Constant-time verification for cut-and-choose-based signatures

Robert Ransom
rransom.8774@gmail.com

September 27, 2020

Abstract

In most post-quantum signature protocols, the verification procedure leaks information about which signature is being verified, and/or which public key is being used to verify the signature, to timing and other side-channel attacks. In some applications, this information leak is a breach of user privacy or system security.

One class of signature protocols, based on the parallel composition of many runs of one or more interactive cut-and-choose protocols, can be modified to enable constant-time verification at low cost by fixing the multiset of challenges which will be chosen at the cut-and-choose step and randomizing only their order based on the hash of the input message. As a side benefit, this technique naturally makes the size and structure of signatures a fixed system parameter, even if the underlying cut-and-choose protocol has different response sizes for each possible challenge at the cut-and-choose step.

When applied to a 5-pass "$q2$" interactive protocol, this technique requires essentially no extra rounds due to how fixed-weight binary vectors interact with the Kales–Zaverucha structural attack. Alternatively, when the data which must be transmitted for one of the two possible challenge values is significantly shorter than the other, or can be made so using standard and/or specialized compression techniques, a longer, lower-weight challenge vector can be used to obtain shorter signatures at the cost of more rounds of the underlying interactive protocol, with a much shallower computation-vs.-size tradeoff than the precomputation tree approach used in Picnic2, MUDFISH, and SUSHSYFISH.

As an example, these techniques reduce MQDSS signatures to under 15 kB and PKP-DSS signatures to under 14 kB with NIST Category 1 security against both secret key recovery and signature forgery. Further improvements in design and parameters allow PKP-DSS signatures under 10 kB with a security level and performance acceptable for almost all interactive authentication.

The asymptotic ROM proof of security published with MQDSS remains applicable to the optimized system, but the QROM proofs by Don et al. turn out to be invalid even for unmodified MQDSS.

1 Introduction

Most of the public-key cryptographic protocols in use today rely, directly or indirectly, on the difficulty of computing discrete logarithms in cyclic groups. If a device capable of solving the discrete logarithm problem, such as a quantum computer capable of performing Shor's algorithm on an input of non-trivial size, is built, then all of these cryptographic protocols will be broken. Post-quantum cryptography aims to develop public-key protocols which would not be broken by such an advance.

Symmetric cryptography, key generation, and public-key encryption, decryption, encapsulation, and decapsulation are generally expected to resist leakage of the secret information they process to side-channel attacks. Specifically, any information leaked to an attacker must be independent of the secret key and any plaintext or MAC being processed. For software implementations, the overall time is almost always leaked; in many cases (e.g. virtualized server hosting), the memory access pattern and execution path are leaked to an attacker as well, even if they do not affect the overall time enough to be observed over the network.

For signature schemes, the focus for timing and other side-channel attacks and defenses has been on the signing procedure, because that operation uses a secret key. Signature verification involves only one or more public keys and signatures; since public keys have "public" in their name and signatures are not obviously secret, designers and implementors generally assume that it is acceptable to disclose the inputs to verification to an attacker. This assumption is not always justified.

As a general example, in any privacy-enhancing technology system which attempts to conceal the signer of a message, or which of a set of known signed messages is being verified, from parties which can observe its delivery across a network to the recipient, signature verification obviously must run in time independent of its inputs.

More commonly used systems may also require certificates and signatures that can be parsed and verified in constant time. For example, the SSH protocol places the client authentication protocol, including the client's username, within an encrypted connection, and also allows the use of certificates for client authentication. SSH explicitly aims to protect the user's identity from eavesdropping. Corporate or government VPN servers which must use client certificates for authentication are similar to SSH.

The Encrypted Server Name Indication extension deployed for TLS has similar secrecy goals for the server's hostname and certificate; it explicitly intends to conceal which of several hostnames the client is attempting to access from network eavesdroppers. The signature contained in a certificate is intended for long-term use, and leaking information about a long-term signature at least partially defeats the surrounding protocol's secrecy objectives.

Where client identity information is made available to a network observer, this can leak not only the identities and locations of specific users, but also information such as the policy for certificate lifetime, and how many different security compartments each user has certificates to obtain access to. These facts may aid an outsider in gaining access or targeting attacks against personnel.

It should be noted that the protocol verification systems used to prove network protocols such as TLS and SSH "secure" do not account for the fact that verifying a signature leaks a moderate-entropy identifier of the signature to the attacker. Since signature protocols which do have this leakage are likely to be put into use, verification models must be updated to account for signature disclosure where it is not explicitly prevented.

For most signature schemes currently in use, the timing leakage from verification is small, generally within the level of timing variation that unscrupulous implementors have tried to pass off as "constant-time", and similar to or less than the timing variation caused by other parts of the surrounding system. Even in the contexts where this timing leakage could be a problem, the attacker usually cannot obtain enough samples to be worthwhile as an attack. However, most signature schemes based on cut-and-choose protocols have much larger variation in verification time, and are likely to leak a nearly unique identifier of the signature to an attacker with access to timing information alone from a single verification operation. The cut-and-choose-based signature protocols which do not have large variation in verification time do not consider constant-time verification to be a design goal, and thus do not have truly constant-time implementations.

Fortunately, the most significant change needed to enable constant-time verification of signatures based on cut-and-choose protocols naturally enables a large decrease in signature size, which is the main performance concern for those signature protocols.

2 Background

This section summarizes the standard definitions and some of the relevant history of related prior work.

The definitions below focus on the core intuitive part of each concept, and are stated in informal terms such as "efficient" algorithms applied to any input, rather than the usual approach of adding formalistic details such as "key relations" or the class of algorithm that produced an input, then giving informal proofs. The reason for this is that none of those details have resulted in serious errors in proofs. The cause of broken proofs is a failure to understand the intuitive core of a concept correctly.

In addition, the explicit statement of a "set of guesses" and a success predicate for a 3-pass protocol appears to be new material, and this concept is needed in section 3.2 below to evaluate the security level of protocols with fixed-weight challenge vectors at the cut-and-choose step.

2.1 Identification protocols

In an identification protocol, the prover knows a secret key, the verifier knows the corresponding public key, and the prover wishes to prove knowledge of the secret key to the verifier without disclosing any information about it beyond the public key. It is always assumed that computing the secret key from the public key is infeasible.

This paper will only consider identification protocols which are "public coin": roughly, the verifier's messages are sampled directly from the uniform distribution on some set of possible challenges. In a public coin protocol, the prover may be assumed to send the first message.

Most post-quantum identification protocols are based on the "cut-and-choose" principle: The prover generates a randomly blinded instance of the hard problem relating its secret key to its public key, cuts the solution to the blinded instance and its relation to the original problem into a few pieces, and commits to each piece; then the verifier chooses which piece or pieces it wants the prover to open. The amount of data sent in the response may vary depending on the verifier's challenge; the computations to verify the response usually vary. Additionally, the number of possible challenges is small, so to achieve a useful level of security, the prover and verifier must perform the protocol many times. This paper will focus on cut-and-choose identification protocols and the theoretical and practical techniques used with them.

The security properties expected of an identification protocol are defined in terms of the set and distribution of "transcripts" of protocol runs. In an identification protocol with $2n+1$ communication passes, the transcript is of the form $(c_1, h_1, \ldots, c_n, h_n, r)$, where $c_i \in \mathcal{C}_i$ and $r \in \mathcal{R}$ are sent by the prover and $h_i \in \mathcal{H}_i$ are sent by the verifier. The messages c_i serve as commitments; the h_i are challenges; and r is the final response. If the verifier participating in a transcript would have accepted the protocol run as valid, the transcript is said to be successful.

(Note that this follows the presentation in [DFM20, §5] and differs from the standard presentation of the 5-pass protocol structure, in which the second commitment step is instead described as an extra response.)

Identification protocols are traditionally proved to have three properties:

- "*Perfect correctness*": When the prover follows the protocol honestly, an honest verifier will always accept the interaction as valid.
- "*Honest-verifier zero knowledge*", or "*HVZK*": There exists an efficient "simulator" algorithm which generates fake transcripts, without knowledge of the secret key, that are indistinguishable from transcripts of a valid interaction between an honest prover and an honest verifier.
- "*Soundness*": A cheating prover which does not know the secret key cannot cause an honest verifier to accept an interaction with it as valid with probability greater than some $\varepsilon < 1$. This probability is called the "soundness error".

There are minor variations in the details and the names used, but the concepts above are essentially the standard ones.

When an identification protocol is to be used as the basis for another protocol, such as a signature scheme, any proof of security for the surrounding protocol requires the identification protocol to have another property, some form of "*special soundness*": There exists an efficient "extractor" algorithm which can extract either the secret key or a break of one of the identification protocol's other underlying security assumptions from any set of successful transcripts satisfying some condition. Exactly what that condition is depends on what kind of special soundness is being proved.

Every correct proof of soundness for a cut-and-choose identification protocol, with soundness error ε, has three steps:

- First, the proof repeatedly rewinds a hypothetical cheating prover algorithm to the point before it receives each challenge, to obtain one transcript for each of the $\#(\mathcal{H}_1 \times \cdots \times \mathcal{H}_n)$ possible sequences of challenges. The set of these transcripts and their prefixes forms a tree; every transcript shares the same initial commitment c_1, and each subsequent commitment is a function of the sequence of challenges preceding it.

  Let N denote the size of this set of transcripts. Not all of these transcripts are valid; the cheating prover algorithm is permitted to fail for some sequences of challenges.

- Then, to obtain a contradiction, the proof assumes that the cheating prover algorithm succeeds with probability greater than ε. The pigeonhole principle is used to show that, if more than εN out of the set of N transcripts collected in the first step are valid, then there is a subset of valid transcripts having some pattern of challenges and commitments required for the next step.

- Finally, the protocol is proved to have some form of special soundness, and that the valid transcripts collected are sufficient to recover the secret key or a break of some other security property (usually the binding property of a generic string commitment protocol).

  The statement of the theorem then discards every piece of information contained in the proof except the soundness error ε. Anyone who wants to write a proof of security relying on the protocol's special soundness is left to either repeat the relevant part of the proof, or assert the required form of special soundness and handwave away its proof.

Every major proof error that I have encountered in writing this paper is a result, either directly or indirectly, of the tradition of proving identification protocols to be sound with some numerical soundness error, rather than explicitly stating the form of special soundness which they provide.

For 3-pass identification protocols, where transcripts are of the form $(c, h, r) \in \mathcal{C} \times \mathcal{H} \times \mathcal{R}$, proofs of the HVZK property also contain more information than the standard definition would suggest:

- The HVZK simulator can be split into two algorithms, one which generates the commitment c and some state information st, and one which takes (h, st) as input and computes the appropriate r, if possible. The second algorithm is deterministic.
- There exists a set $\mathcal{G}$ of possible guesses, and the first algorithm in the HVZK simulator takes $g \in \mathcal{G}$ as input.
- There exists a predicate $S\colon \mathcal{G} \times \mathcal{H} \to \mathbf{Bool}$ indicating whether a challenge $h \in \mathcal{H}$ can be successfully answered by a simulator state (c, st) with a given guess $g \in \mathcal{G}$. Whenever $S(g, h)$ is true, the second algorithm in the simulator outputs an r for which (c, h, r) is a successful transcript.

