Smaller **Sphincs**$^+$

or, Honey, I Shrunk the Signatures

Scott Fluhrer[1] Quynh Dang[2]

[1] Cisco Systems
[2] National Institute of Standards and Technology, USA
January 14, 2025

Abstract

NIST published **FIPS 205** [1] based on the specification of **Sphincs**$^+$[2]. A formula to determine the security strength of a given parameter set is listed in [2]. It is quite complex to use that formula to get the security degradation behavior based on different increases in the number of signatures (called 2^m in this paper) per signing key. The task would become even more complex when we need to compare the security degradation characteristics of many parameter sets. In this paper, we provide a simpler formula to determine the security strengths of a given parameter set at various numbers of signatures produced by one signing key. With this new formula, the task of comparing parameter sets, especially, their security degradation characteristics become easy and that allowed us to search for best parameter sets for users to consider to use and for standard bodies to consider for standardization.

We generated **Sphincs**$^+$parameter sets, then grouped them based on a Selection Set which has 4 variables as following:

- The security level of the parameter sets (either 128, 192 or 256 bits).
- The maximum number of signatures allowed to be generated under 1 signing key (either 2^{10}, 2^{20}, 2^{30}, 2^{40} or 2^{50}) and retain the full security level.
- How much overuse can happen and still retain a lower security level (either 112, 128 or 192 bits).
- The limit for the number of hashes per signing operation (either $100,000$, $1,000,000$, $10,000,000$, $100,000,000$, $600,000,000$ or $1,000,000,000$).

The parameter sets under any Selection Set presented in this paper give us different trade-offs between signing time, verification time, signature size and *overuse safety*. We evaluated the parameter sets and point out the ones we thought may be of interest to the users and standard bodies. It is easy for users to compare those parameter sets and determine which one would work best for their specific use case.

Contents

1 Introduction **2**
1.1 Terminology note 2

2 Previous Work **3**

3 Background of Sphincs$^+$ **3**

4 Sphincs$^+$overview **4**

5 Sphincs$^+$Usage Limitations **5**

6 Tuning Sphincs$^+$ **6**

7 Overuse **7**

8 Small Parameter Sets **8**
8.1 Note on Performance . . . 9
8.2 Differences between the parameter sets of this version of its previous version. . . . 10
8.3 Initial comments about overuse of small parameter sets . . . 10

9 Level 1 Security Parameter Sets **12**
9.1 Level 1 security at 2^{20} signatures . . . 12
9.2 Level 1 security at 2^{30} signatures . . . 15
9.3 Level 1 security at 2^{40} signatures . . . 17
9.4 Level 1 Security at 2^{50} signatures . . . 19

10 Level 3 Security Parameter Sets **20**
10.1 Level 3 security at 2^{20} signatures . . . 21
10.2 Level 3 security at 2^{30} signatures . . . 22
10.3 Level 3 Security at 2^{40} signatures . . . 23
10.4 Level 3 security at 2^{50} signatures . . . 24

11 Conclusions **25**

A References **26**

B Additional Parameter Sets at Security Levels 1 and 3 **27**
B.1 Additional Parameter Sets at Security Level 1 . . . 27
B.2 Additional Parameter Sets at Security Level 3 . . . 29

C Parameter Sets at 2^{10} Signature Limit at Security Levels 1 and 3 **32**
C.1 Security Level 1 at 2^{10} Signature Limit . . . 32
C.2 Security Level 3 at 2^{10} Signature Limit . . . 34

D Level 5 Security Parameter Sets **35**
D.1 Security Level 5 at 2^{10} Signature Limit . . . 35
D.2 Security Level 5 at 2^{20} Signature Limit . . . 36
D.3 Security Level 5 at 2^{30} Signature Limit . . . 38
D.4 Security Level 5 at 2^{40} Signature Limit . . . 39
D.5 Security Level 5 at 2^{50} Signature Limit . . . 39

1 Introduction

NIST has published **FIPS 205** [1], which is the specification of **SLH-DSA** (also known as **Sphincs**$^+$). When NIST released its original call for proposals for the post-quantum standardization project, they specified that signature systems would need to be usable at full security for 2^{64} signatures per private key. Hence, the parameter sets specified in **SLH-DSA** are tuned to have full security after that many signatures. However, it has been noted that in many cases, we don't have any need for that many signatures, and that parameter sets tuned for fewer signatures would be shorter and/or more efficient to process. This paper examines such possible alternative parameter sets.

Keywords: Postquantum Signatures, **Sphincs**$^+$, **SLH-DSA**, **FIPS 205**, Stateless Hash-based Signature Algorithms.

1.1 Terminology note

When we refer to **Sphincs**$^+$restricted to the parameter sets specified in **FIPS 205** [1], we will refer to the system as **SLH-DSA**. When we refer to a more general parameter set, we will refer to the system as **Sphincs**$^+$.

2 Previous Work

There have been several attempts to optimize the **Sphincs**$^+$structure.

One early attempt was **Gravity-Sphincs**[3], which was a submission to Round 1 of the NIST PQC competition. It introduces such innovations such as a large top-level Merkle tree (in order to allow faster signing using precomputation), and **Probabilistic Oracle Signature Scheme (PORS)**, a few-time signature scheme that could provide for shorter signatures at times (at the cost of complexity and having variable signature size). More interestingly, **Gravity-Sphincs** all provided alternative parameter sets targeted at smaller signature limits, in this case, 2^{50} signatures, in addition to the 2^{64} limit parameter sets requested by NIST.

Sphincs$^+$C [4] is another attempt to improve the **Sphincs**$^+$structure, this time by optimizing the structure itself, rather than the parameter sets. There are two main improvements. The first is to require certain bits from the initial randomized hash to be a specific value. This increases the amount of effort an attacker would need per forgery attempt, which (by effectively dividing p from Equation 1) by a constant factor, allows the signer to use less expensive parameters. The other improvement is to the **Winternitz One-Time Signature Plus (WOTS)** scheme and is based on the fact that the distribution of the checksum values is far from an uniformly random distribution. Certain checksum values have much higher probabilities to be hit than the others. Therefore, a checksum value with the highest hit probability is chosen as a constant checksum and the signing process keeps hashing the message to be signed with different counter values until the checksum of a hash value hits the constant checksum. Similarly, one or more of the **Winternitz** digits can be eliminated by keeping hashing until a predetermined part of a hash value to be all zeros. While both do improve the performance, they both require larger changes to the implementation than we are willing to consider for this study.

Sphincs $-\alpha$ [5] specifies a new **One-Time-Signature (OTS)** scheme called **Balanced WOTS+ (WOTS Plus)**. The scheme uses a different message encoding method instead of the checksum in **Sphincs**$^+$. The method produces 1 less **Winternitz** digit (hash chain) for most parameter sets. The **Balanced WOTS+** also adds a variant of **Forest of Random Subsets (FORS)**, a few times signature scheme, which has a slight improvement in security. In our opinion, the changes are substantial from **Sphincs**$^+$, and larger than what we are willing to consider.

In addition, there has been work[6] and [7] which limited the changes to the parameters specified in **Sphincs**$^+$(similar to our work). However, those papers did not provide a full analysis of security degradation nor a simple formula to determine/approximate the security strength at each number of messages signed per a private key, which is critical in determining how safely parameter sets are in comparisons with each other when overuse happens. The paper provided only a handful of parameter sets, with only a limited number of trade-offs between signature size, signing time, verification time and security degradation. Our paper provides a much larger set of options which give users and standard bodies more freedom when deciding what parameter sets are best to use and to standardize respectively.

3 Background of Sphincs$^+$

A hash based signature is a signature scheme that relies solely on the strength of an underlying hash function. In this system, the signer has a number of hash preimages, and when he generates a signature, he reveals several of those preimages (which the verifier can validate). Because the signer has a finite number of hash preimages, there is always a bound on the number of signatures he can safely generate. However, in practice this bound can be made quite large.

One easy way to generate signatures with hashes is to use a one-time signature scheme, such as **Lamport**[8] or **Winternitz**[9], possibly tying them together in a Merkle tree[9] (so we use the same public key to sign multiple messages). There are published standards to do exactly this, such as **LMS**[10] and **XMSS**[11]. These do have a drawback - if we use the same one-time private key to sign two distinct messages, these is a drastic reduction in the security[12][13]. Hence, the security of the system depends, not only on the hash function but also on the

implementation keeping track of which private keys it has signed with already. This state management is not impossible, but it is a nontrivial task.

One improvement to this was the **Sphincs** [14] signature system, which had 41 kilobyte signatures, and had a practical (if slow) signature generation time. It eliminated the requirement for the signer to record state securely. Instead on relying solely on a one-time signature scheme, it relied on a large (2^{60} set of few-time signatures (specifically, the **HORST (HORS with trees)** structure, which is an optimization of **Hash to Obtain Random Subset (HORS)** [15] that was introduced along with **Sphincs**. As a few-time signature, a single **HORST** key could be used to sign several messages without significant security loss. On the other hand, there are a finite number of **HORST** structures (and one is used to sign every message), and there is some security loss if a **HORST** structure signs too many messages. The suggested parameters in **Sphincs** was set so that it could safely generate 2^{50} signatures from a single private key, while still retaining 256 bits of security (NIST Level 5).

To go into more detail about how **Sphincs** works, the digital signature system worked by having a tree structure based on hashes with 256 bit outputs. At the top, there is a 60-level hypertree of Merkle trees, which is essentially an **XMSS**^{MT} structure[11]. At the bottom of the hypertree, there are 2^{60} **HORST** few time signature structures. Each such **HORST** structure could safely sign several messages.

When signing a message, the signer would select a random **HORST** structure, use it to sign the message, and then sign the **HORST** root with the authentication path through the hypertree.

4 Sphincs$^+$overview

Sphincs$^+$[1][16] is a refinement and generalization of the **Sphincs** structure.

Sphincs$^+$relies on a tree structure based on hashes with n bit outputs. At the top, there is an h level hypertree of Merkle trees. The h levels are made up of d layers, each of which consists of a h/d-height Merkle tree, and each leaf of the Merkle tree is a one-time signature. At the bottom of the hypertree, there are 2^h one-time signatures. Beneath each bottom-level one-time signature, there is a **FORS** structure (for a total of 2^h such structures), and each one-time signature signs the root its **FORS** structure. Each **FORS** structure consists of k sets, each of which consists of $t = 2^a$ private key values. These sets use an a-level Merkle tree to allow the authentication of these private values. The k roots of the Merkle trees are hashed together to form the root of the **FORS** structure.

To verify a **Sphincs**$^+$signature, the verifier takes the message and a value R found in the signature, and hashes them together. This hash is used to select one of the 2^h **FORS** structures, and for each of the k sets within that **FORS**, one of the t private values. For each of these private values, the verifier takes the value in the signature, and hashes it. It then takes the a-level authentication path found in the signature to derive an intermediate root for this set. Once all the intermediate roots for the sets have been computed, those are hashed together to form the root for the **FORS** structure. The verifier then uses the one-time signature (also found in the **Sphincs**$^+$signature) to process this root. The verifier then processes the authentication path to compute the root of the intermediate Merkle tree. It then interates through a number of one-time signatures and authentication paths, until it gets to the root of the top level Merkle tree, which is the root of the entire hypertree. This root of the hypertree is compared to the expected root value in the public key. If the two compare the same, the signature is accepted as valid.

There are several parameters that can be adjusted to specify the **Sphincs**$^+$parameter set:

- n: the length of each hash in bits.
- h: the height of the hypertree.
- d: the number of layers of one-time signatures and Merkle trees within the hypertree.
- w: the **Winternitz** parameter to use within the one-time signatures.

- k: the number of sets within a **FORS**.
- a: the $2^a = t$ private values within each **FORS** set.

A general overview of this structure is show in Figure 1. This drawing shows how the various parameters [1] modify the **Sphincs**$^+$structure.

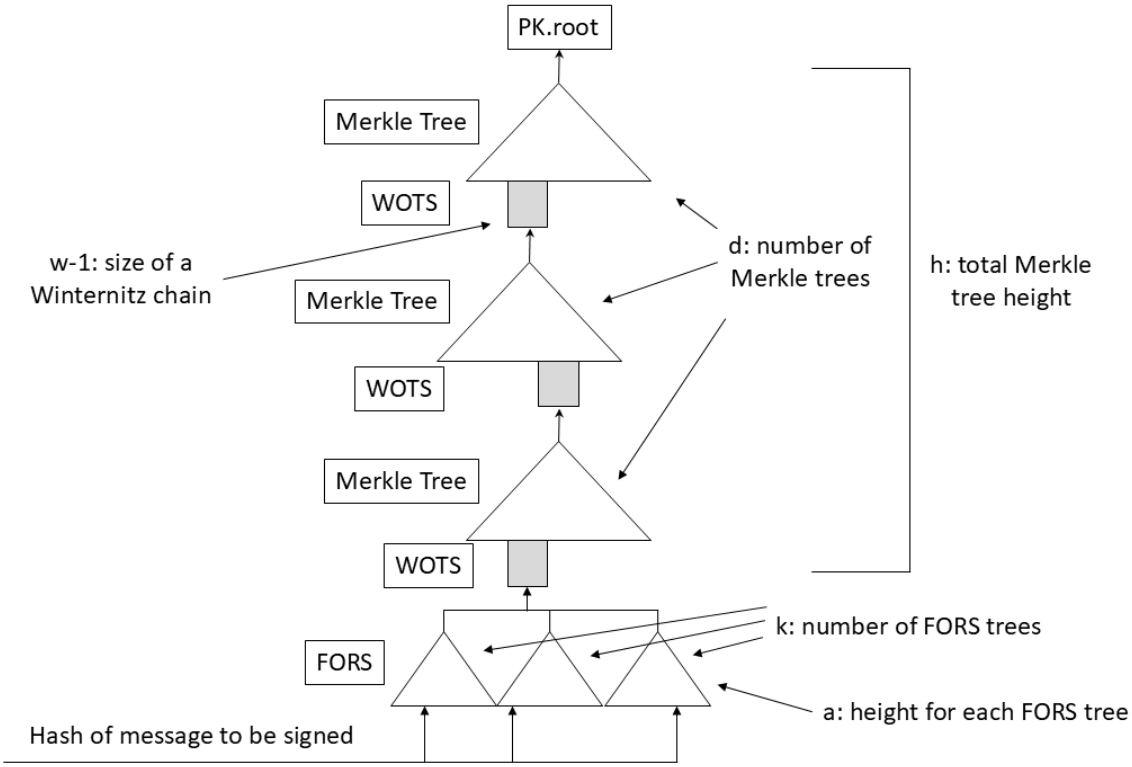

Figure 1: The overall design of the **Sphincs**$^+$, showing how the parameters are used.

5 Sphincs$^+$Usage Limitations

There are a number of ways to attack a **Sphincs**$^+$signature system, such as side channel attacks to recover an internal secret and fault attacks to forge a signature [17]. However, if we assume a correctly running black-box signer, there are effectively only two attack strategies available to the attacker. The first attack strategy is to find a hash preimage or second preimage against the underlying hash function, and use that to generate a forgery. Because we only use the hash functions (see [18] [19]) which have the expected preimage and second-preimage resistance strength of 2^n bits, the expected effort required for this attack is 2^n hash evaluations.

The second attack strategy is to piece together a valid signature for a new message from hashes found in previously published valid signatures. If we go through the **Sphincs**$^+$structure, we find that the only opportunity for this lies within the **FORS** structure. Each valid signature reveals k private key values from k sets within a **FORS** structure. If the adversary finds a message that hashes to a **FORS** structure and k values within that structure that are all revealed somewhere in the valid signatures in that **FORS**, he can combine them to form a valid signature for that message.

[1] Other than n, which affects every hash.

If p is the probability that a new message hashes to k values (one in each of the k t-element sets) that all appear somewhere in the known valid signatures, then the expected effort taken by this process is p^{-1} hashes.

If the hash length is n bits (and so the attack effort for former attack is 2^n hashes), then we want the latter attack to take at least as much effort, that is $p^{-1} \geq 2^n$ or $p \leq 2^{-n}$.

This is the process the adversary would use to perform this attack: he first obtains a large pile of valid signatures (that is, messages and signatures signed by the private key holder). Then, he selects a message (and randomizer R value), and hashes them into a hypertree leaf (which specifies the **FORS**) and k **FORS** leaves. He then goes through his pile of valid signatures, finds the ones that specify the same **FORS**, and from the ones he has, see if all k **FORS** leaves appear within them. If all k **FORS** leaves do appear, then he has all the leaf preimages and authentication paths he needs, and so can generate a signature for the message he picked.

When computing the probability of success of this process, we first need to consider "of the valid signatures that the attacker has, how many specify the same **FORS** as this message". We model the process of hashing a message to a hypertree leaf as a random process[2]; if the hypertree height is h, then the probability of a specific valid signature hashing to the same hypertree leaf is 2^{-h}. Each valid signature can be modeled as selecting a hypertree leaf independently, and so if he has 2^m valid signatures, with m large, then the probability distribution of the number of valid signatures that hash to that specific hypertree leaf is quite close to a Poisson distribution with mean $\lambda = 2^{m-h}$. That is, the probability of the adversary having exactly g valid signatures that hash to that hypertree leaf is approximately $\lambda^g e^{-\lambda}/g!$, see [20].

The next step is "if the adversary has exactly g valid signatures for that **FORS**, what is the probability that all k private values that he needs appear somewhere within those valid signatures?". If we model the hash of a message to **FORS** leaves as a random process[3], after hashing a random or chosen message, one target value in each of the k t-element sets is selected. For a valid (message, signature) pair that also uses that **FORS**, the probability of not hitting the target element in one t-element set is $(1 - t^{-1})$. For g given (message, signature) pairs, the probability of not hitting the target element in one t-element set is $(1 - t^{-1})^g$. So, the probability of having at least 1 hit is $(1 - (1 - t^{-1})^g)$. It requires at least 1 hit for each and every of the k sets to allow the adversary to forge a **FORS** signature. Therefore, this success probability is $(1 - (1 - t^{-1})^g)^k$.

