From: Patrick Longa <plonga@microsoft.com> via pqc-forum <pqc-forum@list.nist.gov>
To: Doge Protocol <dogeprotocol1@gmail.com>, pqc-forum <pqc-forum@list.nist.gov>
Subject: RE: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Tuesday, June 14, 2022 04:41:52 PM ET

Hi all,

Hertzbleed is the name of this new and powerful side-channel timing attack by Wang et al. (https://www.hertzbleed.com/) that exploits the dynamic frequency scaling technology that is used by many modern processors.

The full implications for constant-time protected software ***in general*** are still unclear. But for now, it seems that the limitation of the practical attack is that it requires a sufficiently long computation of zeroes (or ones), and these computations should be somehow correlated to the secret key. In the case of SIKE, the attack is successful against it using a chosen ciphertext attack exploiting strategies that resemble zero-value attacks on standard ECC implementations (see https://eprint.iacr.org/2022/054).

To protect SIKE one needs to include a public-key validation step in the decapsulation function that rejects malformed, invalid public keys that produce such long series of zeroes. The countermeasure is now included in the SIDH library (https://github.com/microsoft/PQCrypto-SIDH), and the authors of the attack have confirmed its effectiveness. The countermeasure injects a 5%-6% slowdown to the whole protocol.

Note that these zero value attacks are not novel themselves, and were known to be exploitable via “physical” attacks such as power analysis. What is novel (and alarming) is that power consumption of certain computations can be leaked remotely via frequency scaling and execution time.

Best regards,

Patrick

From: pqc-forum@list.nist.gov <pqc-forum@list.nist.gov> **On Behalf Of** Doge Protocol
Sent: Tuesday, June 14, 2022 12:42 PM
To: pqc-forum <pqc-forum@list.nist.gov>
Subject: [EXTERNAL] [pqc-forum] HertzBleed : power side channel attacks on SIKE

You don't often get email from dogeprotocol1@gmail.com. Learn why this is important

Came across this today https://www.hertzbleed.com/herzbleed.pdf

The paper describes a side channel chosen-ciphertext attack on SIKE (even on constant-time imementations), allowing full key extraction via remote timing.

Has this been already discussed in this forum? Didn't see any previous references in this forum, hence sharing.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 01:07:21 AM ET
Attachments: smime.p7m

If Turbo Boost and other clock-frequency adjustments are active, then the power and temperature of the program being run are amply documented to influence the clock speed, so the auditor has to assume that whatever secrets leak through power also leak through timing. This applies to all cryptosystems (and other operations on secret data), not just SIKE.

Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem, and nicely composes with the auditability of the usual constant-time coding discipline. Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.

Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?

Per-cryptosystem code modifications are much harder to audit. Small tweaks can slow attacks down but seem unlikely to eliminate the attacks. Maybe there's enough capacitance in the leak mechanisms that software willing to incur the complications of 2-share masking will stop attacks, but how is the auditor supposed to evaluate this even for one CPU, never mind all the other CPUs where software will run?

Saying that a simple model explains a simple attack against an unprotected target doesn't answer the question. There are many CHES papers demonstrating leaks that are explained by more sophisticated models that take much more effort to build, even for relatively simple embedded CPUs.

We already know what happens when implementors try to achieve security against timing attacks without having any realistic way to ensure that they're succeeding: they add ad-hoc countermeasures that stop the current attack and are broken by the next attack. Meanwhile there's indirect security damage from these complications slowing down correctness audits, formal verification, etc., as if we didn't have enough cryptographic code bugs already!

The constant-time approach simply cuts off the information flow at the heart of the security problem. Yes, this takes work, but it's clear what has to be done. For example:

* If there's any path in code from secrets to instructions that take a data-dependent number of cycles, the code has to be fixed. Many important pieces of code have already done this, often with automated tools checking the results.

* The risk of CPU manufacturers adding timing variations has to be addressed through promises in the instruction set. ARM has started doing this.

* If cycles take time that depends on the data---in other words, if the CPU frequency varies depending on the data---then that has to be fixed too.

https://www.hertzbleed.com/ claims that running the CPU at constant speed "has an extreme system-wide performance impact", apparently trying to convince the reader that the auditable solution shouldn't even be considered. I'm skeptical that this performance claim is coming from system-wide measurements:

* My servers, including compute servers, have been running at constant speed for many years. Usually this is top nominal speed without Turbo Boost. Sometimes I set it lower when the top nominal speed seems to be creating hardware stress; dead hardware is a big setback in computation per dollar. Even if I didn't care about the

cost and hassle of hardware replacements, I wouldn't get much out of Turbo Boost: servers running optimized code on all cores are pretty much the worst possible case for Turbo Boost.

* I've also happily set my laptop to constant speed for many years, more and more often choosing the minimum speed that the laptop supports. It's increasingly rare that I'm waiting for my laptop to compute something, and I don't like the laptop heating up, especially from background activity on browser pages.

* If people are waiting for video processing on their laptops and want every possible speedup, the biggest speed boost they can get is from optimized code running on all cores---which again is a bad case for Turbo Boost.

If there's supposed to be an "extreme system-wide" performance penalty for the only auditable path to security, where's the quantification of this penalty?

If the fix is constant CPU frequencies rather than tweaks to SIKE, then it's wrong for https://www.hertzbleed.com to be saying "the attack on SIKE" rather than "the attack on variable CPU frequencies". (Compare this to https://cr.yp.to/papers.html#cachetiming, which explicitly addressed the question of why it was blaming AES, and spent eight sections on the security justification for this attribution.) Best bet is that we'll see subsequent demos of similar attacks for many other cryptosystems. So I don't think NIST should penalize SIKE.

---D. J. Bernstein

From: Bo Lin <bolinsco@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 04:47:28 AM ET

The paper reads "Attack scenario. We target a **static key** version of SIKE where **a single secret key is used to decrypt several ciphertexts**..." This means it does not apply to the standard SIKE which changes its secret keys (skA and skB) every time. In general, for ephemeral keys, only the "single trace" attack is valid because if an adversary is not able to figure out a secret with a single trace, they need to try again. However, the prosed attack in this paper can be a starting point to study single trace attack by using template or machine learning.

Attacks against implementation are important for products to be deployed into regulated markets because a cryptosystems may fail in its security function not because of a wrong choice of cryptographic mechanisms or protocols, but because of an insecure implementation. The implementation security is related to many factors, especially the semiconductor technology. The security evaluation of side-channel attacks and fault injection attacks has been conducted for many year if we think about the 12 billion payment cards in circulation (EMVCo Reports 12 Billion EMV<sup>®</sup></sup> Chip Cards in Global Circulation - EMVCo).

Because implementation security is tightly coupled with device manufacturing technology and with targeted markets, secure coding for implementation security can be left for product building. For NIST standardisation, crypto security has to be focused.

On Wed, 15 Jun 2022 at 06:07, D. J. Bernstein <djb@cr.yp.to> wrote:

> If Turbo Boost and other clock-frequency adjustments are active, then
> the power and temperature of the program being run are amply documented
> to influence the clock speed, so the auditor has to assume that whatever
> secrets leak through power also leak through timing. This applies to all
> cryptosystems (and other operations on secret data), not just SIKE.
>
> Running the CPU at a constant speed, independent of the data being
> processed, is the obvious, straightforward, auditable way to eliminate

> this problem, and nicely composes with the auditability of the usual constant-time coding discipline. Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.
>
> Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?
>
> Per-cryptosystem code modifications are much harder to audit. Small tweaks can slow attacks down but seem unlikely to eliminate the attacks. Maybe there's enough capacitance in the leak mechanisms that software willing to incur the complications of 2-share masking will stop attacks, but how is the auditor supposed to evaluate this even for one CPU, never mind all the other CPUs where software will run?
>
> Saying that a simple model explains a simple attack against an unprotected target doesn't answer the question. There are many CHES papers demonstrating leaks that are explained by more sophisticated models that take much more effort to build, even for relatively simple embedded CPUs.
>
> We already know what happens when implementors try to achieve security against timing attacks without having any realistic way to ensure that they're succeeding: they add ad-hoc countermeasures that stop the current attack and are broken by the next attack. Meanwhile there's indirect security damage from these complications slowing down correctness audits, formal verification, etc., as if we didn't have enough cryptographic code bugs already!
>
> The constant-time approach simply cuts off the information flow at the heart of the security problem. Yes, this takes work, but it's clear what has to be done. For example:
>
> * If there's any path in code from secrets to instructions that take

> a data-dependent number of cycles, the code has to be fixed. Many important pieces of code have already done this, often with automated tools checking the results.
>
> * The risk of CPU manufacturers adding timing variations has to be addressed through promises in the instruction set. ARM has started doing this.
>
> * If cycles take time that depends on the data---in other words, if the CPU frequency varies depending on the data---then that has to be fixed too.
>
> https://www.hertzbleed.com/ claims that running the CPU at constant speed "has an extreme system-wide performance impact", apparently trying to convince the reader that the auditable solution shouldn't even be considered. I'm skeptical that this performance claim is coming from system-wide measurements:
>
> * My servers, including compute servers, have been running at constant speed for many years. Usually this is top nominal speed without Turbo Boost. Sometimes I set it lower when the top nominal speed seems to be creating hardware stress; dead hardware is a big setback in computation per dollar. Even if I didn't care about the cost and hassle of hardware replacements, I wouldn't get much out of Turbo Boost: servers running optimized code on all cores are pretty much the worst possible case for Turbo Boost.
>
> * I've also happily set my laptop to constant speed for many years, more and more often choosing the minimum speed that the laptop supports. It's increasingly rare that I'm waiting for my laptop to compute something, and I don't like the laptop heating up, especially from background activity on browser pages.
>
> * If people are waiting for video processing on their laptops and want every possible speedup, the biggest speed boost they can get is from optimized code running on all cores---which again is a bad case for Turbo Boost.

> If there's supposed to be an "extreme system-wide" performance penalty
> for the only auditable path to security, where's the quantification of
> this penalty?
>
> If the fix is constant CPU frequencies rather than tweaks to SIKE, then
> it's wrong for https://www.hertzbleed.com to be saying "the attack on
> SIKE" rather than "the attack on variable CPU frequencies". (Compare
> this to https://cr.yp.to/papers.html#cachetiming, which explicitly
> addressed the question of why it was blaming AES, and spent eight
> sections on the security justification for this attribution.) Best bet
> is that we'll see subsequent demos of similar attacks for many other
> cryptosystems. So I don't think NIST should penalize SIKE.
>
> ---D. J. Bernstein
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/20220615050635.113967.qmail%40cr.yp.to.

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CAL3OXVxgDKtfYR4HSJTS1fDh4EvUWjQL3v%2BumhcqCGrnCLMD8w%40mail.gmail.com.

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: Bo Lin <bolinsco@gmail.com>
CC: Moody, Dustin (Fed)' via pqc-forum <pqc-forum@list.nist.gov>, D. J. Bernstein <djb@cr.yp.to>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 07:06:29 AM ET

Hi Bo Lin and Dan,

> On Jun 15, 2022, at 10:47 AM, Bo Lin <bolinsco@gmail.com> wrote:
>
> The paper reads "Attack scenario. We target a **static key** version of SIKE where **a single secret key is used to decrypt several ciphertexts**..." This means it does not apply to the standard SIKE which changes its secret keys (skA and skB) every time. In general, for ephemeral keys, only the "single trace" attack is valid because if an adversary is not able to figure out a secret with a single trace, they need to try again. However, the prosed attack in this paper can be a starting point to study single trace attack by using template or machine learning.

SIKE is claimed to be CCA2-secure, and indeed CCA2-security is a useful feature in protocol design. Not all protocols will be ephemeral-only. For example, triple-DH-like systems use long-term KEM keypairs for authentication.

> Attacks against implementation are important for products to be deployed into regulated markets because a cryptosystems may fail in its security function not because of a wrong choice of cryptographic mechanisms or protocols, but because of an insecure implementation. The implementation security is related to many factors, especially the semiconductor technology. The security evaluation of side-channel attacks and fault injection attacks has been conducted for many year if we think about the 12 billion payment cards in circulation (EMVCo Reports 12 Billion EMV<sup>®</sup> Chip Cards in Global Circulation - EMVCo).
>
> Because implementation security is tightly coupled with device manufacturing technology and with targeted markets, secure coding for implementation security

> can be left for product building. For NIST standardisation, crypto security has to be focused.

I agree in part. However, crypto standards must consider real-world implementation constraints, because they will be deployed in the real world. If a specific system did have a difficult-to-mitigate side-channel attack, I would expect that to affect its priority for standardization. For example, I would not expect NIST to standardize a symmetric cipher with sboxes that are far more efficient to implement with a large lookup table, unless there were other very compelling reasons to do so.

In the case of SIKE, it isn’t being considered in the third round anyway. I expect that by the time it is considered for standardization, someone will have implemented and tested mitigations for this issue. Unless those mitigations are surprisingly expensive, I don’t expect this to be a serious problem long-term, but I’ve been wrong before.

> On Wed, 15 Jun 2022 at 06:07, D. J. Bernstein <djb@cr.yp.to> wrote:
>
>> Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem, and nicely composes with the auditability of the usual constant-time coding discipline. Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.
>>
>> Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?

I assume you’re yelling at clouds here, but just to check: are you seriously arguing that CPUs, even laptop and phone CPUs, should universally be run at a constant frequency (not even just a workload-responsive but not thermally-responsive frequency) to mitigate crypto side-channel attacks? In other words, are you seriously arguing that manufacturers of laptops, servers and phones should throw out at least the following 5 optimizations?

* Power- and thermally-adaptive frequency scaling (boost clocking).

* Software workload-adaptive frequency scaling — that is, downclocking below nominal base frequency when idle to save power, and clock-gating and power-gating idle cores.

* Frequency scaling in response to vectorization or other power-intensive features — eg, downclocking when AVX-256 or AVX-512 is in use.

* Heterogeneous cores, such as performance cores and efficiency cores.

* Extending clock periods or dropping cycles for hardware voltage droop mitigation. This lowers the required supply voltage and saves energy, and is present since at least Intel Nehalem.

This is not going to happen industry-wide over a side-channel attack on SIKE, or even on several cryptosystems. Even just throwing out power- and thermally-adaptive scaling, the biggest culprit here, isn’t remotely going to happen. It’s like asking CPU manufacturers to completely throw out speculative execution over Spectre.

We can’t bend the world to our will on this, so like it or not, we will have to consider sufficiently-high-bandwidth power side channels in software design.

> If there's supposed to be an "extreme system-wide" performance penalty
> for the only auditable path to security, where's the quantification of
> this penalty?