  (This is not quite an equivalence; the simulator may be able to compute a successful transcript even if $\neg S(g, h)$, by accidentally guessing the secret key. However, this happens with negligible probability, and most "proofs" completely ignore low-probability events of this sort.)
- It is traditional to show that the soundness error ε proved for any identification protocol is "tight", i.e. that the HVZK simulator can succeed with probability ε. For 3-pass protocols, the special soundness is also tight with respect to this notion of guesses: The extractor algorithm can extract a break of the protocol from a set of successful transcripts $\{(c, h_i, r_i)\}_{i \in I}$ (indexed for convenience) with the same commitment value c whenever there is no guess g such that $\{h_i | i \in I\} \subseteq \{h | S(g, h)\}$.

Some examples of 3-pass interactive cut-and-choose protocols:

- the Fiat–Shamir identification protocol [FS87, §2], in which $\mathcal{H} = \{0, 1\}^k$ for some k, $\mathcal{G} = \mathcal{H}$, and $S(g, h)$ if and only if $g = h$. (This protocol assumes that integer factorization is hard.)
- Blum's proof of knowledge of a Hamiltonian cycle [Blu87], in which $\mathcal{H} = \{\text{iso}, \text{cycle}\}$, $\mathcal{G} = \mathcal{H}$, and $S(g, h)$ if and only if $g = h$.
- Stern's proof of knowledge of a fixed-weight codeword [Ste94b, §1][Ste96a, §2.1][Ste96b, fig. 1], in which $\mathcal{H} = \{0, 1, 2\}$, $\mathcal{G} = \{0, 1, 2\}$, and $S(g, h)$ if and only if $g \neq h$.
- Stern's 3-pass proof of knowledge of a solution to a constrained linear equations problem instance [Ste94a, §3][Ste96c, fig. 1], in which $\mathcal{H} = \{0, 1, 2\}$, $\mathcal{G} = \{0, 1, 2\}$, and $S(g, h)$ if and only if $g \neq h$.
- Pointcheval's 3-pass HVZK proof of knowledge of a solution to a permuted perceptrons problem instance [Poi95, §7.1], in which $\mathcal{H} = \{0, 1, 2, 3\}$, $\mathcal{G} = \{0, 1, 2, 3\}$, and $S(g, h)$ if and only if $g \neq h$.

- Sakumoto–Shirai–Hiwatari 3-pass proof of knowledge of a solution to a multivariate quadratic problem instance [SSH11, §3][SSH13, fig. 4][SSH15, fig. 4], in which $\mathcal{H} = \{0, 1, 2\}$, $\mathcal{G} = \{0, 1, 2\}$, and $S(g, h)$ if and only if $g \neq h$.

To achieve a useful level of security, these protocols are run many times; signature schemes require that the repetition be in parallel. The set of guesses for the parallel composition of r instances of an identification protocol is $\mathcal{G}_{\text{rep}} := \prod_{i=1}^{r} \mathcal{G}$, and $S_{\text{rep}}(\mathbf{g}, \mathbf{h})$ if and only if $S(g_i, h_i)$ for every i, regardless of how the vector of challenges is sampled. When each challenge in the vector is sampled independently, the soundness error for the parallel composition is given by the standard formula $\varepsilon_{\text{rep}} = \varepsilon^r$.

2.2 5-pass identification protocols

The canonical 5-pass interactive identification protocol has two commitment steps and two challenge steps; transcripts are of the form $(c_1, h_1, c_2, h_2, r) \in \mathcal{C}_1 \times \mathcal{H}_1 \times \mathcal{C}_2 \times \mathcal{H}_2 \times \mathcal{R}$.

For most 5-pass protocols, a fake prover can choose c_1 such that successful transcripts for any $(h_1, h_2) \in \mathcal{H}_1 \times \mathcal{H}_2$ can be generated without the use of the secret key. (If this is not the case, then the second commitment pass c_2 can be omitted.) The constraint which makes these protocols useful is that for any fixed choice of c_1, there is only a small set of possible values of h_1 for which the fake prover can generate one c_2 for which all values of h_2 can be successfully answered.

Unfortunately, this means that the HVZK and special soundness properties in the 5-pass case are not easily expressed in terms of a set of guesses as they are in the 3-pass case. However, the concept of a set of guesses remains useful in evaluating the security of signature schemes based on these protocols, as discussed in section 3.2 below.

The most useful 5-pass protocols are "$q2$ *protocols*", defined in [CHR$^+$16, §4.2, Definition 4.5]: they have $\#(\mathcal{H}_1) = q$ for some q and $\#(\mathcal{H}_2) = 2$. For simplicity, this paper will assume $\mathcal{H}_2 = \{0, 1\}$.

The most useful type of special soundness for 5-pass protocols is the existence of a "$q2$-*extractor*". Three definitions of $q2$-extractor have appeared in the literature, two of which do not correctly express the intended concept. This paper will use a simplified statement of the correct definition stated in [BFK$^+$19a, §2.3, Definition 6]: a $q2$-extractor is a fast algorithm which recovers either the secret key or a break of the binding property of a commitment protocol from any set of four transcripts of the form

$$\begin{aligned}\{(c_1, h_1^{(1)}, c_2^{(1)}, 0, r^{(1a)}),\\ (c_1, h_1^{(1)}, c_2^{(1)}, 1, r^{(1b)}),\\ (c_1, h_1^{(2)}, c_2^{(2)}, 0, r^{(2a)}),\\ (c_1, h_1^{(2)}, c_2^{(2)}, 1, r^{(2b)})\}\end{aligned}$$

with $h_1^{(1)} \neq h_1^{(2)}$. The other definitions will be discussed in section 4.1 below.

Some examples of 5-pass interactive cut-and-choose protocols:

- Shamir's proof of knowledge of a solution to a permuted kernel problem instance [Sha90a][Sha90b], a $q2$ protocol where q is the number of elements of a finite ring R. Has a $q2$-extractor if and only if R is a field.
- Stern's proof of knowledge of a simplex code of dimension v [Ste94b, §5][Ste96a, §4.2][Ste96b, fig. 2], a $q2$ protocol where $q = 2^v$.
- Stern's 5-pass proof of knowledge of a solution to a constrained linear equations problem instance [Ste94a, §4][Ste96c, fig. 2], a $q2$ protocol where q is the number of elements of a finite field.
- Pointcheval's 5-pass HVZK proof of knowledge of a solution to a permuted perceptrons problem instance [Poi95, §7.3], not a $q2$ protocol.
- Pointcheval's 5-pass "light" proof of knowledge of a solution to a permuted perceptrons problem instance [Poi95, §7.5], a $q2$ protocol where q is the number of non-zero elements of a finite field. No HVZK simulator known.
- Sakumoto–Shirai–Hiwatari 5-pass proof of knowledge of a solution to a multivariate quadratic problem instance [SSH11, §4][SSH13, fig. 12][SSH15, fig. 12], a $q2$ protocol where q is the number of elements of a finite field.

2.3 The Fiat–Shamir transform

One of the defining properties of a zero-knowledge interactive protocol is that, if the verifier is honest in choosing the challenges independent of the prover's commitments, then nothing at all is proved to any party which did not participate in the protocol. Most currently deployed protocols instead require a non-interactive proof of knowledge and/or signature protocol.

The Fiat–Shamir transform [FS87, §3][SF88, columns 5–6] converts an HVZK interactive proof protocol into a non-interactive proof protocol, by having the prover simulate a dishonest verifier which computes each challenge as a hash of the preceding commitments and challenges and an optional message. It was initially proposed for use with the 3-pass Fiat–Shamir identification protocol, which allows a single round of the basic protocol to have as high a security level as desired.

Interactive cut-and-choose protocols, on the other hand, must be run multiple times to reach a useful security level. In an interactive use, this can be done either serially or in parallel. In a signature protocol, serial composition does not increase security, as a forger can tweak each commitment one at a time until its corresponding challenge can be answered. Thus, signatures based on cut-and-choose protocols always use the parallel composition.

The original publications of the Fiat–Shamir transform, which proposed its use with an identification scheme for which $\mathcal{G} = \mathcal{H}$ and S is the equality predicate, stated that a forger can expect to succeed in $\#(\mathcal{H})$ operations, and this is consistent with experience. More generally, if $S(g, h)$ occurs with probability p when h is sampled from the distribution induced by the hash function, then forgery is as difficult as preimage search on an input set of $1/p$ elements.

The Fiat–Shamir transform for 3-pass protocols was later proved secure in the Random Oracle Model using the "forking lemma" [PS00]; unfortunately, this strategy results in proofs which are too loose to be quantitatively useful for the range of parameters which provides good security in practice. Thus, even though proofs of security are expected for this class of construction, they are generally disregarded for the purpose of choosing parameters to be used in practice.

Shamir mentioned in his original PKP publications [Sha90a][Sha90b] that the 5-pass PKP-based identification protocol could be turned into a non-interactive signature scheme using the Fiat–Shamir transform. "However, PKP-based signatures are much longer than Fiat–Shamir signatures, and their practical significance is unclear." At the time, high-end telephone modems ran at 9600 baud, and Shor's algorithm had not been published. The patents covering the PKP-based and related post-quantum identification protocols ([Sha90b][Ste96b][Ste96c]) expired much later than those covering Diffie–Hellman, RSA, and Schnorr signatures; when those protocols did become available for use, their advantages over RSA and ECC were not significant enough compared to their disadvantages for users to consider switching to them.

The first serious proposal that 5-pass identification protocols should actually be used as the basis for signatures was the submission of MQDSS [CHR+], based on the SSH 5-pass MQ-based protocol [SSH11, §4][SSH13, fig. 12][SSH15, fig. 12], to NIST's Post-Quantum Cryptography project. The MQDSS authors published a proof of security in the ROM for the Fiat–Shamir transform for 5-pass $q2$ protocols; as is usual, it was too loose to be quantitatively useful, so the number of parallel runs of the identification protocol used in MQDSS (in version 1.1) was chosen such that the identification protocol would provide the intended security level, as would be appropriate for a signature scheme based on a 3-pass protocol.

Two years later, Kales and Zaverucha, two of the submitters of a competing pair of signature schemes, discovered that the same type of iterative guessing attack which precludes the use of the Fiat–Shamir transform on the serial composition of interactive protocols can be applied to a portion of the first challenge vector in a signature scheme based on the parallel composition of 5-pass identification protocols [KZ19][KZ20]. Their attack spends about half of the time available to it guessing the first challenge for as many parallel runs as possible, then spends the rest of its time guessing the second challenge for all other parallel runs. This attack is what determines the minimum number of parallel runs to achieve a target security level in a signature system based on a 5-pass protocol.