If we combine those two results, it gives us the probability of the attacker being able to generate a forgery to the selected message with a single query as:

$$p = \sum_{g=0}^{2^m} \frac{\lambda^g e^{-\lambda}}{g!} \cdot (1\text{–}(1\text{–}t^{-1})^g)^k \tag{1}$$

We note that once g is several multiples of the mean λ, the terms drop off exponentially fast (because the $\lambda^g e^{-\lambda}/g!$ term drops by a factor of λ/g for each successive term and the $(1\text{–}(1\text{–}t^{-1})^g)^k$ term is upwards bounded by 1), and so this can be efficiently approximated.

We say that a **Sphincs**$^+$parameter set is 'tuned' to 2^m signatures if $p \approx 2^{-n}$; that is, if the adversary has 2^m signatures available to him, then the expected work effort for either attack strategy is approximately the same.

6 Tuning Sphincs$^+$

The **SLH-DSA** parameter sets are tuned for generating a maximum of 2^{64} signatures. NIST selected this limit because they considered any adversary requesting more signatures to be a Denial of Service attack. On the other

[2]Because the valid signer selects R randomly at signing time, this assumption is valid, even if the attacker specifies the messages being signed.

[3]Again, because R is selected randomly during a valid signature, this is a reasonable assumption.

hand, no real **SLH-DSA** signature system will ever generate anywhere close to 2^{64} valid signatures per key pair. Even if each signature could be generated in 1 µsec, it would take over 500,000 years to generate that many signatures. Even if we were to have a large number of parallel signers with the same public key, it would appear unlikely to ever come close to 2^{64} signatures. In addition, generally a signing key will get replaced or retired over time.

In addition, tuning **Sphincs**$^+$in an overly conservative manner is not free. As we will see below, tuning **Sphincs**$^+$can reduce the signature size and signature generation time.

Given that 2^{64} is overly conservative, a reasonable question to ask is "what would a more appropriate limit be?". That turns out to be usage dependent. For examples:

- A certificate authority root typically signs only a handful of certificates; a small limit would be appropriate.
- A software signing key would generate a signature for every software update. Even if the software is updated frequently, a limit in the thousands or tens of thousands would be appropriate.
- A signature used as a part of an authentication process would be used more frequently, for example, once for every authentication. In this case, a higher limit than those two cases above would be appropriate. Generally, the private keys for this type of signatures are updated after a few years or sooner and the limit of 2^{40} signatures per signing key may be more than sufficient.

Because different use cases have different requirements, we will consider a number of usage limits, specifically $2^{20}, 2^{30}, 2^{40}$ and 2^{50} signatures per signing key limits.

7 Overuse

We can then consider what happens if we 'overuse' a parameter set, that is, generate more signatures than it is tuned for. Unlike a one-time-signature (where 'overuse' would be using the same private key to generate two different signatures), the security of **Sphincs**$^+$doesn't drop off drastically under overuse, but instead is rather more gradual. For a given parameter set, when m increases, the attacker's success probability increases, meaning the security strength goes down. For example, if p is around 2^{-80}, the security strength is only 80 bits.

For example, we can consider how the **SLH-DSA** parameter sets fare under overuse conditions. Figure 2 shows how they behave under various overuse conditions (as computed using Equation 1). Since the choice of the hash function does not affect the security under overuse, we leave that as unspecified. For example, 128S stands for both SLH-DSA-SHA2-128s and SLH-DSA-SHAKE-128s parameter sets and 192F stands for both SLH-DSA-SHA2-192f and SLH-DSA-SHAKE-192f parameter sets, see [1] .

If we examine this graph, we note several things. First, there remains quite a bit of security strength even with moderate overuse. For example, if we look at the 256F parameter sets, they retain about 100 bits of security strength even after 2^{74} signatures, that is, 1,000 times as many signatures as they were designed for. In addition, while the security strength of all the parameters drop, they do not drop at the same rate. For example, at 2^{74} signatures, both 128S and 192F have about the same security strength.

We also noticed that the 128S and 192S parameter sets' security strengths degrade slower than their counter-part 128F and 192F parameter sets' do respectively. However, the order is reversed for the 256S and 256F parameter sets. However, that turns out to be due to the range of signature counts we selected; if we were to extend the graph a bit further (to the 2^{76} signature level), the 256S and 256F parameter set security levels would cross, and the 256S parameter set would have a higher remaining security level.

Of course, practically speaking, no one will ever generate anywhere close to 2^{64} signatures for one private key, much less go over that. So, considering overuse for the **SLH-DSA** parameter sets is an academic exercise. However, when we consider parameter sets with lower usage limits, it is plausible that they might be misused by going over their tuned limits. Hence it is important to consider how they behave on moderate overuse.

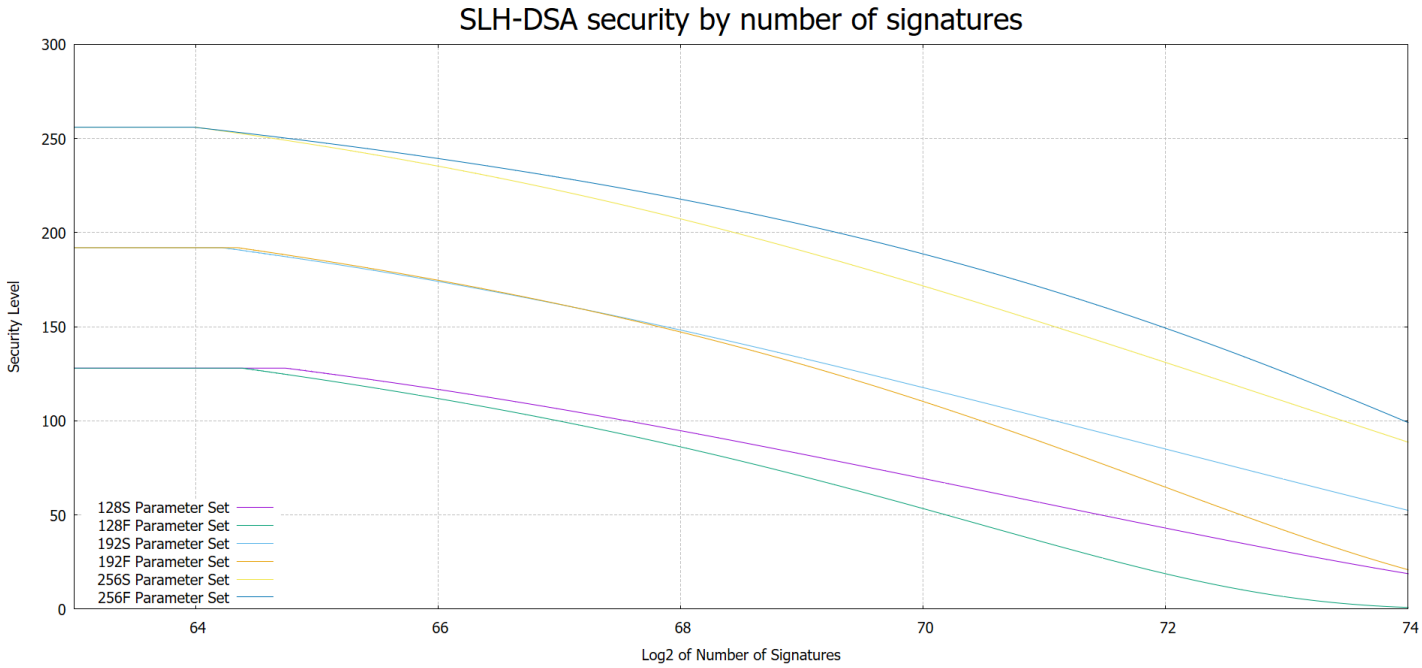

Figure 2: Overuse behavior of **SLH-DSA** parameter sets

8 Small Parameter Sets

When we consider how a **Sphincs**$^+$parameter set works, there are a number of desirable properties they may have, and to some extent, we need to select a trade-off between them. These properties[4] that are involved in the trade-offs include:

- The target security level
- The number of signatures the parameter set is tuned for
- The fall-off behavior when the parameter set is overused
- Signature generation time
- Signature verification time
- Signature size

We consider overuse behavior to be important, because (unlike the parameter sets defined in **SLH-DSA**) many signature limits are sufficiently small that it is not implausible that a parameter set will be misused by signing more messages than expected, and thus we would prefer to reduce the damage in that case.

We did a computerized search of the various potential parameter sets, with the focus on security levels 128 and 192 bits[5]; with the tuned number of signatures being $2^{20}, 2^{30}, 2^{40}$ and 2^{50}, and various signature generation times in term of the numbers of hashes such as 100,000, 1,000,000, 10,000,000, and 100,000,000. If our implementation is able to compute a hash in one microsecond, then this corresponds signature generation times of 0.1, 1, 10 and

[4]We don't list key generation time. The bulk of the computation done during the key generation process consists of building the top level merkle tree, and hence it is always faster than signature generation (which also effectively rebuilds that merkle tree as one of its tasks). Hence, if signature generation is 'fast enough', so is key generation.

[5]We currently do not include any parameter sets at the 256-bit security level in the main body of the document, as those are of interest only in extremely conservative situations, and those scenarios might want to avoid all issues around overuse.

100 seconds respectively[6][7]. Even though taking 100 seconds to generate a signature may seem excessive, the processing within **Sphincs**$^+$signature generation is parallelizable, and some implementations may be able to use that to accelerate the signing processs (perhaps using SIMD (AVX) instructions or multithreading).

For each parameter set, we list how many signatures can be generated and still retain a lesser security level (112 bits for parameter sets tuned to Level 1 security; 128 bits for parameter sets tuned to Level 3 security). This shows how well the parameter sets behave under overuse.

There have been several proposed ways to optimize the **Sphincs**$^+$architecture, such as:

- Making the top Merkle tree larger (and store sections of it in the private key); this either allows us to make the hypertree taller at the same amount of signer computation, allowing us to use shorter/fewer **FORS** trees, or allowing us to eliminate one of the internal **WOTS+** signatures, either of which reduces the signature size. **Gravity-Sphincs**[3] was designed with this optimization in mind.
- Having the signer perform some expensive operations before generating the hash that selects the **FORS**/Hypertree nodes; because the forger would need to perform those operations, that reduces the number of bits of security that the core part of **Sphincs**$^+$needs to provide (allowing it to shrink). It turns out that having some moderate expense is a win, even in terms of signature generation time. **Sphincs**$^+$**C** [4] implementes this optimization.
- Allowing an even larger range of w values, such as non-powers of two (which requires a base conversion to support).
- Changing how the **Winternitz** signature works from the current 'take the bits from the previous hash and append an inverted checksum' to a 'use the hash to generate a constant checksum' approach; this can reduce the number of **Winternitz** chains we need by 1 or 2 (or even 3 with a bit of signer processing), cutting the signature size. **Sphincs**$^+$**C** [4] implements this optimization.

We decided that these were larger changes to the architecture than what we wanted to suggest. Except for supporting w values other than 16 (but still powers of two between 2 and 256), all our changes involve modifying the existing values of the **Sphincs**$^+$parameter set (and hence can be expected that the effort to support this on an existing **SLH-DSA** implementation to be relatively minimal).

8.1 Note on Performance

In this paper, signature generation time is a factor in our parameter set searches and evaluations. Our formulae compute the number of hashes needed to generate a signature because the vast bulk of the computation is to compute the hashes. So to get a general performance estimate, we focus in only on the hashes in this paper.
A signing algorithm can be implemented in a Hardware Secure Module (HSM). However, generally, HSMs are not as fast as what are found on high-end CPUs.

We had discussions with some HSM vendors about the time to compute short hashes, and our understanding is that generally about one million short hashes can be generated per second in a HSM, with some HSMs being a bit faster than this and others a bit slower. We could not find a published reference for this estimate. In addition, this assumes current HSMs; if someone designs an HSM with a considerably faster CPU, parallel processing or hardware acceleration for the hash function, it may perform considerably better.

To give some context about signing times, we provide some performance data of the **SLH-DSA** parameter sets in **FIPS 205** below 1 by using the estimate of one million short hashes being computed per a second by a HSM. 128s stands for both SLH-DSA-SHA2-128s and SLH-DSA-SHAKE-128s. 128f stands for both SLH-DSA-SHA2-128f

[6]This is a bit of a simplification; in practice, we compute hashes of strings of differing lengths, which means that just counting hash computations doesn't give a precise reflection of the time taken. On the other hand, the ratio of differing length hashes is approximately the same for different parameter sets, and so this is still a useful for comparing parameter sets.

[7]Computing hashes take up the bulk of the time during the signature generation process.

and SLH-DSA-SHAKE-128f. Similarly for the other **SLH-DSA** Parameter set values in the table.

SLH-DSA Parameter set	Signature Generation:Hashes/Seconds	Signature Verification: Hashes
128s	2,186,222/2.18	2214
128f	105,196/0.10	6481
192s	3,767,275/3.76	3183
192f	169,260/0.16	9393
256s	3,280,869/3.28	4692
256f	345,839/0.34	9549

Table 1: Performance of **SLH-DSA** Parameter Sets

Generally, the biggest negative feature of the parameter sets in **FIPS 205** is their large signature sizes. The parameter sets whose signature generation times are large may be unacceptable for some applications. In our searches for parameter sets, we used the signing time limits of 100,000 hashes (0.1 seconds)[8], 1,000,000 (1 second), 10,000,000 (10 seconds) and 100,000,000 (100 seconds), 600,000,000 (600 seconds). We did not use the limit of 1,000,000,000 (1,000 seconds) except for the Selection Sets (192, 128, 2^{20}, 1,000,000,000) (see 10.1) and (256, 192, 2^{20}, 1,000,000,000) (see D) to see if we could find arguably better parameter sets than the parameter sets **Sphincs$^+$-192s-q20** and **Sphincs$^+$-256s-q20** recommended in [7] in some aspects.

8.2 Differences between the parameter sets of this version of its previous version.

If you look at the listed parameter sets in this paper closely, you would note that many parameter sets are different from the previous version [21] under several Selection Sets. This is due to a previous miscalculation on the cost of the **FORS** trees. The computed costs were erroneously computed to be somewhat cheaper, and hence some parameter sets were deemed easier to generate than they should have been. This paper provides the generated parameter sets after those errors were corrected, and show correct computational amounts.

The programs we used to generate the parameter sets are stored at this repository: `https://github.com/sfluhrer/sphincs-param-set-search`.

8.3 Initial comments about overuse of small parameter sets

For an initial example, we will consider three parameter sets that all have Level 1 security, tuned for 2^{20} signatures. They all take approximately 100,000 hashes to generate a signature[9]. They vary somewhat in verification time and signature size; these parameter sets are:

- Set A - n=16 h=20 d=4 a=9 k=19 w=16 - Signature size = 5616
- Set B - n=16 h=25 d=5 a=8 k=18 w=16 - Signature size = 5808
- Set C - n=16 h=28 d=7 a=10 k=13 w=16 - Signature size = 6672

Just looking at the signature size, Set A would look superior, however we can also consider the fall-off behavior, shown in Figure 3.

As we can see from the graph, Set A loses its security relatively quickly on overuse. However, Set B loses it rather more slowly (still retaining 100 bits of security after 2^{25} signatures), and Set C loses it even slower still.

We look for parameter sets whose signature sizes are smaller than the smallest signature size in **SLH-DSA**, which is 7,856 bytes for security Level 1 and 16,224 for security Level 3. To facilitate the discussions in this paper, we

[8] Unsing the one million hashes per second assumption we mentioned above.

[9] Hence if we can compute about a million hashes per second, they all take about 0.1 second to generate a signature.

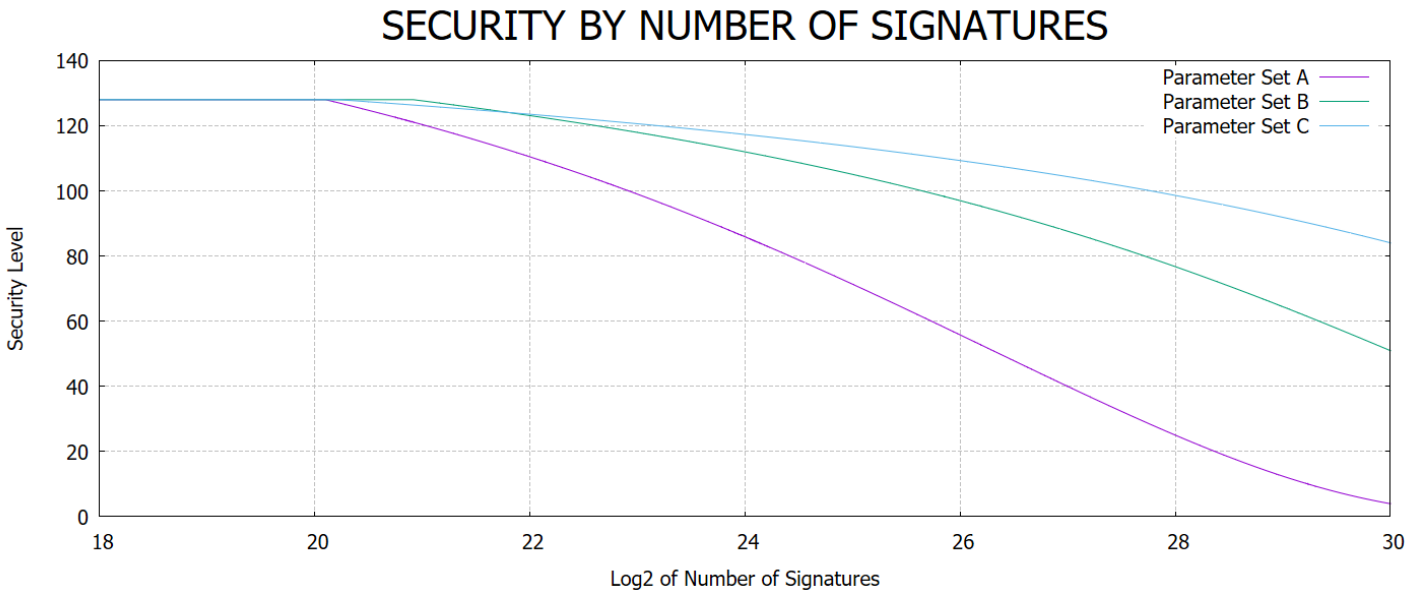

Figure 3: **Overuse characteristics of the three example parameter sets**

call those numbers the *bound-size* and the *192-size* respectively.

