parameter set to use. A controversial assessment of category 5 runs the risk that the submission will not meet the needs of any users who actually want category 5. We reiterate what we've been saying since the beginning of the PQC process that, barring future cryptanalysis, category 1 parameters are probably enough to thwart any purely computational attack in the near to medium term future, and category 3 is almost certainly enough.

We do try to respond promptly to questions brought to us, but it does take a little bit of time for our team to discuss and draft a response. The updated specifications and implementations are due on October 1st. We've tried to be flexible throughout the process. Teams can always contact us to ask for a bit more time if they feel they need it.

Dustin and Ray
The NIST PQC team
Two years ago an NTRU Prime update talk announced the new "factored" NTRU Prime software, a wrapper around "modules with separate tests and optimizations". I gave a talk today announcing, among other things, computer verification for most of these modules that the existing "avx" implementation produces the same output as the existing "ref" implementation for _all_ possible inputs:

https://cr.yp.to/talks.html#2021.09.03

This is a big step towards full verification of the optimized NTRU Prime software. Next steps are matching "avx" to "ref" for the other modules (notably multiplication, where another tool has gotten through some important parts of the code but not yet everything) and matching the C code to the Sage reference code.

The "saferewrite" tool used for this verification has a broad range of applicability beyond NTRU Prime. The first example in the talk is how saferewrite catches both of the array-comparison problems that were announced in the official Frodo software. However, to enable this analysis, I had to define a Frodo array-comparison module and write reference code for that module, so that saferewrite could compare the official code to my reference code. This wasn't a big deal since this particular module is so simple, but an analogous analysis for larger components of Frodo (short of taking the entire KEM as a monolith!) would require additional work to write reference code for those modules.

For NTRU Prime, this work is already done.

For each module, saferewrite compiles each implementation with clang -O1 and with gcc -O3, uses the angr symbolic-execution toolkit to convert each binary into unrolled code in a much simpler language, and uses the Z3 theorem prover (via angr's claripy) to verify equivalence or find a counterexample. The automatic equivalence chains look like this (although this pattern isn't optimal in general):

```
opt clang -O1 = ref clang -O1 = avx clang -O1
||
opt gcc -O3 = ref gcc -O3 = avx gcc -O3
```

There could be compiler bugs affecting outputs, but to evade detection these bugs would have to have the same effect on every node in the diagram simultaneously. (It would also be possible to hook a direct Python-to-the-simpler-language conversion into the picture.) There could be unrolling bugs, but saferewrite also runs the binaries on some random inputs (plus all-0 and all-1 to make sure every bit is touched) and checks that these match the unrolled code; also, angr has been heavily exercised in a variety of reverse-engineering applications. There could be bugs in saferewrite itself, but reviewing saferewrite is a much smaller task than reviewing a ton of optimized post-quantum code.

The examples supplied in the saferewrite package include deliberately buggy code to exercise saferewrite's tests, in particular producing 16 analyses printing "differentfrom" counterexamples (which I've checked by hand), providing some evidence that saferewrite is working as desired. Some of these bugs are also found by random tests, but some aren't. More advanced fuzzing can do better than random tests but has no hope of finding typical cryptographic overflow bugs.
Seeing C code working with two compilers doesn't mean that the same code will work with further compilers, but if analyses are fast enough then it's realistic to re-apply the analyses whenever the compiler changes. I tried the saferewrite analysis of 107 implementations of 27 functions, times 2 compilers, on a dual EPYC 7742; it finished in 8 minutes of wall-clock time, using 20 cores on average, using under 200GB of RAM.

I'm filing this as an OFFICIAL COMMENT regarding NTRU Prime because it's directly relevant to the official NISTPQC evaluation criteria, notably the following:

The algorithms can be implemented securely and efficiently on a wide variety of platforms, including constrained environments, such as smart cards.

Various NISTPQC submissions have provided fast AVX2 software, fast M4 software, etc., but the primary evidence for "securely" is that the software is constant-time. (I'll skip discussion of the broken masked implementations.) The problem is that the same optimizations add massive complexity to the software, and this complexity is a security threat:

* [https://arxiv.org/abs/2107.04940](https://arxiv.org/abs/2107.04940) studied the vulnerabilities announced between 2010 and 2020 in eight well-known cryptographic libraries, and found 73 vulnerabilities in the cryptographic computations, including 11 known to be exploitable ("severe"), along with "evidence of a strong correlation between the complexity of these libraries and their (in)security". (There were also hundreds of further bugs, such as buffer overflows.)

* Post-quantum software is newer, more complicated, and much harder to thoroughly review. Superficial reviews of post-quantum software have caught one devastating bug after another; the only reasonable prediction is that more serious reviews will find many more bugs.

Is it reasonable to say that an algorithm "can be implemented securely and efficiently" if fast implementations are so complex that the experts are getting them wrong? If the answer is pointing to an implementation and saying "No bugs are known in this implementation", then why should we think that the code is correct, rather than thinking that security reviewers are overloaded and that this answer is pure selection bias?

There's stronger evidence for "securely and efficiently" when optimized modules are verified to match much simpler reference implementations. Covering more modules will further strengthen this evidence.

Another relevant NISTPQC evaluation criterion is the following:

Factors that might hinder or promote widespread adoption of an algorithm or implementation will be considered in the evaluation process, including, but not limited to, ...

The availability of modularized implementations, and the availability of verification tools applicable to some of those modules, certainly help promote widespread adoption of those implementations and the algorithm.

---Dan (speaking for myself)
Thank you for this, Dan - really interesting. I am hopeful of having something to share in the next couple of months on my own verification and validation efforts with Classic McEliece using SAW/Cryptol, and certainly much of your thinking here seems to align with conclusions and thoughts I've reached.

(Increasingly I think a hybrid approach, using multiple toolchains that complement each other's weaknesses, with a clear analysis and synthesis of the truth-claims that each makes. is the way forward - though this mean, of course, you have more tools that you need to validate and verify...)

- Wrenna

On Fri, 3 Sept 2021 at 21:53, D. J. Bernstein <djb@cr.yp.to> wrote:

Two years ago an NTRU Prime update talk announced the new "factored" NTRU Prime software, a wrapper around "modules with separate tests and optimizations". I gave a talk today announcing, among other things, computer verification for most of these modules that the existing "avx" implementation produces the same output as the existing "ref" implementation for _all_ possible inputs:

https://cr.yp.to/talks.html#2021.09.03

This is a big step towards full verification of the optimized NTRU Prime software. Next steps are matching "avx" to "ref" for the other modules (notably multiplication, where another tool has gotten through some important parts of the code but not yet everything) and matching the C code to the Sage reference code.

The "saferewrite" tool used for this verification has a broad range of applicability beyond NTRU Prime. The first example in the talk is how saferewrite catches both of the array-comparison problems that were announced in the official Frodo software. However, to enable this analysis, I had to define a Frodo array-comparison module and write reference code for that module, so that saferewrite could compare the official code to my reference code. This wasn't a big deal since this particular module is so simple, but an analogous analysis for larger components of Frodo (short of taking the entire KEM as a monolith!) would require additional work to write reference code for those modules. For NTRU Prime, this work is already done.

For each module, saferewrite compiles each implementation with clang -O1 and with gcc -O3, uses the angr symbolic-execution toolkit to convert each binary into unrolled code in a much simpler language, and uses the Z3 theorem prover (via angr's claripy) to verify equivalence or find a counterexample. The automatic equivalence chains look like this