Comparing proofs of security for lattice-based encryption

Daniel J. Bernstein

Primary objective of this paper:
Make a **complete plan**
for **thorough security reviews**
of 36 target KEMs.

Much harder: Do the reviews!
Complete plan is framework
to evaluate which pieces are done,
and to coordinate further efforts.
KEMs vary in what's needed.

The target KEMs (all proposed for wide deployment, IND-CCA2):

`frodo`	`640`, `976`, `1344`.
`kyber`	`512`, `768`, `1024`.
`lac`	`128`, `192`, `256`.
`newhope`	`512`, `1024`.
`ntru`	`hps2048509`, `hps2048677`, `hps4096821`, `hrss701`.
`ntrulpr`	`653`, `761`, `857`.
`round5n1`	`1`, `3`, `5`.
`round5nd`	`1.0d`, `3.0d`, `5.0d`, `1.5d`, `3.5d`, `5.5d`.
`saber`	`light`, `main`, `fire`.
`sntrup`	`653`, `761`, `857`.
`threebears`	`baby`, `mama`, `papa`.

ng proofs of security
ce-based encryption

. Bernstein

---

objective of this paper:
complete plan
ough security reviews
rget KEMs.

arder: Do the reviews!
te plan is framework
ate which pieces are done,
coordinate further efforts.
ary in what's needed.

The target KEMs (all proposed for wide deployment, IND-CCA2):

`frodo`	640, 976, 1344.
`kyber`	512, 768, 1024.
`lac`	128, 192, 256.
`newhope`	512, 1024.
`ntru`	`hps2048509`, `hps2048677`, `hps4096821`, `hrss701`.
`ntrulpr`	653, 761, 857.
`round5n1`	1, 3, 5.
`round5nd`	`1.0d`, `3.0d`, `5.0d`, `1.5d`, `3.5d`, `5.5d`.
`saber`	`light`, `main`, `fire`.
`sntrup`	653, 761, 857.
`threebears`	`baby`, `mama`, `papa`.

One cat

`frodo`
`kyber`
`lac`
`newhope`
`ntru`
`ntrulpr`
`round5n`
`round5n`
`saber`
`sntrup`
`threebe`

of security
ncryption

n

---

of this paper:
plan
urity reviews
s.

the reviews!
framework
pieces are done,
further efforts.
t's needed.

1

The target KEMs (all proposed for wide deployment, IND-CCA2):

frodo	640, 976, 1344.
kyber	512, 768, 1024.
lac	128, 192, 256.
newhope	512, 1024.
ntru	hps2048509, hps2048677, hps4096821, hrss701.
ntrulpr	653, 761, 857.
round5n1	1, 3, 5.
round5nd	1.0d, 3.0d, 5.0d, 1.5d, 3.5d, 5.5d.
saber	light, main, fire.
sntrup	653, 761, 857.
threebears	baby, mama, papa.

2

One categorization

frodo
kyber
lac
newhope
ntru
ntrulpr
round5n1
round5nd
saber
sntrup
threebears

y

per:

ews

ws!

done,
forts.
d.

1

The target KEMs (all proposed for wide deployment, IND-CCA2):

frodo	640, 976, 1344.
kyber	512, 768, 1024.
lac	128, 192, 256.
newhope	512, 1024.
ntru	hps2048509, hps2048677, hps4096821, hrss701.
ntrulpr	653, 761, 857.
round5n1	1, 3, 5.
round5nd	1.0d, 3.0d, 5.0d, 1.5d, 3.5d, 5.5d.
saber	light, main, fire.
sntrup	653, 761, 857.
threebears	baby, mama, papa.

2

One categorization of the K

frodo	Product N
kyber	Product N
lac	Product N
newhope	Product N
ntru	Quotient N
ntrulpr	Product N
round5n1	Product N
round5nd	Product N
saber	Product N
sntrup	Quotient N
threebears	Product N

The target KEMs (all proposed for wide deployment, IND-CCA2):

frodo	640, 976, 1344.
kyber	512, 768, 1024.
lac	128, 192, 256.
newhope	512, 1024.
ntru	hps2048509, hps2048677, hps4096821, hrss701.
ntrulpr	653, 761, 857.
round5n1	1, 3, 5.
round5nd	1.0d, 3.0d, 5.0d, 1.5d, 3.5d, 5.5d.
saber	light, main, fire.
sntrup	653, 761, 857.
threebears	baby, mama, papa.

One categorization of the KEMs:

frodo	Product NTRU.
kyber	Product NTRU.
lac	Product NTRU.
newhope	Product NTRU.
ntru	Quotient NTRU.
ntrulpr	Product NTRU.
round5n1	Product NTRU.
round5nd	Product NTRU.
saber	Product NTRU.
sntrup	Quotient NTRU.
threebears	Product NTRU.

get KEMs (all proposed
deployment, IND-CCA2):

	640, 976, 1344.
	512, 768, 1024.
	128, 192, 256.
e	512, 1024.
ps2048509,	hps2048677,
hps4096821,	hrss701.
r	653, 761, 857.
n1	1, 3, 5.
nd	1.0d, 3.0d, 5.0d,
	1.5d, 3.5d, 5.5d.
	light, main, fire.
	653, 761, 857.
ears	baby, mama, papa.