For most cases, under a Selection Set of (security level, retained lesser security level, maximum number of signatures, the upper bound of hash operations for signing), we are going to pick 2 parameter sets to discuss their security degradation behaviors which are called "*overuse safety*" in the parameter set tables later in this paper: one with the smallest signature and the other with the highest *overuse safety*. For example, a Selection Set: (128, 112, 2^{20}, 100,000) means we search for parameter sets offering 128 bits of security, the retained lesser security at 112 bits (at what total number of signatures per a signing key the retained security level still remains), the maximum number of signatures a signing key is allowed to generate before its security strength starts to fall below 128 bits, signing times are not more than 100,000 hash operations.

When a parameter set A loses its retained lesser security level faster than a parameter set B, we may just say that B has "less risk" than A or A has "more risk" than B.

In addition, we are going to search for a third parameter set which does not have either the largest *overuse safety* or the smallest signature size, but has either one of the 2 properties below.

1) It has noticeably larger *overuse safety* than the smallest signature parameter set has and its signature size is marginally larger than the latter.
2) Its *overuse safety* is marginally smaller than the largest *overuse safety* parameter set, but its signature size is noticeably smaller than the latter.

Of course, this is a subjective matter for what "marginally" and "noticeably" mean. So, we'll need to look at specific instances in order to have sound opinions.

To generate the raw list of parameter sets for us to consider, we went through the possible combinations exhaustively and selected the parameter sets that had the required security level at the minimum number of signatures (as well as requiring fewer hash computations to generate a signature than our bound), sorted them by signature size, and scanned them (in increasing signature size order). We then selected the parameter sets which were "better" than any of the ones previously found, where "better" is either "better overuse characteristic" or "better W value[10]"

[10]W=16 is considered the best, because that's what **SLH-DSA** uses, hence making plugging in these parameter sets into an existing **SLH-DSA** implementation the easiest; W=4 or W=256 is second best (because it makes the mapping of the hash to digit conversion

To determine the overuse characteristics, we computed the number of signatures it would take to reduce the security to a lesser security level (112 bits for Level 1 security parameter sets, 128 bits for Level 3 security parameter sets); the more signatures can be generated before reducing security to that lesser security level, the better (safer).

We then went through the lists of parameter sets manually, and selected ones which appeared to us to be attractive to discuss in this paper.

9 Level 1 Security Parameter Sets

9.1 Level 1 security at 2^{20} signatures

Here are the search results for parameter sets tuned for Level 1 security after 2^{20} signatures, and an upper bound of 100,000 hash operations during the signing process. This Selection Set is (128, 112, 2^{20}, 100,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
A-1	16	20	4	5	8	23	4	26	1	32	5888	100%	89576	1353	21.69	3
A-2	16	25	5	5	7	21	4	23	1	32	5904	100%	97961	1600	23.25	9
A-3	16	25	5	5	7	22	4	24	1	32	6032	102%	98344	1608	23.87	14
A-4	16	25	5	5	7	23	4	25	1	32	6160	104%	98727	1616	24.39	20
A-5	16	25	5	5	7	24	4	25	1	32	6288	106%	99110	1624	24.85	28
A-6	16	25	5	5	7	25	4	26	1	32	6416	108%	99493	1632	25.24	37
A-7	16	25	5	5	7	26	4	27	1	32	6544	111%	99876	1640	25.59	48
A-8	16	28	7	4	9	15	4	21	1	32	6784	115%	85965	2147	26.79	110
A-9	16	28	7	4	9	16	4	22	1	32	6944	117%	87500	2157	27.83	227
A-10	16	28	7	4	9	17	4	24	1	32	7104	120%	89035	2167	28.65	401
A-11	16	32	8	4	9	14	4	21	1	32	7248	123%	93421	2422	29.39	670
A-12	16	32	8	4	9	15	4	22	1	32	7408	125%	94956	2432	30.79	1770

Table 2: Selection Set (128, 112, 2^{20}, 100,000)

The "sig bytes" is the size of the signature in bytes. The "% size" is the signature size, compared to the smallest signature on the list.

"sign time" and "verify time" are the numbers of hashes computed during signature generation and verification respectively. Now, in practice, not all hash computations are the same, as some hashes have larger inputs than others, however in practice, that does not affect the relative numbers appreciably.

The 'sigs at level 112' is the logarithm (to the base of 2) of the number of signatures that can be generated and retain a security level of 112 bits.

The '*overuse safety*' is the number of signatures (times the target level) that can be generated and retain a security level of 112 bits. For example, for the parameter set A-12, if you generated $1770 \times 2^{20} \approx 1,856,000,000$ signatures, you would still have a security level of 112 bits.

The first parameter set A-1 (16, 20, 4, 5, 8, 23, 4, 26) is $(7,856 - 5,888 =)$ 1,968 bytes smaller than the *bound-size*. This is a roughly 25% improvement in the signature size. However, it has the smallest *overuse safety* among the parameter sets As while having the most signature size improvement.

simple), and other W values the worst (because digit values will span bytes at times).

On the other hand, the signatures for parameter set A-2 is only marginally larger than A-1's signature, but has noticeably larger *overuse safety* than A-1's.

On the other hand, the last parameter set A-12 (16, 32, 8, 4, 9, 15, 4, 22) has the largest *overuse safety*, but has the least improvement in signature size. The improvement is less than 500 bytes from the *bound-size*.

Even the best of the above parameter sets has a signature size of $5,888$ bytes, this may be too large for many applications. So, we are going to look at how we can shrink the signature size by spending more time during signature generation.

Now, we are going to list the search result for an upper bound on the signer time of 1,000,000 hashes in the table below. This Selection Set is (128, 112, 2^{20}, 1,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
B-1	16	21	3	7	13	11	6	21	1	32	3968	100%	860917	2484	22.92	7
B-2	16	24	4	6	12	11	7	21	1	32	4032	101%	823796	5549	24.12	17
B-3	16	25	5	5	13	10	8	21	1	32	4096	103%	983348	11692	25.24	37
B-4	16	24	4	6	13	11	7	22	1	32	4208	106%	958964	5560	25.92	60
B-5	16	30	5	6	12	10	7	19	1	32	4256	107%	983668	6887	27.86	232
B-6	16	28	4	7	12	11	6	21	1	32	4288	108%	922612	3249	28.12	278
B-7	16	18	2	9	13	13	4	26	1	32	4336	109%	894964	764	22.00	4
B-8	16	30	6	5	10	12	8	20	1	32	4336	109%	921969	13994	26.82	112
B-9	16	30	6	5	11	11	8	21	1	32	4336	109%	952690	13994	27.62	196
B-10	16	24	3	8	12	11	4	20	1	32	4368	110%	566773	1012	24.12	17
B-11	16	27	3	9	10	13	4	22	1	32	4416	111%	903155	1015	25.76	54
B-12	16	27	3	9	12	11	4	22	1	32	4416	111%	998389	1015	27.12	139
B-13	16	32	4	8	10	12	5	19	1	32	4432	111%	956403	1962	28.82	451
B-14	16	32	4	8	11	11	5	20	1	32	4432	111%	987124	1962	29.62	786
B-15	16	30	5	6	12	11	7	21	1	32	4464	112%	995955	6900	30.12	1112
B-16	16	30	6	5	10	13	8	22	1	32	4512	113%	925040	14005	28.76	433
B-17	16	30	6	5	11	12	8	22	1	32	4528	114%	958833	14006	29.63	792
B-18	16	27	3	9	11	13	4	23	1	32	4624	116%	943091	1028	27.99	254
B-19	16	30	6	5	10	14	8	23	1	32	4688	118%	928111	14016	30.09	1089
B-20	16	27	3	9	10	15	4	24	1	32	4768	120%	909297	1037	28.08	270
B-21	16	32	4	8	13	9	4	19	1	32	4784	120%	796662	1284	29.48	714
B-22	16	32	4	8	12	10	4	19	1	32	4848	122%	698357	1288	29.86	929
B-23	16	32	4	8	13	10	4	21	1	32	5008	126%	821237	1298	32.24	4837

Table 3: Selection Set (128, 112, 2^{20}, 1,000,000)

In comparison to the previous A parameter sets' signatures, these are considerably smaller. In addition, with this amount of computation available during signature generation, the setting w=32 and above becomes viable; this reduces the signature size somewhat. However, this comes at a cost of significantly increasing the signature verification time. Because they can give smaller signatures at times, we included both w=16 and w=256 options, as well as intermediate options when appropriate.

The parameter set B-1 (16, 21, 3, 7, 13, 11, 6, 21) has the smallest signature size which is 3,968 bytes and the parameter set B-23 (16, 32, 4, 8, 13, 10, 4, 21) has the largest signature size, 5,008 bytes.

The signature size reductions of B-1 and B-23 from the *bound-size* are $(7,856 - 3,968)/7,856$ ($\approx 50\%$) and $(7,856 - 5,008)/7,856$ ($\approx 36\%$), respectively. The former parameter set has a large signature size reduction, while

the latter has a noticeable but smaller reduction. However, the latter has significantly higher *overuse safety* than the former.

We now consider the same situation, except for an upper bound on the signing time of 10,000,000 hash function evaluations. Below is the parameter sets under this Selection Set: (128, 112, 2^{20}, 10,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
C-1	16	24	3	8	16	8	8	19	1	32	3440	100%	5113336	7077	24.90	29
C-2	16	27	3	9	14	9	8	21	1	32	3472	100%	7523319	7079	26.92	121
C-3	16	27	3	9	16	8	8	21	1	32	3488	101%	8653816	7080	27.90	238
C-4	16	30	3	10	15	8	7	20	1	32	3552	103%	9050104	4195	28.94	491
C-5	16	27	3	9	15	9	8	22	1	32	3616	105%	7965687	7088	28.69	413
C-6	16	30	3	10	17	7	6	20	1	32	3664	106%	7477241	2465	29.21	592
C-7	16	30	3	10	14	9	7	21	1	32	3664	106%	8706039	4202	29.92	968
C-8	16	30	3	10	16	8	7	21	1	32	3680	106%	9836536	4203	30.90	1910
C-9	16	24	2	12	16	8	4	20	1	32	3696	107%	6176761	724	24.90	29
C-10	16	32	4	8	13	9	8	19	1	32	3696	107%	4941814	9380	29.48	714
C-11	16	32	4	8	17	7	8	19	1	32	3696	107%	7473144	9380	31.21	2368
C-12	16	26	2	13	14	9	4	20	1	32	3712	107%	9650168	725	25.92	60
C-13	16	26	2	13	13	10	4	21	1	32	3792	110%	9453559	730	26.24	75
C-14	16	24	2	12	17	8	4	21	1	32	3824	111%	7749625	732	26.42	85
C-15	16	26	2	13	14	10	4	22	1	32	3952	114%	9699319	740	27.97	250
C-16	16	24	2	12	17	9	4	24	1	32	4112	119%	8142840	750	28.32	319
C-17	16	30	3	10	17	7	4	20	1	32	4192	121%	4478969	1001	29.21	592
C-18	16	26	2	13	14	11	4	24	1	32	4192	121%	9748470	755	29.35	652
C-19	16	33	3	11	13	9	4	20	1	32	4240	123%	3674103	1004	30.48	1428

Table 4: Selection Set (128, 112, 2^{20}, 10,000,000)

When we are able to spend this amount of time during the signature generation process, parameter sets with small signatures tend to have a comparatively large hypertree and large **FORS** sets; both tend to lend themselves to superior overuse characteristics.

As we can see, parameter set C-1 (16, 24, 3, 8, 16, 8, 8, 19) has the smallest signature size in the table. C-1 is also 528 (3968-3440) bytes smaller than the smallest signature (B-1) in the previous table. C-1 also has larger 'sigs at level 112' than B-1 by a factor of $2^{24.9-22.92} = 2^{1.98}$.

The second parameter set C-2 (16, 27, 3, 9, 14, 9, 8, 21) has the second smallest signature size which is only 32 bytes larger than C-1. However, C-2 has a meaningful larger 'sigs at level 112' than C-1 does. The difference is $2^{26.92-24.9} = 2^{2.02}$ times. In addition, C-2 has a smaller combined signature plus public key size (3,504 = 3472 + 32) than the smallest option in *FIPS 204, Module-Lattice-Based Digital Signature Standard (ML-DSA)* [22]. This smallest option is the parameter set ML-DSA-44, which has a combined signature and public key size of 3,732 (1312 + 2420) bytes.

The parameter set C-9 (16, 24, 2, 12, 16, 8, 4, 20) is only 3696 − 3472 = 224 bytes larger than C-2 in signature size, but the former's Verify Time is only around 10% of C-2's verification time, however C-9 has worse *overuse safety*.

The parameter set C-11 (16, 32, 4, 8, 17, 7, 8, 19) has the lowest risk. Its signature size is only 256 bytes (3696 − 3472 = 224) larger than C-2's signature size, but its *overuse safety* is significantly larger than C-2's by

the factor of $2^{31.21-26.92} = 2^{4.29}$ times.

Now, searching for the parameter sets with 100,000,000 hash evaluations, we get:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
D-1	16	24	2	12	21	6	8	20	1	32	3088	100%	75513856	4768	25.68	51
D-2	16	26	2	13	18	7	8	20	1	32	3136	101%	81035256	4771	27.05	132
D-3	16	26	2	13	21	6	7	20	1	32	3216	104%	81821696	2850	27.68	205
D-4	16	26	2	13	19	7	8	21	1	32	3248	105%	86540280	4778	28.50	362
D-5	16	28	2	14	18	7	7	20	1	32	3264	105%	93650936	2853	29.05	530
D-6	16	28	2	14	21	6	6	20	1	32	3344	108%	88145920	1700	29.68	820
D-7	16	26	2	13	20	7	8	22	1	32	3360	108%	97550328	4785	29.74	855
D-8	16	30	3	10	20	6	8	20	1	32	3376	109%	33036282	7073	30.10	1097
D-9	16	17	1	17	20	8	4	23	1	32	3536	114%	98828280	468	22.92	7
D-10	16	30	2	15	17	7	4	19	1	32	3632	117%	39583740	720	29.21	592
D-11	16	30	2	15	20	6	4	19	1	32	3632	117%	55705600	720	30.10	1097

Table 5: Selection Set (128, 112, 2^{20}, 100,000,000)

We first note that all the parameter sets listed here have a combined public key size plus signature size of 3,120 to 3,664 bytes, which is again smaller than the smallest option in ML-DSA.

We also note that parameter set D-8 (16, 30, 3, 10, 20, 6, 8, 20) has signatures that are only 288 bytes larger than D-1 (16, 24, 2, 12, 21, 6, 8, 20) 's signatures, and D-8 has considerably better *overuse safety* (as well as being comparatively cheap to generate for this group).

[7] had a recommended parameter set **Sphincs$^+$-128s-q20** which has the signature size of 3,264 bytes. We wanted to know if there were more attractive parameter sets in some aspects so we increased the signing time bound to 600,000,000 hashes and we found the parameter sets below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
AAA-1	16	30	2	15	24	5	8	19	1	32	3072	100%	553779200	4767	30.75	1722
AAA-2	16	18	1	18	24	6	4	21	1	32	3264	106%	449314816	451	23.25	9
AAA-3	16	19	1	19	24	6	4	21	1	32	3280	106%	596639744	452	24.25	19
AAA-4	16	19	1	19	22	7	4	23	1	32	3456	112%	382730240	463	24.93	30
AAA-5	16	18	1	18	23	7	4	24	1	32	3552	115%	323485696	469	24.97	31
AAA-6	16	19	1	19	23	7	4	24	1	32	3568	116%	470810624	470	25.97	62
AAA-7	16	34	2	17	23	5	4	21	1	32	3600	117%	273154048	718	33.00	8192

Table 6: Selection Set (128, 112, 2^{20}, 600,000,000)

AAA-1(16, 30, 2, 15, 24, 5, 8, 19) has the smallest signature size and second best *overuse safety*. AAA-2 is the parameter set **Sphincs$^+$-128s-q20** in [7]. AAA-1 is superior to AAA-2 in signature size and *overuse safety*.