Intel and AMD’s recent high-end laptop offerings have these specs:

Core i7-1280P (Alder Lake):

E-core base 1.3 GHz, turbo to 3.6 GHz

P-core base 1.8 GHz, turbo to 4.8 GHz

AMD Ryzen 7 6800H: Base clock 3.2 GHz, boost up to 4.7

AMD Ryzen 7 6800U: Base clock 2.7 GHz, boost up to 4.7

So to first order, on an i7-1280P’s performance cores, the performance advantage of thermally-adaptive boost is 2.66x. Of course, how much that affects your specific workload will depend on that workload, as well as your laptop’s heatsinks and fans. Could they change the base frequency spec and demand better cooling from ultrabooks, to reduce this ratio? Sure. Are they going to do that over a side-channel in SIKE? Absolutely not.

Note that boosting is primarily aimed at interactive workloads, not long-running compute-intensive ones. That is, it’s primarily for launching an app or performing a CPU-intensive query during an otherwise lower-power session. By doing this, it makes a laptop or phone much more responsive within a given thermal envelope. So contrary to your assertion, it’s a great fit for your desired cool-but-responsive laptop that’s not used for heavy compute, at least if you can reduce the temperature threshold.

It’s also useful for long-term CPU-intensive applications like gaming or video editing, if the system has more thermal headroom (e.g. a cleaner fan, a more thermally conductive tabletop, or a less heavily loaded GPU) than what was used to choose the spec.

Regards,

— Mike

From: Carl Mitchell <carl.mitchell@gomotive.com> via pqc-forum <pqc-forum@list.nist.gov>
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 07:58:24 AM ET

Hi Mike and Dan

Downclocking to a single fixed frequency shouldn't be required at all times, but rather only when performing a cryptographic operation. ISA support could provide a "clock fence" similar to current memory fences, but for clock speed. Or hardware could be dedicated to cryptographic processing as is commonly done with AES and SHA-2.

Regards,

Carl

On Wed, Jun 15, 2022, 07:06 Mike Hamburg <mike@shiftleft.org> wrote:

> Hi Bo Lin and Dan,
>
> On Wed, 15 Jun 2022 at 06:07, D. J. Bernstein <djb@cr.yp.to> wrote:
>
>> Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem, and nicely composes with the auditability of the usual constant-time coding discipline. Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.
>>
>> Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?
>
> I assume you're yelling at clouds here, but just to pqc-forum@list.nist.gov: are you seriously arguing that CPUs, even laptop and phone CPUs, should universally be run at a constant

> frequency (not even just a workload-responsive but not thermally-responsive frequency) to mitigate crypto side-channel attacks?

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: Carl Mitchell <carl.mitchell@gomotive.com>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 08:53:06 AM ET

Hi Carl,
That’s a good idea, though there are some issues with it, such as denial of service. Maybe the biggest issue is that the CPU’s temperature changes will persist, likely significantly, after the crypto operation had finished. So the fence might also have to persist for a while, or something.

Possibly the easiest reliable approach is to offload the crypto processing to a dedicated, and therefore hopefully smaller and more efficient, crypto accelerator. This could run at a fixed (or at least key-independent) clock frequency, and either have dedicated side-channel mitigations or just use little enough power that it won’t significantly affect the main CPU’s clocking.

Regards,

— Mike

> On Jun 15, 2022, at 1:58 PM, 'Carl Mitchell' via pqc-forum <pqc-forum@list.nist.gov> wrote:
>
> Hi Mike and Dan
>
> Downclocking to a single fixed frequency shouldn't be required at all times, but rather only when performing a cryptographic operation. ISA support could provide a "clock fence" similar to current memory fences, but for clock speed. Or hardware could be dedicated to cryptographic processing as is commonly done with AES and SHA-2.
>
> Regards,
>
> Carl
>
> On Wed, Jun 15, 2022, 07:06 Mike Hamburg <mike@shiftleft.org> wrote:

> Hi Bo Lin and Dan,
>
> On Wed, 15 Jun 2022 at 06:07, D. J. Bernstein <djb@cr.yp.to> wrote:
>
> > Running the CPU at a constant speed, independent of the data
> > being
> > processed, is the obvious, straightforward, auditable way to
> > eliminate
> > this problem, and nicely composes with the auditability of the usual
> > constant-time coding discipline. Constant speed isn't the typical OS
> > default these days, but there are tools to set it up. See, e.g.,
> > https://bench.cr.yp.to/supercop.html for Linux instructions to
> > disable
> > Turbo Boost, Turbo Core, and downclocking.
> >
> > Compared to running the CPU at a constant speed (and blaming the
> > attack
> > on variable CPU frequencies), why is it supposed to be better to try
> > to
> > modify code for specific cryptosystems (and to blame the attack on
> > the
> > unmodified code, or on the cryptosystems)?
>
> I assume you're yelling at clouds here, but just to pqc-forum@list.nist.gov: are you seriously arguing that CPUs, even laptop and phone CPUs, should universally be run at a constant frequency (not even just a workload-responsive but not thermally-responsive frequency) to mitigate crypto side-channel attacks?

From: Doge Protocol <dogeprotocol1@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum <pqc-forum@list.nist.gov>
CC: mi...@shiftleft.org <mike@shiftleft.org>, Moody, Dustin (Fed)' via pqc-forum <pqc-forum@list.nist.gov>, D. J. Bernstein <djb@cr.yp.to>, boli...@gmail.com <bolinsco@gmail.com>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 12:01:47 PM ET

>>>case of SIKE, it isn’t being considered in the third round anyway.
It is an alternate candidate in Round 3.

On Wednesday, June 15, 2022 at 4:06:24 AM UTC-7 mi...@shiftleft.org wrote:

> Hi Bo Lin and Dan,
>
> > On Jun 15, 2022, at 10:47 AM, Bo Lin <boli...@gmail.com> wrote:
> >
> > The paper reads "Attack scenario. We target a **static key** version of SIKE where **a single secret key is used to decrypt several ciphertexts**..." This means it does not apply to the standard SIKE which changes its secret keys (skA and skB) every time. In general, for ephemeral keys, only the "single trace" attack is valid because if an adversary is not able to figure out a secret with a single trace, they need to try again. However, the prosed attack in this paper can be a starting point to study single trace attack by using template or machine learning.
>
> SIKE is claimed to be CCA2-secure, and indeed CCA2-security is a useful feature in protocol design. Not all protocols will be ephemeral-only. For example, triple-DH-like systems use long-term KEM keypairs for authentication.
>
> > Attacks against implementation are important for products to be deployed into regulated markets because a cryptosystems may fail in its security function not because of a wrong choice of cryptographic mechanisms or protocols, but because of an insecure implementation. The implementation security is related to many factors, especially the semiconductor technology. The security evaluation of side-channel attacks and fault injection attacks has been conducted

> > for many year if we think about the 12 billion payment cards in circulation (EMVCo Reports 12 Billion EMV<sup>®</sup> Chip Cards in Global Circulation - EMVCo).
> >
> > Because implementation security is tightly coupled with device manufacturing technology and with targeted markets, secure coding for implementation security can be left for product building. For NIST standardisation, crypto security has to be focused.
>
> I agree in part. However, crypto standards must consider real-world implementation constraints, because they will be deployed in the real world. If a specific system did have a difficult-to-mitigate side-channel attack, I would expect that to affect its priority for standardization. For example, I would not expect NIST to standardize a symmetric cipher with sboxes that are far more efficient to implement with a large lookup table, unless there were other very compelling reasons to do so.
>
> In the case of SIKE, it isn't being considered in the third round anyway. I expect that by the time it is considered for standardization, someone will have implemented and tested mitigations for this issue. Unless those mitigations are surprisingly expensive, I don't expect this to be a serious problem long-term, but I've been wrong before.
>
> > On Wed, 15 Jun 2022 at 06:07, D. J. Bernstein <d...@cr.yp.to> wrote:
> >
> > > Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem, and nicely composes with the auditability of the usual constant-time coding discipline. Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.
> > >
> > > Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?
>
> I assume you're yelling at clouds here, but just to check: are you seriously arguing that CPUs, even laptop and phone CPUs, should universally be run at a constant frequency (not

> even just a workload-responsive but not thermally-responsive frequency) to mitigate crypto side-channel attacks? In other words, are you seriously arguing that manufacturers of laptops, servers and phones should throw out at least the following 5 optimizations?
>
> * Power- and thermally-adaptive frequency scaling (boost clocking).
>
> * Software workload-adaptive frequency scaling — that is, downclocking below nominal base frequency when idle to save power, and clock-gating and power-gating idle cores.
>
> * Frequency scaling in response to vectorization or other power-intensive features — eg, downclocking when AVX-256 or AVX-512 is in use.
>
> * Heterogeneous cores, such as performance cores and efficiency cores.
>
> * Extending clock periods or dropping cycles for hardware voltage droop mitigation. This lowers the required supply voltage and saves energy, and is present since at least Intel Nehalem.
>
> This is not going to happen industry-wide over a side-channel attack on SIKE, or even on several cryptosystems. Even just throwing out power- and thermally-adaptive scaling, the biggest culprit here, isn’t remotely going to happen. It’s like asking CPU manufacturers to completely throw out speculative execution over Spectre.
>
> We can’t bend the world to our will on this, so like it or not, we will have to consider sufficiently-high-bandwidth power side channels in software design.
>
>> If there's supposed to be an "extreme system-wide" performance penalty
>> for the only auditable path to security, where's the quantification of
>> this penalty?
>
> Intel and AMD’s recent high-end laptop offerings have these specs:
>
> Core i7-1280P (Alder Lake):
>
> E-core base 1.3 GHz, turbo to 3.6 GHz
>
> P-core base 1.8 GHz, turbo to 4.8 GHz
>
> AMD Ryzen 7 6800H: Base clock 3.2 GHz, boost up to 4.7

> AMD Ryzen 7 6800U: Base clock 2.7 GHz, boost up to 4.7
>
> So to first order, on an i7-1280P's performance cores, the performance advantage of thermally-adaptive boost is 2.66x. Of course, how much that affects your specific workload will depend on that workload, as well as your laptop's heatsinks and fans. Could they change the base frequency spec and demand better cooling from ultrabooks, to reduce this ratio? Sure. Are they going to do that over a side-channel in SIKE? Absolutely not.
>
> Note that boosting is primarily aimed at interactive workloads, not long-running compute-intensive ones. That is, it's primarily for launching an app or performing a CPU-intensive query during an otherwise lower-power session. By doing this, it makes a laptop or phone much more responsive within a given thermal envelope. So contrary to your assertion, it's a great fit for your desired cool-but-responsive laptop that's not used for heavy compute, at least if you can reduce the temperature threshold.
>
> It's also useful for long-term CPU-intensive applications like gaming or video editing, if the system has more thermal headroom (e.g. a cleaner fan, a more thermally conductive tabletop, or a less heavily loaded GPU) than what was used to choose the spec.
>
> Regards,
>
> — Mike

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/b7365e02-d768-4686-b926-3a25fb665010n%40list.nist.gov.

From: Deirdre Connolly <durumcrustulum@gmail.com> via pqc-forum@list.nist.gov
To: Patrick Longa <plonga@microsoft.com>
CC: Doge Protocol <dogeprotocol1@gmail.com>, pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 12:15:43 PM ET

> To protect SIKE one needs to include a public-key validation step in the decapsulation function that rejects malformed, invalid public keys that produce such long series of zeroes. The countermeasure is now included in the SIDH library (https://github.com/microsoft/PQCrypto-SIDH), and the authors of the attack have confirmed its effectiveness. The countermeasure injects a 5%-6% slowdown to the whole protocol.

Would the long series of zeroes be caught as malformed with existing pubkey checks? I'm assuming not as a change was applied, but just wanted to check.

On Tue, Jun 14, 2022, 4:40 PM 'Patrick Longa' via pqc-forum <pqc-forum@list.nist.gov> wrote:

> Hi all,
>
> Hertzbleed is the name of this new and powerful side-channel timing attack by Wang et al. (https://www.hertzbleed.com/) that exploits the dynamic frequency scaling technology that is used by many modern processors.
>
> The full implications for constant-time protected software ***in general*** are still unclear. But for now, it seems that the limitation of the practical attack is that it requires a sufficiently long computation of zeroes (or ones), and these computations should be somehow correlated to the secret key. In the case of SIKE, the attack is successful against it using a chosen ciphertext attack exploiting strategies that resemble zero-value attacks on standard ECC implementations (see https://eprint.iacr.org/2022/054).
>
> To protect SIKE one needs to include a public-key validation step in the decapsulation function that rejects malformed, invalid public keys that produce such long series of zeroes. The countermeasure is now included in the SIDH library (https://github.com/microsoft/PQCrypto-SIDH), and the authors of the attack have confirmed its effectiveness. The countermeasure injects a 5%-6% slowdown to the whole protocol.

> Note that these zero value attacks are not novel themselves, and were known to be exploitable via “physical” attacks such as power analysis. What is novel (and alarming) is that power consumption of certain computations can be leaked remotely via frequency scaling and execution time.
>
> Best regards,
>
> Patrick
>
> ---
>
> **From:**pqc-forum@list.nist.gov <pqc-forum@list.nist.gov> **On Behalf Of** Doge Protocol
> **Sent:** Tuesday, June 14, 2022 12:42 PM
> **To:** pqc-forum <pqc-forum@list.nist.gov>
> **Subject:** [EXTERNAL] [pqc-forum] HertzBleed : power side channel attacks on SIKE
>
> > You don't often get email from dogeprotocol1@gmail.com. Learn why this is important
>
> Came across this today https://www.hertzbleed.com/herzbleed.pdf
>
> The paper describes a side channel chosen-ciphertext attack on SIKE (even on constant-time imementations), allowing full key extraction via remote timing.
>
> Has this been already discussed in this forum? Didn't see any previous references in this forum, hence sharing.
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/e82d7833-9bee-4879-a676-1c1756154a28n%40list.nist.gov.
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/

> CY4PR21MB082144D3076E6F9F806C74BFDAAA9%40CY4PR21MB0821.namprd21.prod.outlook.com.

--

You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CAFR824zmZUuh4qD-sdfyti6aAO2Y6a9udU50j53Z%3D04%2BaSW8Sw%40mail.gmail.com.

From: Hovav Shacham <hovav@hovav.net> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 01:07:36 PM ET

On Wed, Jun 15, 2022, at 11:14 AM, Deirdre Connolly wrote:

> Would the long series of zeroes be caught as malformed with existing pubkey checks? I'm assuming not as a change was applied, but just wanted to check.

Unfortunately no: The 0 doesn't show up until ladder round i (where m_i is the key bit being targeted) in our CCA attack and the section-3.1 variant of De Feo et al.'s attack (https://tches.iacr.org/index.php/TCHES/article/view/9701/9232). Even checking for 0s during the ladder isn't enough, because the section-3.2 variant of De Feo et al.'s attack has the 0 wait until early in isogeny computation to make an appearance.

The attacker has a lot of freedom in picking the points P and Q, so my guess is that simple key checks, if added, could be bypassed as in Galbraith-Petit-Shani-Ti (https://www.iacr.org/archive/asiacrypt2016/10031307/10031307.pdf). Instead the right (SIKE-specific) mitigation is the full input validation described in section 5 of De Feo et al.