Quantitatively, for a $q2$ protocol with each challenge sampled independently from the uniform distribution, the probability of guessing the first challenge in at least N out of r parallel runs is $\sum_{i=N}^{r} \left(\frac{1}{q}\right)^{-i} \left(\frac{q-1}{q}\right)^{r-i} \binom{r}{i}$, and the probability of guessing the second challenge in all of the remaining runs is $2^{-(r-N)}$. Each of these tasks is roughly a preimage search problem, and the attack scales accordingly. When the probability of success at each of these steps is so small that it is not expected to be feasible (given the target security level), the amount of work to achieve an event of probability p can be approximated as $1/p$; this gives the condition specified in [CHR+20, §8] subject to which r is minimized. (For lower signature security levels, where it is expected that an attacker could come close to performing enough operations to forge a signature within the time period that it would be accepted, the probability of success as a function of the number of hash compression function evaluations should be considered.)

2.4 Choosing an identification protocol

MQDSS [CHR+16][CHR+] is based on the 5-pass identification protocol published in 2011 by Sakumoto, Shirai, and Hiwatari [SSH11, §4]. They claimed two main practical advantages for their MQ-based identification protocols over the prior art [SSH11, §5.2]:

- Their protocols do not require secret random permutations.

  At the time, applying secret random permutations on most mainstream processors without leaking information to side-channel attacks would have had a high performance or software development cost. However, the Classic McEliece [BCL+19] and NTRU Prime [BCLvV18][BCLvV19a] submissions to the NIST PQC project included a reasonably fast constant-time sorting routine, and used it to sample and apply secret random permutations. This sorting routine has been extracted into a library [Ber] and extensively optimized for AMD64 processors, and theoretical bounds on the divergence from uniformity of the distributions from which these submissions sample fixed-weight vectors and permutations have been published [Ber18].

  As a result of this work, the need for secret random permutations is no longer a serious obstacle, and indeed if signature verification is to be performed in constant time, the sorting routine is required regardless of the underlying identification protocol.

- Their protocols have lower communication cost than any of the prior HVZK identification protocols which they evaluated, including slightly lower communication than Shamir's PKP-based protocol [Sha90a][Sha90b], when used for interactive identification.

  However, the difficulty of the MQ problem decreases significantly as field size increases, while PKP can easily reach 128-bit or higher security levels over fields of size 251 and larger. When a $q2$ protocol is to be converted to a signature scheme, the Kales–Zaverucha attack [KZ19][KZ20] makes increasing the field size q more important to limiting signature size than decreasing the communication cost per identification protocol run.

The main technical disadvantage of the Sakumoto–Shirai–Hiwatari MQ-based protocols is that they require a much larger system parameter, and thus have a much greater computation cost than the older protocols. As a result, the existing PKP-DSS software is faster than MQDSS when a vector unit is not available [BFK$^+$19b, §5.5][Beu19b], in addition to achieving shorter signatures.

In addition, the Sakumoto–Shirai–Hiwatari identification protocols are covered by active U.S. patents [SSH13][SSH15], and the patent holder intends to collect royalties on the use of MQDSS [SRC$^+$17]. Even if MQDSS were competitive with PKP-based signatures on technical grounds, these patents would prevent the general public from adopting it for practical use.

Accordingly, this paper will use PKP-DSS and Shamir's 1989 PKP-based protocol as its main example for performance comparison. However, the theoretical discussions will focus on MQDSS, as that is the protocol for which the claimed proofs of security were written.

2.5 Sampling challenges from non-uniform distributions

The general approach of constraining the weight of challenge vectors for the purpose of improving one or more performance measurements has been discovered multiple times:

- Sampling a binary vector of challenges with constrained weight was proposed in the original Fiat–Shamir paper and patent. [FS87, §2.4] and [SF88, column 4, lines 47 to 69] starts with the example of $k = 5$ challenge bits per round, sampled uniformly and independently, and $t = 4$ protocol rounds, for a 2^{-20} probability of impersonation. Each 1 bit in the challenge vector corresponds to a modular multiplication by one of the prover's secret square roots.

  "Even better performance can be obtained by increasing k to 18 (a 1152 byte ROM). If we use e_{ij} vectors with at most three 1's in them, we have a choice of 988 possible vectors in each iteration." Here the benefit of using a challenge vector with constrained weight was that it allowed an improvement in the soundness of each round while decreasing the computational cost. The reduction in number of rounds to reach a given security level also achieved a reduction in communication cost. [SF88] goes on to state, "This is the preferred mode of the invention vis-a-vis identification."

- Leichtle proposed the use of rejection sampling of a challenge vector to transform a signature protocol with variable signature length into one with fixed signature length [Lei18, §7.1.8], and applied it to a signature protocol based on Stern's 3-pass 2/3 proof of knowledge of a fixed-weight codeword. The paper then reduces the problem of simulating a transcript of the parallel composition of the identification protocol with limited maximum signature length to simulating a transcript with unrestricted signature length, and cites a general proof of security to show that this proves that his fixed-length transformation does not reduce security.

Leichtle's rejection sampling transform leaves the absolute expected number of random oracle calls needed to forge a signature unchanged, but increases the expected number of random oracle calls by the honest signer to generate a signature. Thus, this transform reduces the ratio of costs between forgery and honest signing.

- Conceptually, the identification protocol used in the Picnic2 signature scheme [KKW18] performs two cut-and-choose steps; the first selects one of a set of "preprocessing phases" to use in an MPC protocol, and discloses the random seeds from which the others were computed so that the verifier can check that they were computed honestly. As usual, this protocol has a non-trivial soundness error for any practical set of parameters, so it must be repeated many times in parallel. However, the first step of this simplified protocol requires too large an overhead in both computation and communication to be useful, even with standard compression techniques; when m preprocessing phases are generated and sent, the soundness error is at least $1/m$ per protocol run.

  In the actual (potentially useful in practice) Picnic2 signature scheme and identification protocol, the cut-and-choose of the preprocessing phases is shared over the overall protocol, and which of these are to be used for the second step of the identification protocol are chosen by a fixed-weight vector [KKW18, §2.3, "Beating parallel repetition"].

  In Picnic2, the purpose of sampling a vector of challenges from a non-uniform distribution is specifically to reduce the size of the signature; it spends considerable extra computation time for that purpose. The second step of the underlying identification protocol in Picnic2 runs in time proportional to n to achieve a soundness error of $1/n$ per run, and so does the verification procedure for each of the preprocessing phases which is not chosen for use in the second protocol step. The overall effect is an exponential cost tradeoff of computation for reduced size, as in SPHINCS+[ABD+19].

  Additionally, in Picnic2, each run of the second step of the identification protocol varies in size and runs a different verification procedure for each challenge; that property, combined with the use of a "seed tree" to compress the first step, results in variable-length signatures which do not permit constant-time verification.

- The MUDFISH/SUSHSYFISH paper [Beu20] generalizes the overall strategy of Picnic2 to the class of identification protocols based on a single-use "trusted setup string", of which the "preprocessing phase" of the KKW MPC protocol is one example. As in Picnic2, the transform used to eliminate the need for trusted setup strings from an identification protocol uses a fixed-weight challenge vector [Beu20, §7, "Beating parallel repetition"] and results in an exponential speed-vs.-size tradeoff; unlike Picnic2, the transform specified in the final version of the paper does not use a seed tree, so the first step of the signature scheme does not guarantee variable-size signatures or variable-time verification.

MUDFISH and SUSHSYFISH then apply that setup-elimination transform to two new identification protocols based on the MQ and inhomogeneous PKP problems. Since constant-time verification was not a design goal, their implementations [Beu19a] do not achieve it, and some of the low-level details of SUSHSYFISH as implemented appear to be incompatible with efficient constant-time verification. However, the overall protocols could potentially support constant-time verification, and the timing leaks from the current implementation are probably not much worse than the pre-quantum signature verification routines currently deployed.

The goal of constant-time verification appears to be new, as is trying to achieve it in the 5-pass case with reasonable performance (i.e. no exponential tradeoff).

3 Changes to enable constant-time verification

Traditionally, every challenge in an identification protocol is sampled independently from the uniform distribution. Allowing constant-time verification requires sampling the vector of cut-and-choose challenges from a distribution with a fixed multiset of vector elements. This obviously reduces the size of the set from which the vectors are sampled.

In addition, all vectors of challenges must be sampled in constant time, and this will usually introduce a non-uniformity into the resulting distributions. Both changes must be taken into account when choosing parameters for the resulting signature protocol to reach a target security level.

3.1 Sampling the first challenge round in constant time

In most of the published $q2$ protocols, q is the order of a finite field over which the protocol operates. Since the CPUs most widely used as benchmarking platforms for performance evaluation include fast SIMD operations for integer multiplication, but not for multiplication in small binary fields, most implementations choose q to be a prime number rather than a power of 2. This raises the issue of how to convert the sequence of bits produced by a hash function into a sequence of challenges in $\mathbb{F}_q$. The standard options for this are:

- rejection sampling of vector elements, from an unbounded stream of XOF output. Not constant-time. Used in MQDSS for sampling of all vectors mod $q = 31$, including the sequence of first challenges.
- rejection sampling of vector elements, from a XOF output of bounded length. Can be made to run in constant time, by sorting. Has non-zero probability of failing outright. Used in PKP-DSS for sampling of all vectors mod q (value depends on parameter set), including the sequence of first challenges.

- reduction of chunks of bitstream mod q, from a XOF output of fixed length. Can easily be made to run in constant time, using code which will already be required for field operations in $\mathbb{F}_q$. No possibility of failure, but the outputs in $\mathbb{F}_q$ will be non-uniform.
- division of b-bit chunks of bitstream by $2^b/q$. Faster than computing a remainder when q is much smaller than 2^b. Used in NTRU Prime to sample a vector from $\{0,1,2\}^n$ [Ber18, §2.4]. Has the same amount of non-uniformity as reduction mod q, and when $\mathbb{F}_q$ is used as a field, requires additional code not used for any other purpose.

Some non-uniformity is acceptable for the vector of challenges, so reduction mod q is the best constant-time challenge sampling option for signature schemes based on $q2$ protocols. Quantitatively, when reducing a b-bit value mod q, there are at most $P_{\max} = \lceil 2^b/q \rceil$ preimages [Ber18, Theorem 2.2] and at least $P_{\min} = \lfloor 2^b/q \rfloor$ preimages; the probabilities of sampling field elements are either $p_{\max} = P_{\max}/2^b$ or $p_{\min} = P_{\min}/2^b$. Given these probabilities, in the Kales-Zaverucha attack, the probability of guessing the first challenge in at least N out of r parallel runs is at most $\sum_{i=N}^{r} p_{\max}^{-i}(1-p_{\min})^{r-i}\binom{r}{i}$. This non-uniformity is taken into account below.