9.2 Level 1 security at 2^{30} signatures

Below is the parameter sets under Selection Set: (128, 112, 2^{30}, 1,000,000). The parameter sets we found are below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
E-1	16	32	4	8	11	13	5	22	1	32	4816	100%	999410	1986	32.99	7

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
E-2	16	32	4	8	10	15	5	23	1	32	4960	102%	965616	1995	33.08	8
E-3	16	35	5	7	13	10	5	22	1	32	5056	104%	820468	2422	35.24	37
E-4	16	30	6	5	11	15	8	26	1	32	5104	105%	977262	14042	32.76	6
E-5	16	36	6	6	13	10	6	22	1	32	5136	106%	836339	4792	36.24	75
E-6	16	32	4	8	13	11	4	22	1	32	5232	108%	845812	1312	33.92	15
E-7	16	35	5	7	13	11	5	23	1	32	5280	109%	845043	2436	36.92	121
E-8	16	36	6	6	13	11	6	23	1	32	5360	111%	860914	4806	37.92	242
E-9	16	42	7	6	13	9	6	21	1	32	5392	111%	910195	5553	39.48	714
E-10	16	36	9	4	13	10	8	22	1	32	5424	112%	909584	20923	36.24	75
E-11	16	32	4	8	11	14	4	24	1	32	5456	113%	661489	1326	33.99	15
E-12	16	32	4	8	13	12	4	24	1	32	5456	113%	870387	1326	35.10	34
E-13	16	42	7	6	12	10	6	21	1	32	5456	113%	811890	5557	39.86	929
E-14	16	40	5	8	12	10	4	20	1	32	5536	114%	842228	1577	37.86	232
E-15	16	40	8	5	13	10	7	23	1	32	5584	115%	934385	10942	40.24	1209
E-16	16	36	9	4	13	11	8	23	1	32	5648	117%	934159	20937	37.92	242
E-17	16	40	5	8	13	10	4	22	1	32	5696	118%	965108	1587	40.24	1209

Table 7: Selection Set (128, 112, 2^{30}, 1,000,000)

The parameter set E-1 (16, 32, 4, 8, 11, 13, 5, 22) has the smallest signature size from the table even though it does not have the smallest *overuse safety*. The parameter sets E-15 (16, 40, 8, 5, 13, 10, 7, 23) and E-17 (16, 40, 5, 8, 13, 10, 4, 22) have the largest *overuse safety*. E-1's signature size is (5584 - 4816 =) 768 bytes smaller than E-15's. But, the *overuse safety* of E-15 (and E-17) is significantly bigger than E-1's by a factor of $2^{40.24-32.99} = 2^{7.25}$.

To search for smaller parameter sets, we increased the upper bound to 10,000,000 hash operations to generate a signature. Below are several of such parameter sets under the Selection Set (128, 112, 2^{30}, 10,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
F-1	16	36	4	9	14	9	8	22	1	32	3904	100%	9883638	9393	35.92	60
F-2	16	36	4	9	13	10	8	23	1	32	3984	102%	9687029	9398	36.24	75
F-3	16	40	5	8	17	7	8	20	1	32	4112	105%	8653303	11693	39.21	592
F-4	16	40	5	8	14	9	8	21	1	32	4256	109%	6343157	11702	39.92	968
F-5	16	40	5	8	16	8	8	21	1	32	4272	109%	7473654	11703	40.90	1910
F-6	16	36	3	12	14	9	4	21	1	32	4432	113%	7348215	1016	35.92	60
F-7	16	36	3	12	16	8	4	21	1	32	4448	113%	8478712	1017	36.90	119
F-8	16	36	3	12	15	9	4	22	1	32	4576	117%	7790583	1025	37.69	206
F-9	16	36	3	12	14	10	4	23	1	32	4672	119%	7397366	1031	37.97	250
F-10	16	36	3	12	16	9	4	23	1	32	4720	120%	8675319	1034	39.10	548
F-11	16	36	3	12	15	10	4	24	1	32	4832	123%	7888886	1041	39.37	661
F-12	16	48	4	12	14	8	4	21	1	32	4944	126%	9601015	1294	44.12	17805

Table 8: Selection Set (128, 112, 2^{30}, 10,000,000)

There are parameter sets in this table which have noticeable reductions in signature sizes and overuse risks over the parameter sets D-i$_s$ and E-i$_s$. For example, the parameter set F-1 (16, 36, 4, 9, 14, 9, 8, 22)'s signature size is (4816 - 3904 =) 912 bytes smaller than the signature of the parameter set E-1 which has the smallest signature among all parameter sets D-i$_s$ and E-i$_s$. F-1 also has a significant increase in *overuse safety* from E-1 by the factor of $2^{35.92-32.99} = 2^{2.93}$.

One can also choose parameter sets which have even much higher overuse safeties, but larger signature sizes. One of such parameter sets is the parameter set F-5 (16, 40, 5, 8, 16, 8, 8, 21) which has a much larger *overuse safety* than F-1's *overuse safety*, but F-5's signature is only 368 (4272-3904) bytes larger than F-1's signature.

To search for smaller parameter sets, we increased the upper bound to 100,000,000 hash operations to generate a signature. Below are several of such parameter sets under the Selection Set (128, 112, 2^{30}, 100,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
G-1	16	36	3	12	21	6	8	21	1	32	3568	100%	94396416	7085	37.68	205
G-2	16	39	3	13	20	6	7	21	1	32	3664	102%	84983808	4202	39.10	548
G-3	16	36	3	12	19	7	8	22	1	32	3696	103%	67657728	7093	38.50	362
G-4	16	39	3	13	18	7	7	22	1	32	3776	105%	71614464	4209	40.05	1060
G-5	16	36	3	12	20	7	8	23	1	32	3808	106%	78667776	7100	39.74	855
G-6	16	40	4	10	20	6	8	21	1	32	3824	107%	37756920	9388	40.10	1097
G-7	16	32	2	16	19	7	4	21	1	32	3888	108%	84672504	736	34.50	22
G-8	16	32	2	16	20	7	4	22	1	32	4000	112%	95682552	743	35.74	53
G-9	16	32	2	16	19	8	4	23	1	32	4208	117%	86245368	756	36.85	115
G-10	16	39	3	13	20	6	4	21	1	32	4336	121%	32686074	1010	39.10	548
G-11	16	45	3	15	19	6	4	21	1	32	4336	121%	64684032	1010	43.02	8306

Table 9: Selection Set (128, 112, 2^{30}, 100,000,000)

G-1 (16, 36, 3, 12, 21, 6, 8, 21) has the smallest signatures, and a relatively decent *overuse safety*. G-4 (16, 39, 3, 13, 18, 7, 7, 22) has larger signatures by only 208(3776 - 3568) bytes, but much larger *overuse safety*.

9.3 Level 1 security at 2^{40} signatures

Increasing the number of signatures to 2^{40}, we set the number of hashes to 1,000,000 and many parameter sets which have smaller signature sizes than the *bound-size*. Below are several of such parameter sets under the Selection Set (128, 112, 2^{40}, 1,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
H-1	16	42	7	6	13	11	6	24	1	32	5840	100%	959345	5581	43.92	15
H-2	16	49	7	7	11	11	5	23	1	32	6048	103%	872177	3326	46.62	98
H-3	16	48	8	6	10	13	6	24	1	32	6144	105%	827374	6345	46.76	108
H-4	16	48	8	6	12	11	6	24	1	32	6144	105%	922608	6345	48.12	278
H-5	16	40	5	8	12	13	4	25	1	32	6160	105%	879089	1616	42.65	6
H-6	16	44	11	4	12	11	8	23	1	32	6176	105%	946509	25544	44.12	17
H-7	16	49	7	7	12	11	5	24	1	32	6224	106%	939761	3337	49.12	556
H-8	16	40	5	8	11	15	4	26	1	32	6336	108%	811503	1627	42.76	6
H-9	16	48	8	6	12	12	6	25	1	32	6352	108%	934895	6358	49.59	770
H-10	16	40	5	8	12	14	4	26	1	32	6368	109%	891376	1629	43.47	11
H-11	16	48	6	8	9	14	4	22	1	32	6384	109%	884719	1876	45.39	41
H-12	16	44	11	4	12	12	8	24	1	32	6384	109%	958796	25557	45.59	48
H-13	16	48	6	8	10	13	4	23	1	32	6432	110%	903152	1879	46.76	108
H-14	16	48	6	8	12	11	4	23	1	32	6432	110%	998386	1879	48.12	278
H-15	16	49	7	7	11	13	5	25	1	32	6432	110%	884463	3350	49.99	1016
H-16	16	49	7	7	12	12	5	25	1	32	6432	110%	952048	3350	50.59	1541

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
H-17	16	48	6	8	11	13	4	24	1	32	6640	113%	943088	1892	48.99	508
H-18	16	48	6	8	10	15	4	25	1	32	6784	116%	909294	1901	49.08	541
H-19	16	48	6	8	11	14	4	26	1	32	6832	116%	949231	1904	49.99	1016
H-20	16	52	13	4	10	13	8	24	1	32	6880	117%	998793	30162	50.76	1734
H-21	16	48	6	8	11	15	4	27	1	32	7024	120%	955374	1916	50.76	1734

Table 10: Selection Set (128, 112, 2^{40}, 1,000,000)

All of the parameter sets H-i$_s$ have smaller signatures than the *bound-size*. The biggest and smallest signature size reductions are 2,016 (7,856 - 5,840) and 832 (7,856 - 7,024) bytes which come with the parameter sets H-1 (16, 42, 7, 6, 13, 11, 6, 24) and H-21 (16, 48, 6, 8, 11, 15, 4, 27) respectively. H-1 is about 1,000 bytes smaller than H-20 (16, 52, 13, 4, 10, 13, 8, 24), but H-20 has a much larger *overuse safety* than H-1 by the factor of $2^{50.76-43.92} = 2^{6.84}$.

The parameter set I-i$_s$ in the table below have smaller signature sizes than the parameter set H-i$_s$. I-i$_s$ have an upper bound of 10,000,000 hash operations for signing process and they were generated according to the Selection Set (128, 112, 2^{40}, 10,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
I-1	16	40	5	8	16	9	8	23	1	32	4544	100%	7670261	11720	43.10	8
I-2	16	45	5	9	16	8	7	23	1	32	4592	101%	8459254	6908	45.90	59
I-3	16	40	5	8	15	10	8	24	1	32	4656	102%	6883828	11727	43.37	10
I-4	16	48	6	8	14	9	8	22	1	32	4672	102%	7523316	14015	47.92	242
I-5	16	48	6	8	16	8	8	22	1	32	4688	103%	8653813	14016	48.90	477
I-6	16	50	5	10	15	8	6	22	1	32	4784	105%	8660982	4025	48.94	491
I-7	16	48	6	8	15	9	8	23	1	32	4816	105%	7965684	14024	49.69	826
I-8	16	50	5	10	14	9	6	23	1	32	4896	107%	8316917	4032	49.92	968
I-9	16	48	6	8	14	10	8	24	1	32	4912	108%	7572467	14030	49.97	1002
I-10	16	54	6	9	13	9	7	23	1	32	4912	108%	8484852	8252	51.48	2856
I-11	16	56	7	8	13	9	8	22	1	32	4944	108%	8482291	16319	53.48	11425
I-12	16	44	4	11	16	8	4	23	1	32	5136	113%	6176759	1306	44.90	29
I-13	16	48	4	12	14	9	4	23	1	32	5184	114%	9650166	1309	47.92	242
I-14	16	48	4	12	13	10	4	24	1	32	5264	115%	9453557	1314	48.24	302
I-15	16	48	4	12	14	10	4	25	1	32	5424	119%	9699317	1324	49.97	1002
I-16	16	48	4	12	14	11	4	27	1	32	5664	124%	9748468	1339	51.35	2610

Table 11: Selection Set (128, 112, 2^{40}, 10,000,000)

I-1 (16, 40, 5, 8, 16, 9, 8, 23) is the smallest signature. I-1 is 1,296 (5840 - 4544) bytes smaller than H-1, but I-1 has lower *overuse safety* than H-1's by a small factor of $2^{43.92-43.10} = 2^{0.82}$.

An interesting parameter set is I-11 (16, 56, 7, 8, 13, 9, 8, 22) which has the lowest security risk among all H and I parameter sets and it is smaller than all signatures from H parameter sets.

Increasing the number of hashes to 100,000,000, we found the parameter sets below:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
J-1	16	44	4	11	21	6	8	23	1	32	3984	100%	75513848	9398	45.68	51
J-2	16	48	4	12	18	7	8	23	1	32	4064	102%	81035256	9403	49.05	530
J-3	16	48	4	12	19	7	8	24	1	32	4176	104%	86540280	9410	50.50	1448
J-4	16	45	3	15	21	6	4	22	1	32	4528	113%	92995584	1022	46.68	102
J-5	16	45	3	15	19	7	4	23	1	32	4656	116%	66256892	1030	47.50	181
J-6	16	45	3	15	20	7	4	24	1	32	4768	119%	77266944	1037	48.74	427
J-7	16	45	3	15	21	7	4	25	1	32	4880	122%	99287040	1044	49.87	935
J-8	16	56	4	14	19	6	4	23	1	32	5072	127%	46268408	1302	54.02	16612

Table 12: Selection Set (128, 112, 2^{40}, 100,000,000)

J-1 (16, 44, 4, 11, 21, 6, 8, 23) has the smallest signatures, while J-3 (16, 48, 4, 12, 19, 7, 8, 24) has better *overuse safety*. Even though J-8 (16, 56, 4, 14, 19, 6, 4, 23) has outstanding *overuse safety*, its relatively large signature size makes it unattractive in our opinion.

9.4 Level 1 Security at 2^{50} signatures

To increase the maximum number of signatures per a signing key is a way to search for safer parameter sets. On the other hand, that also increases the signature size. Therefore, such parameter sets tend to have less signature size reductions.

K-i parameter sets in the table below were produced when we set the hash operation limit for signing to 1,000,000. The Selection Set was (128, 112, 2^{50}, 1,000,000).

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
K-1	16	56	8	7	10	13	5	25	1	32	6784	100%	959470	3793	54.76	27
K-2	16	56	8	7	11	12	5	25	1	32	6800	100%	993263	3794	55.63	49
K-3	16	56	8	7	10	14	5	26	1	32	6960	102%	962541	3804	56.09	68
K-4	16	56	8	7	11	13	5	26	1	32	6992	103%	999406	3806	56.99	127
K-5	16	56	8	7	10	15	5	27	1	32	7136	105%	965612	3815	57.08	135
K-6	16	60	10	6	8	17	6	25	1	32	7264	107%	997352	7905	58.08	270
K-7	16	52	13	4	9	17	8	27	1	32	7312	107%	984965	30189	52.65	6
K-8	16	48	6	8	11	17	4	30	1	32	7408	109%	967660	1940	51.93	3
K-9	16	60	10	6	8	18	6	26	1	32	7408	109%	998119	7914	59.00	512
K-10	16	52	13	4	9	18	8	28	1	32	7472	110%	986500	30199	53.31	9
K-11	16	60	10	6	8	19	6	27	1	32	7552	111%	998886	7923	59.74	855
K-12	16	48	6	8	11	18	4	31	1	32	7600	112%	973803	1952	52.38	5
K-13	16	49	7	7	12	14	4	28	1	32	7632	112%	675566	2200	52.47	5
K-14	16	56	8	7	13	10	4	25	1	32	7632	112%	821233	2446	56.24	75
K-15	16	60	10	6	13	10	5	25	1	32	7696	113%	820463	4692	60.24	1209
K-16	16	56	8	7	13	11	4	26	1	32	7856	115%	845808	2460	57.92	242
K-17	16	56	8	7	11	14	4	28	1	32	8080	119%	661485	2474	57.99	254
K-18	16	63	9	7	13	9	4	23	1	32	8080	119%	868593	2720	60.48	1428

Table 13: Selection Set (128, 112, 2^{50}, 1,000,000)

The smallest signature parameter set is K-1 (16, 56, 8, 7, 10, 13, 5, 25) whose signatures are 1,072 (7,856 - 6,784) bytes smaller than the *bound-size* and that is about 14% reduction. This parameter set might be good for some

applications where the signature size reduction is impactful for performance. The parameter sets K-1 through K-15 have smaller signatures than the *bound-size*.

As seen, there were no parameter sets which had significant signature size reductions produced under the Selection Set (128, 112, 2^{50}, 1,000,000), we tried with the Selection Set (128, 112, 2^{50}, 10,000,000). L-i$_s$ in the table below are several parameter sets that we found.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
L-1	16	54	6	9	16	8	7	24	1	32	5072	100%	9836533	8262	54.90	29
L-2	16	56	7	8	14	9	8	23	1	32	5088	100%	8703475	16328	55.92	60
L-3	16	56	7	8	16	8	8	23	1	32	5104	100%	9833972	16329	56.90	119
L-4	16	56	7	8	15	9	8	24	1	32	5232	103%	9145843	16337	57.69	206
L-5	16	56	7	8	14	10	8	25	1	32	5328	105%	8752626	16343	57.97	250
L-6	16	64	8	8	13	9	8	23	1	32	5360	105%	9662450	18632	61.48	2856
L-7	16	55	5	11	16	8	4	24	1	32	5872	115%	7327734	1598	55.90	59
L-8	16	55	5	11	15	9	4	25	1	32	6000	118%	6639605	1606	56.69	103
L-9	16	55	5	11	17	8	4	25	1	32	6000	118%	8900598	1606	57.42	171
L-10	16	55	5	11	16	9	4	26	1	32	6144	121%	7524341	1615	58.10	274
L-11	16	55	5	11	15	10	4	27	1	32	6256	123%	6737908	1622	58.37	330
L-12	16	55	5	11	17	9	4	28	1	32	6288	123%	9293813	1624	59.32	639
L-13	16	55	5	11	16	10	4	28	1	32	6416	126%	7720948	1632	59.58	765
L-14	16	66	6	11	13	9	4	24	1	32	6448	127%	7127028	1880	63.48	11425

Table 14: Selection Set (128, 112, 2^{50}, 10,000,000)

The parameter set L-1 (16, 54, 6, 9, 16, 8, 7, 24)'s signature size is 2,784 (7,856 - 5,072) bytes smaller than the *bound-size*. This is more than 35% reduction in size.

Another attractive parameter set is L-14 (16, 66, 6, 11, 13, 9, 4, 24) because its risk of security degradation to below 112 bits is extremely low and significantly lower than L-1's and its signature size is 1,408 (7,856 - 6,448) bytes smaller than the *bound-size*.

Many L parameter sets are good for consideration. There are trade-offs between "sig bytes", "sign time", "verify time" and "*overuse safety*". One downside of these parameter sets is their slow signature generation operations.

When we set the hash limit to 100,000,000, we got the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
M-1	16	60	5	12	17	7	8	23	1	32	4432	100%	97165304	11713	59.21	592
M-2	16	60	5	12	18	7	8	24	1	32	4544	102%	99917816	11720	61.05	2120
M-3	16	60	4	15	17	7	4	23	1	32	5232	118%	76414968	1312	59.21	592
M-4	16	60	4	15	20	6	4	23	1	32	5232	118%	92536824	1312	60.10	1097

Table 15: Selection Set (128, 112, 2^{50}, 100,000,000)

M-1 (16, 60, 5, 12, 17, 7, 8, 23) has the smallest signature and good *overuse safety*. There are interesting trade-offs between the 4 M parameter sets.