One categorization of the KEMs:

frodo	Product NTRU.
kyber	Product NTRU.
lac	Product NTRU.
newhope	Product NTRU.
ntru	Quotient NTRU.
ntrulpr	Product NTRU.
round5n1	Product NTRU.
round5nd	Product NTRU.
saber	Product NTRU.
sntrup	Quotient NTRU.
threebears	Product NTRU.

An overs

Plan: Ve
make su

(all proposed
nt, IND-CCA2):
640, 976, 1344.
512, 768, 1024.
128, 192, 256.
512, 1024.
09, hps2048677,
96821, hrss701.
653, 761, 857.
1, 3, 5.
0d, 3.0d, 5.0d,
5d, 3.5d, 5.5d.
ght, main, fire.
653, 761, 857.
aby, mama, papa.

One categorization of the KEMs:

frodo	Product NTRU.
kyber	Product NTRU.
lac	Product NTRU.
newhope	Product NTRU.
ntru	Quotient NTRU.
ntrulpr	Product NTRU.
round5n1	Product NTRU.
round5nd	Product NTRU.
saber	Product NTRU.
sntrup	Quotient NTRU.
threebears	Product NTRU.

An oversimplified

Plan: Verify the s
make sure there a

sed
CA2):
1344.
1024.
2, 256.
1024.
48677,
ss701.
1, 857.
1, 3, 5.
5.0d,
5.5d.
fire.
1, 857.
papa.

One categorization of the KEMs:

`frodo`	Product NTRU.
`kyber`	Product NTRU.
`lac`	Product NTRU.
`newhope`	Product NTRU.
`ntru`	Quotient NTRU.
`ntrulpr`	Product NTRU.
`round5n1`	Product NTRU.
`round5nd`	Product NTRU.
`saber`	Product NTRU.
`sntrup`	Quotient NTRU.
`threebears`	Product NTRU.

An oversimplified plan

Plan: Verify the security pro
make sure there are no mist

One categorization of the KEMs:

`frodo`	Product NTRU.
`kyber`	Product NTRU.
`lac`	Product NTRU.
`newhope`	Product NTRU.
`ntru`	Quotient NTRU.
`ntrulpr`	Product NTRU.
`round5n1`	Product NTRU.
`round5nd`	Product NTRU.
`saber`	Product NTRU.
`sntrup`	Quotient NTRU.
`threebears`	Product NTRU.

An oversimplified plan

Plan: Verify the security proofs—make sure there are no mistakes.

One categorization of the KEMs:

`frodo`	Product NTRU.
`kyber`	Product NTRU.
`lac`	Product NTRU.
`newhope`	Product NTRU.
`ntru`	Quotient NTRU.
`ntrulpr`	Product NTRU.
`round5n1`	Product NTRU.
`round5nd`	Product NTRU.
`saber`	Product NTRU.
`sntrup`	Quotient NTRU.
`threebears`	Product NTRU.

An oversimplified plan

Plan: Verify the security proofs—make sure there are no mistakes.

Why verification is important: e.g., Asiacrypt 2004 Rogaway "OCB2" was standardized in 2009, completely broken in 2018. The attack exploited proof error.

One categorization of the KEMs:

`frodo`	Product NTRU.
`kyber`	Product NTRU.
`lac`	Product NTRU.
`newhope`	Product NTRU.
`ntru`	Quotient NTRU.
`ntrulpr`	Product NTRU.
`round5n1`	Product NTRU.
`round5nd`	Product NTRU.
`saber`	Product NTRU.
`sntrup`	Quotient NTRU.
`threebears`	Product NTRU.

An oversimplified plan

Plan: Verify the security proofs—make sure there are no mistakes.

Why verification is important: e.g., Asiacrypt 2004 Rogaway "OCB2" was standardized in 2009, completely broken in 2018. The attack exploited proof error.

I did some sanity checks (*tiny* part of full verification!) and found unproven theorems claimed by `frodo`, `round5n1`, `round5nd`, `saber`; also wrong hypotheses for `newhope` theorem.

egorization of the KEMs:

	Product NTRU.
	Product NTRU.
	Product NTRU.
e	Product NTRU.
	Quotient NTRU.
r	Product NTRU.
n1	Product NTRU.
nd	Product NTRU.
	Product NTRU.
	Quotient NTRU.
ears	Product NTRU.

An oversimplified plan

Plan: Verify the security proofs—make sure there are no mistakes.

Why verification is important: e.g., Asiacrypt 2004 Rogaway "OCB2" was standardized in 2009, completely broken in 2018. The attack exploited proof error.

I did some sanity checks (*tiny* part of full verification!) and found unproven theorems claimed by `frodo`, `round5n1`, `round5nd`, `saber`; also wrong hypotheses for `newhope` theorem.

Strategy
explain a
to a tho
that con

n of the KEMs:

Product NTRU.
Product NTRU.
Product NTRU.
Product NTRU.
Quotient NTRU.

Product NTRU.
Product NTRU.
Product NTRU.

Product NTRU.
Quotient NTRU.
Product NTRU.

An oversimplified plan

Plan: Verify the security proofs—make sure there are no mistakes.