3.2 Fixed-multiset challenge vectors

Sampling challenge vectors for the cut-and-choose step of a signature protocol to have a fixed multiset of elements in random order obviously decreases the size of the challenge space for a given number of parallel runs compared to sampling each vector element uniformly at random. The question is, by how much?

Let $\mathrm{FWV}(r_0, r_1)$ denote the set of fixed-weight vectors of length $r_0 + r_1$ and weight r_1. Let $\mathrm{FWV}(r_0, r_1, r_2)$ denote the set of vectors in $\{0,1,2\}^{r_0+r_1+r_2}$ with r_0 occurrences of 0, r_1 occurrences of 1, and r_2 occurrences of 2; note that $\#(\mathrm{FWV}(r_0, r_1, r_2)) = \#(\mathrm{FWV}(r_0 + r_1, r_2) \times \mathrm{FWV}(r_0, r_1))$. Let $\mathrm{wt}_x(v)$ and $\mathrm{wt}_Y(v)$ denote the number of elements of the vector v equal to x or contained in the set Y, respectively.

For the random-order challenge vectors, δ will denote the divergence bound from [Ber18, §3, §4, §5] for sampling by sorting. All numerical examples will assume that the number of random bits b is chosen such that the sort operates on 32-bit values; for binary vectors, this means $b = 31$. Note that δ is a function of the number of parallel runs r as well as b.

$\mathcal{H}_{\mathsf{id}}$, $\mathcal{G}_{\mathsf{id}}$, and S_{id} will refer to the corresponding sets for the cut-and-choose challenge step of each single run of the underlying identification protocol. Note that the definition allows $\mathcal{G}$ to have extra elements which never succeed; this will be used below.

The simplest case to analyze is that where the underlying protocol has $\mathcal{H}_{\mathsf{id}} = \{0,1\}$; then $\mathcal{G}_{\mathsf{id}} = \mathcal{H}_{\mathsf{id}}$ and S_{id} is the equality predicate. Let r denote the number of parallel runs of the underlying protocol. Previously, the set of possible challenge vectors would have been $\mathcal{H}_{\mathsf{id}}^r$ of size 2^r, sampled from the uniform distribution, so the probability of forging a signature would have been $(2^r)^{-1}$. With challenges sampled from $\mathrm{FWV}(r_0, r_1)$ with a divergence from uniformity of at most δ, the number of possible challenge vectors is $\binom{r_0+r_1}{r_1}$, and the probability of guessing a challenge vector correctly is $\delta\binom{r_0+r_1}{r_1}^{-1}$. As a numerical example, for $r = 128$ and $r_0 = r_1 = 64$, the probability of guessing the challenge vector is slightly less than 2^{-124}; to reach the 128-bit preimage security level with $|r_0 - r_1| \leq 1$, r must be increased to 132.

The next simplest case, and the most important one given the identification protocols which it applies to, is the second challenge pass of a $q2$ protocol. Assume that the Kales–Zaverucha attack has guessed the first challenge correctly for N of the r parallel runs, let r_0 and r_1 be fixed system parameters such that $r = r_0 + r_1$, and let $\mathcal{H}$, $\mathcal{G}$, and S refer to the second challenge pass. Without loss of generality, assume that the N runs guessed by the K–Z attack are at the end of the vector. Then $\mathcal{H} = \mathrm{FWV}(r_0, r_1)$, but $\mathcal{G} = \{0, 1\}^{r-N}$ and $S(g, h) \iff (\forall_{i=0}^{r-N} g_i = h_i)$, ignoring the last N elements of h. Now different guesses $g \in \mathcal{G}$ have widely varying probabilities of success; out of $\binom{r}{r_1}$ elements of $\mathcal{H}$ sampled with divergence δ from uniformity, $S(g, h)$ for $\binom{N}{r_1 - \mathrm{wt}_1(g)}$ values of h. The overall time to break a set of parameters, again using the approximation that an event of probability p is achieved in $1/p$ operations, is thus:

$$\min_{N=0}^{r}\left(\left(\sum_{i=N}^{r} p_{\max}^{-i}(1-p_{\min})^{r-i}\binom{r}{i}\right)^{-1} + \delta \max_{w=0}^{r_1}\left(\frac{\binom{r}{r_1}}{\binom{N}{r_1-w}}\right)\right)$$

Empirically, at the Category 1 security level, switching from independent, uniform challenges to fixed-weight vectors does not increase the required number of identification protocol runs at all. When the second commitment pass and response for one of the challenge values is reduced to one random seed per protocol run, and the number of runs is increased to optimize for signature size, the weight of the challenge with larger response in the challenge vector is about $r/3$.

In the case of 3-pass identification protocols with soundness error 2/3, $\mathcal{G}_{\mathsf{id}} = \mathcal{H}_{\mathsf{id}} = \{0, 1, 2\}$ and $S_{\mathsf{id}}(g, h) \iff g \neq h$. Then $\mathcal{H} = \mathrm{FWV}(r_0, r_1, r_2)$, $\mathcal{G} = \{0, 1, 2\}^r$, and $S(g, h) \iff (\forall_{i=0}^{r} g_i \neq h_i)$. For any chosen $g \in \mathcal{G}$, $h \in \mathcal{H}$, let $w_{i,j}$ denote the number of indices k where $g_k = i$ and $h_k = j$. Then $\sum_{j=0}^{2} w_{i,j} = \mathrm{wt}_i(g)$ and $\sum_{i=0}^{2} w_{i,j} = r_j$, and g will succeed if and only if $w_{i,i} = 0$ for all i. For a fixed g and fixed, valid $w_{i,j}$ with $w_{i,i} = 0$, there are

$$\binom{\mathrm{wt}_0(g)}{w_{0,1}}\binom{\mathrm{wt}_1(g)}{w_{1,0}}\binom{\mathrm{wt}_2(g)}{w_{2,0}}$$

corresponding values of h. To determine the number of challenges for which a guess g succeeds, one sums over all possible arrays $w_{i,j}$ (three small-integer degrees of freedom); to determine the security level of a parameter set (r_0, r_1, r_2), one calculates the probability of a guess g most likely to succeed (two small-integer degrees of freedom, $\mathrm{wt}_0(g)$ and $\mathrm{wt}_1(g)$), accounting for sampling divergence. Optimizing for size requires a search over further degrees of freedom.

This calculation is simple enough to describe using elementary combinatorial methods, but will be unreasonably slow to evaluate without further optimization or the use of analytic techniques. This effort is unlikely to be worthwhile for general-purpose signature use, as an optimal parameter set for the $\varepsilon = 2/3$ structure must have almost equal quantities of the two challenges which are answered with larger responses.

Pointcheval's 3-pass identification protocol based on the permuted perceptrons problem, with soundness error 3/4, can be handled analogously to the protocols with soundness error 2/3, and his 5-pass zero-knowledge protocol combines puncturing due to the Kales–Zaverucha attack with the 2/3 case. As in the case of 3-pass protocols with 2/3 soundness error, these protocols are unlikely to be competitive as identification or signature protocols, so there is no reason to go into further detail.

4 Effect on provable security

To support constant-time verification of a general $q2$-protocol-based signature, both challenge vectors must be sampled from non-uniform distributions, whereas the proofs of security claimed to apply to such protocols have assumed that the Fiat–Shamir random oracles will sample each challenge independently and uniformly. As discussed below, the "proofs of security" which are possible for this class of protocol are of no, or even negative, practical value, but the cryptography community expects them, regardless of whether the proofs are correct or the statements allegedly proved are useful. Additionally, the MQDSS authors explicitly requested a proof of security for these changes.

Adapting the MQDSS proofs of security to handle fixed-weight challenge vectors was made more difficult by the fact that no one has written a completely correct proof of security of any kind for the original MQDSS.

The original random oracle model (ROM) proof of security in [CHR$^+$16, §4.3] and [CHR$^+$, §A] has a minor error in the statement of its "forking lemma" [CHR$^+$16, §4.3, Lemma 4.10][CHR$^+$, §A.1, Lemma A.2] but takes essentially the correct steps and contains enough information in the proof to invoke a (correctly defined) $q2$-extractor.

However, the quantum random oracle model (QROM) proofs of security in [DFM20] are entirely inapplicable to MQDSS.

4.1 Errors in prior MQDSS "proofs of security"

Incorrect definitions of the $q2$-extractor concept: The concept of $q2$-extractor was introduced in [CHR$^+$16] to abstract away the underlying interactive protocol from their proof of security for the surrounding signature protocol. The MQDSS authors published (essentially) two different definitions of the concept; the final version of the PKP-DSS paper [BFK$^+$19a, §2.3, Definition 6] specifies a third definition.

The important difference between the various definitions of $q2$-extractor is the pattern which a set of transcripts must follow to permit a break of the identification protocol to be extracted:

- The first definition of $q2$-extractor, in the original MQDSS paper [CHR$^+$16, §4.2, Definition 4.6], specified a sequence of four transcripts of the form "$\mathsf{trans}^{(i)} = (\mathsf{com}, \mathsf{ch}_1^{(i)}, \mathsf{resp}_1^{(i)}, \mathsf{ch}_2^{(i)}, \mathsf{resp}_2^{(i)})$, $i \in \{1, 2, 3, 4\}$, with

$$\mathsf{ch}_1^{(1)} = \mathsf{ch}_1^{(2)} \neq \mathsf{ch}_1^{(3)} = \mathsf{ch}_1^{(4)},$$
$$\mathsf{ch}_2^{(1)} = \mathsf{ch}_2^{(3)} \neq \mathsf{ch}_2^{(2)} = \mathsf{ch}_2^{(4)},$$

  valid with respect to pk".

  This definition fails on the protocol side: Because it does not place any restrictions on the message $\mathsf{resp}_1^{(i)}$ sent in the second commitment pass, there is no $q2$-extractor according to this definition for the SSH MQ-based 5-pass protocol. Kales and Zaverucha provided an efficient algorithm that can produce a set of four successful transcripts of this form as part of their structural attack on the 5-pass Fiat–Shamir transform [KZ20, §3.2, p. 10].

- The second definition of $q2$-extractor appears in two trivially different versions, a post-quantum definition in the SOFIA paper [CHR$^+$17b, §2, Definition 2.9], by the same authors as MQDSS, and a definition limited to classical algorithms in all versions of the MQDSS NIST submission document [CHR$^+$, §1.2.2, Definition 1.13]. This pair of definitions specifies a sequence of "four valid transcripts with respect to pk:

$$\mathsf{trans}^{(1)} = (\mathsf{com}, \mathsf{ch}_1, \mathsf{resp}_1, \mathsf{ch}_2, \mathsf{resp}_2),\ \mathsf{trans}^{(3)} = (\mathsf{com}, \mathsf{ch}'_1, \mathsf{resp}'_1, \mathsf{ch}_2, \mathsf{resp}_2),$$
$$\mathsf{trans}^{(2)} = (\mathsf{com}, \mathsf{ch}_1, \mathsf{resp}_1, \mathsf{ch}'_2, \mathsf{resp}'_2),\ \mathsf{trans}^{(4)} = (\mathsf{com}, \mathsf{ch}'_1, \mathsf{resp}'_1, \mathsf{ch}'_2, \mathsf{resp}'_2)$$

  where $\mathsf{ch}_1 \neq \mathsf{ch}'_1$ and $\mathsf{ch}_2 \neq \mathsf{ch}'_2$".