10 Level 3 Security Parameter Sets

Similarly, we searched for parameter sets with 192 bits of security (which is NIST level 3) at different maximum numbers of signatures. However, instead of finding the bounds on total number of signatures per a signing key for

the retention of 112 bits of security, we found the bounds for the retention of 128 bits of security. That means that when overuse happens, at what bound of the total number of generated signatures per a key pair of a particular parameter set still retains 128 bits of security.

The smallest signature parameter sets at 192-bit security level in **FIPS 205** are SLH-DSA-SHA2-192s and SLH-DSA-SHAKE-192s and their signature size is 16,224 bytes. Again, we call this signature size, the *192-size* in this document to facilitate for easy discussions.

10.1 Level 3 security at 2^{20} signatures

We are now going to consider parameter sets under the Selection Set (192, 128, 2^{20}, 1,000,000). They are the N-i$_s$ in the table below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
N-1	24	24	4	6	13	16	6	30	3	48	9240	100%	950767	4606	28.68	410
N-2	24	25	5	5	12	17	7	30	3	48	9528	103%	823597	9853	29.00	512
N-3	24	24	4	6	13	17	6	32	3	48	9576	103%	975342	4620	29.23	600
N-4	24	20	4	5	11	23	8	35	3	48	9624	104%	993512	13614	25.11	34
N-5	24	20	5	4	13	18	8	33	3	48	9672	104%	974988	16919	25.70	51
N-6	24	28	4	7	12	16	5	28	3	48	9720	105%	885743	2930	31.39	2683
N-7	24	24	3	8	12	18	4	30	3	48	9888	107%	849390	1487	28.51	364
N-8	24	28	7	4	12	16	8	28	3	48	10056	108%	942284	23541	31.39	2683
N-9	24	24	3	8	12	19	4	32	3	48	10200	110%	861677	1500	28.95	494
N-10	24	24	3	8	12	20	4	33	3	48	10512	113%	873964	1513	29.34	648
N-11	24	28	4	7	12	16	4	28	3	48	10584	114%	615407	1874	31.39	2683

Table 16: Selection Set (192, 128, 2^{20}, 1,000,000)

Interestingly, N-6 (24, 28, 4, 7, 12, 16, 5, 28), N-8 (24, 28, 7, 4, 12, 16, 8, 28) and N-11 (24, 28, 4, 7, 12, 16, 4, 28) have the same risk level of falling below 128 bits of security (because they share the same h, a, k parameters). But N-6 has the smallest signature size among them. N-1 (24, 24, 4, 6, 13, 16, 6, 30) has the smallest signature size, but its risk is significantly higher than N-6's.

Let's see how much reduction in signature sizes when using the upper bound of 10,000,000 hash function evaluations for a signing. O-i$_s$ are the parameter sets that we found under the Selection Set (192, 128, 2^{20}, 10,000,000)

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
O-1	24	24	3	8	17	12	8	29	3	48	7656	100%	9831924	10229	30.24	1209
O-2	24	22	2	11	16	13	4	30	3	48	8304	108%	5906420	1063	28.04	263
O-3	24	24	2	12	16	13	4	30	3	48	8352	109%	9256948	1065	30.04	1052

Table 17: Selection Set (192, 128, 2^{20}, 10,000,000)

O-1 (24, 24, 3, 8, 17, 12, 8, 29) has the smallest signature size and the best *overuse safety*.

When we increased the hash limit to 100,000,000, we got the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
P-1	24	24	2	12	18	11	8	29	3	48	6864	100%	63193080	6893	30.28	1243
P-2	24	26	2	13	21	9	4	28	3	48	7848	114%	70025208	1044	32.73	6793

Table 18: Selection Set (192, 128, 2^{20}, 100,000,000)

P-1 (24, 24, 2, 12, 18, 11, 8, 29) has the best signature size, and decent *overuse safety*. P-2 has larger *overuse safety* than P-1. However, the *overuse safety* of P-1 is very large for its 2^{20} signature limit.

[7] had a recommended parameter set **Sphincs$^+$-192s-q20** which has the signature size of 7,008 bytes. We wanted to know if there were more attractive parameter sets in some aspects so we increased the signing time bound to 1,000,000,000 hashes and we found the parameter sets below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
PPP-1	24	30	2	15	23	8	8	27	3	48	6600	100%	637665280	6882	36.96	127487
PPP-2	24	20	1	20	21	10	4	30	3	48	7008	106%	920649728	651	28.18	290
PPP-3	24	19	1	19	24	9	4	30	3	48	7104	107%	881852416	655	28.77	436
PPP-4	24	20	1	20	22	10	4	31	3	48	7248	109%	983564288	661	29.19	584
PPP-5	24	20	1	20	21	11	4	32	3	48	7536	114%	926941184	673	29.35	652
PPP-6	24	19	1	19	24	10	4	33	3	48	7704	116%	932184064	680	30.20	1176

Table 19: Selection Set (192, 128, 2^{20}, 1,000,000,000)

PPP-2 is the parameter set **Sphincs$^+$-192s-q20**. PPP-1 is superior to PPP-2 in signing time, *overuse safety* and signature size.

10.2 Level 3 security at 2^{30} signatures

Now, searching under the Selection Set (192, 128, 2^{30}, 1,000,000), we found the parameter sets Q-i$_s$ below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
Q-1	24	30	5	6	12	20	6	34	3	48	11064	100%	942698	5737	35.34	40
Q-2	24	36	6	6	11	19	6	32	3	48	11256	101%	953066	6800	39.66	809
Q-3	24	36	6	6	11	20	6	33	3	48	11544	104%	959209	6812	40.10	1097
Q-4	24	32	4	8	11	21	4	33	3	48	11736	106%	966634	1922	36.48	89
Q-5	24	32	4	8	11	22	4	35	3	48	12024	108%	972777	1934	36.81	112
Q-6	24	36	9	4	9	24	8	32	3	48	12264	110%	995586	30239	38.59	385
Q-7	24	35	5	7	12	17	4	31	3	48	12288	111%	732397	2303	39.00	512
Q-8	24	35	5	7	13	16	4	31	3	48	12360	111%	916718	2306	39.68	820
Q-9	24	35	5	7	13	17	4	33	3	48	12696	114%	941293	2320	40.23	1200

Table 20: Selection Set (192, 128, 2^{30}, 1,000,000)

Q-2 (24, 36, 6, 6, 11, 19, 6, 32) 's signature size is about 200 bytes larger than Q-1's, but Q-2's risk of security degradation to below 128 bits is a factor of $2^{4.3}$ times smaller than Q-1's.

Now, searching under the Selection Set (192, 128, 2^{30}, 10,000,000), we found the parameter sets R-i$_s$ below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
R-1	24	32	4	8	16	13	8	30	3	48	8592	100%	9373682	13571	38.04	263
R-2	24	36	4	9	15	13	7	31	3	48	8760	101%	9146354	7930	40.92	1937
R-3	24	32	4	8	16	14	8	32	3	48	9000	104%	9570289	13588	38.78	439
R-4	24	40	5	8	12	16	8	29	3	48	9096	105%	8718830	16895	43.39	10734
R-5	24	33	3	11	17	12	4	31	3	48	9672	112%	9744372	1478	39.24	604
R-6	24	33	3	11	16	14	4	33	3	48	10200	118%	7778290	1500	39.78	879
R-7	24	44	4	11	15	12	4	30	3	48	10584	123%	7880691	1874	47.96	254975

Table 21: Selection Set (192, 128, 2^{30}, 10,000,000)

R-2 (24, 36, 4, 9, 15, 13, 7, 31)'s signature size is only 168 bytes larger than R-1's signature size, but R-2 has a noticeable larger *overuse safety*.

Under the Selection Set (192, 128, 2^{30}, 100,000,000), we found the parameter sets below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
S-1	24	36	3	12	19	10	8	29	3	48	7560	100%	97542136	10225	42.11	4420
S-2	24	30	2	15	18	12	4	31	3	48	8664	114%	63045624	1078	37.29	156
S-3	24	30	2	15	20	11	4	32	3	48	8736	115%	88211448	1081	38.34	324
S-4	24	30	2	15	20	12	4	34	3	48	9240	122%	91357176	1102	39.32	639
S-5	24	36	3	12	21	9	4	29	3	48	9312	123%	66674680	1463	42.73	6793

Table 22: Selection Set (192, 128, 2^{30}, 100,000,000)

S-1 (24, 36, 3, 12, 19, 10, 8, 29) has the smallest signatures and very good *overuse safety*. Other than fast verify times, the other parameter sets don't have enough to justify their larger signatures in our opinion.

10.3 Level 3 Security at 2^{40} signatures

Now, searching under the Selection Set (192, 128, 2^{40}, 1,000,000), we found the parameter sets T-i$_s$ below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
T-1	24	40	8	5	13	18	6	36	3	48	13560	100%	999913	9006	45.70	51
T-2	24	45	9	5	12	17	6	32	3	48	13752	101%	836137	10069	49.00	512
T-3	24	45	9	5	12	18	6	33	3	48	14064	103%	848424	10082	49.51	729
T-4	24	48	8	6	12	16	5	31	3	48	14232	104%	885739	5642	51.39	2683
T-5	24	42	6	7	12	19	4	35	3	48	14304	105%	861674	2745	46.95	123
T-6	24	42	6	7	12	20	4	36	3	48	14616	107%	873961	2758	47.34	162
T-7	24	49	7	7	12	16	4	31	3	48	14760	108%	929516	3122	52.39	5367

Table 23: Selection Set (192, 128, 2^{40}, 1,000,000)

T-1 (24, 40, 8, 5, 13, 18, 6, 36) has the smallest signature size and T-7 (24, 49, 7, 7, 12, 16, 4, 31) has the smallest risk of security degradation to below 128 bits.

Now, searching under the Selection Set (192, 128, 2^{40}, 10,000,000), we found the parameter sets U-i$_s$ below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
U-1	24	40	5	8	15	15	8	34	3	48	9864	100%	9996783	16927	46.36	82
U-2	24	42	6	7	16	13	8	32	3	48	10080	102%	7669232	20239	48.04	263
U-3	24	45	5	9	11	19	7	34	3	48	10176	103%	9952235	9880	48.66	404
U-4	24	48	6	8	16	12	7	30	3	48	10392	105%	8260593	11780	53.14	9026
U-5	24	49	7	7	16	12	8	31	3	48	10464	106%	8324848	23558	54.14	18053
U-6	24	44	4	11	16	13	4	33	3	48	11280	114%	9256946	1903	50.04	1052

Table 24: Selection Set (192, 128, 2^{40}, 10,000,000)

U-1 (24, 40, 5, 8, 15, 15, 8, 34) has the smallest signature size; while both U-4 (24, 48, 6, 8, 16, 12, 7, 30) and U-5 (24, 49, 7, 7, 16, 12, 8, 31) have very good *overuse safety* values.

When increasing the number of hashes to 100,000,000, we get the following parameter sets:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
V-1	24	44	4	11	18	11	8	32	3	48	8592	100%	63193076	13571	50.28	1243
V-2	24	45	3	15	18	11	4	31	3	48	9792	113%	89063416	1483	51.28	2486

Table 25: Selection Set (192, 128, 2^{40}, 100,000,000)

V-1 (24, 44, 4, 11, 18, 11, 8, 32) has the smallest signatures and very good *overuse safety*. V-2 (24, 45, 3, 15, 18, 11, 4, 31) has higher *overuse safety* than V-1 has; however at the cost of noticeably larger signatures.

10.4 Level 3 security at 2^{50} signatures

Now, searching under the Selection Set (192, 128, 2^{50}, 1,000,000), we found the parameter sets W-i$_s$ below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
W-1	24	50	10	5	12	20	6	37	3	48	15624	100%	942693	11202	55.34	40
W-2	24	55	11	5	12	17	6	34	3	48	15624	100%	975527	12257	59.00	512
W-3	24	55	11	5	12	18	6	35	3	48	15936	101%	987814	12270	59.51	729
W-4	24	49	7	7	12	21	4	39	3	48	16320	104%	990951	3187	54.69	25
W-5	24	60	12	5	11	18	6	33	3	48	16440	105%	946917	13346	63.16	9152
W-6	24	56	8	7	11	19	4	35	3	48	16632	106%	954344	3558	59.66	809
W-7	24	56	8	7	11	20	4	36	3	48	16920	108%	960487	3570	60.10	1097

Table 26: Selection Set (192, 128, 2^{50}, 1,000,000)

Only W-1, W-2 and W-3 have smaller signature sizes than the *192-size*, but with relatively small percentages in the size reductions.
Now, searching under the Selection Set (192, 128, 2^{50}, 10,000,000), we found the parameter sets X-i$_s$ below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
X-1	24	56	8	7	16	12	8	32	3	48	11256	100%	9177071	26894	61.14	2256
X-2	24	55	5	11	12	17	4	34	3	48	12768	113%	8585197	2323	59.00	512
X-3	24	55	5	11	13	16	4	34	3	48	12840	114%	8769518	2326	59.68	820

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
X-4	24	55	5	11	15	14	4	35	3	48	12840	114%	9752560	2326	60.70	1663

Table 27: Selection Set (192, 128, 2^{50}, 10,000,000)

X-1 (24, 56, 8, 7, 16, 12, 8, 32) has the smallest signature size and the biggest *overuse safety*. The downside is its verification time which is about 10 times the other 3 parameter sets' verification times.

Increasing the hash limit to 100,000,000, we found the parameter sets below:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
Y-1	24	55	5	11	18	11	8	33	3	48	9480	100%	76828656	16911	61.28	2486
Y-2	24	56	4	14	19	10	4	32	3	48	11064	116%	69337080	1894	62.11	4420

Table 28: Selection Set (192, 128, 2^{50}, 100,000,000)

Y-1 (24, 55, 5, 11, 18, 11, 8, 33) has the smallest signature size and very good *overuse safety*. Y-2 (24, 56, 4, 14, 19, 10, 4, 32) has higher *overuse safety* than Y-1. Since Y-1's *overuse safety* is very good, the bigger signature size of Y-2 may make it less attractive then Y-1 in many use cases.

11 Conclusions

In this paper, we figured out what is the success probability of a forgery attack with a random or chosen message by an attacker when they are given 2^m valid signatures. This success probability is called p in the equation (1). For any given parameter set, when m increases, p increases. From here, we plotted the security degradation behavior of a given parameter set, especially at what value of m, the security strength falls below the retained security level which is either 112 or 128 bits of security.

We generated **Sphincs**$^+$parameter sets, then grouped them based on a Selection Set which has 4 variables as following:

- The security level of the parameter sets (either 128 or 192 bits).
- The retained security strength when overuse happens (either 112 or 128 bits).
- The maximum number of signatures allowed to be generated under 1 signing key (either 2^{20}, 2^{30}, 2^{40} or 2^{50}).
- The limit for the number of hashes per signing operation (either $100,000$, $1,000,000$, $10,000,000$, $100,000,000$, $600,000,000$ or $1,000,000,000$).

We provided many good parameter sets under each Selection Set and generally they have different trade-offs between signing time, verification time, signature size and *overuse safety*, in this paper. We evaluated the parameter sets and pointed out the ones we thought are special for the users and standard bodies to pay more attention to.

We searched and found parameter sets which are more attractive than the 2 parameter sets **Sphincs$^+$-128s-q20** and **Sphincs$^+$-192s-q20** recommended in [7] in our opinion, and they are the parameter sets **AAA-1** and **PPP-1** in this paper.

We generated other parameter sets which were not discussed in the main body of this paper. They are listed in the Appendices 11.

A References

References

[1] National Institute of Standards and Technology. Stateless hash-based digital signature standard. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 205, August 2024. https://doi.org/10.6028/NIST.FIPS.205.

[2] Jean-Philippe Aumasson, Daniel J. Bernstein, Ward Beullens, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, and Bas Westerbaan. SPHINCS$^+$ – Submission to the NIST post-quantum project, v.3.1, 2022.

[3] Jean-Philippe Aumasson and Guillaume Endignoux. Improving stateless hash-based signatures. In *Topics In Cryptology - Ct-Rsa 2018*, pages 219–242, New York, NY, 3028. Springer New York.

[4] Eyal Ronen Mikhail Kudinov, Andreas Hülsing and Eylon Yogev. Sphincs+c: Compressing sphincs+ with (almost) no cost. eprint archive 2022/778, June 2022. https://eprint.iacr.org/2022/778.

[5] Kaiyi Zhang, Hongrui Cui, and Yu Yu. Sphincs-α: A compact stateless hash-based signature scheme. *IACR Cryptol. ePrint Arch.*, page 59, 2022.

[6] Stefan Kölbl and Jade Philipoom. A note on sphincs+ parameter sets. eprint archive 2022/1725, 2022. https://eprint.iacr.org/2022/1725.

[7] Stefan Kölbl and Jade Philipoom. A note on sphincs+ parameter sets, April 2024. https://csrc.nist.gov/csrc/media/Events/2024/fifth-pqc-standardization-conference/documents/papers/a-note-on-sphincs-plus-parameter-sets.pdf.

[8] Leslie Lamport. Constructing digital signatures from a one way function. Technical Report CSL-98, October 1979. This paper was published by IEEE in the Proceedings of HICSS-43 in January, 2010.

[9] Ralph C. Merkle. A certified digital signature. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 218–238, New York, NY, 1990. Springer New York.

[10] David McGrew, Michael Curcio, and Scott Fluhrer. Leighton-Micali Hash-Based Signatures. RFC 8554, April 2019.

[11] Andreas Hülsing, Denis Butin, Stefan-Lukas Gazdag, Joost Rijneveld, and Aziz Mohaisen. XMSS: eXtended Merkle Signature Scheme. RFC 8391, May 2018.