As I understand it, that full validation is now implemented for PQCrypto-SIDH 5.3.1 and for CIRCL 1.2.0 by

https://github.com/microsoft/PQCrypto-SIDH/commit/75ed5b09bd06a19cdad7660bef10e820074f808f and

https://github.com/cloudflare/circl/commit/10923e8d736009130170c06c1bdbd81bee4de56c ,

respectively.

-hs.

From: Phillip Hallam-Baker <phill@hallambaker.com> via pqc-forum@list.nist.gov
To: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 03:07:34 PM ET

On Wed, Jun 15, 2022 at 1:07 AM D. J. Bernstein <djb@cr.yp.to> wrote:

> ...
>
> Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?
>
> ...

This is merely another data point showing that the 'constant speed' assumption is unworkable. CPUs do not operate at a constant speed, that stopped being the case back in. the 80s when self-timed circuits came in.

I have never been a fan of the Montgomery Ladder and I certainly don't look to rely on that type of technique to avoid timing analysis. The technique I would use instead is Kocher blinding which is now conveniently out of patent for many crypto systems.

For ECDH systems, to calculate a.P, generate a random nonce x, then calculate x.P + (a-x mod g).P where g is the group order. Perform the two calculations in parallel on separate cores and this doesn't increase latency unless the CPU is already fully loaded.

That is an approach that I would trust, because it doesn't rely on any assumptions about the implementation other than that the side channels are not so wide that the correlation can be detected on a single pass. Using a Montgomery ladder on top of Kocher blinding makes some sense but I see that as an additional control rather than the primary.

It seems to me that requiring that a PQC system support the type of composability that allows Kocher blinding to be employed would be the best defense against this class of attacks using current hardware.

The only 'constant time' approach that I would be confident in would be one that is implemented as a separate on-chip dedicated crypto-processor. And we might well come to needing to use that type of approach at some point since it is the only robust defense against ROWHAMMER, SPECTRE etc. attacks.

Which is why we desperately need agreement on standard crypto systems, both conventional and PQC so that Intel, AMD, ARM etc. can be told exactly what to build.

If you have a PKI that is designed around the use of this type of technique such as the Mesh Threshold Key Infrastructure, you can then make use of device bound keys that make use of a random seed that cannot be extracted from the device. Distinct keys for different users/applications etc. can then be created through threshold combination techniques.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 04:31:13 PM ET
Attachments: smime.p7m

Mike Hamburg writes:

> are you seriously arguing that CPUs, even laptop and phone CPUs,
> should universally be run at a constant frequency (not even just a
> workload-responsive but not thermally-responsive frequency)

Those are two separate issues.

"Constant" refers to dependence on the secret data being processed. Some data is designated as secret; some—including timing—is designated as public; the whole point of the security mechanism is to cut off all information flow from the secret data to the public data, eliminating the multiple-decade mess of evaluating whether nonzero information flow from secrets to timing is exploitable.

Users generally aren't trying to hide _whether their devices are idle_; i.e., the idle times are public. Having a device run at the lowest possible power when it's idle isn't creating the prohibited information flow. This is an example of "workload-responsive". So, no, "constant" isn't in conflict with "workload-responsive".

More examples:

* There's data flow from the program's instruction pointer to whether the device is idle. That's allowed by "constant", since the instruction pointer is also designated as public.

* Secret-dependent branches create data flow from the secrets to the instruction pointer (and thus to idle time etc.). That's a classic example of what's _not_ allowed by "constant".

* The hardware mechanisms currently under discussion are creating

data flow from power and temperature (secret) to clock frequency (public). This is also not allowed by "constant".

The auditability of the constant-time security mechanism comes from being able to look through each system component and---using the labels that say what's secret and what's public---evaluate locally whether the data flow is following the rules. This takes work---systems are large, and the necessary labels often have to be traced manually---but it's clear what to do, and many years of work have gone into doing it.

There are tools available right now to run CPUs at constant frequency. There are tools available right now to check that software is following the data-flow rules for the cycle counts to be constant. There are some NISTPQC KEM implementations available right now that pass those tools. There are CPUs available right now where all available evidence is that all of this works, cutting off all data flow from secrets to timing and thus producing complete security against timing attacks. There are also increasingly useful commitments for the future from CPU manufacturers.

What's the alternative plan for stopping the attacks? Again, we already know what happens when implementors try to achieve security against timing attacks without having any realistic way to ensure that they're succeeding: they add ad-hoc countermeasures that stop the current attack and are broken by the next attack.

People suggesting that SIKE should be penalized, or using the "attack on SIKE" terminology, are assigning blame to a specific cryptosystem rather than to the use of non-constant CPU frequencies. But this assignment of blame doesn't appear to be backed by a measurable action plan. Why should anyone believe that revised SIKE software, or software for other KEMs, is safe when there's no serious proposal of an audit mechanism? Are we supposed to allow data flow from power to timing and simply _hope_ that the power variations aren't creating enough timing variation to be exploitable?

> * Power- and thermally-adaptive frequency scaling (boost clocking).

That's the heart of the security problem at hand, yes.

If there's supposed to be an argument that there's so much benefit to the end user from power-dependent clock boosts that this justifies sacrificing security, then the benefit should be carefully quantified. I think many users will decide that, no, security is more important.

There's also a hidden cost to power-dependent clock boosts, namely the damage that overclocking does to equipment lifetime. A user with a painfully slow phone has other motivations to replace the equipment, but the world is getting past that phase even in low-income countries.

> * Software workload-adaptive frequency scaling — that is, downclocking below nominal base frequency when idle to save power, and clock-gating and power-gating idle cores.

No, that's a separate issue. See above.

> * Frequency scaling in response to vectorization or other power-intensive features — eg, downclocking when AVX-256 or AVX-512 is in use.

Is the user trying to hide which programs are running? ("My software for corporate espionage uses AVX-512.")

Typical security policies today don't try to achieve this: the current goal is to protect the _data_ that the software is handling, without getting distracted by trying to hide the software too. Constant-time labeling formalizes this by designating the software as public and the data as secret.

> * Heterogeneous cores, such as performance cores and efficiency cores.

Why do you think these are prohibited? The question of which core a job is using has to be data-independent, of course.

> * Extending clock periods or dropping cycles for hardware voltage

> droop mitigation. This lowers the required supply voltage and saves energy, and is present since at least Intel Nehalem.

What saves more energy is running at slightly lower clock speeds where this corner case doesn't happen.

> This is not going to happen industry-wide over a side-channel attack on SIKE, or even on several cryptosystems.

Actually, it has been normal for many years (even though not universal) for computer manufacturers to give users control over clock frequencies. The user can turn off Turbo Boost, just like turning off hyperthreading; this doesn't require any sort of "industry-wide" action.

The current histrionics about the "extreme system-wide performance impact" of turning off Turbo Boost are about as well founded as picking benchmarks where hyperthreading gains 2x and leaping to the conclusion that hyperthreading can't possibly be turned off.

> So to first order, on an i7-1280P’s performance cores, the performance advantage of thermally-adaptive boost is 2.66x.

Not even close.

Multithreaded software typically gains a factor very close to 6 on those 6 performance cores, and vectorization typically gains a factor around 4. This 24x reduction in cycle counts eats up most of the power that would have been used by Turbo Boost, so maybe the final speedup is _only_ 10x, but this is a massive performance win—and one of the major trends in software engineering, with the relevant tools constantly becoming easier to use.

The only reasonable expectation, then, is that bottlenecks that matter for the user will be handled by multithreaded vectorized code—which also means that Turbo Boost is gaining relatively little. Many major pieces of code have already made this switch. (Video games made the switch many years ago.)

Cherry-picking unoptimized code is exaggerating the overall gain that the user sees today from Turbo Boost, and the exaggeration becomes more and more severe as more and more code takes advantage of multithreading and vectorization. Are we supposed to tell future users that, yes, we know how unoptimized code is skewing our performance evaluations away from what matters to them, but we nevertheless insist on making decisions for them on the basis of those performance evaluations?

> Note that boosting is primarily aimed at interactive workloads, not long-running compute-intensive ones. That is, it’s primarily for launching an app or performing a CPU-intensive query during an otherwise lower-power session.

This distinction between "interactive workloads" and "compute-intensive ones" has little relevance to the performance issues under discussion. Anything long enough for the user to be waiting for is _far_ above the startup costs of a thread.

If the unspecified "CPU-intensive query" isn't yet vectorized and multithreaded, maybe that's a hint that it's not something typical users are spending much time waiting for---i.e., it's a corner case, not the alleged "extreme system-wide performance impact".

> By doing this, it makes a laptop or phone much more responsive within a given thermal envelope.

Much _less_ responsive than handling the same bottlenecks with vectorized multithreaded code, by far the most important scenario for users within the lifetime of a post-quantum standard.

> So contrary to your assertion, it’s a great fit for your desired cool-but-responsive laptop that’s not used for heavy compute, at least if you can reduce the temperature threshold.

Um, which assertion is this supposed to be disputing? Here's what I actually wrote about my desired laptop: "I've also happily set my laptop

to constant speed for many years, more and more often choosing the
minimum speed that the laptop supports. It's increasingly rare that I'm
waiting for my laptop to compute something, and I don't like the laptop
heating up, especially from background activity on browser pages."

———D. J. Bernstein

From: gard...@gmail.com <gardinerm@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum <pqc-forum@list.nist.gov>
CC: D. J. Bernstein <djb@cr.yp.to>, pqc-...@list.nist.gov <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 06:13:21 PM ET

Dan,

The fact that turbo boost, hyper threading, etc. can be disabled on an individual laptop is not really relevant in a world where the majority of workloads are processed in a public cloud. You would need to convince the cloud providers this is a systemic security problem and that reducing performance per watt is the only reasonable way to fix it.

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 06:37:59 PM ET

> On Jun 15, 2022, at 10:30 PM, D. J. Bernstein <djb@cr.yp.to> wrote:
>
> Mike Hamburg writes:
>
>> are you seriously arguing that CPUs, even laptop and phone CPUs, should universally be run at a constant frequency (not even just a workload-responsive but not thermally-responsive frequency)
>
> Those are two separate issues.
>
> "Constant" refers to dependence on the secret data being processed. Some data is designated as secret; some---including timing---is designated as public; the whole point of the security mechanism is to cut off all information flow from the secret data to the public data, eliminating the multiple-decade mess of evaluating whether nonzero information flow from secrets to timing is exploitable.

Ah. I was confused, sorry. For clarity, you might want to use a different term for this (say, “key-independent”), because you used the word “constant” several times in your post, and in this one, to refer specifically to frequencies that don’t change. You also used its antonym “variable” referring to things that do change. Obviously, “not changing” is a subset of “not dependent on the key”, but it would be helpful to distinguish these terms when moving from

> My servers, including compute servers, have been running at constant speed for many years.

…

> I've also happily set my laptop to constant speed for many years,
>
> more and more often choosing the minimum speed that the laptop
>
> supports.

to

> If the fix is constant CPU frequencies rather than tweaks to SIKE…

Sorry to miss that “constant” had switched meanings.

> People suggesting that SIKE should be penalized, or using the "attack on SIKE" terminology, are assigning blame to a specific cryptosystem rather than to the use of non-constant CPU frequencies. But this assignment of blame doesn't appear to be backed by a measurable action plan.

I disagree in part with this reasoning, but it’s not about blame. It’s about,

will it be difficult to produce real-world software that runs SIKE on today’s

and tomorrow’s real computers and phones that’s secure against attack?

If the answer is yes, then that’s an issue that makes SIKE less desirable,

no matter whose fault it is.

However, it seems that this issue is not excessively costly to mitigate, so

personally I don’t think it makes SIKE significantly less desirable.

> Why
> should anyone believe that revised SIKE software, or software for other KEMs, is safe when there's no serious proposal of an audit mechanism?

> Are we supposed to allow data flow from power to timing and simply _hope_ that the power variations aren't creating enough timing variation to be exploitable?

From my POV, the ideal solution is to use an accelerator. The accelerator can be designed to mitigate data flow to externally visible signals (timing or even power, RF etc). Then we would have security without undoing decades of optimization.

It would be an uphill battle to get an appropriate accelerator into most chips, and of course you then have to trust them manufacturers not to backdoor it. But this is less of an uphill battle than getting everyone to turn off turbo boost.

> > * Heterogeneous cores, such as performance cores and efficiency cores.

> Why do you think these are prohibited? The question of which core a job is using has to be data-independent, of course.

As with most of these, I just didn’t understand that “constant” didn’t mean constant. P- and E-cores run your jobs at a different frequency in load-responsive ways, and possibly data-dependent ways if the scheduler is aware of the processor’s thermal state.

> > * Extending clock periods or dropping cycles for hardware voltage droop mitigation. This lowers the required supply voltage and saves energy, and is present since at least Intel Nehalem.

> What saves more energy is running at slightly lower clock speeds where this corner case doesn't happen.

Obviously, but using droop mitigation uses less power for a given level of performance, whereas running at a lower frequency does not. Nehalem’s droop mitigation gave about 5% frequency uplift at a given voltage. Not a huge benefit, but pretty big for such a small mechanism, and these things compound. I have no idea how much it adds in latest-generation chips.

>> This is not going to happen industry-wide over a side-channel attack on SIKE, or even on several cryptosystems.

> Actually, it has been normal for many years (even though not universal) for computer manufacturers to give users control over clock frequencies. The user can turn off Turbo Boost, just like turning off hyperthreading; this doesn't require any sort of "industry-wide" action.
>
> The current histrionics about the "extreme system-wide performance impact" of turning off Turbo Boost are about as well founded as picking benchmarks where hyperthreading gains 2x and leaping to the conclusion that hyperthreading can't possibly be turned off.

But most users do not turn off turbo boost or hyperthreading, and neither you nor I nor NIST can command them to do so. And it's no good to design crypto libraries that will be insecure (under practical, remote attacks) in the common case.

Or did you have a plan for getting everyone to drop this feature?

>> So to first order, on an i7-1280P’s performance cores, the performance advantage of thermally-adaptive boost is 2.66x.

> Not even close.
>
> Multithreaded software typically gains a factor very close to 6 on those 6 performance cores, and vectorization typically gains a factor around 4. This 24x reduction in cycle counts eats up most of the power that would have been used by Turbo Boost, so maybe the final speedup is _only_ 10x, but this is a massive performance win---and one of the major trends in software engineering, with the relevant tools constantly becoming easier to use.
>
> The only reasonable expectation, then, is that bottlenecks that matter for the user will be handled by multithreaded vectorized code---which also means that Turbo Boost is gaining relatively little. Many major pieces of code have already made this switch. (Video games made the switch many years ago.)