  In MQDSS, this definition fails on the signature security reduction side: There is no reason, without looking at protocol details which this definition is meant to abstract away, that there should be only one valid response resp_2 for each of the two possible values of ch_2. Further, when the signature scheme is optimized to omit the second commitment value from signatures and recompute it from the final response, the resulting identification protocol may not satisfy this condition on responses.

- The third definition of $q2$-extractor appears in the final version of the PKP-DSS paper [BFK$^+$19a, §2.3, Definition 6]. This version specifies a sequence of "four transcripts $(\mathsf{com}, c^{(i)}, \mathsf{rsp}_1^{(i)}, b^{(i)}, \mathsf{rsp}_2^{(i)})$ for i from 1 to 4, such that

$$c^{(1)} = c^{(2)} \neq c^{(3)} = c^{(4)}$$
$$\mathsf{rsp}_1^{(1)} = \mathsf{rsp}_1^{(2)} \quad \mathsf{rsp}_1^{(3)} = \mathsf{rsp}_1^{(4)}$$
$$b^{(1)} = b^{(3)} \neq b^{(2)} = b^{(4)}\text{''}.$$

This definition correctly captures the concept; it holds for the protocols used in MQDSS and PKP-DSS (over finite fields), and it can be used by a correct security reduction in the ROM for a signature scheme.

When the second challenge is sampled from $\{0, 1\}$, the definition given above in section 2.2 is equivalent and easier to read.

A further fact should be noted here. The MQDSS authors did point out in the SOFIA paper [CHR$^+$17b, §2, p. 7, above Definition 2.9] that they were giving a different definition of $q2$-extractor than the original paper specified: "In the following we give a post-quantum version of $q2$-Extractor that fixes two slight technical shortcomings of the definition in [CHR$^+$16]. On the one hand, we add the algorithm that actually generates the transcripts to the definition, on the other hand we use the notion of key relation to capture what kind of secret key the extractor returns."

The SOFIA paper did not mention the change in the set of transcripts required for extraction, nor did the MQDSS specifications. Their authors clearly knew that (part of) that change was needed to allow correct proofs, but they have never explicitly stated that fact. By silently revising their definition without ever explicitly stating a correction, they led Don et al. to rely on their previous, incorrect, definition of $q2$-extractor in [DFM20, §A, Definition 26].

No proof of soundness against a quantum adversary: Two different proofs of soundness-like properties for the Sakumoto–Shirai–Hiwatari 5-pass identification protocol have appeared:

- A sketch of a proof in [SSH11, §4, Theorem 5] that the protocol is "argument of knowledge" for the key relation, i.e. sound against computationally bounded attackers, with knowledge error $\frac{1}{2} + \frac{1}{2q}$. The sketch shows a $q2$-extractor (under the correct definition) for the identification protocol, but does not show how this pattern of challenges is obtained from the attack algorithm.

- A proof in [CHR$^+$16, §3, Theorem 3.1] that the protocol is sound against computationally bounded attackers, with soundness error $\frac{1}{2} + \frac{1}{2q}$. This proof explicitly rewinds the attack algorithm. It then shows a $q2$-extractor (under the correct definition), though it adds the unnecessary condition that both of the values $\mathbf{t}_1$ and $\mathbf{e}_1$ sent in the second commitment pass differ when the challenges differ (not true if $\mathbf{r}_0 = \mathbf{0}$).

Since they rely on rewinding, either explicitly or in a gap in the proof sketch, they are not currently believed to hold for quantum algorithms. Accordingly, [DFM20, §5.2, Corollary 13] is not applicable to MQDSS.

5-pass identification protocols do not have computationally unique responses: Don et al. prove the security of the multi-round Fiat–Shamir transform under the assumption that the interactive protocol has "computationally unique responses" [DFM20, §7.1, Definition 22]: "if given a partial transcript $(x, a_1, c_1, \ldots a_i, c_i)$ it is computationally hard to find two accepting conversations that both extend the partial transcript but differ in (at least) a_{i+1} (here we consider z to be equal to a_{n+1})". (In their notation, the first item in a transcript is the public key x, the commitment rounds are denoted a_i instead of this paper's c_i, and the challenge rounds are denoted c_i instead of h_i.)

As with the original definition of $q2$-extractor, the attack algorithm by Kales and Zaverucha constructs a counterexample for the runs for which it fails to guess the first challenge correctly. This time, two transcripts are sufficient to prove that the definition does not apply to the protocol.

The essential step of the proof of [DFM20, §7.1, Theorem 23] is a use of this "computationally unique responses" property. Thus, that theorem cannot be applied to MQDSS as the first paragraph of the section implies.

QROM proof of security uses wrong version of $q2$-extractor: Don et al. give a separate sketch of a proof of security for $q2$-extractable identification schemes in [DFM20, §A]. The definition they rely on is the original, flawed, definition of $q2$-extractor from [CHR+16, §4.2, Definition 4.6], which does not constrain the second commitment round. They then rely on this error in the definition by assuming that they can obtain the set of transcripts to be provided to the $q2$-extractor by rerunning the attack algorithm four times from the beginning, rather than rewinding it.

It is not clear from that paper exactly what the proof is, but as with computationally unique responses, an incorrect step this fundamental is likely to break the proof.

4.2 Probability of $q2$-extraction

To permit constant-time verification of $q2$-based signatures, three changes are made to the distributions from which challenges are sampled:

- Each element of the first challenge vector is sampled i.i.d. from a slightly non-uniform distribution Ω_1, by reducing a b-bit value modulo q. What is important is the maximum probability over Ω_1^2 of sampling the same challenge value twice. Let $p_{\max} := \lceil 2^b/q \rceil$ be the probability of the most likely challenge; then $p_{\max}^2$ is the probability that extraction fails in any position due to the first challenge pass.

  In an asymptotic proof, if q increases as a function of the security level, b must increase as well.

- The second challenge vector is sampled from a non-uniform distribution, by sorting. Let δ be the divergence bound for fixed-weight binary vectors from [Ber18, §3].

  Note that, in that paper's notation, δ is a function of both $b + 1$ (the size of words being sorted) and n (the vector length). In an asymptotic proof, n increases as a function of the security level, so b must increase as well.

- The second challenge vector is sampled as a whole, not i.i.d. element-wise. This requires some combinatorics.

The following will use the notation $[1\,..\,n]$ from [FS09] for the "integer interval" from 1 to n. In addition, this section will use $[1\,..\,n]_w$ to denote the set of all w-element subsets of the integer interval $[1\,..\,n]$.

As a first step, consider the overlap between (positions of ones in) fixed-weight vectors, or equivalently intersections between pairs of sets sampled from $\Omega := [1\,..\,n]_{w_1} \times [1\,..\,n]_{w_2}$.

For any given k-element set $I \in [1\,..\,n]_k$, there are $\binom{n-k}{(w_1-k)+(w_2-k)}\binom{(w_1-k)+(w_2-k)}{(w_1-k)}$ pairs $(S_1, S_2) \in \Omega$ such that $S_1 \cap S_2 = I$. Since there are $\binom{n}{k}$ elements of $[1\,..\,n]_k$, the probability that a pair sampled uniformly from Ω has intersection of size k is

$$p_{\cap(n,w_1,w_2)}(k) := \frac{\binom{n}{k}\binom{n-k}{(w_1-k)+(w_2-k)}\binom{(w_1-k)+(w_2-k)}{(w_1-k)}}{\binom{n}{w_1}\binom{n}{w_2}}.$$

It may be verified experimentally that the sum over all k is equal to 1.

Where the positions of ones in two fixed-weight vectors do *not* overlap, they mark differences between the vectors. The number of positions where two fixed-weight vectors of weights w_1 and w_2 and overlapping in k positions differ can be obtained as $(w_1 - k) + (w_2 - k)$, or $2(w - k)$ if $w = w_1 = w_2$. Let $p_{\triangle(n,w)}(k')$ denote the probability that two vectors of equal weight w have k' differences.

The probability that $q2$-extraction will be permitted by the vectors of second challenges in exactly k positions is then

$$p_2(k) := \sum_{d_1=0}^{r}\left(\sum_{d_2=0}^{r}\left(p_{\cap(r,d_1,d_2)}(k)\ p_{\triangle(r,r_1)}(d_1)\ p_{\triangle(r,r_1)}(d_2)\right)\right),$$

and given each k, the probability that extraction fails due to the first challenge for all of those k runs is at most $p_{\max}^{2k}$. The overall upper bound on the probability that $q2$-extraction will fail is the weighted average

$$p_{\text{fail}} \le \delta \sum_{k=0}^{n}\left(p_{\max}^{2k}\ p_2(k)\right).$$

The MQDSS spec states [CHR$^+$20, §A, Lemma A.3, p. 78] that the probability that extraction fails with uniform, i.i.d. challenges is $\left(\frac{3q+1}{4q}\right)^r$. Using the MQDSS version 2.1 parameter set for Category 1 [CHR$^+$20, §8.1], with $q = 31$ and $r = 184$, as an example, the extraction failure probability is about $2^{-73.5}$, whereas a size-optimized parameter set with $r = 200$ and $r_1 = 70$ results in an extraction failure probability of about $2^{-70.0}$.

For comparison, the additional parameter set proposed in the MQDSS spec over $\mathbb{F}_{64}$ at the same security level has $q = 64$ and $r = 171$ [CHR$^+$20, §8.2], and its extraction failure probability is about $2^{-69.7}$.

An asymptotic proof requires more advanced techniques. The rest of this subsection will use further terminology from [FS09].

There are two approaches, one longer but potentially easier to check for mistakes, and one direct method.

The longer approach can be sketched as follows:

- Prove the asymptotic mean and standard deviation of the distribution of overlaps between two fixed-weight vectors, using the exponential MGF $\exp(z(1+uv_1v_2+v_1+v2))$ in which v_1 and v_2 mark w_1 and w_2, and u marks the number of overlaps. Also prove that the distribution is concentrated.

  Empirically, for vectors of length n and equal weight λn, the mean overlap is $\lambda^2 n$.

- If the asymptotic mean is only proved for equal weights, prove that increasing the weight of either vector increases the number of overlaps, and vice versa.

- Prove asymptotic upper and lower bounds on some fraction of the probability mass, e.g. 1/2.

- Use the upper and lower bounds to prove a lower bound on the number of extractable runs, for e.g. $(1/2)^3$ of possible signatures.

- Multiply by $p_{\max}^{2k}$ and δ.