[12] Leon Groot Bruinderink and Andreas Hülsing. "oops, i did it again" – security of one-time signatures under two-message attacks. In Carlisle Adams and Jan Camenisch, editors, *Selected Areas in Cryptography – SAC 2017*, pages 299–322, Cham, 2018. Springer International Publishing.

[13] Scott Fluhrer. Oops, i did it again revisited; another look at reusing one-time signatures. eprint archive 2023/1905, December 2023. https://eprint.iacr.org/2023/1905.

[14] Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn. Sphincs: Practical stateless hash-based signatures. In *EUROCRYPT (1)*, pages 368–397. Springer, 2015.

[15] Leonid Reyzin and Natan Reyzin. Better than biba: Short one-time signatures with fast signing and verifying. In Lynn Batten and Jennifer Seberry, editors, *Information Security and Privacy*, pages 144–153, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[16] Stefan Kölbl Ruben Niederhagen Joost Rijneveld Daniel J. Bernstein, Andreas Hülsing and Peter Schwabe. The sphincs$^+$ signature framework. In *CCS '19: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages "2129–2146", "New York, NY", 2019. "Association for Computing Machinery".

[17] Aymeric Genêt. On Protecting SPHINCS+ Against Fault Attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(2):80–114, Mar. 2023. https://doi.org/10.46586/tches.v2023.i2.80-114.

[18] National Institute of Standards and Technology. SHA-3 standard: Permutation-based hash and extendable-output functions. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 202, August 2015. https://doi.org/10.6028/NIST.FIPS.202.

[19] National Institute of Standards and Technology. Secure hash standard (SHS). (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 180-4, August 2015. https://doi.org/10.6028/NIST.FIPS.180-4.

[20] Andrew Chamberlain. Deriving the poisson distribution from the binomial distribution. https://medium.com/@andrew.chamberlain/deriving-the-poisson-distribution-from-the-binomial-distribution-840cc16~:text=So%20we've%20shown%20that,And%20that%20completes%20the%20proof, 2016. Accessed: 11/18/2024.

[21] Scott Fluhrer and Quynh Dang. Smaller sphincs+ or, honey, i shrunk the signatures. eprint archive 2024/018, January 2024. https://eprint.iacr.org/2024/018.pdf.

[22] National Institute of Standards and Technology. Module-lattice-based digital signature standard. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 204, August 2024. https://doi.org/10.6028/NIST.FIPS.204.

B Additional Parameter Sets at Security Levels 1 and 3

We also generated other sets of parameter sets where the signing bound was 100,000 hashes. However, except for the Level 1, 2^{20}-signature parameter sets, we didn't find any attractive parameter sets because either we felt that the signature size reductions were not significant over the *bound-size*, the signatures were even larger than the *bound-size* or the *192-size* for the security Level 1 or Level 3 respectively.

Here are those parameter sets:

B.1 Additional Parameter Sets at Security Level 1

For Level 1, 2^{30} signature limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
1	16	32	8	4	9	17	4	25	1	32	7728	100%	98026	2452	32.65	6
2	16	32	8	4	9	18	4	26	1	32	7888	102%	99561	2462	33.31	9
3	16	36	9	4	8	17	4	22	1	32	8080	104%	93961	2720	34.08	16
4	16	36	9	4	8	18	4	23	1	32	8224	106%	94728	2729	35.00	32
5	16	36	9	4	8	19	4	24	1	32	8368	108%	95495	2738	35.74	53
6	16	36	9	4	8	20	4	25	1	32	8512	110%	96262	2747	36.35	81
7	16	36	9	4	8	21	4	26	1	32	8656	112%	97029	2756	36.86	116
8	16	36	9	4	8	22	4	27	1	32	8800	113%	97796	2765	37.31	158
9	16	40	10	4	7	20	4	24	1	32	8816	114%	97573	3012	37.50	181
10	16	40	10	4	7	21	4	25	1	32	8944	115%	97956	3020	38.25	304
11	16	40	10	4	7	22	4	26	1	32	9072	117%	98339	3028	38.87	467
12	16	40	10	4	7	23	4	27	1	32	9200	119%	98722	3036	39.39	670
13	16	40	10	4	7	24	4	27	1	32	9328	120%	99105	3044	39.85	922
14	16	40	10	4	7	25	4	28	1	32	9456	122%	99488	3052	40.24	1209

Table 29: Selection Set (128, 112, 2^{30}, 100,000)

For Level 1, 2^{40} signature limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
1	16	40	10	4	6	40	4	36	1	32	10736	100%	97553	3132	41.22	2
2	16	40	10	4	6	41	4	37	1	32	10848	101%	97744	3139	41.36	2
3	16	40	10	4	6	42	4	38	1	32	10960	102%	97935	3146	41.48	2
4	16	40	10	4	9	19	3	28	1	32	11056	102%	88358	2082	41.85	3
5	16	40	10	4	6	43	4	39	1	32	11072	103%	98126	3153	41.60	3
6	16	40	10	4	6	44	4	39	1	32	11184	104%	98317	3160	41.71	3
7	16	40	10	4	9	20	3	29	1	32	11216	104%	89893	2092	42.32	4
8	16	42	14	3	9	17	4	26	1	32	11248	104%	89028	4148	42.65	6
9	16	44	11	4	9	16	3	24	1	32	11376	105%	89672	2241	43.83	14
10	16	42	14	3	9	18	4	27	1	32	11408	106%	90563	4158	43.31	9
11	16	44	11	4	9	17	3	26	1	32	11536	107%	91207	2251	44.65	25
12	16	42	14	3	9	19	4	28	1	32	11568	107%	92098	4168	43.85	14
13	16	45	15	3	9	16	4	25	1	32	11696	108%	91988	4422	44.83	28
14	16	44	11	4	9	18	3	27	1	32	11696	108%	92742	2261	45.31	39
15	16	45	15	3	9	17	4	27	1	32	11856	110%	93523	4432	45.65	50
16	16	44	11	4	9	19	3	28	1	32	11856	110%	94277	2271	45.85	57
17	16	45	15	3	9	18	4	28	1	32	12016	111%	95058	4442	46.31	79
18	16	48	12	4	9	15	3	24	1	32	12016	111%	94056	2420	46.79	110
19	16	48	16	3	9	15	4	24	1	32	12144	113%	94948	4696	46.79	110
20	16	45	15	3	9	19	4	29	1	32	12176	113%	96593	4452	46.85	115
21	16	48	12	4	9	16	3	25	1	32	12176	113%	95591	2430	47.83	227
22	16	48	16	3	9	16	4	25	1	32	12304	114%	96483	4706	47.83	227
23	16	48	12	4	9	17	3	27	1	32	12336	114%	97126	2440	48.65	401
24	16	48	16	3	9	17	4	27	1	32	12464	116%	98018	4716	48.65	401
25	16	48	12	4	9	18	3	28	1	32	12496	116%	98661	2450	49.31	634
26	16	48	16	3	9	18	4	28	1	32	12624	117%	99553	4726	49.31	634
27	16	52	13	4	9	14	3	23	1	32	12656	117%	98440	2599	49.39	670
28	16	51	17	3	9	15	4	24	1	32	12752	118%	99443	4980	49.79	885
29	16	52	13	4	9	15	3	24	1	32	12816	119%	99975	2609	50.79	1770
30	16	51	17	3	8	18	4	25	1	32	12944	120%	90224	4992	50.00	1024

Table 30: Selection Set (128, 112, 2^{40}, 100,000)

For Level 1, 2^{50} signature limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
1	16	52	13	4	8	20	3	27	1	32	13296	100%	92290	2639	52.35	5
2	16	52	13	4	8	21	3	28	1	32	13440	101%	93057	2648	52.86	7
3	16	51	17	3	8	22	4	29	1	32	13520	101%	93292	5028	52.31	4
4	16	52	13	4	8	22	3	29	1	32	13584	102%	93824	2657	53.31	9
5	16	51	17	3	8	23	4	30	1	32	13664	102%	94059	5037	52.69	6
6	16	56	14	4	8	17	3	25	1	32	13664	102%	95908	2801	54.08	16
7	16	54	18	3	8	19	4	27	1	32	13696	103%	95486	5285	53.74	13
8	16	56	14	4	8	18	3	26	1	32	13808	103%	96675	2810	55.00	32
9	16	54	18	3	8	20	4	28	1	32	13840	104%	96253	5294	54.35	20
10	16	56	14	4	8	19	3	27	1	32	13952	104%	97442	2819	55.74	53
11	16	54	18	3	8	21	4	29	1	32	13984	105%	97020	5303	54.86	29
12	16	57	19	3	8	17	4	25	1	32	14016	105%	98447	5551	55.08	33

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
13	16	56	14	4	8	20	3	28	1	32	14096	106%	98209	2828	56.35	81
14	16	54	18	3	8	22	4	30	1	32	14128	106%	97787	5312	55.31	39
15	16	57	19	3	8	18	4	26	1	32	14160	106%	99214	5560	56.00	64
16	16	56	14	4	8	21	3	29	1	32	14240	107%	98976	2837	56.86	116
17	16	57	19	3	8	19	4	27	1	32	14304	107%	99981	5569	56.74	106
18	16	56	14	4	8	22	3	30	1	32	14384	108%	99743	2846	57.31	158
19	16	60	15	4	7	20	3	26	1	32	14576	109%	96448	2997	57.50	181
20	16	57	19	3	7	24	4	29	1	32	14640	110%	94600	5590	56.85	115
21	16	60	15	4	7	21	3	27	1	32	14704	110%	96831	3005	58.25	304
22	16	60	20	3	7	20	4	27	1	32	14736	110%	97563	5842	57.50	181
23	16	60	15	4	7	22	3	28	1	32	14832	111%	97214	3013	58.87	467
24	16	60	20	3	7	21	4	28	1	32	14864	111%	97946	5850	58.25	304
25	16	60	15	4	7	23	3	29	1	32	14960	112%	97597	3021	59.39	670
26	16	60	20	3	7	22	4	29	1	32	14992	112%	98329	5858	58.87	467
27	16	60	15	4	7	24	3	29	1	32	15088	113%	97980	3029	59.85	922
28	16	60	20	3	7	23	4	30	1	32	15120	113%	98712	5866	59.39	670
29	16	60	15	4	7	25	3	30	1	32	15216	114%	98363	3037	60.24	1209
30	16	60	20	3	7	24	4	30	1	32	15248	114%	99095	5874	59.85	922
31	16	60	20	3	7	25	4	31	1	32	15376	115%	99478	5882	60.24	1209

Table 31: Selection Set (128, 112, 2^{50}, 100,000)

For Level 3, 2^{20} signature limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
1	24	20	5	4	8	38	4	41	3	48	14832	100%	94584	2409	23.85	14
2	24	24	6	4	7	37	4	37	3	48	15048	101%	92696	2776	25.94	61
3	24	24	6	4	7	38	4	38	3	48	15240	102%	93079	2784	26.10	68
4	24	25	5	5	7	35	3	35	3	48	15384	103%	99483	1652	26.59	96
5	24	24	6	4	7	39	4	39	3	48	15432	104%	93462	2792	26.25	76
6	24	25	5	5	7	36	3	36	3	48	15576	105%	99866	1660	26.77	109
7	24	24	6	4	7	40	4	39	3	48	15624	105%	93845	2800	26.39	83
8	24	24	6	4	7	41	4	40	3	48	15816	106%	94228	2808	26.52	91
9	24	28	7	4	6	40	4	34	3	48	15984	107%	99252	3173	28.08	270
10	24	28	7	4	6	41	4	35	3	48	16152	108%	99443	3180	28.26	306
11	24	28	7	4	6	42	4	36	3	48	16320	110%	99634	3187	28.42	342
12	24	28	7	4	6	43	4	37	3	48	16488	111%	99825	3194	28.57	380
13	24	27	9	3	9	23	4	30	3	48	17208	116%	94195	3940	29.22	596
14	24	27	9	3	9	24	4	31	3	48	17448	117%	95730	3950	29.59	770
15	24	28	7	4	9	23	3	30	3	48	17472	117%	95557	2143	30.22	1192
16	24	27	9	3	9	25	4	33	3	48	17688	119%	97265	3960	29.91	962
17	24	27	9	3	9	26	4	34	3	48	17928	120%	98800	3970	30.20	1176

Table 32: Selection Set (192, 128, 2^{20}, 100,000)

B.2 Additional Parameter Sets at Security Level 3

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
1	24	32	8	4	8	33	3	38	3	48	20784	100%	94170	2483	35.11	34
2	24	33	11	3	8	31	4	36	3	48	20976	100%	95753	4813	35.73	53
3	24	33	11	3	8	32	4	37	3	48	21192	101%	96520	4822	35.92	60
4	24	33	11	3	8	33	4	38	3	48	21408	103%	97287	4831	36.11	69
5	24	36	9	4	8	28	3	33	3	48	21408	103%	98942	2711	38.01	257
6	24	33	11	3	8	34	4	39	3	48	21624	104%	98054	4840	36.28	77
7	24	36	12	3	8	28	4	34	3	48	21624	104%	99995	5198	38.01	257
8	24	36	9	4	8	29	3	34	3	48	21624	104%	99709	2720	38.27	308
9	24	39	13	3	7	31	4	34	3	48	22824	109%	96935	5606	39.68	820
10	24	40	10	4	7	30	3	33	3	48	22824	109%	97563	2972	40.39	1341
11	24	39	13	3	7	32	4	34	3	48	23016	110%	97318	5614	39.94	982
12	24	39	13	3	7	33	4	35	3	48	23208	111%	97701	5622	40.17	1152

Table 33: Selection Set (192, 128, 2^{30}, 100,000)

For Level 3, 2^{40} signature limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
1	24	40	10	4	6	70	3	59	3	48	28824	100%	99443	3222	42.82	7
2	24	40	10	4	6	71	3	60	3	48	28992	100%	99634	3229	42.86	7
3	24	40	10	4	6	72	3	60	3	48	29160	101%	99825	3236	42.91	7
4	24	42	14	3	8	33	3	39	3	48	30672	106%	85556	4107	45.11	34
5	24	42	14	3	8	34	3	40	3	48	30888	107%	86323	4116	45.28	38
6	24	42	14	3	8	35	3	41	3	48	31104	107%	87090	4125	45.44	43
7	24	42	14	3	8	36	3	42	3	48	31320	108%	87857	4134	45.59	48
8	24	45	15	3	8	29	3	36	3	48	31488	109%	86791	4343	47.27	154
9	24	45	15	3	8	30	3	37	3	48	31704	109%	87558	4352	47.51	182
10	24	45	15	3	8	31	3	38	3	48	31920	110%	88325	4361	47.73	212
11	24	45	15	3	8	32	3	39	3	48	32136	111%	89092	4370	47.92	242
12	24	45	15	3	8	33	3	40	3	48	32352	112%	89859	4379	48.11	276
13	24	45	15	3	8	34	3	41	3	48	32568	112%	90626	4388	48.28	310
14	24	48	16	3	8	27	3	34	3	48	32736	113%	89560	4597	49.72	843
15	24	48	16	3	8	28	3	35	3	48	32952	114%	90327	4606	50.01	1031
16	24	40	10	4	8	38	2	44	3	48	33432	115%	94099	2414	43.85	14
17	24	40	10	4	8	39	2	45	3	48	33648	116%	94866	2423	43.98	15
18	24	40	20	2	8	38	4	44	3	48	33672	116%	94569	8564	43.85	14
19	24	42	21	2	8	33	4	39	3	48	33864	117%	94005	8930	45.11	34
20	24	42	21	2	8	34	4	40	3	48	34080	118%	94772	8939	45.28	38
21	24	44	11	4	8	30	2	36	3	48	34224	118%	94458	2549	46.51	91
22	24	42	21	2	8	35	4	41	3	48	34296	118%	95539	8948	45.44	43
23	24	44	11	4	8	31	2	37	3	48	34440	119%	95225	2558	46.73	106
24	24	44	22	2	8	30	4	37	3	48	34488	119%	94975	9314	46.51	91
25	24	44	11	4	8	32	2	38	3	48	34656	120%	95992	2567	46.92	121
26	24	44	22	2	8	31	4	38	3	48	34704	120%	95742	9323	46.73	106
27	24	44	11	4	8	33	2	39	3	48	34872	120%	96759	2576	47.11	138
28	24	44	22	2	8	32	4	39	3	48	34920	121%	96509	9332	46.92	121
29	24	44	11	4	8	34	2	40	3	48	35088	121%	97526	2585	47.28	155
30	24	44	22	2	8	33	4	40	3	48	35136	121%	97276	9341	47.11	138
31	24	44	11	4	8	35	2	41	3	48	35304	122%	98293	2594	47.44	173
32	24	46	23	2	8	28	4	35	3	48	35328	122%	96712	9707	48.01	257

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
33	24	46	23	2	8	29	4	36	3	48	35544	123%	97479	9716	48.27	308
34	24	46	23	2	8	30	4	37	3	48	35760	124%	98246	9725	48.51	364
35	24	46	23	2	8	31	4	38	3	48	35976	124%	99013	9734	48.73	424
36	24	48	12	4	8	27	2	34	3	48	36096	125%	98652	2729	49.72	843
37	24	46	23	2	8	32	4	39	3	48	36192	125%	99780	9743	48.92	484
38	24	48	12	4	8	28	2	35	3	48	36312	125%	99419	2738	50.01	1031
39	24	48	24	2	8	27	4	34	3	48	36384	126%	99216	10109	49.72	843
40	24	48	24	2	8	28	4	35	3	48	36600	126%	99983	10118	50.01	1031

Table 34: Selection Set (192, 128, 2^{40}, 100,000)