I don't know about you, but at my day job I spend a fair bit of time waiting for unvectorized, lightly-threaded code. Word and Outlook, Microsoft Teams, their auto-update features, the corporate mandated virus scanner, some stupid web app, etc. It's not a giant time drain, but it's slightly annoying every time I cmd-tab over to Outlook and it takes several seconds to come up.

Even more obviously "computational" workloads are not always parallel. When I type "sage" into a terminal and it spins up, I'm fairly certain that most of what it's doing is not vectorized, and probably not parallel either due to the infamous GIL. When I run scripts in sage, they are usually also not bottlenecked by vectorized or multithreaded portions of libraries. When I run the C compiler, it is mostly not vectorized (it may be multithreaded, if only due to make -j). Usually the code I'm compiling isn't production-grade or heavily optimized (I usually do prototyping work), so often it isn't vectorized either.

What’s more, much of this code *can’t* be vectorized, or at least it’s extremely difficult to do so and gains a lot less than a factor of 4. Traversing the DOM of some webapp, or the CFG of a C program, just fundamentally isn’t a vector workload.

And even threaded, vectorized code benefits from boost. I don’t have the all-core boost spec for the Core i7 1280P I mentioned earlier, but last generation’s flagship Core i7-1165G7 had a base clock of 2.8 GHz, all-core boost to 4.1, single-core boost to 4.7. Sure, it can’t boost as high or presumably as long on all cores, but there’s still a large benefit.

> Cherry-picking unoptimized code is exaggerating the overall gain that the user sees today from Turbo Boost, and the exaggeration becomes more and more severe as more and more code takes advantage of multithreading and vectorization. Are we supposed to tell future users that, yes, we know how unoptimized code is skewing our performance evaluations away from what matters to them, but we nevertheless insist on making decisions for them on the basis of those performance evaluations?

My overarching point in this thread is that “we" aren’t making those decisions.

> > Note that boosting is primarily aimed at interactive workloads, not long-running compute-intensive ones. That is, it’s primarily for launching an app or performing a CPU-intensive query during an otherwise lower-power session.
>
> This distinction between "interactive workloads" and "compute-intensive ones" has little relevance to the performance issues under discussion. Anything long enough for the user to be waiting for is _far_ above the

> startup costs of a thread.
>
> If the unspecified "CPU-intensive query" isn't yet vectorized and multithreaded, maybe that's a hint that it's not something typical users are spending much time waiting for---i.e., it's a corner case, not the alleged "extreme system-wide performance impact".

This is only true in an ideal world. In the real world, there are huge swathes of code that users wait for that isn’t vectorized or multithreaded, and (see above) possibly can’t be vectorized. I don’t expect this to stop being true in the near future, especially not to the point that manufacturers just pull that feature from new designs or users turn it off.

> > So contrary to your assertion, it’s a great fit for your desired cool-but-responsive laptop that’s not used for heavy compute, at least if you can reduce the temperature threshold.
>
> Um, which assertion is this supposed to be disputing? Here's what I actually wrote about my desired laptop: "I've also happily set my laptop to constant speed for many years, more and more often choosing the minimum speed that the laptop supports. It's increasingly rare that I'm waiting for my laptop to compute something, and I don't like the laptop heating up, especially from background activity on browser pages."

I suppose it wasn’t actually an assertion, but an implication that running at constant (I take your statement here to mean “unchanging”) frequency is desirable for keeping laptops responsive but cool and quiet.

Fundamentally, the point of boosting is to minimize waiting while keeping the machine under a certain temperature. That’s ordinarily set to avoid damaging the processor, but the same mechanism can keep the machine cool to the touch instead. And it will give more performance than running at an unchanging frequency.

But if your laptop is always instantly responsive to your commands when running at, say, 600 MHz (a typical minimum frequency, though your laptop may be different), then whatever you’re doing is apparently working.

Regards,

— Mike

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 07:00:52 PM ET
Attachments: smime.p7m

Phillip Hallam-Baker writes:

> This is merely another data point showing that the 'constant speed' assumption is unworkable. CPUs do not operate at a constant speed, that stopped being the case back in. the 80s when self-timed circuits came in.

Sorry, are you claiming that typical Intel CPU cores run at data-dependent speed when overclocking and downclocking are disabled as explained in, e.g., https://bench.cr.yp.to/supercop.html?

There are some corner cases. For example, the "check for adequate cooling" step isn't scripted. If a CPU fan fails then the CPU will react with lower clock speeds, maybe probing for higher clock speeds after it cools down. Fortunately, this hardware failure is visible to the OS, which is free to react by (1) alerting the user to replace the fan and (2) forcing minimum clock speed until reboot to protect the hardware.

[blinding]

> That is an approach that I would trust, because it doesn't rely on any assumptions about the implementation other than that the side channels are not so wide that the correlation can be detected on a single pass.

That's not a safe assumption. Single-trace attacks have been demonstrated against various cryptographic systems. Random example: https://eprint.iacr.org/2020/371.

> It seems to me that requiring that a PQC system support the type of composability that allows Kocher blinding to be employed would be the best defense against this class of attacks using current hardware.

The more general concept of 2-share masking applies to every computation, although it's usually more expensive than the special case of 2-share blinding. SABER was early in providing an implementation with full 2-share masking. That implementation was then successfully attacked (https://eprint.iacr.org/2021/079). As far as I know, the attack wasn't because of an implementation mistake, but because in general 2-share masking is insufficient.

In simple models that limit how much the attacker sees, more shares increase the attack cost exponentially. It's not clear at this point whether the same is true for real attackers. Stretching out a computation in time and/or space---as one has to do in general for more shares---also gives the attackers more time and/or space for probing the computation.

Analyzing security against invasive side channels is _really difficult_, a major topic of the CHES series of conferences over the last two decades. From an auditing perspective, this is a nightmare compared to what's achievable regarding timing attacks.

---D. J. Bernstein

--

You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/20220615230018.173150.qmail%40cr.yp.to.

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 08:30:19 PM ET

> On Jun 16, 2022, at 00:37, Mike Hamburg <mike@shiftleft.org> wrote:
>
> From my POV, the ideal solution is to use an accelerator. The accelerator can be designed to mitigate data flow to externally visible signals (timing or even power, RF etc). Then we would have security without undoing decades of optimization.

Hm, or maybe this is my own bias showing. Dan works on proving that code is correct and constant-time, and he thinks the ideal solution is to make chips constant-frequency (for some definition of constant) to preserve those guarantees. I work on side-channel-resistant accelerators for situations where almost no CPU would be good enough, and I think the ideal solution is to use accelerators instead of the CPU.

— Mike

From: Deirdre Connolly <durumcrustulum@gmail.com> via pqc-forum@list.nist.gov
To: Hovav Shacham <hovav@hovav.net>
CC: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 08:54:35 PM ET

Ah; for some reason I assumed the CLN check (including the supersingular check) was being done in SIKE decaps, but apparently not? Ah well

On Wed, Jun 15, 2022 at 1:06 PM Hovav Shacham <hovav@hovav.net> wrote:

> On Wed, Jun 15, 2022, at 11:14 AM, Deirdre Connolly wrote:
>
> > Would the long series of zeroes be caught as malformed with existing pubkey checks? I'm assuming not as a change was applied, but just wanted to check.
>
> Unfortunately no: The 0 doesn't show up until ladder round i (where m_i is the key bit being targeted) in our CCA attack and the section-3.1 variant of De Feo et al.'s attack (https://tches.iacr.org/index.php/TCHES/article/view/9701/9232). Even checking for 0s during the ladder isn't enough, because the section-3.2 variant of De Feo et al.'s attack has the 0 wait until early in isogeny computation to make an appearance.
>
> The attacker has a lot of freedom in picking the points P and Q, so my guess is that simple key checks, if added, could be bypassed as in Galbraith-Petit-Shani-Ti (https://www.iacr.org/archive/asiacrypt2016/10031307/10031307.pdf). Instead the right (SIKE-specific) mitigation is the full input validation described in section 5 of De Feo et al.
>
> As I understand it, that full validation is now implemented for PQCrypto-SIDH 5.3.1 and for CIRCL 1.2.0 by
>
> https://github.com/microsoft/PQCrypto-SIDH/commit/75ed5b09bd06a19cdad7660bef10e820074f808f and
>
> https://github.com/cloudflare/circl/commit/10923e8d736009130170c06c1bdbd81bee4de56c ,
>
> respectively.

> -hs.
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/dce735d7-181a-43c1-8010-c8d41ee3e26e%40www.fastmail.com.

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CAFR824zUq2BcPmQQb6H_AOWdaazLouc%3DCAUvfqWeLOAGdVf5nw%40mail.gmail.com.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 15, 2022 09:23:03 PM ET
Attachments: smime.p7m

Mike Hamburg writes:

> Sorry to miss that “constant” had switched meanings.

To be clear, it's the same meaning, always referring to cutting off all information flow from secrets. (Modulo declassification of the outputs, obviously; it's the job of the cryptosystem to ensure that its outputs are secure.)

For example, if a server CPU is always running at its base clock speed, then there's no information flow from secrets to the clock speed, so it's useless for attacks to inspect the clock speed, whether directly or indirectly via timings.

As a fancier example, since I'm not trying to hide when my laptop cores are idle, I don't care whether the cores switch to some super-low clock frequency when they're idle. There's again no information flow from secrets to the clock speed.

> It’s about, will it be difficult to produce real-world software that
> runs SIKE on today’s and tomorrow’s real computers and phones that’s
> secure against attack? If the answer is yes, then that’s an issue that
> makes SIKE less desirable, no matter whose fault it is.

But "less desirable" is a comparison to other cryptosystems, and without a real auditing mechanism there's no way to carry out this comparison!

Suppose some CPU misfeature means that an attacker can extract every byte of data stored at memory position 0 mod 4096, and someone decides to give an attack demo involving the official SIKE code. Penalizing SIKE on this basis, rather than blaming the CPU, would be unprincipled, prone to error (there's no reason to believe other software is immune), and prone to abuse (NISTPQC should be protected against being manipulated through choices of demos). How is this different from penalizing SIKE for HertzBleed?

> However, it seems that this issue is not excessively costly to mitigate,

You mean via the SIKE code changes that have happened? What's the basis for this evaluation?

If you're talking about the attack demo that worked against SIKE before the code changes and didn't work afterwards: The demo uses the sort of models and signal processing that you'd expect from attacks twenty years ago, far behind the state of the art appearing these days at CHES. That's adequate for saying "this is broken" but woefully inadequate for saying "this is secure". Best bet is that much more will be broken---except for users who set constant clock speeds.

> Nehalem’s droop mitigation gave about 5% frequency uplift at a given
> voltage. Not a huge benefit, but pretty big for such a small mechanism,
> and these things compound.

Sure, CPU manufacturers put a lot of effort into finding every little speedup they can.

> But most users do not turn off turbo boost or hyperthreading, and
> neither you nor I nor NIST can command them to do so.

Actually, NIST standards often explicitly impose requirements on users of the standards, and are based on security evaluations that assume that users follow those requirements. It doesn't make any sense to object to a security upgrade by saying that most users haven't upgraded yet.

> And it's no good to design crypto libraries that will be insecure
> (under practical, remote attacks) in the common case.

The common case is that buffer overflows allow complete remote system compromise, which, of course, can be artificially specialized to compromising the crypto libraries. Ergo, it's no good to design any crypto libraries?

The goal is a secure system. Getting to this goal requires work in many components of the system. When a specific information leak can be

* straightforwardly and auditably fixed in component X (the OS choice of CPU clock speed), or

* _maybe_ addressed through a complicated collection of unauditable changes in component Y (the crypto library),

it's clear what should be done. Saying that unfixed X is common, ergo it's no good to deploy Y without a fix, ergo the fix has to be Y's responsibility, is failing to address the relative merits of fixing X and fixing Y.

> I don’t know about you, but at my day job I spend a fair bit of time waiting
> for unvectorized, lightly-threaded code. Word and Outlook, Microsoft Teams,
> their auto-update features, the corporate mandated virus scanner, some stupid
> web app, etc. It’s not a giant time drain, but it’s slightly annoying every time
> I cmd-tab over to Outlook and it takes several seconds to come up.

Sounds like you should complain to Microsoft. I'm willing to bet, however, that they've heard about multi-core processors and vectors, and are already working through their list of top bottlenecks.

If your usage patterns are too far away from most people, then it might be a while before you see multithreading in the things you care the most about, but post-quantum standards should be looking farther ahead than that.

> And even threaded, vectorized code benefits from boost. I don’t have the
> all-core boost spec for the Core i7 1280P I mentioned earlier, but last generation’s
> flagship Core i7-1165G7 had a base clock of 2.8 GHz, all-core boost to 4.1,
> single-core boost to 4.7. Sure, it can’t boost as high or presumably as long on

> all cores, but there’s still a large benefit.

Those are _maximum_ numbers from the documentation. People who measure multithreaded vectorized code consistently find much lower Turbo Boost benefits.

> > Um, which assertion is this supposed to be disputing? Here's what I
> > actually wrote about my desired laptop: "I've also happily set my laptop
> > to constant speed for many years, more and more often choosing the
> > minimum speed that the laptop supports. It's increasingly rare that I'm
> > waiting for my laptop to compute something, and I don't like the laptop
> > heating up, especially from background activity on browser pages."
> I suppose it wasn’t actually an assertion, but an implication that running
> at constant (I take your statement here to mean “unchanging”) frequency
> is desirable for keeping laptops responsive but cool and quiet.

The ambiguity of "responsive" is doing a lot of work here.

My original text clearly indicated that even at low speed the laptop is already so fast that I'm rarely waiting for it. You slide to saying that I want something "responsive", and that Turbo Boost is designed to "minimize waiting" (subject to a temperature limit), ergo I want Turbo Boost. But, no, this is a nano-optimization to something that I already made clear _isn't_ a problem for me; even worse, a nano-optimization that frivolously turns background browser activity into CPU heat that I don't want.

> Fundamentally, the point of boosting is to minimize waiting while keeping
> the machine under a certain temperature. That’s ordinarily set to avoid
> damaging the processor

Why would a CPU manufacturer care about the long-term hardware damage caused by frequent overheating, as opposed to merely setting limits to avoid immediate failures?

For server CPUs, there's an answer to this, namely the threat that dead CPUs are called out in a year or two in reliability reports from big

server operators. Unsurprisingly, CPUs aimed at the server market have
lower thermal limits and lower frequency limits.

---D. J. Bernstein

--

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Thursday, June 16, 2022 04:41:51 AM ET

> On Jun 16, 2022, at 3:22 AM, D. J. Bernstein <djb@cr.yp.to> wrote:
>
> Mike Hamburg writes:
>> Sorry to miss that “constant” had switched meanings.
>
> To be clear, it's the same meaning, always referring to cutting off all
> information flow from secrets.

It quite clearly had a more specific meaning many of the times you used it. But I’m not here to argue semantics. I’m happy to use your meaning of “key-independent” for this discussion, and thus discard any objection involving heterogeneous cores (when scheduled in a power- and otherwise key-independent manner), load-dependent scaling etc.