The shorter approach uses the exponential MGF

$$\mathbf{F}(\mathbf{z}) := \exp(z((1+v_1)(1+v_2)(1+v_3)(1+v_4) + (u-1)(v_1+v_2)(v_3+v_4))),$$

where u marks extractable runs, to solve the whole problem at once. Obtain an asymptotic approximation for $f(u) := [z^r][v_1^{r_1}][v_2^{r_1}][v_3^{r_1}][v_4^{r_1}]\mathbf{F}(\mathbf{z})$ and the PGF $p(u) := \frac{f(u)}{f(1)}$, then evaluate at $u = p_{\max}^2$ to obtain the asymptotic probability of extraction failure.

Both problems should be solvable using a generalization of the large powers method covered in [FS09, § VIII. 8.] to the multivariate case.

Since asymptotic proofs of security say nothing of practical value, and the preceding subsection provides overwhelming evidence that no one will ever read or verify the proofs, I will leave the details to the theoreticians.

4.3 Comments on provable security

- The Kales–Zaverucha attack is sufficient to show that reductionist proofs of security for signature schemes based on the probability of extraction success or failure can never be useful for parameter selection in practice. The probability of extraction failure varies wildly for parameter sets with the same security level and different values of q in the range used by MQDSS.

- Public coin identification protocols are not known to be sound in a quantum world at all. Attacking a signature scheme requires solving a set of equations involving a hash function, and this set of equations remains the same with and without the assumption of quantum computing.

- [DFM20, §5.2, Remark 14] claims to show that the Fiat–Shamir transform applied to the *sequential* composition of cut-and-choose protocols should be expected to have a non-trivial provable security bound. This construction was implicitly known to be broken back to [FS87] and [Sch96, §5.1],

  In general, the field of cryptography has done a terrible job of teaching its fundamentals to the current generation. The material in the following section shows further examples of this.

5 Generic optimization considerations

Given that some applications require signature protocols based on 5-pass $q2$ identification protocols to choose the second challenge vector to have fixed weight, the natural next step is to take advantage of the side benefit that the signature will have fixed size even if the underlying identification protocol has differing communication costs depending on the cut-and-choose challenge h_2.

In particular, for one of the two cut-and-choose challenges, the total communication cost per run can be reduced to one random seed and one commitment; then, the challenge vector's length and weight can be optimized together to minimize signature size for a given security level. (The value which would be sent in the clear during the second commitment pass is recomputed from the random seed.)

Many of the considerations in optimizing these signature protocols are independent of the details of the identification protocol they are based on. These were lessons learned in the 1990s, which have since been all but forgotten. Most—not all—have since been rediscovered in the context of the Picnic NIST submission [CDG+20].

Domain separation: When a cryptographic protocol uses more than one random oracle or hash function, or uses a single hash in more than one way, it is standard practice to map the set of inputs for each use into a separate domain, usually by including some context tag value in the input to the underlying hash function. Some examples of the consequences of failing to separate the domains properly in the context of KEMs are shown in [BDG20], along with quotations from a small part of the extensive prior art on this topic.

Preventing multi-target preimage search: It has long been known that generic preimage search circuits can find a secret preimage of one of 2^k outputs of a single function in about $1/2^k$ of the time needed to find a preimage for a single output. ([Ber05] describes two of the applicable parallel algorithms.) There are two standard ways to defend against this: either use inputs long enough that the target security level will be achieved even if the attacker obtains many function outputs, or add a nonce which is collision-resistant at the target security level. Where the key length is limited, as in signature compression, adding a nonce (shared over the whole signature) is the preferred choice.

Salting and collision-resilience for message hashes: Another standard practice for Fiat–Shamir signatures is to prefix an unpredictable salt to the message before hashing it, so that a party who supplies a message to be signed cannot precompute two messages with colliding hashes. For Fiat–Shamir signatures based on a single run of a non-cut-and-choose identification protocol, this salt can be the commitment value; since changing the commitment would invalidate the response, this is sufficient to provide collision-resilience as well.

Where the commitment is a sequence of hashes, this approach is not sufficient; in this case, collision-resilience can only be obtained by including the salt with the message hash in every use within the signature scheme.

Collision-resilience for commitment hashes: It is also beneficial to protect the commitment hashes in a cut-and-choose-based signature protocol against precomputed collision attacks. In this case, the relying party for collision-resilience is the verifier, so the verifier must be certain that the salt could not have been predicted. To ensure this property, first, the signature scheme must include the message hash in the commitment inputs, rather than a salt freely chosen by the attacker. Second, the application must include some string which the verifier knows was unpredictable into the message to be signed.

For example, the salt in Picnic2's commitment inputs [st20, §7.1, p. 23] is provided directly in the signature, so it adds no extra security.

Implementing collision-resilience correctly for the commitment hashes is beneficial in another way: If a collision search on the commitment hashes cannot begin until the message is known, an interactive protocol in which the verifier provides a nonce to be signed can accept a lower security level, allowing for shorter signatures.

Omitting the commitment randomization string: The MQDSS team asserted in version 2.0 of their specification that a commitment randomization string is needed to prove that their signature scheme is "EU-CMA" [CHR$^+$19, Introduction, "New in Version 2.0"], adding $2k$ bits per protocol run at a preimage security level of k bits. This is not true; in fact, adding the randomization strings strictly reduces security in practice.

- $2k$ bits is twice as long as could possibly be necessary at the k-bit security level.
- Signatures are generated deterministically, as a function of the message and a k-bit secret sk. Both the commitment opening and the randomization string are subject to a single preimage search on the same k-bit string. (Though this is not useful, as it is easier to recover sk from $S_{\mathbf{F}}$.)
- The chain of properties they believe they need is "computationally hiding commitments" $\implies$ "computational HVZK" $\implies$ "computational EU-CMA", and adding a randomization string would give them computationally hiding commitments.

  Here it is simpler to show computational HVZK under the assumption that the adversary does not know the secret key. Knowledge of the honest prover's openings for a single run is sufficient to recover the secret key, and one opening is already disclosed; an attacker who can distinguish whether the other commitment was honestly generated has queried the random oracle on the value which the honest prover would have used. (Theoreticians may wish to expand this sketch into a multi-page oracle extraction proof.)

- Signatures are generated deterministically, so anyone who has the signer's secret key can distinguish the honest prover's transcripts from simulated transcripts. Apart from the fact that the identification protocol's secret key is a one-way function of the signer's seed, this essentially rules out HVZK against an attacker which knows the secret key.

- Including randomization strings in the commitment inputs enables generic collision attacks on c_0 and c_1 separately which would otherwise have had to maintain a functional relation between the openings to be useful.

6 Optimizing PKP-DSS

The basic Permuted Kernel Problem is, given parameters m, n, and q, a matrix $\mathbf{A} \in \mathbb{F}_q^{m \times n}$, and a vector $\mathbf{v} \in \mathbb{F}_q^n$, to find a permutation π such that $\mathbf{A}(\mathbf{v}_\pi) = \mathbf{0}$. (The notation used here is from the PKP-DSS paper, which conforms to modern sensibilities better than Shamir's notation.)

Shamir's original publication of the PKP-based identification protocol mentioned that "the homogeneous equations can be replaced by non-homogeneous equations" [Sha90a, §3]. [Sha90b, column 5] states that "[t]he permutations π and σ can be chosen from any publicly known subgroup", explicitly describes the non-homogeneous case and how it can be implemented as a special case of subgroup homogeneous PKP, and mentions that, because $m < n$, the non-homogeneous case allows a smaller public key.

PKP-DSS obtains the same key size benefit in a different way, at the cost of imposing additional constraints on $\mathbf{v}$. I prefer the simpler, older approach of directly using the non-homogeneous case.

For the sake of simplicity and generality, this section will consider the subgroup inhomogeneous PKP problem directly, where $\pi \in G \leq S_n$ and $\mathbf{A}(\mathbf{v}_\pi) = \mathbf{t}$.

6.1 Permutation operations

For consistency with implementations which will use 0-based arrays, vector and matrix indices will be considered to be 0-based here, and S_n will act on the set of non-negative integers less than n.

Sorting $\{(\pi^{-1}(i), \mathbf{v}_i, \sigma(i))\}_i$ results in $\{(i, (\mathbf{v}_\pi)_i, (\sigma \circ \pi)(i))\}_i$. Either $\mathbf{v}$ or σ can be omitted, but a signer with memory to spare can benefit from operating on both at once, which is possible for the parameter sets proposed below.

6.2 Revising the protocol

First, Shamir's original PKP-based identification protocol, in the notation above:

Shamir's PKP-based identification protocol

$\mathcal{P}(\mathsf{sk}, \mathsf{pk})$		$\mathcal{V}(\mathsf{pk})$
$(\sigma, \mathbf{r}) \leftarrow_{\$} G \times \mathbb{F}_q^n$		
$c_0 \leftarrow H(\sigma, \mathbf{Ar})$		
$c_1 \leftarrow H(\pi\sigma, \mathbf{r}_\sigma)$		
	$\xrightarrow{(c_0, c_1)}$	
		$\alpha \leftarrow_{\$} \mathbb{F}_q$
	$\xleftarrow{\alpha}$	
$\mathbf{z} \leftarrow \mathbf{r}_\sigma + \alpha \mathbf{v}_{\pi\sigma}$		
	$\xrightarrow{\mathbf{z}}$	
		$b \leftarrow_{\$} \{0, 1\}$
	$\xleftarrow{b}$	
if $b = 0$ **then** rsp $\leftarrow \sigma$		
if $b = 1$ **then** rsp $\leftarrow \pi\sigma$		
	$\xrightarrow{\text{rsp}}$	
		if $b = 0$ **then return** $c_0 \overset{?}{=} H(\sigma, \mathbf{A}(\mathbf{z}_{\sigma^{-1}}) - \alpha \mathbf{t})$
		if $b = 1$ **then return** $c_1 \overset{?}{=} H(\pi\sigma, \mathbf{z} - \alpha \mathbf{v}_{\pi\sigma})$

PKP-DSS [BFK$^+$19a, §3.2] compresses rsp to a random seed by generating $\pi\sigma$ and r_σ from independent seeds, and constraining the public key further so that σ can be recovered uniquely from $(r_\sigma, \mathbf{z})$.

To obtain smaller signatures, I generate $(\pi\sigma, r_\sigma)$ from a single seed; generate the $\mathbf{z}$ values, hash them to obtain the second challenge pass vector, and send the hash; and send either the seed or $(\mathbf{z}, \sigma)$ for each run.

Parameter sets below will replace b with $1 - b$ to maintain the convention that the $b = 1$ case has longer proofs, as would be the case for an optimized MQDSS.

6.3 Encoding of vectors and permutations

Recursive vector encoding algorithms based on trees of arbitrary depth have been studied since 2008:

- Mihai Patrascu's "Succincter" included a forest-based representation for a vector of trits [Pat08, §2.1]. The payload may be considered to be stored at leaf nodes; higher-level nodes contain only "spills" passed up from below. The paper chooses parameters to obtain theoretically near-optimal size.