Why verification is important: e.g., Asiacrypt 2004 Rogaway "OCB2" was standardized in 2009, completely broken in 2018. The attack exploited proof error.

I did some sanity checks (*tiny* part of full verification!) and found unproven theorems claimed by `frodo`, `round5n1`, `round5nd`, `saber`; also wrong hypotheses for `newhope` theorem.

Strategy to elimin
explain all of the t
to a thoroughly au
that completely ve

EMs:

NTRU.
NTRU.
NTRU.
NTRU.
NTRU.
NTRU.

NTRU.
NTRU.
NTRU.

NTRU.
NTRU.
NTRU.

An oversimplified plan

Plan: Verify the security proofs—make sure there are no mistakes.

Why verification is important: e.g., Asiacrypt 2004 Rogaway "OCB2" was standardized in 2009, completely broken in 2018. The attack exploited proof error.

I did some sanity checks (*tiny* part of full verification!) and found unproven theorems claimed by `frodo`, `round5n1`, `round5nd`, `saber`; also wrong hypotheses for `newhope` theorem.

Strategy to eliminate proof
explain all of the target proc
to a thoroughly audited prog
that completely verifies proc

An oversimplified plan

Plan: Verify the security proofs—
make sure there are no mistakes.

Why verification is important:
e.g., Asiacrypt 2004 Rogaway
"OCB2" was standardized in
2009, completely broken in 2018.
The attack exploited proof error.

I did some sanity checks
(*tiny* part of full verification!)
and found unproven theorems
claimed by `frodo`, `round5n1`,
`round5nd`, `saber`; also wrong
hypotheses for `newhope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

An oversimplified plan

Plan: Verify the security proofs—
make sure there are no mistakes.

Why verification is important:
e.g., Asiacrypt 2004 Rogaway
"OCB2" was standardized in
2009, completely broken in 2018.
The attack exploited proof error.

I did some sanity checks
(*tiny* part of full verification!)
and found unproven theorems
claimed by `frodo`, `round5n1`,
`round5nd`, `saber`; also wrong
hypotheses for `newhope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.

An oversimplified plan

Plan: Verify the security proofs—
make sure there are no mistakes.

Why verification is important:
e.g., Asiacrypt 2004 Rogaway
"OCB2" was standardized in
2009, completely broken in 2018.
The attack exploited proof error.

I did some sanity checks
(*tiny* part of full verification!)
and found unproven theorems
claimed by `frodo`, `round5n1`,
`round5nd`, `saber`; also wrong
hypotheses for `newhope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?

An oversimplified plan

Plan: Verify the security proofs—
make sure there are no mistakes.

Why verification is important:
e.g., Asiacrypt 2004 Rogaway
"OCB2" was standardized in
2009, completely broken in 2018.
The attack exploited proof error.

I did some sanity checks
(*tiny* part of full verification!)
and found unproven theorems
claimed by `frodo`, `round5n1`,
`round5nd`, `saber`; also wrong
hypotheses for `newhope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

An oversimplified plan

Plan: Verify the security proofs—
make sure there are no mistakes.

Why verification is important:
e.g., Asiacrypt 2004 Rogaway
"OCB2" was standardized in
2009, completely broken in 2018.
The attack exploited proof error.

I did some sanity checks
(*tiny* part of full verification!)
and found unproven theorems
claimed by `frodo`, `round5n1`,
`round5nd`, `saber`; also wrong
hypotheses for `newhope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

simplified plan

erify the security proofs—
re there are no mistakes.

rification is important:
acrypt 2004 Rogaway
was standardized in
ompletely broken in 2018.
ack exploited proof error.

me sanity checks
rt of full verification!)
nd unproven theorems
by `frodo`, `round5n1`,
nd, `saber`; also wrong
ses for `newhope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

Why cal

What "s
is not ac

plan

ecurity proofs—
re no mistakes.

s important:
04 Rogaway
dardized in
broken in 2018.
ted proof error.

checks
erification!)
en theorems
`round5n1`,
; also wrong
`whope` theorem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

Why call this "ove

What "security pr
is not actually sec

oofs—
akes.

t:
ay
n
2018.
error.

!)
ns
1,
ng
orem.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

Why call this "oversimplified

What "security proofs" prov
is not actually security.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Even with correct proofs,
there are still risks of attacks.
We all rely on cryptanalysis
for analyzing remaining risks.

Strategy to eliminate proof errors:
explain all of the target proofs
to a thoroughly audited program
that completely verifies proofs.

My assessment of this strategy:

- Status today: ≈0% completed.
- Progress is painful and slow.
  Will we even reach 1% before
  post-quantum standardization?
- Easier-to-use proof tools
  could make strategy work.

Backup strategies: Clean up
proofs. Check proofs by hand.
Track bug categories, as in code.

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Even with correct proofs,
there are still risks of attacks.
We all rely on cryptanalysis
for analyzing remaining risks.

Revised plan:

1. Verify the "security proofs".
2. Verify the cryptanalysis
of the risks left by the proofs.

Again clean up; check by hand;
track failure categories.

y to eliminate proof errors:
all of the target proofs
roughly audited program
mpletely verifies proofs.

ssment of this strategy:
s today: ≈0% completed.
ess is painful and slow.
ve even reach 1% before
uantum standardization?
-to-use proof tools
make strategy work.

strategies: Clean up
Check proofs by hand.
ug categories, as in code.