>> It’s about, will it be difficult to produce real-world software that
>> runs SIKE on today’s and tomorrow’s real computers and phones that’s
>> secure against attack? If the answer is yes, then that’s an issue that
>> makes SIKE less desirable, no matter whose fault it is.
>
> But "less desirable" is a comparison to other cryptosystems, and without
> a real auditing mechanism there's no way to carry out this comparison!
>
> Suppose some CPU misfeature means that an attacker can extract every
> byte of data stored at memory position 0 mod 4096, and someone decides
> to give an attack demo involving the official SIKE code. Penalizing SIKE
> on this basis, rather than blaming the CPU, would be unprincipled, prone
> to error (there's no reason to believe other software is immune), and
> prone to abuse (NISTPQC should be protected against being manipulated
> through choices of demos). How is this different from penalizing SIKE
> for HertzBleed?

Any decision based on HertzBleed is premature, obviously, since the attack came out this week. (I could imagine circumstances where such an attack could delay an imminent announcement for more study, but I don’t think they apply either in the real world or in your example.)

The hypothetical scenario where I think it would make sense for SIKE to be “penalized” is if, after other authors had studied the attack, it appeared that SIKE were uniquely vulnerable to it – for example, because its main loop can get stuck at zero based on a key bit, whereas the same might not be true for eg Saber – and fixing the issue (with fixes that stand up to peer review) turned out to be excessively expensive.

Obviously it would be better, other things being equal, to have 100% reliable audits of all potential security issues. But I still don’t think users will pay the cost of that.

>> However, it seems that this issue is not excessively costly to mitigate,
>
> You mean via the SIKE code changes that have happened? What's the basis
> for this evaluation?

Again, this is my personal guess about this issue. Any real judgment is premature.

> Actually, NIST standards often explicitly impose requirements on users
> of the standards, and are based on security evaluations that assume that
> users follow those requirements. It doesn't make any sense to object to
> a security upgrade by saying that most users haven't upgraded yet.

It doesn’t make sense for NIST to require such a costly and narrow security upgrade, at least not for the general public. For HSM manufacturers, sure.

I also think that they shouldn’t base their decision on everyone having a crypto accelerator core in their laptop, even though this would arguably be cheaper, more secure and much more auditable.

>> And it's no good to design crypto libraries that will be insecure
>> (under practical, remote attacks) in the common case.
>
> The common case is that buffer overflows allow complete remote system
> compromise, which, of course, can be artificially specialized to
> compromising the crypto libraries. Ergo, it's no good to design any
> crypto libraries?

Perhaps my statement was too absolute.

But if one crypto library, or cryptosystem, were hypothetically much
more vulnerable to buffer overflows (e.g. if most existing and anticipated
near future implementations relied on some famously buggy ASN.1
parser), then this would surely be a strike against it?

>> I don’t know about you, but at my day job I spend a fair bit of time waiting
>> for unvectorized, lightly-threaded code. Word and Outlook, Microsoft Teams,
>> their auto-update features, the corporate mandated virus scanner, some stupid
>> web app, etc. It’s not a giant time drain, but it’s slightly annoying every time
>> I cmd-tab over to Outlook and it takes several seconds to come up.
>
> Sounds like you should complain to Microsoft. I'm willing to bet,
> however, that they've heard about multi-core processors and vectors, and
> are already working through their list of top bottlenecks.

Ah yes. The trajectory of Microsoft’s software clearly indicates a passion for
performance optimization. I’d bet Office 366 solves all these issues.

> But no, this is a nano-optimization to something that I already
> made clear _isn't_ a problem for me; even worse, a nano-optimization
> that frivolously turns background browser activity into CPU heat that I
> don't want.

Sure, you do you, sorry to offer advice. But for people with similar
(but not identical!) goals to you, the feature you’re bashing is useful.

— Mike

From: Bo Lin <bolinsco@gmail.com> via pqc-forum@list.nist.gov
To: Mike Hamburg <mike@shiftleft.org>
CC: Moody, Dustin (Fed)' via pqc-forum <pqc-forum@list.nist.gov>, D. J. Bernstein <djb@cr.yp.to>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Thursday, June 16, 2022 06:48:52 AM ET

Hi, Mike,
Thanks for your comments on my opinion on dynamic keys and the standardization.

If a static key is used, surely, no matter what countermeasures are implemented, implementation security of side-channel and fault injection must be assessed. It applied to any cryptosystems to be deployed, especially, in regulated markets.

For NIST standardization to take implementation security into account, it may not feasible because because as I mentioned before, implementation security is very much coupled with hardware features. Say, if a part does not leak, then what a side channel adversary can do? (I know, it depends on detection tools, just for an example here.) On the other hand, let me take the time-invariant implementation of the three-point Montgomery ladder as an example here, which is in hot discussion now in this thread. Say, if it is time-invariant, i.e., no time information leakage, but the skA scanning was implemented in if-statement which leaks, then the time-invariant implementation doesn't matter because an adversary can just target the leaky if-statement.

This is what I think the standardization should put the implementation security aside. Implementation security can be considered after the target hardware being selected and characterized. So, a product builder (such as an HSM provider) can protect the potentially vulnerable area.

Regards,

Bo

On Wed, 15 Jun 2022 at 12:06, Mike Hamburg <mike@shiftleft.org> wrote:

> Hi Bo Lin and Dan,
>
> > On Jun 15, 2022, at 10:47 AM, Bo Lin <bolinsco@gmail.com> wrote:

> The paper reads "Attack scenario. We target a **static key** version of SIKE where **a single secret key is used to decrypt several ciphertexts**..." This means it does not apply to the standard SIKE which changes its secret keys (skA and skB) every time. In general, for ephemeral keys, only the "single trace" attack is valid because if an adversary is not able to figure out a secret with a single trace, they need to try again. However, the prosed attack in this paper can be a starting point to study single trace attack by using template or machine learning.

SIKE is claimed to be CCA2-secure, and indeed CCA2-security is a useful feature in protocol design. Not all protocols will be ephemeral-only. For example, triple-DH-like systems use long-term KEM keypairs for authentication.

> Attacks against implementation are important for products to be deployed into regulated markets because a cryptosystems may fail in its security function not because of a wrong choice of cryptographic mechanisms or protocols, but because of an insecure implementation. The implementation security is related to many factors, especially the semiconductor technology. The security evaluation of side-channel attacks and fault injection attacks has been conducted for many year if we think about the 12 billion payment cards in circulation (EMVCo Reports 12 Billion EMV<sup>®</sup></sup> Chip Cards in Global Circulation - EMVCo).
>
> Because implementation security is tightly coupled with device manufacturing technology and with targeted markets, secure coding for implementation security can be left for product building. For NIST standardisation, crypto security has to be focused.

I agree in part. However, crypto standards must consider real-world implementation constraints, because they will be deployed in the real world. If a specific system did have a difficult-to-mitigate side-channel attack, I would expect that to affect its priority for standardization. For example, I would not expect NIST to standardize a symmetric cipher with sboxes that are far more efficient to implement with a large lookup table, unless there were other very compelling reasons to do so.

In the case of SIKE, it isn't being considered in the third round anyway. I expect that by the time it is considered for standardization, someone will have implemented and tested

> mitigations for this issue. Unless those mitigations are surprisingly expensive, I don’t expect this to be a serious problem long-term, but I’ve been wrong before.
>
> On Wed, 15 Jun 2022 at 06:07, D. J. Bernstein <djb@cr.yp.to> wrote:
>
>> Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem, and nicely composes with the auditability of the usual constant-time coding discipline. Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.
>>
>> Compared to running the CPU at a constant speed (and blaming the attack on variable CPU frequencies), why is it supposed to be better to try to modify code for specific cryptosystems (and to blame the attack on the unmodified code, or on the cryptosystems)?
>
> I assume you’re yelling at clouds here, but just to check: are you seriously arguing that CPUs, even laptop and phone CPUs, should universally be run at a constant frequency (not even just a workload-responsive but not thermally-responsive frequency) to mitigate crypto side-channel attacks? In other words, are you seriously arguing that manufacturers of laptops, servers and phones should throw out at least the following 5 optimizations?
>
> * Power- and thermally-adaptive frequency scaling (boost clocking).
>
> * Software workload-adaptive frequency scaling — that is, downclocking below nominal base frequency when idle to save power, and clock-gating and power-gating idle cores.
>
> * Frequency scaling in response to vectorization or other power-intensive features — eg, downclocking when AVX-256 or AVX-512 is in use.
>
> * Heterogeneous cores, such as performance cores and efficiency cores.
>
> * Extending clock periods or dropping cycles for hardware voltage droop mitigation. This lowers the required supply voltage and saves energy, and is present since at least Intel Nehalem.

> This is not going to happen industry-wide over a side-channel attack on SIKE, or even on several cryptosystems. Even just throwing out power- and thermally-adaptive scaling, the biggest culprit here, isn't remotely going to happen. It's like asking CPU manufacturers to completely throw out speculative execution over Spectre.
>
> We can't bend the world to our will on this, so like it or not, we will have to consider sufficiently-high-bandwidth power side channels in software design.
>
> > If there's supposed to be an "extreme system-wide" performance penalty
> > for the only auditable path to security, where's the quantification of
> > this penalty?
>
> Intel and AMD's recent high-end laptop offerings have these specs:
>
> Core i7-1280P (Alder Lake):
>
> E-core base 1.3 GHz, turbo to 3.6 GHz
>
> P-core base 1.8 GHz, turbo to 4.8 GHz
>
> AMD Ryzen 7 6800H: Base clock 3.2 GHz, boost up to 4.7
>
> AMD Ryzen 7 6800U: Base clock 2.7 GHz, boost up to 4.7
>
> So to first order, on an i7-1280P's performance cores, the performance advantage of thermally-adaptive boost is 2.66x. Of course, how much that affects your specific workload will depend on that workload, as well as your laptop's heatsinks and fans. Could they change the base frequency spec and demand better cooling from ultrabooks, to reduce this ratio? Sure. Are they going to do that over a side-channel in SIKE? Absolutely not.
>
> Note that boosting is primarily aimed at interactive workloads, not long-running compute-intensive ones. That is, it's primarily for launching an app or performing a CPU-intensive query during an otherwise lower-power session. By doing this, it makes a laptop or phone much more responsive within a given thermal envelope. So contrary to your assertion, it's a great fit for your desired cool-but-responsive laptop that's not used for heavy compute, at least if you can reduce the temperature threshold.

> It’s also useful for long-term CPU-intensive applications like gaming or video editing, if the system has more thermal headroom (e.g. a cleaner fan, a more thermally conductive tabletop, or a less heavily loaded GPU) than what was used to choose the spec.
>
> Regards,
>
> — Mike

--

From: Taylor R Campbell <campbell+nist-pqc-forum@mumble.net> via Taylor R Campbell <campbell@mumble.net>
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Tuesday, June 21, 2022 09:27:14 AM ET

> Date: Thu, 16 Jun 2022 03:22:31 +0200
> From: "D. J. Bernstein" <djb@cr.yp.to>
>
> Suppose some CPU misfeature means that an attacker can extract every byte of data stored at memory position 0 mod 4096, and someone decides to give an attack demo involving the official SIKE code. Penalizing SIKE on this basis, rather than blaming the CPU, would be unprincipled, prone to error (there's no reason to believe other software is immune), and prone to abuse (NISTPQC should be protected against being manipulated through choices of demos). How is this different from penalizing SIKE for HertzBleed?

This misfeature isn't merely hypothetical -- on Intel SGX, four generations of microarchitectures inadvertently exposed the first eight bytes of every cache line in the SGX secure enclave to the GPU, requiring all secrets in memory to be spread across noncontiguous 56-byte chunks starting at addresses congruent to 8 mod 64:

https://gcc02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fwww.intel.com%2Fcontent%2Fwww%2Fus%2Fen%2Fsecurity-center%2Fadvisory%2Fintel-sa-00219.html&amp;data=05%7C01%7Cyi-kai.liu%40nist.gov%7Cc25cf14ebf8944e11a8508da5389be04%7C2ab5d82fd8fa4797a93e054655c61dec%7C1%7C0%7C637914148341958799%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&amp;sdata=0QTQhfTe4Ud4jmjciZ4IR4pDksTto6y4L0GveLNc8XM%3D&amp;reserved=0

(No comment on whether it is feasible to convince all CPU designers and users on the planet to discard Turbo Boost on the grounds that Microsoft could have just make Outlook faster instead for the same benefit.)

That said, even assuming HertzBleed is an unavoidable fact of life, it seems unlikely to me that anyone has yet studied many cryptosystems or implementations to _rule out_ the adversary's power to influence the Hamming weight of secret-dependent intermediate quantities in the computation, or the Hamming distance of secret-dependent intermediate quantities across two computations -- so surely it is premature to penalize SIKE for that reason.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Tuesday, June 21, 2022 04:08:34 PM ET
Attachments: smime.p7m

Taylor R Campbell writes:

> (No comment on whether it is feasible to convince all CPU designers
> and users on the planet to discard Turbo Boost on the grounds that
> Microsoft could have just make Outlook faster instead for the same
> benefit.)

Regarding "same": Multithreading and vectorization typically improve response time by 10x. Exact numbers depend on variables such as the number of cores, but the benefit is much larger than the benefit of Turbo Boost. If Outlook's CPU usage is a real bottleneck for typical users then it's surprising that Microsoft hasn't already taken action.

It's also important to realize that, for applications that have already upgraded to multithreading and vectorization, the Turbo Boost speedup pretty much vanishes. (For examples and a review of why this happens, see https://timing.attacks.cr.yp.to/overclocking.html.) Basically, Turbo Boost is a temporary kludge sacrificing hardware reliability to produce somewhat less embarrassing response times _in unoptimized software_.

Should decisions regarding post-quantum standards, hopefully standards with many years of useful service, be driven by the sort of marketing numbers we've seen in this thread, such as "2.8 GHz, all-core boost to 4.1, single-core boost to 4.7"? Those numbers are already exaggerating today's speedup: they're cherry-picking the _best possible_ situation for Turbo Boost, and ignoring the much lower speedups that Turbo Boost brings to modern vectorized multithreaded software. Future users will see even more cores and more pervasive vectorization; how are we supposed to explain to those users that we're selecting unvectorized single-threaded software in making decisions for them?

Regarding "convince all CPU designers": Because Turbo Boost isn't a clear win, it has been normal for many years (although not universal) for computer manufacturers to give users options to disable Turbo Boost, as I mentioned earlier. So a security standard prohibiting Turbo Boost can already be widely deployed on today's computers. Yes, there are cases where hardware has to be upgraded, but this shouldn't stop deployment of a security tool for hardware that already supports it.