- Dodis, Patrascu, and Thorup developed a vector encoding stated for a general element bound (uniform throughout the vector), based on a heap [DPT10, §4.2]. The tree structure here is not quite balanced (three node types at each level), and payload is stored at internal nodes, making it more complex than the original and not a strict generalization.

- The NTRU Prime round 2 NIST submission included a vector encoding [BCLvV19b, §3.1] which is structurally a generalization of the Succincter trit vector, but with a few parameters fixed for implementation convenience. Payload is stored at leaf nodes only, and when all vector positions have the same upper bound, there is at most one unbalanced node at each level, and the encoding routine can be vectorized.

  The NTRU Prime round 2 vector encoder is already specified for vectors with an arbitrary sequence of upper bounds; this functionality is not used in NTRU Prime itself, but it enables further optimizations in this cryptosystem.

There are two further obvious generalizations for the encoder used in NTRU Prime:

- As specified, the encoder stores the root node of the encoding tree in the output byte vector. The root node could instead be passed out as an integer spill value ≤ 16384; when multiple vectors must be packed, this can save space without complicating low-memory or vectorized implementations.

- Where public keys must be represented as human-transportable strings, the encoder can output base 36 or base 37, or fill in an output vector of elements with varying upper bounds, instead of encoding to a sequence of bytes.

The encoding of permutations requires a tradeoff of computation vs. size. The permutation array can be encoded as in memory, packing all permutation vector elements with n as an upper bound, or the array can be squished down in $O(n^2)$ operations to fit in about $\lg(n!)$ bits. My recommendation is to use un-squished permutations for interactive-only parameter sets (below Category 1) and squished permutations for higher settings.

The reference implementation [Ran20b] uses a simpler, byte-oriented encoding for all hash function inputs, so signatures for different settings of the permutation squishing and root merging options can be interconverted.

6.4 Choice of parameter sizes

Four different sets of parameter sizes, each containing one tuple of (q, n, m) for each of the NIST security levels Category 1, Category 3, and Category 5, have been published by the PKP-DSS authors:

- $(379, 51, 16)$, $(521, 70, 23)$, and $(661, 90, 33)$ in the first two versions of ePrint 2018/714 (20180801:194820 [FKMR$^+$18a] and 20180928:145725 [FKMR$^+$18b]). They chose (q, n) such that "$n \approx \mathcal{O}(\sqrt{p})$", so that a random element of $\text{Ker}(\mathbf{A})$ could be sampled. (At this point, they were still using Shamir's homogeneous PKP protocol.)

  On the other hand, they kept the modulus q as small as possible to defend against a claimed 2001 attack algorithm by Joux and Jaulmes, in which the total number of operations is inversely proportional to some power of q.

  All versions of ePrint 2018/714 state that m is chosen such that "$n! \approx p^m$", so that the average number of permutations which map any given vector into $\text{Ker}(\mathbf{A})$ can be expected to be approximately 1. However, this first set of parameter sizes was chosen with much smaller values of m, so that random vectors in $\text{Ker}(\mathbf{A})$ could be sampled in a tolerable amount of time.

- $(977, 61, 28)$, $(1409, 87, 42)$, and $(1889, 111, 55)$ in ePrint 2018/714 version 20181201:152523 [FKMR$^+$18c], which also contained an early draft of the security analysis later published as [KMRP19]. According to this analysis, the Joux–Jaulmes attack is nowhere near as efficient as an earlier (1993) algorithm by Patarin and Chauvaud, in which only one of the attack steps takes time inversely proportional to a power of q.

- $(251, 69, 41)$, $(509, 94, 54)$, and $(4093, 106, 47)$ in ePrint 2018/714 version 20190410:111616 [FKMR$^+$19]. This version also changed the key generation method and key format to their new variant of inhomogeneous PKP.

  The rationale stated for selection of these sizes was not updated until the following version (20190925:113903) [BFK$^+$19c], which states that they evaluated the security levels of sizes with q constrained to be a prime number close to a power of 2, then chose (q, n, m) to minimize signature size (given the rest of their design).

- $(997, 61, 28)$, $(1409, 87, 42)$, and $(1889, 111, 55)$, used for SUSHSYFISH in all versions of ePrint 2019/490 [Beu20], and attributed to ePrint 2018/714 (no version number specified). 997 appears to be a typo of 977.

The known attacks on PKP appear to scale as a generalized birthday attack, which would make them less feasible than collision attacks at the same nominal RAM model attack cost. Accordingly, I consider $(977, 61, 28)$ and $(1409, 87, 42)$ to either reach the Category 2 and Category 4 security levels or come close enough that they should be used with those symmetric primitive sizes anyway.

In addition, I have chosen three more parameter sets for potential use:

- $(797, 55, 25)$ for “Category 1A”, i.e. 160-bit preimage security and 256-bit collision-resistant hashes.
- $(977, 64, 30)$ for Category 2, with a low-cost added margin of security against attacks on PKP.
- $(1789, 111, 55)$ for Category 5, with slightly smaller signatures than the 1889 parameter set.

7 Optimized signature sizes

Optimized signature sizes for PKP-DSS are listed below. The “minimal?” column indicates whether the signature parameters have minimal size given the values in the first six columns.

The signature security level “b112git” has 112-bit security against the preimage-like forgery attack on the Fiat–Shamir transform, and 160-bit collision-resistant hashes, as are currently used in the ubiquitous Git version control system. This security level is likely to remain acceptable for interactive authentication longer than Git’s SHA-1 hashes will.

q	n	m	keysec	sigsec	squished?	*bytes*	r_0	r_1	minimal?
797	55	25	Cat. 1A	b80	yes	6882	67	35	yes
797	55	25	Cat. 1A	b80	no	7197	67	35	no
797	55	25	Cat. 1A	b112git	yes	9578	99	47	yes
797	55	25	Cat. 1A	b112git	no	10001	99	47	no
797	55	25	Cat. 1A	Cat. 1	yes	12966	105	57	yes
797	55	25	Cat. 1A	Cat. 1A	yes	16190	127	73	yes
977	61	28	Cat. 1	Cat. 1	yes	13145	108	55	yes
977	61	28	Cat. 2	Cat. 1	yes	14009	108	55	yes
977	61	28	Cat. 2	Cat. 1A	yes	17523	135	69	yes
977	61	28	Cat. 2	Cat. 2	yes	20956	158	84	yes
977	64	30	Cat. 2	Cat. 1	yes	14394	108	55	yes
977	64	30	Cat. 2	Cat. 1A	yes	18006	135	69	yes
977	64	30	Cat. 2	Cat. 2	yes	21542	166	81	yes
1409	87	42	Cat. 4	Cat. 2	yes	26780	178	76	yes
1409	87	42	Cat. 4	Cat. 3	yes	30738	160	82	yes
1409	87	42	Cat. 4	Cat. 4	yes	40860	216	108	yes
1789	111	55	Cat. 5	Cat. 3	yes	34988	176	76	yes
1789	111	55	Cat. 5	Cat. 5	yes	51492	232	102	yes

The software used to generate these parameters is in [Ran20a].

8 Further directions

The design above is sufficient to demonstrate the concept and provide a signature scheme efficient enough for practical use, but some further topics should be studied:

- Investigate hash functions and/or tree hashing modes along the general lines of Rumba20 [Ber07] which combine commutative operations with hash calls. Determine whether there is any performance benefit to using one of these modes to avoid fully sorting lists of hashes.

- Investigate the effect of using commitments short enough to permit N-collision attacks [FS09, § II. 3.2. "Birthday paradox and coupon collector problem", p. 114–118][KN67] in a $q2$-based signature scheme, in particular a $q2$ protocol in which one of the cut-and-choose challenges is compressed to a random seed. I conjecture that this reduces the effective value of q by only a factor of N.

  If this is secure, evaluate whether there is any overall benefit to signature size in reducing commitment sizes, and/or what values of q would justify this tradeoff.

- Clarify the security analysis of PKP presented in [KMRP19]. Evaluate the effect on difficulty of solving the IPKP problem as used here rather than restricting to full-group homogeneous PKP.

  An analysis should also consider the sizes of physically plausible machines with sorting hardware and processing elements distributed throughout the system's memory.

- Evaluate the difficulty of solving the IPKP problem with some elements of $\mathbf{v}$ repeated.

- Find parameters for a linear code such that both IPKP and permutation equivalence are hard, if this is possible. Such a code would be potentially useful to build a signature scheme with blinded public keys.

- Look for large-modulus attacks on PKP analogous to "overstretched" lattice attacks. First, construct backdoored parameters $(\mathbf{A}, \mathbf{v})$ with $q \geq 2^{n+1}$ for which the secret key π can easily be recovered from $\mathbf{w}$, to use as an example. Then, study the extent to which these backdoors could be hidden, and conversely the probability that some random $(\mathbf{A}, \mathbf{v})$ can enable these attacks.

- Evaluate the difficulty of PKP over characteristic-2 tower fields of sizes 256, 4096, and 65536. These fields will be tempting to hardware-oriented designers, but will enable both the algebraic attacks which wrecked CTRU [Vat08] and potentially SAT-based attacks due to the large number of equations which can be obtained over $\mathbb{F}_2$. Someone should evaluate the attacks before they can have a practical impact.

- The signature sizes achieved from the 5-pass PKP protocol by this paper are slightly less than those achieved by the "Middle" parameter sets of SUSHSYFISH in [Beu20], and will likely do so with much lower signing and verification times. However, the SUSHSYFISH sizes include commitment randomization strings which provide no practical benefit, and encode each vector and permutation element by padding to a power of two.

The SUSHSYFISH approach should be re-evaluated using the same optimization techniques applied here, and considering the full range of keypair and signature security level options.

9 Acknowledgements

Ward Beullens provided helpful comments on an earlier draft, which led me to take a closer look at [DFM20]. Sakumoto, Shirai, and Hiwatari patented their MQ-based identification protocols [SSH13][SSH15], prompting me to research prior art further and find a better protocol to use instead.

References

[ABD+19] Jean-Philippe Aumasson, Daniel J. Bernstein, 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, and Peter Schwabe. SPHINCS+ [web page], 2019. `http://sphincs.org/`.

[BCL+19] Daniel J. Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, and Wen Wang. Classic McEliece: NIST submission [web page], 2019. `https://classic.mceliece.org/nist.html`.

[BCLvV18] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU Prime: reducing attack surface at low cost. In Carlisle Adams and Jan Camenisch, editors, *Selected Areas in Cryptography—SAC 2017, 24th international conference, Ottawa, ON, Canada, August 16–18, 2017, revised selected papers*, volume 10719 of *Lecture Notes in Computer Science*, pages 235–260. Springer, 2018. `https://cr.yp.to/papers.html#ntruprime`.