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Even with correct proofs,
there are still risks of attacks.
We all rely on cryptanalysis
for analyzing remaining risks.

Revised plan:

1. Verify the "security proofs".
2. Verify the cryptanalysis
of the risks left by the proofs.

Again clean up; check by hand;
track failure categories.

Are atta
How tho
of space
How tho
claimed
that wor
Do the c
match th

Long his
NSA ove
$L(1/2)$ c
for facto
TLS Trip
without

ate proof errors:
target proofs
udited program
erifies proofs.

this strategy:
0% completed.
ful and slow.
ach 1% before
tandardization?
oof tools
tegy work.

Clean up
ofs by hand.
ies, as in code.

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Even with correct proofs,
there are still risks of attacks.
We all rely on cryptanalysis
for analyzing remaining risks.

Revised plan:

1. Verify the "security proofs".
2. Verify the cryptanalysis
of the risks left by the proofs.

Again clean up; check by hand;
track failure categories.

Are attack-cost ar
How thorough is e
of space of optimi
How thorough is t
claimed barriers to
that work for simil
Do the cryptanaly
match the proof ri

Long history of fai
NSA overstated D
$L(1/2)$ optimality
for factorization w
TLS Triple-DES-C
without Triple-DE

errors:
ofs
gram
ofs.

egy:
leted.
w.
fore
tion?

d.
code.

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Even with correct proofs,
there are still risks of attacks.
We all rely on cryptanalysis
for analyzing remaining risks.

Revised plan:

1. Verify the "security proofs".
2. Verify the cryptanalysis
of the risks left by the proofs.

Again clean up; check by hand;
track failure categories.

Are attack-cost analyses cor
How thorough is exploration
of space of optimizations?
How thorough is the study o
claimed barriers to speedups
that work for similar problem
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.
NSA overstated DES attack
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was b
without Triple-DES attack;

Why call this "oversimplified"?

What "security proofs" prove
is not actually security.

Even with correct proofs,
there are still risks of attacks.
We all rely on cryptanalysis
for analyzing remaining risks.

Revised plan:
1. Verify the "security proofs".
2. Verify the cryptanalysis
of the risks left by the proofs.

Again clean up; check by hand;
track failure categories.

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

l this "oversimplified"?

security proofs" prove
ctually security.

th correct proofs,
e still risks of attacks.
ely on cryptanalysis
yzing remaining risks.

plan:
y the "security proofs".
y the cryptanalysis
sks left by the proofs.

ean up; check by hand;
lure categories.

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bot

Plan wit
Verify cr

ersimplified"?

oofs" prove
urity.

proofs,
s of attacks.
ptanalysis
ining risks.

urity proofs".
tanalysis
the proofs.

heck by hand;
ories.

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bother with

Plan without proo
Verify cryptanalysi

d”?

ve

s.

s.

fs”.

fs.

nd;

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bother with proofs?

Plan without proofs is simpl

Verify cryptanalysis of the K

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

Are attack-cost analyses correct?
How thorough is exploration
of space of optimizations?
How thorough is the study of
claimed barriers to speedups
that work for similar problems?
Do the cryptanalytic targets
match the proof risks? etc.

Long history of failures: e.g.,
NSA overstated DES attack cost;
$L(1/2)$ optimality conjecture
for factorization was wrong;
TLS Triple-DES-CBC was broken
without Triple-DES attack; etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

ck-cost analyses correct?
prough is exploration
of optimizations?
prough is the study of
barriers to speedups
rk for similar problems?
cryptanalytic targets
he proof risks? etc.

story of failures: e.g.,
erstated DES attack cost;
optimality conjecture
prization was wrong;
ple-DES-CBC was broken
Triple-DES attack; etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks no

A "secu
security
against
assumin
for unde

nalyses correct?
exploration
zations?
he study of
o speedups
lar problems?
tic targets
sks? etc.

lures: e.g.,
ES attack cost;
conjecture
as wrong;
BC was broken
S attack; etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled ou

A "security proof"
security level λ for
against all attacks
assuming security
for underlying prob

rect?

of

s

ms?

,

cost;

e

roken

etc.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled out by proof

A "security proof" guarante
security level λ for system X
against all attacks of type T
assuming security level λ'
for underlying problem P.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled out by proofs

A "security proof" guarantees
security level λ for system X
against all attacks of type T
assuming security level λ'
for underlying problem P.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled out by proofs

A "security proof" guarantees
security level λ for system X
against all attacks of type T
assuming security level λ'
for underlying problem P.

Risk #1: P does not reach
security level λ'.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled out by proofs

A "security proof" guarantees
security level λ for system X
against all attacks of type T
assuming security level λ'
for underlying problem P.

Risk #1: P does not reach
security level λ'.

Risk #2 (looseness): λ is below
claimed security level of X.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled out by proofs

A "security proof" guarantees
security level λ for system X
against all attacks of type T
assuming security level λ'
for underlying problem P.

Risk #1: P does not reach
security level λ'.