---D. J. Bernstein

P.S. The "extreme system-wide performance impact" claim has now been quietly removed, without an erratum. It was never backed by numbers in the first place. One wonders whether the source of that claim ever bothered measuring the impact of turning off Turbo Boost.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Tuesday, June 21, 2022 07:32:29 PM ET
Attachments: smime.p7m

Intel's concurrent paper on these attacks includes graphs showing how quickly an experiment extracted AES-128 key bits from the AES-NI hardware in various Intel CPUs:

   https://arxiv.org/abs/2206.07012

So, along with penalizing SIKE for the "attacks on SIKE", NIST should withdraw the AES standard in light of Intel's paper, right?

Or could it _possibly_ be that there will be many more of these demos, that the "attacks on SIKE" naming is horribly unfair to SIKE, and that the blame should actually be assigned to a different layer of the computer system, namely the CPU being configured to continually broadcast sensor measurements for the sake of an overhyped speedup?

——D. J. Bernstein

From: Wrenna Robson <wren.robson@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 22, 2022 11:40:16 AM ET

Have NIST publicly confirmed they will be penalizing SIKE for this?

On Wed, 22 Jun 2022 at 00:32, D. J. Bernstein <djb@cr.yp.to> wrote:

```
>
> Intel's concurrent paper on these attacks includes graphs showing how
> quickly an experiment extracted AES-128 key bits from the AES-NI
> hardware in various Intel CPUs:
>
>    https://gcc02.safelinks.protection.outlook.com/?url=https%3A%2F%2Farxiv.org%2Fabs%2F2206.07012&amp;data=05%7C01%7Cyi-kai.liu%40nist.gov%7Caf971feac9584d3aa8a408da54657f0c%7C2ab5d82fd8fa4797a93e054655c61dec%7C1%7C0%7C637915092162630404%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&amp;sdata=sKG5gTBt7bm90PeGTHd%2FEbxGq9lqhdPKBP6uRCdBCOY%3D&amp;reserved=0
>
> So, along with penalizing SIKE for the "attacks on SIKE", NIST should
> withdraw the AES standard in light of Intel's paper, right?
>
> Or could it _possibly_ be that there will be many more of these demos,
> that the "attacks on SIKE" naming is horribly unfair to SIKE, and that
> the blame should actually be assigned to a different layer of the
> computer system, namely the CPU being configured to continually
> broadcast sensor measurements for the sake of an overhyped speedup?
>
> ——D. J. Bernstein
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/20220621233157.640139.qmail%40cr.yp.to.
```

From: Daniel Apon <dapon.crypto@gmail.com> via pqc-forum@list.nist.gov
To: Wrenna Robson <wren.robson@gmail.com>
CC: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 22, 2022 11:43:33 AM ET

Obviously, no.

On Wed, Jun 22, 2022 at 11:40 AM Wrenna Robson <wren.robson@gmail.com> wrote:

> Have NIST publicly confirmed they will be penalizing SIKE for this?
>
> On Wed, 22 Jun 2022 at 00:32, D. J. Bernstein <djb@cr.yp.to> wrote:
> \>
> \> Intel's concurrent paper on these attacks includes graphs showing how
> \> quickly an experiment extracted AES-128 key bits from the AES-NI
> \> hardware in various Intel CPUs:
> \>
> \> https://arxiv.org/abs/2206.07012
> \>
> \> So, along with penalizing SIKE for the "attacks on SIKE", NIST should
> \> withdraw the AES standard in light of Intel's paper, right?
> \>
> \> Or could it _possibly_ be that there will be many more of these demos,
> \> that the "attacks on SIKE" naming is horribly unfair to SIKE, and that
> \> the blame should actually be assigned to a different layer of the
> \> computer system, namely the CPU being configured to continually
> \> broadcast sensor measurements for the sake of an overhyped speedup?
> \>
> \> ---D. J. Bernstein
> \>
> \> --
> \> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> \> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> \> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/

> pqc-forum/20220621233157.640139.qmail%40cr.yp.to.
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CANuKc3iahkH9hrUFtWvxFn01esZzujRn8ijcNsUqJ1A6wOcgag%40mail.gmail.com.

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CAPxHsS%2B_dJDxa8EsoaFZEg0c3fR1URXeuhK2DbXZzhRp6jBbSQ%40mail.gmail.com.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 22, 2022 05:21:42 PM ET
Attachments: smime.p7m

Wrenna Robson writes:

> Have NIST publicly confirmed they will be penalizing SIKE for this?

Given the widespread advertising of "attacks on SIKE", there's a clear _risk_ that NIST will penalize SIKE for this. https://www.hertzbleed.com says "attack on SIKE", as does the paper, so the title of this thread isn't surprising. We've already seen commentators writing things such as "If a specific system did have a difficult-to-mitigate side-channel attack, I would expect that to affect its priority for standardization" in a context strongly suggesting that this is an argument against SIKE.

It's not as if NISTPQC is a transparent process following clear rules that stop people from taking a failure elsewhere in the system and using that failure to selectively target SIKE. On the contrary, NIST's usual pattern has been to release only slow, limited dribbles of information about the NISTPQC decision-making process. NIST's round-2 report contains various erroneous claims that NIST hadn't made in public before the resulting decisions were set in stone. Much more of the NISTPQC decision-making process has been entirely hidden.

Given the lack of transparency, public errors have to be _presumed_ to influence the NIST process, and have to be proactively corrected so that they don't lead to bad decisions, even if NIST never acknowledges the importance of these corrections.

It would, of course, be a time-saver for NIST to promptly say something like this: "Given that CPU frequency variations allow an attack against existing, widely deployed AES hardware, we're going to presume that any system requiring security will disable CPU frequency variations. Evaluating the cost of protecting specific cryptosystems on systems with CPU frequency variations will require years of research, and there is little reason to believe that demonstrations of attacks against unprotected software are helpful for this evaluation. Consequently, while we encourage further research into this topic, attacks based on CPU frequency variations will be disregarded for decisions through the end of round 4 of the NIST Post-Quantum Standardization Project."

———D. J. Bernstein

--

From: Daniel Apon <dapon.crypto@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 22, 2022 05:26:02 PM ET

" Given the widespread advertising of "attacks on SIKE", there's a clear _risk_ that NIST will penalize SIKE for this."

Stop blustering.

On Wed, Jun 22, 2022 at 5:21 PM D. J. Bernstein <djb@cr.yp.to> wrote:

> Wrenna Robson writes:
> > Have NIST publicly confirmed they will be penalizing SIKE for this?
>
> Given the widespread advertising of "attacks on SIKE", there's a clear _risk_ that NIST will penalize SIKE for this. https://www.hertzbleed.com says "attack on SIKE", as does the paper, so the title of this thread isn't surprising. We've already seen commentators writing things such as "If a specific system did have a difficult-to-mitigate side-channel attack, I would expect that to affect its priority for standardization" in a context strongly suggesting that this is an argument against SIKE.
>
> It's not as if NISTPQC is a transparent process following clear rules that stop people from taking a failure elsewhere in the system and using that failure to selectively target SIKE. On the contrary, NIST's usual pattern has been to release only slow, limited dribbles of information about the NISTPQC decision-making process. NIST's round-2 report contains various erroneous claims that NIST hadn't made in public before the resulting decisions were set in stone. Much more of the NISTPQC decision-making process has been entirely hidden.
>
> Given the lack of transparency, public errors have to be _presumed_ to influence the NIST process, and have to be proactively corrected so that they don't lead to bad decisions, even if NIST never acknowledges the importance of these corrections.

> It would, of course, be a time-saver for NIST to promptly say something
> like this: "Given that CPU frequency variations allow an attack against
> existing, widely deployed AES hardware, we're going to presume that any
> system requiring security will disable CPU frequency variations.
> Evaluating the cost of protecting specific cryptosystems on systems with
> CPU frequency variations will require years of research, and there is
> little reason to believe that demonstrations of attacks against
> unprotected software are helpful for this evaluation. Consequently,
> while we encourage further research into this topic, attacks based on
> CPU frequency variations will be disregarded for decisions through the
> end of round 4 of the NIST Post-Quantum Standardization Project."
>
> ---D. J. Bernstein
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/20220622212108.708670.qmail%40cr.yp.to.

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CAPxHsSL-b51wGLSqj-E56n2i67pX%3D6ft1b12%2BP-hxUtxc%3D3UAw%40mail.gmail.com.

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Wednesday, June 22, 2022 07:00:37 PM ET

> On Jun 22, 2022, at 23:21, D. J. Bernstein <djb@cr.yp.to> wrote:
> We've already seen commentators writing things such as
> "If a specific system did have a difficult-to-mitigate side-channel
> attack, I would expect that to affect its priority for standardization"
> in a context strongly suggesting that this is an argument against SIKE.

That commentator was me, but I’ve also made it quite clear in this thread that I don’t think this should affect SIKE’s standardization chances, except under hypothetical conditions that I don’t expect to apply. I also don’t recall anyone else in this thread arguing that it hurts SIKE’s standardization case.

Also, did I miss SIKE getting promoted? I’d thought were were awaiting a choice of lattice schemes, a blessing on McEliece and possibly SPHINCS+ (which IIRC did get promoted?), with SIKE likely to follow in round 4. Since alternates were not prioritized for extra analysis, I’d thought they would not be standardized just yet.

But I must be missing something, because if the decision on SIKE isn’t going to be made for another year or two anyway, then there’s no point in an urgent campaign to prevent NIST from “penalizing“ it. The irrelevance of this attack to SIKE will be clear once followups come out attacking the rest of the finalists plus RSA, ECC and, in a surprise throwback, FEAL.

Regards,
— Mike

From:	D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To:	pqc-forum@list.nist.gov
Subject:	Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date:	Thursday, June 23, 2022 01:07:17 AM ET
Attachments:	smime.p7m

Mike Hamburg writes:

> But I must be missing something, because if the decision on SIKE isn’t
> going to be made for another year or two anyway

NIST has removed candidates in every round. What stops them from responding to high-profile "attacks on SIKE", and expert comments about how this maybe "makes SIKE less desirable", by removing SIKE right now?

Dustin Moody appeared to indicate in a talk earlier this year that NIST had already decided which finalists (plus potentially SPHINCS+) to select for standardization at the end of round 3. But the decision still hasn't been announced. Is the decision not subject to reconsideration in light of attacks? And does that decision include the decision of whether to continue considering SIKE in round 4?

People who think "small keys are great" or "elliptic curves are great" might think that SIKE is such an obvious candidate that _of course_ it will be in round 4 and won't be dragged under the water by these "attacks on SIKE". The same people also seem to have been surprised by NIST not taking SIKE as a finalist. NIST described SIKE as "a strong candidate for future standardization with continued improvements", but what if they're thinking that the SIKE improvements have run out of steam and that 99.9% of applications will prefer lattices?

I would prefer security to be the top priority, but we've never seen a clear statement from NIST regarding the degradation of lattice security against known attacks, never mind addressing more advanced security questions such as how to compare the relative risks of lattices and SIKE. NIST's round-2 report gave a remarkably flimsy security argument for eliminating NewHope, an argument that had appeared in zero previous NIST statements; how is anyone supposed to be confident that something similar won't happen to SIKE?

> then there’s no point in an urgent campaign to prevent NIST from
> “penalizing“ it.

There are various ways that NIST could issue statements ruling out an immediate problem for SIKE here. In the absence of any such statements, the "no point" claim is unsupported, just like the underlying notion that "the decision on SIKE isn’t going to be made for another year or two anyway".

I'm also skeptical that "another year or two" will be enough time to fix the systems-level mistakes that we've seen in this discussion if further discussion is delayed. Simpler errors within NISTPQC have sometimes persisted for years after being pointed out.

> I’ve also made it quite clear in this thread that I don’t think this
> should affect SIKE’s standardization chances,

Sorry, can you please quote where you made this clear? Quotes such as

* "In the case of SIKE … I don’t expect this to be a _serious_ problem _long-term_, _but I've been wrong before_" (emphasis added) and

* "If the answer is yes, then that's an issue that makes SIKE less desirable" and

* "personally I don’t think it makes SIKE _significantly_ less desirable" (emphasis added)

and so on all seem to be indicating that there could be an effect, which is _very_ different from saying that NIST should commit to disregarding these attacks for round-3/4 decisions. Here are more detailed quotes:

* "In the case of SIKE, it isn’t being considered in the third round anyway. I expect that by the time it is considered for

standardization, someone will have implemented and tested mitigations for this issue. Unless those mitigations are surprisingly expensive, I don’t expect this to be a serious problem long-term, but I’ve been wrong before."

* "It's about, will it be difficult to produce real-world software that runs SIKE on today’s and tomorrow’s real computers and phones that’s secure against attack? If the answer is yes, then that’s an issue that makes SIKE less desirable, no matter whose fault it is."

* "However, it seems that this issue is not excessively costly to mitigate, so personally I don’t think it makes SIKE significantly less desirable."

* "The hypothetical scenario where I think it would make sense for SIKE to be “penalized” is if, after other authors had studied the attack, it appeared that SIKE were uniquely vulnerable to it — for example, because its main loop can get stuck at zero based on a key bit, whereas the same might not be true for eg Saber — and fixing the issue (with fixes that stand up to peer review) turned out to be excessively expensive."

If the _attack on AES_ (ahem) has made you shift away from the positions expressed in the above statements, let me suggest that it would be best to issue clear retractions of these statements.

---D. J. Bernstein

P.S. While you're issuing retractions, please acknowledge that my first message in this thread explicitly said "constant speed, independent of the data being processed", and please withdraw your false claims that the word "constant" had switched meanings. Thanks in advance!

--

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Thursday, June 23, 2022 07:40:56 AM ET

Hi Dan, all,

> On Jun 23, 2022, at 7:06 AM, D. J. Bernstein <djb@cr.yp.to> wrote:
>
> Mike Hamburg writes:
>
>> But I must be missing something, because if the decision on SIKE isn't going to be made for another year or two anyway
>
> NIST has removed candidates in every round. What stops them from responding to high-profile "attacks on SIKE", and expert comments about how this maybe "makes SIKE less desirable", by removing SIKE right now?

Ah, I see. Maybe it's a perspective thing: while I don't always agree with NIST's decisions, I don't expect them to respond to a single side-channel attack paper (which already has reasonably cheap proposed mitigations) by immediately removing a system, such that it's necessary to vociferously argue against this possibility ahead of time.

Similarly, I don't expect them to outright remove SPHINCS+ due to the recent attacks based on the state size of SHA-256, so I didn't start a thread arguing that they shouldn't do this.

>> I've also made it quite clear in this thread that I don't think this should affect SIKE's standardization chances,

> Sorry, can you please quote where you made this clear?

You dropped the second half of this sentence, "except under hypothetical conditions that I don’t expect to apply”. I believe the quotes you chose make it pretty clear that:

* Hypothetically, if it turns out that SIKE is unfixably vulnerable to a side-channel attack that makes secure deployment much more difficult, then in my opinion this makes it less desirable to standardize it. (But, I would like to clarify now, IMHO this should only factor into the decision process, not determine it.)