[BCLvV19a] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU Prime: NIST submission [web page], 2019. `https://ntruprime.cr.yp.to/nist.html`.

[BCLvV19b] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU Prime: round 2, March 2019. `https://ntruprime.cr.yp.to/nist/ntruprime-20190330.pdf`.

[BDG20] Mihir Bellare, Hannah Davis, and Felix Günther. Separate Your Domains: NIST PQC KEMs, Oracle Cloning and Read-Only Indifferentiability. Cryptology ePrint Archive, Report 2020/241, 2020. `https://eprint.iacr.org/2020/241`.

[Ber] Daniel J. Bernstein. `djbsort`. `https://sorting.cr.yp.to/`.

[Ber05] Daniel J. Bernstein. Understanding brute force. ECRYPT STVL Workshop on Symmetric Key Encryption, 2005. `https://cr.yp.to/papers.html#bruteforce`.

[Ber07] Daniel J. Bernstein. What output size resists collisions in a xor of independent expansions? Workshop Record of ECRYPT Workshop on Hash Functions 2007, 2007. `https://cr.yp.to/papers.html#expandxor`.

[Ber18] Daniel J. Bernstein. Divergence bounds for random fixed-weight vectors obtained by sorting, April 2018. `https://cr.yp.to/papers.html#divergence`.

[Beu19a] Ward Beullens. FISH. Public GitHub repository, 2019. `https://github.com/WardBeullens/FISH`.

[Beu19b] Ward Beullens. PKPDSS. Public GitHub repository, 2019. `https://github.com/WardBeullens/PKPDSS`.

[Beu20] Ward Beullens. Sigma protocols for MQ, PKP and SIS, and fishy signature schemes. Cryptology ePrint Archive, Report 2019/490, version 20200221:084011, 2020. `https://eprint.iacr.org/2019/490/20200221:084011`.

[BFK$^+$19a] Ward Beullens, Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, version 20191028:100205, 2019. `https://eprint.iacr.org/2018/714/20191028:100205`.

[BFK$^+$19b] Ward Beullens, Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, 2019. `https://eprint.iacr.org/2018/714`.

[BFK$^+$19c] Ward Beullens, Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, version 20190925:113903, 2019. `https://eprint.iacr.org/2018/714/20190925:113903`.

[Blu87] Manuel Blum. How to prove a theorem so no one else can claim it. In *Proceedings of the International Congress of Mathematicians*, pages 1444–1451, 1987. `https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.469.9048`.

[CDG$^+$20] Melissa Chase, David Derler, Steven Goldfeder, Jonathan Katz, Vladimir Kolesnikov, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Xiao Wang, and Greg Zaverucha. The Picnic Signature Scheme — Design Document — Version 2.2, 2020. file `spec/design-v2.2.pdf` in `https://github.com/Microsoft/Picnic`.

[CHR$^+$] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. MQDSS specifications [all versions]. (see [CHR$^+$17a], [CHR$^+$19], or [CHR$^+$20]).

[CHR$^+$16] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-pass MQ-based identification to MQ-based signatures. Cryptology ePrint Archive, Report 2016/708, version 20161204:155428, 2016. `https://eprint.iacr.org/2016/708/20161204:155428`.

[CHR$^+$17a] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. MQDSS specifications [version 1.0], November 2017. `http://mqdss.org/files/mqdss.pdf`.

[CHR$^+$17b] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. SOFIA: MQ-based signatures in the QROM. Cryptology ePrint Archive, Report 2017/680, version 20170718:150037, 2017. `https://eprint.iacr.org/2017/680/20170718:150037`.

[CHR$^+$19] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. MQDSS specifications [version 2.0], March 2019. `http://mqdss.org/files/MQDSS_Ver2.pdf`.

[CHR$^+$20] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. MQDSS specifications [version 2.1], April 2020. `http://mqdss.org/files/mqdssVer2point1.pdf`.

[DFM20] Jelle Don, Serge Fehr, and Christian Majenz. The Measure-and-Reprogram Technique 2.0: Multi-Round Fiat-Shamir and More. Cryptology ePrint Archive, Report 2020/282, version 20200727:092440, 2020. `https://eprint.iacr.org/2020/282/20200727:092440`.

[DPT10] Yevgeniy Dodis, Mihai Patrascu, and Mikkel Thorup. Changing Base without Losing Space. In *Proceedings of the Forty-Second*

ACM Symposium on Theory of Computing, STOC '10, page 593–602, New York, NY, USA, 2010. Association for Computing Machinery. https://cs.nyu.edu/~dodis/ps/prefix.pdf, https://doi.org/10.1145/1806689.1806771.

[FKMR$^+$18a] Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, version 20180801:194820, 2018. https://eprint.iacr.org/2018/714/20180801:194820.

[FKMR$^+$18b] Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, version 20180928:145725, 2018. https://eprint.iacr.org/2018/714/20180928:145725.

[FKMR$^+$18c] Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, version 20181201:152523, 2018. https://eprint.iacr.org/2018/714/20181201:152523.

[FKMR$^+$19] Jean-Charles Faugère, Eliane Koussa, Gilles Macario-Rat, Jacques Patarin, and Ludovic Perret. PKP-Based Signature Scheme. Cryptology ePrint Archive, Report 2018/714, version 20190410:111616, 2019. https://eprint.iacr.org/2018/714/20190410:111616.

[FS87] Amos Fiat and Adi Shamir. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO '86 Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1987. https://link.springer.com/chapter/10.1007%2F3-540-47721-7_12.

[FS09] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009. https://ac.cs.princeton.edu/home/AC.pdf.

[KKW18] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. Cryptology ePrint Archive, Report 2018/475, 2018. https://eprint.iacr.org/2018/475.

[KMRP19] Eliane Koussa, Gilles Macario-Rat, and Jacques Patarin. On the complexity of the Permuted Kernel Problem. Cryptology ePrint Archive, Report 2019/412, 2019. https://eprint.iacr.org/2019/412.

[KN67] M. S. Klamkin and D. J. Newman. Extensions of the birthday surprise. *Journal of Combinatorial Theory*, 3:279–282, 1967. https://www.sciencedirect.com/science/article/pii/S0021980067800759.

[KZ19] Daniel Kales and Greg Zaverucha. Forgery Attacks on MQDSSv2.0. NIST PQC Round 2 official comment, August 2019. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-2/official-comments/MQDSS-round2-official-comment.pdf.

[KZ20] Daniel Kales and Greg Zaverucha. An Attack on Some Signature Schemes Constructed From Five-Pass Identification Schemes. Cryptology ePrint Archive, Report 2020/837, 2020. https://eprint.iacr.org/2020/837.

[Lei18] D. F. Leichtle. Post-quantum signatures from identification schemes. Master's thesis, Universität Stuttgart, Oct 2018. https://research.tue.nl/en/studentTheses/post-quantum-signatures-from-identification-schemes.

[Pat08] Mihai Patrascu. Succincter. In *2008 IEEE 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 305–313, Los Alamitos, CA, USA, Oct 2008. IEEE Computer Society.

[Poi95] David Pointcheval. A New Identification Scheme Based on the Perceptrons Problem. In *Advances in Cryptology — Proceedings of EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 319–328, 1995. http://www.di.ens.fr/~pointche/Documents/Papers/1995_eurocrypt.pdf, http://dx.doi.org/10.1007/3-540-49264-X_26.

[PS00] David Pointcheval and Jacques Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13(3):361–396, June 2000. https://link.springer.com/article/10.1007/s001450010003, https://www.di.ens.fr/~stern/data/St75.pdf.

[Ran20a] Robert Ransom. mqdss-parameters. Public Git repository, 2020. https://gitlab.com/rransom/mqdss-parameters.

[Ran20b] Robert Ransom. pkpsig-python-ref-impl. Public Git repository, 2020. https://gitlab.com/rransom/pkpsig-python-ref-impl.

[Sch96] Bruce Schneier. *Applied Cryptography.* John Wiley & Sons, Inc., second edition, 1996.

[SF88] Adi Shamir and Amos Fiat. US 4,748,668 A — method, apparatus, and article for identification and signature. U.S. patent, May 1988. Expired 2006-07-09.

[Sha90a] Adi Shamir. An Efficient Identification Scheme Based on Permuted Kernels (extended abstract). In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO '89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 606–609. Springer, 1990. `https://link.springer.com/chapter/10.1007/0-387-34805-0_54`.

[Sha90b] Adi Shamir. US 4,932,056 A — method and apparatus for user identification based on permuted kernels. U.S. patent, June 1990. Expired 2009-03-16.

[SRC$^+$17] Simona Samardjiska, Joost Rijneveld, Ming-Shing Chen, Andreas Hülsing, Peter Schwabe, and Tomonori Okuwaki. Intellectual property statements for MQDSS, round 1, November 2017. `https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/ip-statements/MQDSS-Statements.pdf`.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. Public-Key Identification Schemes Based on Multivariate Quadratic Polynomials. In *Advances in Cryptology — CRYPTO 2011 — 31st Annual Cryptology Conference*, volume 6841 of *Lecture Notes in Computer Science*, page 703. Springer, 2011. `https://www.iacr.org/archive/crypto2011/68410703/68410703.pdf`.

[SSH13] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. US 8,522,033 B2 — authentication device, authentication method, program, and signature generation device. U.S. patent, August 2013. Active.

[SSH15] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. US 8,959,355 B2 — authentication device, authentication method, program, and signature generation device. U.S. patent, February 2015. Active.

[st20] Picnic NIST submission team. The Picnic Signature Algorithm — Specification — Version 3.0, 2020. file `spec/spec-v3.0.pdf` in `https://github.com/Microsoft/Picnic`.

[Ste94a] Jacques Stern. Designing Identification Schemes with Keys of Short Size. In Yvo G. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 164–173. Springer, 1994. `https://www.di.ens.fr/users/stern/data/St51.pdf`, `https://link.springer.com/chapter/10.1007/3-540-48658-5_18`.

[Ste94b] Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *Advances in Cryptology — CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21. Springer, 1994. `https://www.di.ens.fr/users/stern/data/St47.pdf`, `https://link.springer.com/chapter/10.1007/3-540-48329-2_2`.

[Ste96a] Jacques Stern. A new paradigm for public key identification. *IEEE Transactions on Information Theory*, 42(6):1757–1768, 1996. `https://www.di.ens.fr/users/stern/data/St55b.pdf`, `https://ieeexplore.ieee.org/document/556672`.

[Ste96b] Jacques Stern. US 5,483,597 A — authentication process for at least one identification device using a verification device and a device embodying the process. U.S. patent, January 1996. Expired 2013-12-30.

[Ste96c] Jacques Stern. US 5,581,615 A — scheme for authentication of at least one prover by a verifier. U.S. patent, December 1996. Expired 2014-12-30.

[Vat08] Nitin Vats. Algebraic Cryptanalysis of CTRU Cryptosystem. In Xiaodong Hu and Jie Wang, editors, *Computing and Combinatorics*, pages 235–244, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.