Risk #2 (looseness): λ is below
claimed security level of X.

Risk #3: There are faster
attacks outside type T.

Why bother with proofs?

Plan without proofs is simpler:
Verify cryptanalysis of the KEMs.

But sometimes the proofs
reduce cost of cryptanalysis.

Sometimes this outweighs
cost to verify proofs: reduces
cost of thorough security review.
Hopefully less chance of disaster.

This paper's verification plan
skips proofs that clearly fail
to reduce cost of cryptanalysis:
e.g., `frodo` seed "reduction".

Risks not ruled out by proofs

A "security proof" guarantees
security level λ for system X
against all attacks of type T
assuming security level λ'
for underlying problem P.

Risk #1: P does not reach
security level λ'.

Risk #2 (looseness): λ is below
claimed security level of X.

Risk #3: There are faster
attacks outside type T.

Risk #4: Proof is incorrect.

ther with proofs?

thout proofs is simpler:
ryptanalysis of the KEMs.

netimes the proofs
cost of cryptanalysis.

nes this outweighs
verify proofs: reduces
thorough security review.
y less chance of disaster.

ber's verification plan
oofs that clearly fail
e cost of cryptanalysis:
odo seed "reduction".

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets

Attack C
security

proofs?

fs is simpler:
is of the KEMs.

e proofs
ptanalysis.

tweighs
fs: reduces
ecurity review.
nce of disaster.

cation plan
clearly fail
cryptanalysis:
"reduction".

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice

Attack OW-Passiv
security of the 36

er:
KEMs.

es
view.
aster.

n

sis:
’
.

Risks not ruled out by proofs

A “security proof” guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanaly

Attack OW-Passive (“OW-C
security of the 36 core PKEs

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA") security of the 36 core PKEs.

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA") security of the 36 core PKEs.

For some targets: Attack IND-CPA security of core PKEs.

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA") security of the 36 core PKEs.

For some targets: Attack IND-CPA security of core PKEs.

For some targets: Attack pseudorandom multipliers.

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA") security of the 36 core PKEs.

For some targets: Attack IND-CPA security of core PKEs.

For some targets: Attack pseudorandom multipliers.

For some targets: KEM proofs are loose. Find faster attacks.

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA") security of the 36 core PKEs.

For some targets: Attack IND-CPA security of core PKEs.

For some targets: Attack pseudorandom multipliers.

For some targets: KEM proofs are loose. Find faster attacks. Also, some KEM "proofs" rely on unproven conjectures.

Risks not ruled out by proofs

A "security proof" guarantees security level λ for system X against all attacks of type T assuming security level λ' for underlying problem P.

Risk #1: P does not reach security level λ'.

Risk #2 (looseness): λ is below claimed security level of X.

Risk #3: There are faster attacks outside type T.

Risk #4: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA") security of the 36 core PKEs.

For some targets: Attack IND-CPA security of core PKEs.

For some targets: Attack pseudorandom multipliers.

For some targets: KEM proofs are loose. Find faster attacks. Also, some KEM "proofs" rely on unproven conjectures.

For all targets: KEM proofs allow non-ROM attacks.

t ruled out by proofs

rity proof" guarantees
level λ for system X
all attacks of type T
g security level λ'
rlying problem P.

: P does not reach
level λ'.

(looseness): λ is below
security level of X.

: There are faster
outside type T.

: Proof is incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core

Key gen

- Table
- Table
- Table

t by proofs

guarantees
r system X
of type T
level λ'
blem P.

not reach

s): λ is below
evel of X.

re faster
pe T.

incorrect.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core PKEs ("

Key generation:

- Table 8.6: Publi
- Table 8.7: Short
- Table 8.8: Publi

s

es
K
T

elow

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplie
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

Targets for lattice cryptanalysis

Attack OW-Passive ("OW-CPA")
security of the 36 core PKEs.

For some targets: Attack
IND-CPA security of core PKEs.

For some targets: Attack
pseudorandom multipliers.

For some targets: KEM proofs
are loose. Find faster attacks.
Also, some KEM "proofs"
rely on unproven conjectures.

For all targets: KEM proofs
allow non-ROM attacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

for lattice cryptanalysis

OW-Passive ("OW-CPA")
of the 36 core PKEs.

e targets: Attack
A security of core PKEs.

e targets: Attack
andom multipliers.

e targets: KEM proofs
e. Find faster attacks.
me KEM "proofs"
unproven conjectures.

argets: KEM proofs
n-ROM attacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

OW-Pas

Quotient
asks for
2003 Na

cryptanalysis

ve ("OW-CPA")
core PKEs.

Attack
of core PKEs.

Attack
ltipliers.

KEM proofs
ster attacks.
"proofs"
conjectures.

EM proofs
ttacks.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

OW-Passive vs. IN

Quotient NTRU (
asks for OW-Passi
2003 Naor: this is

ysis

CPA")

s.

KEs.

ofs

ks.

s.

The core PKEs ("*P*")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

OW-Passive vs. IND-CPA ("

Quotient NTRU (`ntru`, `snt`
asks for OW-Passive cryptan
2003 Naor: this is "falsifiabl

The core PKEs ("P")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`)
asks for OW-Passive cryptanalysis.
2003 Naor: this is "falsifiable".