* But if not, then not, and this is the outcome I expect.

* There’s no point in arguing about it now, because we have time to get a clearer picture.

> If the _attack on AES_ (ahem) has made you shift away from the positions expressed in the above statements, let me suggest that it would be best to issue clear retractions of these statements.

Yes. Here’s my clear retraction: if it turns out that SIKE is unfixably vulnerable, but no more than a variety of other systems, then I don’t think this affects SIKE’s desirability.

Note however that the SIKE paper is a remote attack, and the AES one is local.

Bonus retraction! If enough systems turn out to be vulnerable enough to this attack, then I expect we might be able to convince processor and OS vendors to include mitigations. I don’t expect temperature-responsive boost to go away entirely though.

> ---D. J. Bernstein
>
> P.S. While you're issuing retractions, please acknowledge that my first message in this thread explicitly said "constant speed, independent of the data being processed", and please withdraw your false claims that the word "constant" had switched meanings. Thanks in advance!

I acknowledge that you wrote this (with an article, “_a_ constant speed”, omitted in this demand for retraction). But in the context of that entire email, I believed that you were recommending to run at a literally constant speed, to the extent reasonably and safely possible. (E.g., I did not understand you to be recommending maintaining a fixed frequency even while the machine is asleep, shut down, or critically overheating.)

Anyway, let’s take this part of the argument off list. Nobody appreciates a semantics debate on-list.

Regards,

— Mike

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Thursday, June 23, 2022 03:20:18 PM ET
Attachments: smime.p7m

All available evidence is that these alleged "attacks on SIKE" are actually security failures in a completely different layer of the system, so the "attacks on SIKE" naming is misinformation, which at the time of this writing still hasn't been corrected.

Yes, the demo was specific to SIKE. It's natural for readers to leap to the conclusion that this is a minus for SIKE, as the following quote illustrates: "It’s about, will it be difficult to produce real-world software that runs SIKE on today’s and tomorrow’s real computers and phones that’s secure against attack? If the answer is yes, then that’s an issue that makes SIKE less desirable, no matter whose fault it is."

Some readers will spot the gap in the logic in this quote. The first sentence, the hypothesis about SIKE attacks, says nothing about attacks against other NISTPQC candidates, so how does this justify penalizing specifically SIKE?

All available evidence is that "today's and tomorrow's real computers" will in most cases continue to be plagued by, e.g., exploitable buffer overflows in browsers, so it's difficult to produce real-world software that runs _any_ cryptosystem on these computers and is secure against attack. The hypothesis is thus satisfied for this type of attack, but it would be absurd to claim on this basis that a _specific_ cryptosystem is less desirable.

Someone could also give an end-to-end demo of using an exploitable buffer overflow in a browser specifically to extract SIKE keys. So saying that there's an attack demo specific to SIKE doesn't separate HertzBleed from the buffer-overflow situation.

Cryptographers normally have narrow attack training—learning, e.g., about attacks exploiting RC4 biases, attacks certainly not applicable to AES-CTR---and will naturally _assume_ that an "attack against SIKE" isn't applicable to other NISTPQC candidates. But a systems-level perspective says that, no, this assumption is unjustified.

Saying that modified SIKE code resists the HertzBleed demo, ergo the hypothesis looks like it won't happen for these attacks, (1) doesn't eliminate the logical flaw in leaping from the hypothesis to the conclusion, (2) will sound remarkably shortsighted if attacks are demonstrated "against" the revised SIKE code (just like the attacks "against" AES), and (3) doesn't address the bigger problem of readers naturally leaping from "attacks on SIKE" to penalizing SIKE.

Saying that the "SIKE less desirable" conclusion didn't explicitly mention other NISTPQC candidates---it's just SIKE being less desirable now than it was before, nothing to do with anything else---would also be missing the point. The objective here is to select the best post-quantum submissions for standardization; readers are comparing systems in any case! Claiming a problem specifically for one submission has the natural effect of downgrading that submission, which is wrong when the available evidence says that the problem is _not_ specific to that submission.

Mike Hamburg writes:

> Hypothetically, if it turns out that SIKE is unfixably vulnerable to a
> side-channel attack that makes secure deployment much more difficult,
> then in my opinion this makes it less desirable to standardize it.

So NIST should reconsider the AES standard on the basis of the (very similar) side-channel attack in the new Intel paper? If this isn't a valid analogy, why not?

The targeted AES-NI hardware is baked into the widely deployed CPUs in question. Your previous comments strongly suggest that you don't view turning off Turbo Boost etc. as an acceptable fix. (This view directly flows from misinformation about performance: e.g., "2.8 GHz, all-core boost to 4.1", omitting the word _maximum_ and omitting discussion of how important this is.) AES-NI is very widely used in software for these

CPUs; trying to avoid it would require many software changes and large AES slowdowns (which I'd hope wouldn't be a big part of the user's performance picture, but if we're cherry-picking the maximum Turbo Boost benefit then why shouldn't we cherry-pick AES slowdowns?).

Surely you're not claiming, after the Intel paper, that current AES deployments are secure on these computers. If this example doesn't qualify as "unfixably" or "much more difficult", please clarify what you mean by those phrases.

> Note however that the SIKE paper is a remote attack, and the AES one is local.

Why is this difference supposed to be relevant?

Are you saying that software broken by local attacks can qualify as software running "on today’s and tomorrow’s real computers and phones that’s secure against attack"?

Do you suggest stopping browsers from locally running code that they've downloaded from whatever web sites? I think it's safe to say that most users would find this _vastly_ more painful than disabling Turbo Boost.

> There’s no point in arguing about it now, because we have time to get a clearer picture.

If this is based on the guesses you mentioned earlier (e.g., you don't "expect" NIST to remove SIKE at the end of round 3; SIKE is "likely" to be standardized at the end of round 4), please label it explicitly as a guess to avoid misleading readers. Sure, all your guesses _could_ be correct, but this doesn't justify overstating what's known at this point. Thanks in advance.

———D. J. Bernstein

--

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Thursday, June 23, 2022 05:17:50 PM ET

Hi Dan,

> On Jun 23, 2022, at 9:19 PM, D. J. Bernstein <djb@cr.yp.to> wrote:
> Some readers will spot the gap in the logic in this quote. The first
> sentence, the hypothesis about SIKE attacks, says nothing about attacks
> against other NISTPQC candidates, so how does this justify penalizing
> specifically SIKE?

It doesn't, obviously.

SIKE was selected for the demo because, at least with certain malformed public keys, it changes its power consumption drastically depending on a single key bit. This gives a high signal-to-noise ratio and enables a remote attack. Someone may find a way to make another system do that, or perhaps not. We don't know yet. My guess is that most other systems will see attacks, but that they won't be nearly as effective as the demo on SIKE — but that this will also be true for SIKE after the mitigation. But I don't know. This is just a guess.

The entire discussion is under a hypothetical that somehow the mitigation doesn't work, or otherwise SIKE does turn out to be noticeably more vulnerable than other candidates, just as it appeared to be in the demo. You could replace SIKE with any other system and the argument would be the same. I'm only arguing this hypothetical because I (mis?)understood you and Bo Lin to be arguing that this type of side channel attack would never matter for standardization purposes, even if the hypothetical case were true — either because the attack isn't the attacked cryptosystem's "fault", or because SIKE doesn't need to be CCA-secure anyway (in Bo Lin's email), or for whatever other reason.

>> Note however that the SIKE paper is a remote attack, and the AES one
>> is local.
>
> Why is this difference supposed to be relevant?

It’s funny how shades of attack severity and implementation difficulty and cost are a huge differentiating factor when it comes to elliptic curve shapes or block ciphers. For example, prime-order short Weierstrass curves are “unsafe”, in part because they don’t have a constant-time Montgomery ladder. This is a serious flaw, because while you could use a different constant-time algorithm, the lack of a simple constant-time Montgomery ladder presents a danger that implementers will make the wrong decision, making their code vulnerable to timing attacks.

Oh wait, they’ve had one since 2003! But it doesn’t count because it’s not the fastest implementation, so an implementer would be tempted to do something else, even though it’s insecure. Wait what, since 2017 a constant-time Montgomery ladder is the fastest implementation or at least competitive⁇ Eh well, still “unsafe” because they aren’t encodeable into strings that are indistinguishable from random, a feature that almost nobody needs, but which is provided by Elligator for Montgomery curves. Wait, that feature is clear in BCIMRT 2010, a paper we cited in Elligator? Doesn’t count because of … reasons. Better make a website.

But then when it comes to other systems, there’s no relevant difference between a local attack and a remote one. I mean, in both cases you aren’t perfectly secure, so y’know, same same.

> Do you suggest stopping browsers from locally running code that they've
> downloaded from whatever web sites? I think it's safe to say that most
> users would find this _vastly_ more painful than disabling Turbo Boost.

If it turns out that this allows practical local attacks from Javascript, and the only mitigation is to disable Turbo Boost (or like, DPA-harden AES and everything else in software), then I expect Turbo Boost to go away.

If it turns out that there is a less heavy mitigation that solves nearly all such attacks, I expect that to roll out instead, even if it doesn’t make the world a perfect place.

These are guesses though. Who knows what the future holds? I was pretty surprised that Spectre could be mitigated on existing hardware as well as it was.

>> There’s no point in arguing about it now, because we have time to get
>> a clearer picture.
>
> If this is based on the guesses you mentioned earlier (e.g., you don't
> "expect" NIST to remove SIKE at the end of round 3; SIKE is "likely" to
> be standardized at the end of round 4), please label it explicitly as a
> guess to avoid misleading readers. Sure, all your guesses _could_ be
> correct, but this doesn't justify overstating what's known at this
> point. Thanks in advance.

Are you sure you don’t want to demand another retraction, while you’re bothering to thank me in advance?

— Mike

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Thursday, June 23, 2022 10:02:00 PM ET
Attachments: smime.p7m

Mike Hamburg writes:

> It’s funny how shades of attack severity and implementation difficulty and
> cost are a huge differentiating factor when it comes to elliptic curve
> shapes or block ciphers.

The objective is to get to a secure system, one that enforces the specified security policy. An essential part of comparing security plans is measuring the difficulty of implementing those plans.

I have no idea why you lump "attack severity" into this, especially in response to my question about how "Note however that the SIKE paper is a remote attack, and the AES one is local" is supposed to be relevant. The usual security policy includes stopping local attacks. This is motivated by user demand to run untrusted code locally, for example in browsers, and it's not as if anyone has articulated a plan for changing this.

——D. J. Bernstein

P.S. I should note that measuring implementation difficulty generally requires tremendous attention to detail, and isn't competently handled by——just to take a hypothetical example——the sort of person so sloppy as to publicly make confident implementation-related claims that were already debunked years earlier by https://eprint.iacr.org/2019/1166.

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Friday, June 24, 2022 03:21:16 AM ET
Attachments: smime.p7m

Some NIST ex-employee writes:

> Stop blustering.

Ah, yes, the same source who wrote "Stop propagandizing" when I wrote the following: "Within NISTPQC, MQDSS advertised a loose proof, violated the requirement from the above papers to choose key sizes accordingly, and was broken. Round2 advertised a proof for its CCA conversion, but the proof was incorrect, and Round2 was broken. NISTPQC should respond by adding procedural protections against loose proofs and unreviewed proofs. This could have been done earlier, but better late than never."

That source has repeatedly and flagrantly violated the announced NIST policy: "NIST strongly discourages dishonesty, misrepresentations of science, personal attacks, and any form of hostility." What action has NIST taken in response to these violations?

——D. J. Bernstein

From: Daniel Apon <dapon.crypto@gmail.com> via pqc-forum@list.nist.gov
To: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Friday, June 24, 2022 03:33:27 AM ET

Thanks, Dan. Happy July 4th.

On Fri, Jun 24, 2022 at 3:21 AM D. J. Bernstein <djb@cr.yp.to> wrote:

> Some NIST ex-employee writes:
> \> Stop blustering.
>
> Ah, yes, the same source who wrote "Stop propagandizing" when I wrote the following: "Within NISTPQC, MQDSS advertised a loose proof, violated the requirement from the above papers to choose key sizes accordingly, and was broken. Round2 advertised a proof for its CCA conversion, but the proof was incorrect, and Round2 was broken. NISTPQC should respond by adding procedural protections against loose proofs and unreviewed proofs. This could have been done earlier, but better late than never."
>
> That source has repeatedly and flagrantly violated the announced NIST policy: "NIST strongly discourages dishonesty, misrepresentations of science, personal attacks, and any form of hostility." What action has NIST taken in response to these violations?
>
> ---D. J. Bernstein
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/20220624072047.806186.qmail%40cr.yp.to.

--
You received this message because you are subscribed to the Google Groups "pqc-forum"

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Sunday, June 26, 2022 02:29:27 PM ET
Attachments: smime.p7m

Mike Hamburg writes:

> It quite clearly had a more specific meaning many of the times you used it.

No, "constant" throughout all my messages in the thread always means independent of the data being processed. This concept plays an important role in defending against timing attacks, and is the normal usage of the word "constant" in the same context. This meaning is also stated in the text the first time the word "constant" appears, very close to the top of my first message in the thread:

> Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem, and nicely composes with the auditability of the usual constant-time coding discipline.

The text immediately continues with an example of an action item, of course using "constant" with the same meaning:

> Constant speed isn't the typical OS default these days, but there are tools to set it up. See, e.g., https://bench.cr.yp.to/supercop.html for Linux instructions to disable Turbo Boost, Turbo Core, and downclocking.

It's easy to see how this action item would trigger an objection from someone who (1) isn't proactive about security, (2) has an overinflated idea of the Turbo Boost benefits, and (3) is somehow confident that downclocking can't create data dependence. I'm happy to discuss the contents of this objection. It is, however, _highly_ inappropriate for someone to try to add weight to this objection by pretending that the objection triggered a retroactive change in the meaning of the text.

---D. J. Bernstein

P.S. For students learning the terminology: This usage of "constant" is an example of a more general notion of "constant" used in mathematics, which in turn is the default usage of "constant" across a wide range of technical discussions.

The general notion is of being unaffected by some variables. One then has to specify _which_ variables (e.g., "the data being processed"). Formally, such a specification is distinguishing "parameters" from "inputs" to a function. "Constant" then means that, for each choice of parameters, the value of the function doesn't depend on the input.

For example, if integer variables c,d are both designated as inputs, then c^2 isn't constant; i.e., the function F from \Z^2 to \Z defined by F(c,d) = c^2 isn't constant; i.e., c^2 isn't independent of (c,d).

If, however, c is designated as a parameter and d is designated as an input, then c^2 _is_ constant; i.e., the function G_c from \Z to \Z defined by G_c(d) = c^2 is constant; i.e., c^2 is independent of d.