For Level 3, 2^{50} signature limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
1	24	51	17	3	8	35	3	42	3	48	36144	100%	99999	4941	54.44	21
2	24	54	18	3	7	37	3	41	3	48	37368	103%	91628	5194	55.94	61
3	24	54	18	3	7	38	3	42	3	48	37560	103%	92011	5202	56.10	68
4	24	54	18	3	7	39	3	43	3	48	37752	104%	92394	5210	56.25	76
5	24	54	18	3	7	40	3	43	3	48	37944	104%	92777	5218	56.39	83
6	24	54	18	3	7	41	3	44	3	48	38136	105%	93160	5226	56.52	91
7	24	57	19	3	7	33	3	37	3	48	38280	105%	94399	5434	58.17	288
8	24	57	19	3	7	34	3	38	3	48	38472	106%	94782	5442	58.39	335
9	24	57	19	3	7	35	3	39	3	48	38664	106%	95165	5450	58.59	385
10	24	57	19	3	7	36	3	40	3	48	38856	107%	95548	5458	58.77	436
11	24	57	19	3	7	37	3	41	3	48	39048	108%	95931	5466	58.94	491
12	24	57	19	3	7	38	3	42	3	48	39240	108%	96314	5474	59.10	548
13	24	60	20	3	7	30	3	36	3	48	39384	108%	97553	5682	60.39	1341
14	24	52	13	4	6	56	2	49	3	48	42192	116%	95134	3085	53.97	15
15	24	52	13	4	6	57	2	50	3	48	42360	117%	95325	3092	54.04	16
16	24	54	27	2	6	48	4	44	3	48	42432	117%	97488	11435	55.22	37
17	24	56	14	4	6	43	2	41	3	48	42528	117%	99146	3201	56.57	95
18	24	54	27	2	6	49	4	45	3	48	42600	117%	97679	11442	55.33	40
19	24	56	14	4	6	44	2	41	3	48	42696	118%	99337	3208	56.72	105
20	24	54	27	2	6	50	4	46	3	48	42768	118%	97870	11449	55.44	43
21	24	56	28	2	6	43	4	41	3	48	42864	118%	99804	11811	56.57	95
22	24	56	14	4	6	45	2	42	3	48	42864	118%	99528	3215	56.85	115
23	24	56	28	2	6	44	4	41	3	48	43032	119%	99995	11818	56.72	105
24	24	56	14	4	6	46	2	43	3	48	43032	119%	99719	3222	56.98	126
25	24	56	14	4	6	47	2	44	3	48	43200	119%	99910	3229	57.11	138
26	24	58	29	2	5	54	4	42	3	48	44688	123%	99992	12245	57.05	132
27	24	56	14	4	5	83	2	60	3	48	47256	130%	98818	3398	57.12	139
28	24	56	14	4	5	84	2	61	3	48	47400	131%	98913	3404	57.16	143
29	24	56	28	2	5	82	4	60	3	48	47448	131%	99381	12002	57.08	135
30	24	56	14	4	5	85	2	62	3	48	47544	131%	99008	3410	57.20	147
31	24	56	28	2	5	83	4	60	3	48	47592	131%	99476	12008	57.12	139
32	24	56	14	4	5	86	2	62	3	48	47688	131%	99103	3416	57.25	152
33	24	56	28	2	5	84	4	61	3	48	47736	132%	99571	12014	57.16	143
34	24	56	14	4	5	87	2	63	3	48	47832	132%	99198	3422	57.29	156
35	24	56	28	2	5	85	4	62	3	48	47880	132%	99666	12020	57.20	147
36	24	56	14	4	5	88	2	63	3	48	47976	132%	99293	3428	57.32	159

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
37	24	56	28	2	5	86	4	62	3	48	48024	132%	99761	12026	57.25	152
38	24	56	14	4	5	89	2	64	3	48	48120	133%	99388	3434	57.36	164
39	24	56	28	2	5	87	4	63	3	48	48168	133%	99856	12032	57.29	156
40	24	56	14	4	5	90	2	65	3	48	48264	133%	99483	3440	57.40	168
41	24	56	28	2	5	88	4	63	3	48	48312	133%	99951	12038	57.32	159
42	24	56	14	4	5	91	2	65	3	48	48408	133%	99578	3446	57.44	173
43	24	56	14	4	5	92	2	66	3	48	48552	134%	99673	3452	57.47	177
44	24	56	14	4	5	93	2	67	3	48	48696	134%	99768	3458	57.51	182
45	24	56	14	4	5	94	2	67	3	48	48840	135%	99863	3464	57.54	186
46	24	56	14	4	5	95	2	68	3	48	48984	135%	99958	3470	57.57	190
47	24	54	18	3	8	36	2	44	3	48	52728	145%	86061	4034	57.59	192
48	24	54	18	3	8	37	2	45	3	48	52944	146%	86828	4043	57.72	210
49	24	57	19	3	9	23	2	34	3	48	52968	146%	97001	4146	59.22	596
50	24	57	19	3	9	24	2	35	3	48	53208	147%	98536	4156	59.59	770
51	24	57	19	3	8	31	2	39	3	48	54144	149%	85473	4195	59.73	849
52	24	57	19	3	8	32	2	40	3	48	54360	150%	86240	4204	59.92	968
53	24	57	19	3	8	33	2	41	3	48	54576	150%	87007	4213	60.11	1105

Table 35: Selection Set (192, 128, 2^{50}, 100,000)

C Parameter Sets at 2^{10} Signature Limit at Security Levels 1 and 3

We don't expect that 2^{10} signature limit is good enough for standardization consideration. However, some practitioners may have a different opinion, hence they are listed here.

C.1 Security Level 1 at 2^{10} Signature Limit

For 100,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
aa-1	16	12	2	6	9	17	4	22	1	32	4048	100%	98032	746	12.65	6
aa-2	16	12	3	4	10	15	5	21	1	32	4192	103%	89169	1526	13.08	8
aa-3	16	12	2	6	9	18	4	23	1	32	4208	103%	99567	756	13.31	9
aa-4	16	16	4	4	10	13	5	20	1	32	4352	107%	97394	1957	14.76	27
aa-5	16	15	3	5	10	14	4	21	1	32	4400	108%	96946	1014	15.09	34
aa-6	16	16	4	4	9	16	5	21	1	32	4624	114%	82031	1974	15.83	56
aa-7	16	15	3	5	9	17	4	23	1	32	4656	115%	80047	1030	15.65	50
aa-8	16	16	4	4	9	17	5	23	1	32	4784	118%	83566	1984	16.65	100
aa-9	16	15	3	5	9	18	4	24	1	32	4816	118%	81582	1040	16.31	79
aa-10	16	20	4	5	9	14	4	19	1	32	4816	118%	93425	1286	17.39	167
aa-11	16	20	4	5	9	15	4	20	1	32	4976	122%	94960	1296	18.79	442
aa-12	16	20	4	5	9	16	4	21	1	32	5136	126%	96495	1306	19.83	910
aa-13	16	20	4	5	9	17	4	23	1	32	5296	130%	98030	1316	20.65	1606

Table 36: Selection Set (128, 112, 2^{10}, 100,000)

For a 1,000,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
bb-1	16	14	2	7	12	11	7	19	1	32	3200	100%	823798	2849	14.12	17
bb-2	16	12	2	6	13	11	8	20	1	32	3248	101%	860406	4778	13.92	15
bb-3	16	16	2	8	10	13	6	19	1	32	3328	104%	827380	1699	14.76	27
bb-4	16	16	2	8	14	9	5	18	1	32	3328	104%	902136	1051	15.92	60
bb-5	16	16	2	8	12	11	6	19	1	32	3328	104%	922614	1699	16.12	69
bb-6	16	9	1	9	14	11	4	22	1	32	3360	105%	828407	457	12.35	5
bb-7	16	15	3	5	13	10	8	20	1	32	3360	105%	688310	7072	15.24	37
bb-8	16	18	3	6	9	14	8	19	1	32	3408	106%	906610	7075	15.39	41
bb-9	16	16	2	8	13	10	5	19	1	32	3408	106%	705527	1056	16.24	75
bb-10	16	10	1	10	13	12	4	22	1	32	3424	107%	870390	461	13.10	8
bb-11	16	18	3	6	10	13	8	20	1	32	3456	108%	925043	7078	16.76	108
bb-12	16	18	3	6	11	12	8	20	1	32	3472	108%	958836	7079	17.63	198
bb-13	16	18	3	6	14	9	7	19	1	32	3472	108%	958839	4190	17.92	242
bb-14	16	16	2	8	14	9	4	18	1	32	3552	111%	730104	715	15.92	60
bb-15	16	18	3	6	13	10	7	20	1	32	3552	111%	762230	4195	18.24	302
bb-16	16	20	4	5	12	10	8	18	1	32	3568	111%	712949	9372	17.86	232
bb-17	16	21	3	7	12	10	6	18	1	32	3584	112%	713462	2460	18.86	464
bb-18	16	16	2	8	13	10	4	19	1	32	3632	113%	533495	720	16.24	75
bb-19	16	18	3	6	10	14	8	21	1	32	3632	113%	928114	7089	18.09	272
bb-20	16	18	2	9	13	10	4	21	1	32	3664	114%	821239	722	18.24	302
bb-21	16	18	3	6	11	13	8	21	1	32	3664	114%	964979	7091	18.99	508
bb-22	16	20	4	5	13	10	8	20	1	32	3728	116%	835829	9382	20.24	1209
bb-23	16	18	2	9	13	11	4	22	1	32	3888	121%	845814	736	19.92	968
bb-24	16	24	3	8	13	9	4	18	1	32	4096	128%	652791	995	21.48	2856

Table 37: Selection Set (128, 112, 2^{10}, 1,000,000)

For a 10,000,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
cc-1	16	12	1	12	15	9	6	19	1	32	2896	100%	7184377	927	13.69	12
cc-2	16	12	1	12	17	8	6	19	1	32	2896	100%	9445370	927	14.42	21
cc-3	16	10	1	10	16	9	8	20	1	32	2912	100%	6490105	2470	13.10	8
cc-4	16	16	2	8	18	7	8	18	1	32	2976	102%	7865338	4761	17.05	132
cc-5	16	20	2	10	12	10	8	19	1	32	2992	103%	9564151	4762	17.86	232
cc-6	16	20	2	10	17	7	7	19	1	32	3024	104%	8261626	2838	19.21	592
cc-7	16	18	2	9	14	9	8	20	1	32	3040	104%	5163000	4765	17.92	242
cc-8	16	18	2	9	16	8	8	20	1	32	3056	105%	6293497	4766	18.90	477
cc-9	16	12	1	12	15	9	4	19	1	32	3072	106%	3186681	439	13.69	12
cc-10	16	12	1	12	17	8	4	19	1	32	3072	106%	5447674	439	14.42	21
cc-11	16	20	2	10	14	9	8	20	1	32	3072	106%	9883640	4767	19.92	968
cc-12	16	13	1	13	15	9	4	19	1	32	3088	106%	5488633	440	14.69	25
cc-13	16	13	1	13	17	8	4	19	1	32	3088	106%	7749626	440	15.42	42
cc-14	16	20	2	10	13	10	8	21	1	32	3152	108%	9687031	4772	20.24	1209
cc-15	16	12	1	12	18	8	4	20	1	32	3200	110%	8593402	447	15.70	51
cc-16	16	14	1	14	14	10	4	20	1	32	3200	110%	9699320	447	15.97	62
cc-17	16	13	1	13	16	9	4	20	1	32	3232	111%	6373369	449	16.10	68
cc-18	16	13	1	13	15	10	4	21	1	32	3344	115%	5586936	456	16.37	82
cc-19	16	13	1	13	17	9	4	22	1	32	3376	116%	8142841	458	17.32	159
cc-20	16	14	1	14	14	11	4	22	1	32	3440	118%	9748471	462	17.35	163

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
cc-21	16	20	2	10	17	7	4	19	1	32	3472	119%	3903482	710	19.21	592
cc-22	16	22	2	11	13	9	4	19	1	32	3504	120%	2523128	712	19.48	714
cc-23	16	22	2	11	17	7	4	19	1	32	3504	120%	5054458	712	21.21	2368

Table 38: Selection Set (128, 112, 2^{10}, 10,000,000)

And for a 100,000,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 112	overuse safety
dd-1	16	14	1	14	21	6	7	18	1	32	2688	100%	81821696	1493	15.68	51
dd-2	16	14	1	14	16	8	8	18	1	32	2704	100%	77103096	2457	14.90	29
dd-3	16	12	1	12	22	6	8	19	1	32	2704	100%	94380032	2457	14.99	31
dd-4	16	15	1	15	18	7	7	18	1	32	2720	101%	93650936	1495	16.05	66
dd-5	16	13	1	13	19	7	8	19	1	32	2752	102%	48775164	2460	15.50	45
dd-6	16	15	1	15	21	6	6	18	1	32	2752	102%	88145920	918	16.68	102
dd-7	16	14	1	14	19	7	8	19	1	32	2768	102%	86540280	2461	16.50	90
dd-8	16	16	1	16	21	6	5	18	1	32	2832	105%	96600064	599	17.68	205
dd-9	16	13	1	13	20	7	8	20	1	32	2864	106%	59785212	2467	16.74	106
dd-10	16	14	1	14	20	7	8	20	1	32	2880	107%	97550328	2468	17.74	213
dd-11	16	14	1	14	21	6	4	18	1	32	2912	108%	46956536	429	15.68	51
dd-12	16	15	1	15	21	6	4	18	1	32	2928	108%	56164348	430	16.68	102
dd-13	16	20	2	10	17	7	8	19	1	32	2928	108%	12193786	4758	19.21	592
dd-14	16	20	2	10	20	6	8	19	1	32	2928	108%	28315644	4758	20.10	1097
dd-15	16	16	1	16	21	6	4	18	1	32	2944	109%	74579968	431	17.68	205
dd-16	16	17	1	17	18	7	4	19	1	32	2976	110%	79167480	433	18.05	265
dd-17	16	16	1	16	19	7	4	19	1	32	3072	114%	47841276	439	18.50	362
dd-18	16	17	1	17	19	7	4	20	1	32	3088	114%	84672504	440	19.50	724
dd-19	16	16	1	16	20	7	4	20	1	32	3184	118%	58851324	446	19.74	855
dd-20	16	17	1	17	20	7	4	21	1	32	3200	119%	95682552	447	20.74	1710

Table 39: Selection Set (128, 112, 2^{10}, 100,000,000)

C.2 Security Level 3 at 2^{10} Signature Limit

For a 100,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
ee-1	24	10	2	5	9	31	4	37	3	48	10152	100%	99938	1140	14.29	19
ee-2	24	12	3	4	9	27	4	33	3	48	10464	103%	80709	1511	15.46	44
ee-3	24	12	3	4	9	28	4	34	3	48	10704	105%	82244	1521	15.69	51
ee-4	24	15	3	5	7	35	4	34	3	48	10776	106%	91933	1524	16.59	96
ee-5	24	15	3	5	7	36	4	35	3	48	10968	108%	92316	1532	16.77	109
ee-6	24	16	4	4	7	34	5	33	3	48	10968	108%	99165	2982	17.39	167
ee-7	24	16	4	4	9	24	4	30	3	48	11064	108%	89191	1894	18.59	385
ee-8	24	16	4	4	9	25	4	32	3	48	11304	111%	90726	1904	18.91	481
ee-9	24	16	4	4	9	26	4	33	3	48	11544	113%	92261	1914	19.20	588
ee-10	24	16	4	4	9	27	4	34	3	48	11784	116%	93796	1924	19.46	704
ee-11	24	20	5	4	9	22	4	28	3	48	11904	117%	99208	2287	21.81	3590

Table 40: Selection Set (192, 128, 2^{10}, 100,000)

For a 1,000,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
ff-1	24	14	2	7	13	16	6	28	3	48	7368	100%	950769	2418	18.68	410
ff-2	24	15	3	5	12	17	8	29	3	48	7560	102%	848047	10225	19.00	512
ff-3	24	18	3	6	12	16	7	27	3	48	7608	103%	934256	5991	21.39	2683
ff-4	24	9	1	9	13	19	4	33	3	48	7848	106%	885743	686	15.11	34
ff-5	24	15	3	5	12	18	8	30	3	48	7872	106%	860334	10238	19.51	729
ff-6	24	18	2	9	11	18	4	29	3	48	8088	109%	948207	1054	21.16	2288

Table 41: Selection Set (192, 128, 2^{10}, 1,000,000)

For a 10,000,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
gg-1	24	16	2	8	16	12	8	26	3	48	6552	100%	5768181	6880	21.14	2256
gg-2	24	12	1	12	16	13	4	28	3	48	6840	104%	5906421	644	18.04	263
gg-3	24	13	1	13	16	13	4	28	3	48	6864	104%	9256949	645	19.04	526
gg-4	24	12	1	12	17	13	4	30	3	48	7152	109%	8462325	657	19.09	544
gg-5	24	13	1	13	16	14	4	30	3	48	7272	110%	9453556	662	19.78	879
gg-6	24	12	1	12	17	14	4	32	3	48	7584	115%	8855540	675	19.82	903
gg-7	24	18	2	9	17	11	4	28	3	48	7656	116%	5162998	1036	23.19	9345

Table 42: Selection Set (192, 128, 2^{10}, 10,000,000)

And for a 100,000,000 hash limit, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 128	overuse safety
hh-1	24	13	1	13	20	10	8	27	3	48	6000	100%	85999608	3554	20.16	1144
hh-2	24	16	1	16	19	10	4	26	3	48	6432	107%	69337080	627	22.11	4420

Table 43: Selection Set (192, 128, 2^{10}, 100,000,000)

D Level 5 Security Parameter Sets

In our personal view, one should not go for Level 5 security parameter sets when the important goal is to get a much smaller signature size than the *bound-size* or *192-size*. However, we understand that there could be practitioners who want to use Level 5 security parameter sets. We used the retained security level of 192 bits for the parameter sets below.