The core PKEs ("P")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`)
asks for OW-Passive cryptanalysis.
2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and
systems not named after NTRU)
asks for IND-CPA cryptanalysis.
Lower security than OW-Passive?
Only "somewhat falsifiable".

The core PKEs ("P")

Key generation:

- Table 8.6: Public multiplier G.
- Table 8.7: Short secret a.
- Table 8.8: Public $A \approx aG$.

Encryption: Short secret b;
public ciphertext $B \approx Gb$
(or $B \approx Gb/3$ or $B \approx 3Gb$).

That's it for Quotient NTRU.

More for Product NTRU:

- Table 8.9: Public $C \approx Ab + M$.
- Table 8.10: Secret M.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`)
asks for OW-Passive cryptanalysis.
2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and
systems not named after NTRU)
asks for IND-CPA cryptanalysis.
Lower security than OW-Passive?
Only "somewhat falsifiable".

Compare 2006 Goldreich: "What
concerns us about" DDH is that
"DDH is less simple than DH"
making it "harder to evaluate."

e PKEs ("*P*")

eration:

8.6: Public multiplier G.

8.7: Short secret a.

8.8: Public $A \approx aG$.

on: Short secret b;

phertext $B \approx Gb$

$Gb/3$ or $B \approx 3Gb$).

t for Quotient NTRU.

r Product NTRU:

8.9: Public $C \approx Ab + M$.

8.10: Secret M.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis.
2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis.
Lower security than OW-Passive?
Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

Pseudora

Product

into PK

pseudora

P")

c multiplier G.

t secret a.

c $A \approx aG$.

secret b;

$B \approx Gb$

$B \approx 3Gb$).

ient NTRU.

NTRU:

c $C \approx Ab + M$.

ret M.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis. 2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis. Lower security than OW-Passive? Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

Pseudorandom mu

Product NTRU: c
into PKE that bui
pseudorandomly fr

er G.

.

J.

$+ M$.

11

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis. 2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis. Lower security than OW-Passive? Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

12

Pseudorandom multipliers ("

Product NTRU: convert cor
into PKE that builds multip
pseudorandomly from public

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis. 2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis. Lower security than OW-Passive? Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis. 2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis. Lower security than OW-Passive? Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis. 2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis. Lower security than OW-Passive? Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

I dispute this. Need non-ROM cryptanalysis for all these PKEs. Proofs cover only ROM attacks. Must modify theorem statements.

OW-Passive vs. IND-CPA ("dist")

Quotient NTRU (`ntru`, `sntrup`) asks for OW-Passive cryptanalysis. 2003 Naor: this is "falsifiable".

Product NTRU (`ntrulpr` and systems not named after NTRU) asks for IND-CPA cryptanalysis. Lower security than OW-Passive? Only "somewhat falsifiable".

Compare 2006 Goldreich: "What concerns us about" DDH is that "DDH is less simple than DH" making it "harder to evaluate."

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

I dispute this. Need non-ROM cryptanalysis for all these PKEs. Proofs cover only ROM attacks. Must modify theorem statements.

`frodo` seed "reduction": Useless. Still need non-ROM cryptanalysis.

sive vs. IND-CPA ("dist")

t NTRU (`ntru`, `sntrup`)
OW-Passive cryptanalysis.
or: this is "falsifiable".

NTRU (`ntrulpr` and
not named after NTRU)
IND-CPA cryptanalysis.
ecurity than OW-Passive?
omewhat falsifiable".

e 2006 Goldreich: "What
s us about" DDH is that
s less simple than DH"
it "harder to evaluate."

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

I dispute this. Need non-ROM cryptanalysis for all these PKEs. Proofs cover only ROM attacks. Must modify theorem statements.

`frodo` seed "reduction": Useless. Still need non-ROM cryptanalysis.

More ha

Want th
to provi

The pro
even ass
of the u
The pro
ROM IN

Issue for
and for

ID-CPA ("dist")

`ntru`, `sntrup`)
ve cryptanalysis.
"falsifiable".

`trulpr` and
d after NTRU)
cryptanalysis.
n OW-Passive?
"falsifiable".

ldreich: "What
" DDH is that
ole than DH"
to evaluate."

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

I dispute this. Need non-ROM cryptanalysis for all these PKEs. Proofs cover only ROM attacks. Must modify theorem statements.

`frodo` seed "reduction": Useless. Still need non-ROM cryptanalysis.

More hashing ("R

Want the target K
to provide IND-C

The proofs don't g
even assuming sec
of the underlying
The proofs are lim
ROM IND-CCA2 a

Issue for Product
and for Quotient

“dist”)

rup)
nalysis.
le”.

nd
TRU)
ysis.
ssive?
T
What
that
H”
te.”

Pseudorandom multipliers (“ROM2”)

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

I dispute this. Need non-ROM cryptanalysis for all these PKEs. Proofs cover only ROM attacks. Must modify theorem statements.

`frodo` seed “reduction”: Useless. Still need non-ROM cryptanalysis.

More hashing (“ROM”)

Want the target KEMs to provide IND-CCA2 securi

The proofs don't give this, even assuming security of the underlying PKEs. The proofs are limited to ROM IND-CCA2 attacks.