In the previous paragraph, the constant c^2 is not an _absolute_ constant. An "absolute constant" refers to something unaffected by _all_ variables: in other words, saying "absolute constant" is designating _all_ variables as inputs rather than parameters. The integer 31415 is an absolute constant.

In the context of defending against timing attacks, it's important to ask whether the time taken by a computation is influenced not just by the amount of data provided to the computation but also by the contents of the data. For example, say there are n items of input, where n is a parameter, and define T_n(x) as the time spent on an n-item input x. One then asks whether T_n is constant; i.e., whether the time taken by the computation is constant; i.e., whether the computation is constant-time.

This is how the "constant-time" terminology arose as an example of the usual mathematical terminology. Notice how important it is to specify which variables are inputs: designating n as an input would change which computations qualify as constant-time.

--

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Sunday, June 26, 2022 02:38:15 PM ET

Hello Dan, all,

Consistent with my belief and previous statement that nobody on
this list wants to read a semantic argument, and given that the
off-list semantic argument went on for days and went nowhere, I
will not be further arguing this point.

Regards,
— Mike

```
> On Jun 26, 2022, at 8:28 PM, D. J. Bernstein <djb@cr.yp.to> wrote:
>
> Mike Hamburg writes:
>> It quite clearly had a more specific meaning many of the times you used it.
>
> No, "constant" throughout all my messages in the thread always means
> independent of the data being processed. This concept plays an important
> role in defending against timing attacks, and is the normal usage of the
> word "constant" in the same context. This meaning is also stated in the
> text the first time the word "constant" appears, very close to the top
> of my first message in the thread:
>
>   Running the CPU at a constant speed, independent of the data being
>   processed, is the obvious, straightforward, auditable way to
>   eliminate this problem, and nicely composes with the auditability of
>   the usual constant-time coding discipline.
>
> The text immediately continues with an example of an action item, of
> course using "constant" with the same meaning:
>
>   Constant speed isn't the typical OS default these days, but there are
```

>   tools to set it up. See, e.g., https://gcc02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fbench.cr.yp.to%2Fsupercop.html&amp;data=05%7C01%7Cyi-kai.liu%40nist.gov%7C16bd52e717194e05952808da57a30695%7C2ab5d82fd8fa4797a93e054655c61dec%7C1%7C0%7C637918654955762951%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&amp;sdata=RqymUw5Kt74LazonnxKibicQ9U2%2FN7oa9I%2Fpa9AGVjE%3D&amp;reserved=0
>   for Linux instructions to disable Turbo Boost, Turbo Core, and
>   downclocking.
>
> It's easy to see how this action item would trigger an objection from
> someone who (1) isn't proactive about security, (2) has an overinflated
> idea of the Turbo Boost benefits, and (3) is somehow confident that
> downclocking can't create data dependence. I'm happy to discuss the
> contents of this objection. It is, however, _highly_ inappropriate for
> someone to try to add weight to this objection by pretending that the
> objection triggered a retroactive change in the meaning of the text.
>
> ——D. J. Bernstein
>
> P.S. For students learning the terminology: This usage of "constant" is
> an example of a more general notion of "constant" used in mathematics,
> which in turn is the default usage of "constant" across a wide range of
> technical discussions.
>
> The general notion is of being unaffected by some variables. One then
> has to specify _which_ variables (e.g., "the data being processed").
> Formally, such a specification is distinguishing "parameters" from
> "inputs" to a function. "Constant" then means that, for each choice of
> parameters, the value of the function doesn't depend on the input.
>
> For example, if integer variables c,d are both designated as inputs,
> then c^2 isn't constant; i.e., the function F from \Z^2 to \Z defined by
> F(c,d) = c^2 isn't constant; i.e., c^2 isn't independent of (c,d).
>
> If, however, c is designated as a parameter and d is designated as an
> input, then c^2 _is_ constant; i.e., the function G_c from \Z to \Z
> defined by G_c(d) = c^2 is constant; i.e., c^2 is independent of d.

```
>
> In the previous paragraph, the constant c^2 is not an _absolute_
> constant. An "absolute constant" refers to something unaffected by _all_
> variables: in other words, saying "absolute constant" is designating
> _all_ variables as inputs rather than parameters. The integer 31415 is
> an absolute constant.
>
> In the context of defending against timing attacks, it's important to
> ask whether the time taken by a computation is influenced not just by
> the amount of data provided to the computation but also by the contents
> of the data. For example, say there are n items of input, where n is a
> parameter, and define T_n(x) as the time spent on an n-item input x. One
> then asks whether T_n is constant; i.e., whether the time taken by the
> computation is constant; i.e., whether the computation is constant-time.
>
> This is how the "constant-time" terminology arose as an example of the
> usual mathematical terminology. Notice how important it is to specify
> which variables are inputs: designating n as an input would change which
> computations qualify as constant-time.
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/20220626182836.986632.qmail%40cr.yp.to.

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/94C7354F-5B69-49C2-B7B5-07727331C1D1%40shiftleft.org.
```

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Sunday, June 26, 2022 04:07:57 PM ET
Attachments: smime.p7m

Unfortunately, we're still in the situation of uncorrected public misrepresentations being used to inappropriately add weight to what is, as far as I can tell, unfair treatment of SIKE.

Procedurally, I would _hope_ that these so-called "attacks on SIKE" and the supporting stunts won't have an immediate impact on selections for round 4, but I still see nothing from NIST guaranteeing this.

Mike Hamburg writes:
> given that the off-list semantic argument went on for days and went
> nowhere

Since Mike has decided to issue an incorrect public characterization of an off-list discussion, I believe I'm authorized to give quotes disproving this characterization.

Regarding the text "Running the CPU at a constant speed, independent of the data being processed" etc., the off-list discussion included Mike

* admitting that this "can be read" as "constant speed behavior, where ‘constant’ means 'independent of the data being processed’";

* admitting that this "is a common definition and makes sense, so maybe I should have taken that meaning"; and, most recently,

* _appearing_ to admit that this meaning is "not inconsistent" with anything in any of my messages in the thread.

I'm saying "appearing" because, unfortunately, the last admission didn't clearly and explicitly specify its scope. But this is certainly progress towards correcting Mike's public claims that "quite clearly" the word "had a more specific meaning many of the times you used it" (whatever exactly that alternative meaning was supposed to be) and thus "had switched meanings".

So, no, it's obviously not true that the discussion "went nowhere". To be clear, one part of this, the past tense in "went", does seem to be correct: Mike decided to unilaterally terminate the off-list discussion before reaching resolution, even though he keeps saying that nobody wants to read the dispute on-list.

---D. J. Bernstein

--

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Sunday, June 26, 2022 05:59:17 PM ET

Hi Dan,

> On Jun 26, 2022, at 10:07 PM, D. J. Bernstein <djb@cr.yp.to> wrote:
> Unfortunately, we're still in the situation of uncorrected public
> misrepresentations being used to inappropriately add weight to what is,
> as far as I can tell, unfair treatment of SIKE.

Is anyone here treating SIKE unfairly? I thought everyone in the discussion had agreed that SIKE shouldn’t be penalized based on this paper.

> Since Mike has decided to issue an incorrect public characterization of
> an off-list discussion, I believe I'm authorized to give quotes
> disproving this characterization.

Really? You believe that since I said the discussion "went nowhere" (this is, by the way, an accurate characterization) that you are “authorized” to selectively quote from a multi-thousand word off-list discussion? This is an inexcusable breach of etiquette and you know it.

I will not take this as an excuse to selectively quote your messages from this discussion. But, I will lay out my understanding here. I had hoped to avoid this because I consider it so in-the-weeds as to be spammy, but here goes.

When talking about “running the machine at xxx speed”, I imagined, apparently incorrectly, that you meant one of two things:

(1) Constraining the speed behavior. If asked to formalize this, I would say that the speed behavior is something like the distribution of the time history of speed, possibly as a component of a joint distribution with the time histories of other events on the machine.

It really is a distribution and not a function, since new inputs are arriving continually, and (for example) the speed of a network server also affects what data it processes at what time and in what order.

In this case, “Running the CPU at a constant speed” would mean causing the speed component of the behavior to be constant for that CPU while it is running, and “independent of the data being processed” would mean the usual statistical notion. The appositive construction reinforces that “running the CPU at a constant speed” gives a speed which is “independent of the data being processed”.

(2) Or, choosing the policy for the speed control, which is roughly a function taking (time, load, temperature and power parameters) and perhaps a few other arguments and outputting a target CPU frequency. This is a fairly straightforward understanding, because it’s the part that the user (or at least their software) actually controls, and it’s something that you gave directions for changing.

This function doesn’t take the data being processed as an input, nor does it take inputs which are directly functions of the data being processed, only which covary with the data being processed. So it would be strange to describe it as constant with respect to that data. But it can be both constant with respect to all its inputs (the CPU is a parameter and not an input, since it’s already qualified in the construction “Running the CPU at a constant speed”), and also independent of the data being processed.

I think either of these interpretations is a relatively natural reading and is largely consistent with the rest of your message. Perhaps (1) is more consistent with the term “independent” and (2) is more consistent with your examples. I thus took (2) to be your meaning.

I didn’t consider the following third model, which I now understand to be more in line with your meaning:

(3) “Speed” means a function from (input data, temperature, lots of other stuff, time) to (instantaneous frequency settings of each core at that time), or perhaps without time as an input, to (time histories of the frequency settings of each core). This isn’t the speed controller policy per se: it’s a more complex object with many more inputs, but it filters through that controller.

In that case, I would take “running the CPU at a constant speed, independent of the data being processed” to mean “causing the CPU’s speed, as a function of [all those things], to be constant (in a generalized sense meaning “independent”, since if the CPU is eg load-dependent but not data-dependent, then the behavior is likely to be nondeterministic) with respect to the data inputs, insofar as those inputs do not flow to timing or system controls."

Did I finally get that roughly correct, or did I misunderstand you again and you meant yet another thing?

In any case, please pardon me for misunderstanding your message, my on-list response, and the ensuing off-list semantic argument. I personally still consider (1) and (2) to be more natural readings of your post than (3), but I do not wish to try to convince you or others of it on list.

Regards,
— Mike

From: D. J. Bernstein <djb@cr.yp.to> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Sunday, June 26, 2022 09:10:39 PM ET
Attachments: smime.p7m

Mike Hamburg writes:

> Is anyone here treating SIKE unfairly?

Yes. For a discussion of what's unfair here (given the evidence available), see, e.g., my email dated 23 Jun 2022 21:19:45 +0200, especially the initial paragraphs not quoted in your reply.

> I thought everyone in the discussion had agreed that SIKE shouldn’t be
> penalized based on this paper.

Really? Where? I don't see where various unjustified anti-SIKE snipes have been withdrawn (see, e.g., the quote in the second paragraph in the message mentioned above). I don't see any protections against the damage caused by the "attacks on SIKE" naming. Also, this list isn't an island; I still see https://www.hertzbleed.com saying "attack on SIKE".

More to the point, we've still seen no comments from NIST. What assurances are there that NIST won't immediately take an "attack on SIKE" as a negative for SIKE, possibly even negative enough to push SIKE under the water line for surviving past the end of round 3? It takes extra work to investigate and realize that the available evidence does _not_ support the natural reaction to an "attack on SIKE".

People confident that SIKE is going to make it anyway (or confident that it has drowned already!) _could_ be right. But maybe NIST was thinking "Hmmm, speedups seem to have stalled, applications seem to prefer lattices, not sure SIKE is useful to keep around, but we do like the arguments for SIKE's side-channel resistance"——and then what happens when suddenly there's a side-channel "attack on SIKE"?

> > Since Mike has decided to issue an incorrect public characterization of

> > an off-list discussion, I believe I'm authorized to give quotes
> > disproving this characterization.

> Really? You believe that since I said the discussion "went nowhere"
> (this is, by the way, an accurate characterization) that you are
> “authorized” to selectively quote from a multi-thousand word off-list
> discussion? This is an inexcusable breach of etiquette and you
> know it.

Publish a false characterization of an off-list discussion, and then complain about "an inexcusable breach of etiquette" when there's a response giving quotes disproving the characterization? And, as icing on the cake, repeat the same false characterization, not addressing the glaring contradiction with the quotes? Impressive.

Regarding the false claim that triggered the discussion ("It quite clearly had a more specific meaning many of the times you used it"), I don't see where a defense has been laid out for this claim.

The actual usage of the word "constant" in all of my messages in this thread is normal in discussions of defenses against timing attacks, and (just in case the reader isn't already familiar with it) also noted in the first message the first time the word is used: "Running the CPU at a constant speed, independent of the data being processed, is the obvious, straightforward, auditable way to eliminate this problem" etc.

I'm baffled by now seeing convoluted attempts to argue that the text actually has three different possible readings. Structurally, arguing that other readings are _possible_, or even _plausible_, is failing to allege any problems with the simple, straightforward reading of the text as writing "constant" to mean "independent of the data being processed", and is failing to support the claim that the text "quite clearly" meant something else.

——D. J. Bernstein

--

From: Mike Hamburg <mike@shiftleft.org> via pqc-forum@list.nist.gov
To: D. J. Bernstein <djb@cr.yp.to>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] HertzBleed : power side channel attacks on SIKE
Date: Sunday, June 26, 2022 11:21:14 PM ET

> On Jun 27, 2022, at 3:09 AM, D. J. Bernstein <djb@cr.yp.to> wrote:
>
> Mike Hamburg writes:
>> Is anyone here treating SIKE unfairly?
>
> Yes. For a discussion of what's unfair here (given the evidence
> available), see, e.g., my email dated 23 Jun 2022 21:19:45 +0200,
> especially the initial paragraphs not quoted in your reply.

What? The first three paragraphs of my reply to that email clarify that this is not intended as a snipe against SIKE, and in fact disclaim any such meaning.

To clarify yet again, I do not see this paper as a basis for disqualifying or otherwise penalizing SIKE, nor as evidence that SIKE will be more vulnerable to this sort of attack than other systems.

If after extensive research, some system turns out to be more vulnerable to this kind of attack than other systems, and if it does not appear that Turbo Boost will go away, then I hope NIST will consider that information.

>>> Since Mike has decided to issue an incorrect public characterization of
>>> an off-list discussion, I believe I'm authorized to give quotes
>>> disproving this characterization.
>> Really? You believe that since I said the discussion "went nowhere"
>> (this is, by the way, an accurate characterization) that you are
>> “authorized” to selectively quote from a multi-thousand word off-list
>> discussion? This is an inexcusable breach of etiquette and you
>> know it.

>
> Publish a false characterization of an off-list discussion, and then
> complain about "an inexcusable breach of etiquette" when there's a
> response giving quotes disproving the characterization? And, as icing on
> the cake, repeat the same false characterization, not addressing the
> glaring contradiction with the quotes? Impressive.

I’m not going to relitigate this, Dan. If people are convinced by your argument and your etiquette, let them be convinced.

Regards,
— Mike