D.1 Security Level 5 at 2^{10} Signature Limit

For 2^{10} signature limit and signing cost limit of 1,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
ii-1	32	12	2	6	13	23	6	40	5	64	13600	100%	934122	3218	16.08	67
ii-2	32	14	2	7	11	27	6	40	5	64	13728	100%	903654	3222	16.81	112
ii-3	32	15	3	5	11	26	8	39	5	64	13760	101%	995494	13388	17.37	165

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
ii-4	32	14	2	7	13	22	5	38	5	64	13856	101%	991723	2086	17.57	190
ii-5	32	15	3	5	12	24	7	39	5	64	14240	104%	774312	7820	18.10	274
ii-6	32	16	2	8	12	23	4	37	5	64	14400	105%	832490	1391	18.57	380
ii-7	32	18	3	6	12	23	6	38	5	64	14496	106%	835945	4642	20.57	1520
ii-8	32	16	2	8	12	24	4	38	5	64	14816	108%	844777	1404	19.10	548
ii-9	32	16	2	8	12	25	4	40	5	64	15232	112%	857064	1417	19.56	754
ii-10	32	20	5	4	12	22	8	36	5	64	15264	112%	966792	22073	21.95	3956
ii-11	32	16	2	8	12	26	4	41	5	64	15648	115%	869351	1430	19.96	995
ii-12	32	16	2	8	12	27	4	43	5	64	16064	118%	881638	1443	20.32	1278

Table 44: Selection Set (256, 192, 2^{10}, 1,000,000)

For 2^{10} signature limit and signing cost limit of 10,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
jj-1	32	18	2	9	16	16	7	36	5	64	11808	100%	8259569	5286	21.39	2683
jj-2	32	14	2	7	17	16	8	36	5	64	11872	100%	8520177	9010	18.68	410
jj-3	32	18	2	9	14	19	8	38	5	64	11904	100%	9848814	9011	21.12	2225
jj-4	32	12	1	12	15	19	4	38	5	64	12288	104%	6266863	855	16.49	89
jj-5	32	12	1	12	16	18	4	38	5	64	12352	104%	7938032	857	17.05	132
jj-6	32	13	1	13	14	21	4	39	5	64	12672	107%	9830381	867	17.40	168
jj-7	32	12	1	12	16	19	4	40	5	64	12896	109%	8134639	874	17.69	206
jj-8	32	13	1	13	14	22	4	41	5	64	13152	111%	9879532	882	17.91	240
jj-9	32	12	1	12	16	20	4	42	5	64	13440	113%	8331246	891	18.24	302
jj-10	32	18	2	9	17	15	4	36	5	64	13536	114%	6998002	1364	21.70	3326

Table 45: Selection Set (256, 192, 2^{10}, 10,000,000)

For 2^{10} signature limit and signing cost limit of 100,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
kk-1	32	14	1	14	18	15	7	36	5	64	10848	100%	93618160	2798	18.94	491
kk-2	32	11	1	11	20	14	8	37	5	64	10880	100%	61870068	4660	17.17	144
kk-3	32	14	1	14	19	14	6	36	5	64	10880	100%	69238776	1737	19.06	533
kk-4	32	12	1	12	20	14	8	37	5	64	10912	100%	79699952	4661	18.17	288
kk-5	32	15	1	15	20	13	5	35	5	64	11008	101%	98631672	1171	20.01	1031
kk-6	32	13	1	13	19	15	8	38	5	64	11136	102%	94912496	4668	19.07	537
kk-7	32	22	2	11	19	13	8	35	5	64	11232	103%	56107000	8990	25.80	57052
kk-8	32	15	1	15	20	13	4	35	5	64	11392	105%	76087288	827	20.01	1031

Table 46: Selection Set (256, 192, 2^{10}, 100,000,000)

D.2 Security Level 5 at 2^{20} Signature Limit

For 2^{20} signature limit and signing cost limit of 1,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
ll-1	32	24	4	6	11	27	6	42	5	64	16928	100%	903652	6114	26.81	112
ll-2	32	25	5	5	11	26	7	40	5	64	17056	100%	958756	12824	27.37	165

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
ll-3	32	25	5	5	11	27	7	42	5	64	17440	103%	964899	12836	27.81	224
ll-4	32	24	3	8	11	27	4	41	5	64	17600	103%	990693	1961	26.81	112
ll-5	32	25	5	5	11	28	7	43	5	64	17824	105%	971042	12848	28.19	292
ll-6	32	28	4	7	10	28	5	39	5	64	17824	105%	988131	3862	29.35	652
ll-7	32	24	3	8	11	28	4	42	5	64	17984	106%	996836	1973	27.19	146
ll-8	32	30	5	6	9	31	6	39	5	64	18112	106%	969823	7547	30.38	1332
ll-9	32	28	4	7	12	23	4	39	5	64	19072	112%	832488	2477	30.57	1520

Table 47: Selection Set (256, 192, 2^{20}, 1,000,000)

For 2^{20} signature limit and signing cost limit of 10,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
mm-1	32	24	3	8	14	20	8	38	5	64	13664	100%	7669228	13385	27.81	224
mm-2	32	27	3	9	14	19	7	39	5	64	13760	100%	8604653	7805	30.12	1112
mm-3	32	24	3	8	15	19	8	39	5	64	13792	100%	8553965	13389	28.49	359
mm-4	32	28	4	7	16	16	8	36	5	64	13984	102%	7603183	17714	31.39	2683
mm-5	32	24	2	12	14	20	4	39	5	64	14688	107%	9781229	1400	27.81	224
mm-6	32	24	2	12	14	21	4	41	5	64	15168	111%	9830380	1415	28.40	337
mm-7	32	24	2	12	14	22	4	43	5	64	15648	114%	9879531	1430	28.91	481
mm-8	32	30	3	10	17	15	4	37	5	64	16064	117%	9197553	1913	33.70	13307

Table 48: Selection Set (256, 192, 2^{20}, 10,000,000)

For 2^{20} signature limit and signing cost limit of 100,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
nn-1	32	24	2	12	19	14	8	38	5	64	11936	100%	93339632	9012	29.06	533
nn-2	32	26	2	13	20	13	6	37	5	64	12480	104%	88113144	3183	31.01	2062
nn-3	32	24	2	12	19	15	8	40	5	64	12576	105%	94912496	9032	30.07	1074
nn-4	32	30	2	15	18	14	4	36	5	64	13792	115%	81395696	1372	33.84	14664

Table 49: Selection Set (256, 192, 2^{20}, 100,000,000)

[7] had a recommended parameter set **Sphincs$^+$-256s-q20** which has the signature size of 12,640 bytes. We wanted to know if there were more attractive parameter sets in some aspects so we increased the signing time bound to 1,000,000,000 hashes and we found the parameter sets below.

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
xx-1	32	28	2	14	21	12	8	36	5	64	11552	100%	360775680	9000	32.76	6936
xx-2	32	19	1	19	21	14	4	40	5	64	12640	109%	651165696	866	26.23	75
xx-3	32	19	1	19	23	13	4	41	5	64	12768	110%	890241024	870	27.20	147
xx-4	32	19	1	19	22	14	4	42	5	64	13088	113%	739246080	880	27.26	153
xx-5	32	18	1	18	23	14	4	44	5	64	13504	116%	633864192	893	27.27	154
xx-6	32	19	1	19	23	14	4	44	5	64	13536	117%	915406848	894	28.27	308
xx-7	32	36	2	18	22	11	4	37	5	64	13568	117%	701497344	1365	40.25	1246974

Table 50: Selection Set (256, 192, 2^{20}, 1,000,000,000)

The parameter set **Sphincs$^+$-256s-q20** is xx-2. xx-1 has smaller signature size and sign time, and bigger *overuse safety* than xx-2.

D.3 Security Level 5 at 2^{30} Signature Limit

For 2^{30} signature limit and signing cost limit of 1,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
oo-1	32	35	7	5	11	26	6	41	5	64	21216	100%	805282	10436	37.37	165
oo-2	32	35	7	5	12	24	6	41	5	64	21216	100%	940452	10436	38.10	274
oo-3	32	36	6	6	12	23	5	40	5	64	21312	100%	959206	5623	38.57	380
oo-4	32	36	6	6	12	24	5	41	5	64	21728	102%	971493	5636	39.10	548
oo-5	32	35	5	7	11	26	4	41	5	64	21856	103%	847076	3034	37.37	165
oo-6	32	35	5	7	12	24	4	41	5	64	21856	103%	982246	3034	38.10	274
oo-7	32	40	8	5	11	24	6	39	5	64	22048	103%	885219	11858	41.29	2503
oo-8	32	35	5	7	12	25	4	43	5	64	22272	104%	994533	3047	38.56	377
oo-9	32	35	5	7	11	30	4	47	5	64	23392	110%	871648	3082	38.85	461
oo-10	32	42	6	7	10	26	4	39	5	64	23392	110%	904675	3552	42.27	4938

Table 51: Selection Set (256, 192, 2^{30}, 1,000,000)

For 2^{30} signature limit and signing cost limit of 10,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
pp-1	32	32	4	8	14	21	8	41	5	64	15488	100%	9947114	17761	36.40	84
pp-2	32	32	4	8	15	19	7	40	5	64	15776	101%	6981612	10326	36.49	89
pp-3	32	35	5	7	15	18	8	39	5	64	15808	102%	7341292	22090	38.78	439
pp-4	32	35	5	7	16	17	8	39	5	64	15840	102%	8914157	22091	39.29	625
pp-5	32	36	4	9	15	18	6	40	5	64	16160	104%	7671789	6090	39.78	879
pp-6	32	36	4	9	16	17	6	40	5	64	16192	104%	9244654	6091	40.29	1251
pp-7	32	35	5	7	15	19	8	41	5	64	16320	105%	7439595	22106	39.49	719
pp-8	32	35	5	7	16	18	8	41	5	64	16384	105%	9110764	22108	40.05	1060
pp-9	32	33	3	11	15	19	4	41	5	64	17248	111%	8466413	1950	37.49	179
pp-10	32	33	3	11	15	20	4	43	5	64	17760	114%	8564716	1966	38.09	272
pp-11	32	33	3	11	15	21	4	45	5	64	18272	117%	8663019	1982	38.62	393
pp-12	32	44	4	11	13	19	4	38	5	64	18528	119%	9265132	2460	45.43	44146

Table 52: Selection Set (256, 192, 2^{30}, 10,000,000)

For 2^{30} signature limit and signing cost limit of 100,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
qq-1	32	33	3	11	20	14	8	40	5	64	13760	100%	97529840	13388	39.17	576
qq-2	32	36	3	12	19	14	7	39	5	64	13888	100%	83386360	7809	41.06	2134
qq-3	32	44	4	11	19	13	8	38	5	64	14112	102%	91766768	17718	47.80	228209
qq-4	32	30	2	15	19	16	4	42	5	64	15520	112%	95551472	1426	36.92	121
qq-5	32	42	3	14	19	13	4	37	5	64	16128	117%	73236472	1915	45.80	57052

Table 53: Selection Set (256, 192, 2^{30}, 100,000,000)

D.4 Security Level 5 at 2^{40} Signature Limit

For 2^{40} signature limit and signing cost limit of 1,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
rr-1	32	45	9	5	11	26	6	42	5	64	24416	100%	989728	13328	47.37	165
rr-2	32	45	9	5	11	27	6	44	5	64	24800	101%	995871	13340	47.81	224
rr-3	32	48	8	6	10	28	5	42	5	64	25504	104%	988127	7406	49.35	652
rr-4	32	42	6	7	10	33	4	48	5	64	25856	105%	926172	3629	45.12	34
rr-5	32	48	8	6	10	29	5	44	5	64	25856	105%	991198	7417	49.79	885
rr-6	32	50	10	5	9	31	6	42	5	64	25952	106%	969818	14772	50.38	1332
rr-7	32	42	6	7	10	34	4	49	5	64	26208	107%	929243	3640	45.38	41
rr-8	32	42	6	7	10	35	4	50	5	64	26560	108%	932314	3651	45.61	48
rr-9	32	42	6	7	10	36	4	51	5	64	26912	110%	935385	3662	45.83	56
rr-10	32	42	7	6	12	26	4	45	5	64	27200	111%	800610	4141	45.96	62
rr-11	32	42	6	7	10	37	4	53	5	64	27264	111%	938456	3673	46.03	65
rr-12	32	49	7	7	8	37	4	44	5	64	27264	111%	990679	4143	49.00	512
rr-13	32	49	7	7	8	38	4	45	5	64	27552	112%	991446	4152	49.31	634
rr-14	32	49	7	7	8	39	4	46	5	64	27840	114%	992213	4161	49.60	776
rr-15	32	49	7	7	8	40	4	47	5	64	28128	115%	992980	4170	49.87	935
rr-16	32	48	8	6	12	23	4	42	5	64	28288	115%	832484	4645	50.57	1520

Table 54: Selection Set (256, 192, 2^{40}, 1,000,000)

For 2^{40} signature limit and signing cost limit of 10,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
ss-1	32	42	6	7	15	19	8	42	5	64	17632	100%	8553962	26466	46.49	89
ss-2	32	45	5	9	15	18	6	41	5	64	17888	101%	9147372	7540	48.78	439
ss-3	32	49	7	7	15	17	8	39	5	64	17920	101%	9471723	30794	51.92	3875
ss-4	32	44	4	11	14	20	4	42	5	64	19616	111%	9781227	2494	47.81	224
ss-5	32	44	4	11	14	21	4	44	5	64	20096	113%	9830378	2509	48.40	337
ss-6	32	44	4	11	14	22	4	46	5	64	20576	116%	9879529	2524	48.91	481
ss-7	32	50	5	10	15	17	4	39	5	64	21056	119%	7170029	3009	52.92	7750

Table 55: Selection Set (256, 192, 2^{40}, 10,000,000)

For 2^{40} signature limit and signing cost limit of 100,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
tt-1	32	44	4	11	19	14	8	41	5	64	14752	100%	93339632	17738	49.06	533
tt-2	32	48	4	12	17	15	7	39	5	64	15200	103%	87719920	10308	51.70	3326
tt-3	32	44	4	11	19	15	8	43	5	64	15392	104%	94912496	17758	50.07	1074
tt-4	32	42	3	14	20	14	4	41	5	64	17216	116%	96829424	1949	48.17	288
tt-5	32	42	3	14	20	15	4	44	5	64	17888	121%	99975152	1970	49.13	560
tt-6	32	52	4	13	19	13	4	38	5	64	18592	126%	55640052	2462	55.80	57052

Table 56: Selection Set (256, 192, 2^{40}, 100,000,000)

D.5 Security Level 5 at 2^{50} Signature Limit

For 2^{50} signature limit and signing cost limit of 1,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
uu-1	32	50	10	5	9	43	6	56	5	64	29792	100%	988238	14892	53.50	11
uu-2	32	50	10	5	9	44	6	57	5	64	30112	101%	989773	14902	53.65	12
uu-3	32	50	10	5	9	45	6	58	5	64	30432	102%	991308	14912	53.79	13
uu-4	32	50	10	5	9	46	6	59	5	64	30752	103%	992843	14922	53.93	15
uu-5	32	50	10	5	12	28	5	49	5	64	30880	103%	907869	9226	54.65	25
uu-6	32	54	9	6	12	24	4	43	5	64	31040	104%	913506	5201	57.10	137
uu-7	32	55	11	5	11	26	5	44	5	64	31136	104%	779934	10060	57.37	165
uu-8	32	55	11	5	12	24	5	44	5	64	31136	104%	915104	10060	58.10	274
uu-9	32	54	9	6	12	25	4	45	5	64	31456	105%	925793	5214	57.56	188
uu-10	32	55	11	5	12	25	5	46	5	64	31552	105%	927391	10073	58.56	377
uu-11	32	56	14	4	12	23	6	43	5	64	31552	105%	928158	20531	58.57	380
uu-12	32	54	9	6	12	26	4	46	5	64	31872	106%	938080	5227	57.96	248
uu-13	32	55	11	5	12	26	5	47	5	64	31968	107%	939678	10086	58.96	497
uu-14	32	56	14	4	12	24	6	44	5	64	31968	107%	940445	20544	59.10	548
uu-15	32	60	12	5	12	22	5	41	5	64	32224	108%	946913	10920	61.95	3956
uu-16	32	54	9	6	12	27	4	48	5	64	32288	108%	950367	5240	58.32	319
uu-17	32	60	10	6	12	22	4	41	5	64	32544	109%	957667	5718	61.95	3956

Table 57: Selection Set (256, 192, 2^{50}, 1,000,000)

For 2^{50} signature limit and signing cost limit of 10,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
vv-1	32	56	8	7	13	21	8	43	5	64	19936	100%	9431014	35176	58.98	504
vv-2	32	56	8	7	14	20	8	43	5	64	20128	100%	9897959	35182	59.81	897
vv-3	32	56	8	7	14	21	8	45	5	64	20608	103%	9947110	35197	60.40	1351
vv-4	32	50	5	10	16	19	4	45	5	64	22688	113%	9234411	3060	55.69	51
vv-5	32	50	5	10	16	20	4	47	5	64	23232	116%	9431018	3077	56.24	75
vv-6	32	60	6	10	15	17	4	41	5	64	23520	117%	8269804	3556	62.92	7750

Table 58: Selection Set (256, 192, 2^{50}, 10,000,000)

For 2^{50} signature limit and signing cost limit of 100,000,000 hashes, we get the following:

ID	n	h	d	h'	a	k	lg_w	m	sec cat.	pk bytes	sig bytes	% size	sign time	verify time	sigs at level 192	overuse safety
ww-1	32	55	5	11	15	18	8	42	5	64	16448	100%	90918896	22110	58.78	439
ww-2	32	55	5	11	17	16	8	42	5	64	16448	100%	95440880	22110	59.68	820
ww-3	32	55	5	11	20	13	7	41	5	64	16768	101%	92033008	12815	60.01	1031
ww-4	32	60	6	10	18	14	8	41	5	64	16992	103%	64499696	26446	63.84	14664
ww-5	32	56	4	14	19	14	4	42	5	64	19360	117%	92405744	2486	61.06	2134

Table 59: Selection Set (256, 192, 2^{50}, 100,000,000)