Issue for Product NTRU *and* for Quotient NTRU.

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE
into PKE that builds multiplier G
pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd`
claim that this provably preserves
security assuming PRG/PRF.

I dispute this. Need non-ROM
cryptanalysis for all these PKEs.
Proofs cover only ROM attacks.
Must modify theorem statements.

`frodo` seed "reduction": Useless.
Still need non-ROM cryptanalysis.

More hashing ("ROM")

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

Pseudorandom multipliers ("ROM2")

Product NTRU: convert core PKE into PKE that builds multiplier G pseudorandomly from public seed.

`saber`, `round5n1`, `round5nd` claim that this provably preserves security assuming PRG/PRF.

I dispute this. Need non-ROM cryptanalysis for all these PKEs. Proofs cover only ROM attacks. Must modify theorem statements.

`frodo` seed "reduction": Useless. Still need non-ROM cryptanalysis.

More hashing ("ROM")

Want the target KEMs to provide IND-CCA2 security.

The proofs don't give this, even assuming security of the underlying PKEs. The proofs are limited to ROM IND-CCA2 attacks.

Issue for Product NTRU *and* for Quotient NTRU.

For all target KEMs, need non-ROM IND-CCA2 cryptanalysis.

andom multipliers ("ROM2")

NTRU: convert core PKE
E that builds multiplier G
andomly from public seed.

`round5n1`, `round5nd`
at this provably preserves
assuming PRG/PRF.

e this. Need non-ROM
alysis for all these PKEs.
cover only ROM attacks.
odify theorem statements.

eed "reduction": Useless.
d non-ROM cryptanalysis.

More hashing ("ROM")

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

For all target KEMs, need non-
ROM IND-CCA2 cryptanalysis.

Decrypti

2017 Ho
proofs d
CCA2 at
even if t

Q: numb
δ: failur

ultipliers ("ROM2")

onvert core PKE
lds multiplier G
rom public seed.

, `round5nd`
ovably preserves
PRG/PRF.

ed non-ROM
ll these PKEs.
ROM attacks.
rem statements.

ction": Useless.
M cryptanalysis.

More hashing ("ROM")

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

For all target KEMs, need non-
ROM IND-CCA2 cryptanalysis.

Decryption failures

2017 Hofheinz–Hö
proofs do not rule
CCA2 attacks with
even if the PKEs a

Q: number of has
δ: failure probabili

“ROM2”)

e PKE
lier G
seed.

d
serves
F.

DM
KEs.
acks.
ments.

seless.
alysis.

More hashing (“ROM”)

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

For all target KEMs, need non-
ROM IND-CCA2 cryptanalysis.

Decryption failures (“fail” / “

2017 Hofheinz–Hövelmanns-
proofs do not rule out ROM
CCA2 attacks with probabili
even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

More hashing ("ROM")

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

For all target KEMs, need non-
ROM IND-CCA2 cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz
proofs do not rule out ROM IND-
CCA2 attacks with probability $Q\delta$,
even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

More hashing ("ROM")

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

For all target KEMs, need non-
ROM IND-CCA2 cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz
proofs do not rule out ROM IND-
CCA2 attacks with probability $Q\delta$,
even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs:
`ntru`, `ntrulpr`, `sntrup`. (Also,
simpler ROM IND-CCA2 proof.)

More hashing ("ROM")

Want the target KEMs
to provide IND-CCA2 security.

The proofs don't give this,
even assuming security
of the underlying PKEs.
The proofs are limited to
ROM IND-CCA2 attacks.

Issue for Product NTRU
and for Quotient NTRU.

For all target KEMs, need non-ROM IND-CCA2 cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz
proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$,
even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs:
`ntru`, `ntrulpr`, `sntrup`. (Also,
simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove
$\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

shing ("ROM")

e target KEMs
de IND-CCA2 security.

ofs don't give this,
uming security
nderlying PKEs.
ofs are limited to
ID-CCA2 attacks.

Product NTRU
Quotient NTRU.

arget KEMs, need non-
ID-CCA2 cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs: `ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The oth
Security
without
So need

OM")

KEMs
CA2 security.

give this,
urity
PKEs.
ited to
attacks.

NTRU
NTRU.

Ms, need non-
cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs:
`ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The other 23 KEM
Security goal 2^k
without proof that
So need CCA cryp

ty.

on-

sis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.

δ: failure probability.

$\delta = 0$ proven for 10 KEMs: `ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.

δ: failure probability.

$\delta = 0$ proven for 10 KEMs:

`ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The other 23 KEMs:

Security goal 2^k

without proof that $\delta \leq 2^{-k}$.

So need CCA cryptanalysis.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs:
`ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim that δ is small enough.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs:
`ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim that δ is small enough.
- 15 KEMs conjecture $\delta \leq \cdots$ without claiming proof.

Decryption failures ("fail" / "conj")

2017 Hofheinz–Hövelmanns–Kiltz proofs do not rule out ROM IND-CCA2 attacks with probability $Q\delta$, even if the PKEs are secure.

Q: number of hash calls.
δ: failure probability.

$\delta = 0$ proven for 10 KEMs:
`ntru`, `ntrulpr`, `sntrup`. (Also, simpler ROM IND-CCA2 proof.)

`frodo640`, `kyber512` prove $\delta \leq 2^{-128}$ with security goal 2^{128}.

`frodo976` proves $\delta \leq 2^{-192}$.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim that δ is small enough.
- 15 KEMs conjecture $\delta \leq \cdots$ without claiming proof.
- 5 KEMs have proofs but do not clearly use correct δ definition. (LEDA uses wrong definition.)

ion failures ("fail" / "conj")

ofheinz–Hövelmanns–Kiltz
o not rule out ROM IND-
ttacks with probability $Q\delta$,
he PKEs are secure.

ber of hash calls.
e probability.

roven for 10 KEMs:
`trulpr`, `sntrup`. (Also,
ROM IND-CCA2 proof.)

40, `kyber512` prove
^{28} with security goal 2^{128}.

76 proves $\delta \leq 2^{-192}$.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim
  that δ is small enough.
- 15 KEMs conjecture $\delta \leq \cdots$
  without claiming proof.
- 5 KEMs have proofs but do not
  clearly use correct δ definition.
  (LEDA uses wrong definition.)

What ab

Consider
for each

s (“fail” / “conj”)

övelmanns–Kiltz
out ROM IND-
n probability $Q\delta$,
are secure.

h calls.
ity.

10 KEMs:
ntrup. (Also,
-CCA2 proof.)

512 prove
curity goal 2^{128}.

$\delta \leq 2^{-192}$.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim
  that δ is small enough.
- 15 KEMs conjecture $\delta \leq \cdots$
  without claiming proof.
- 5 KEMs have proofs but do not
  clearly use correct δ definition.
  (LEDA uses wrong definition.)

What about quant

Consider quantum
for each cryptanal

"conj")

-Kiltz
IND-
ty $Q\delta$,

Also,
oof.)

2^{128}.

.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim that δ is small enough.
- 15 KEMs conjecture $\delta \leq \cdots$ without claiming proof.
- 5 KEMs have proofs but do not clearly use correct δ definition. (LEDA uses wrong definition.)

What about quantum attack

Consider quantum computer
for each cryptanalytic target

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim
  that δ is small enough.

- 15 KEMs conjecture $\delta \leq \cdots$
  without claiming proof.

- 5 KEMs have proofs but do not
  clearly use correct δ definition.
  (LEDA uses wrong definition.)

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim
  that δ is small enough.

- 15 KEMs conjecture $\delta \leq \cdots$
  without claiming proof.

- 5 KEMs have proofs but do not
  clearly use correct δ definition.
  (LEDA uses wrong definition.)

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

The other 23 KEMs:
Security goal 2^k
without proof that $\delta \leq 2^{-k}$.
So need CCA cryptanalysis.

Main issues in these 23 KEMs:

- 14 KEMs do not claim
  that δ is small enough.
- 15 KEMs conjecture $\delta \leq \cdots$
  without claiming proof.
- 5 KEMs have proofs but do not
  clearly use correct δ definition.
  (LEDA uses wrong definition.)

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

er 23 KEMs:
goal 2^k
proof that $\delta \leq 2^{-k}$.
CCA cryptanalysis.

ues in these 23 KEMs:

Ms do not claim
is small enough.

Ms conjecture $\delta \leq \cdots$
ut claiming proof.

Ms have proofs but do not
use correct δ definition.
A uses wrong definition.)

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

What ab

Each KE
of single

Ms:

$\delta \leq 2^{-k}$.

otanalysis.

se 23 KEMs:

t claim

nough.

cture $\delta \leq \cdots$

g proof.

roofs but do not

ct δ definition.

ng definition.)

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

What about multi-

Each KEM has qu
of single-user secu

Ms:

..

do not

tion.

ion.)

16

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

17

What about multi-user attac

Each KEM has quantitative
of single-user security level λ

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

What about multi-user attacks?

Each KEM has quantitative claim
of single-user security level λ.

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

What about multi-user attacks?

Each KEM has quantitative claim
of single-user security level λ.

This claim implies quantitative
claim λ' of U-user security.
λ' vs. λ: looseness factor U.

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

What about multi-user attacks?

Each KEM has quantitative claim
of single-user security level λ.

This claim implies quantitative
claim λ' of U-user security.
λ' vs. λ: looseness factor U.

The only risks of this U-user
security claim being broken
come from the single-user
security claim λ being broken.

What about quantum attacks?

Consider quantum computers
for each cryptanalytic target.

When hashing is involved,
analyze three types of attacks:
(1) ROM attacks.
(2) Non-ROM QROM attacks.
(3) Non-QROM attacks.

Sometimes proofs eliminate #1.
Ongoing efforts to extend proofs
to similarly eliminate #2.
Most QROM proofs are loose,
but see 2019 Bindel–Hamburg–
Hülsing–Persichetti.

What about multi-user attacks?

Each KEM has quantitative claim
of single-user security level λ.

This claim implies quantitative
claim λ' of U-user security.
λ' vs. λ: looseness factor U.

The only risks of this U-user
security claim being broken
come from the single-user
security claim λ being broken.

As far as I can tell,
none of the target KEMs
claim higher U-user security.