INTERNET-DRAFT
Intended Status: Experimental
Expires: [date]

J. M. Schanck
Security Innovation & U. Waterloo
W. Whyte
Security Innovation
Z. Zhang
Security Innovation

20 September 2015

Criteria for selection of public-key cryptographic algorithms for quantum-safe hybrid cryptography
draft-whyte-select-pkc-qsh-00.txt

Abstract

Authenticated key exchange mechanisms instantiated with cryptosystems based on integer factorization, finite field discrete log, or elliptic curve discrete log, are believed to be secure now but are vulnerable to a harvest-then-decrypt attack where an attacker who cannot currently break the mechanism records the traffic anyway, then decrypts it at some point in the future when quantum computers become available. The Quantum-safe Hybrid approach is a modular design, allowing any authenticated key exchange mechanism to be protected against the harvest-then-decrypt attack by exchanging additional secret material protected with an ephemeral key for a quantum-safe public key cryptographic algorithm and including that secret material in the Key Derivation Function (KDF) run at the end of the key exchange. This approach has been proposed in TLS as the Quantum-safe Hybrid handshake mechanism for Transport Layer Security protocol (QSH_TLS). This document provides a guideline to criteria for selecting public key encryption algorithms approved for experimental use in the quantum safe hybrid setting.

Status of This Memo

Table of Contents

1. Introduction

Quantum computers pose a significant threat to modern cryptography. The two most widely adopted public key cryptosystems, namely, RSA [PKCS1] and Elliptic Curve Cryptography (ECC) [SECG], will be broken by general purpose quantum computers. RSA is adopted in TLS from Version 1.0 to TLS Version 1.3 [RFC2246], [RFC4346], [RFC5246], [TLS1.3]. ECC is enabled in RFC 4492 [RFC4492] and adopted in TLS version 1.2 [RFC5246] and 1.3 [TLS1.3]. Those two primitives are the only public key cryptography that TLS relies on.

Although these algorithms are currently believed to be secure, data encrypted using these algorithms is vulnerable to a "harvest-then-decrypt" attack where an attacker who cannot currently break the mechanism records the traffic anyway, then decrypts it at some point in the future when quantum computers become available. See section 2 for a detailed account of those attacks.

The Quantum-safe Hybrid approach, which has a concrete proposal as the Quantum-safe Hybrid handshake for Transport Layer Security protocol (QSH_TLS) [QSHTLS], addresses this attack by introducing a quantum-safe public key encapsulation mechanism along with the classical authenticated handshake. QSH_TLS is a modular design that allows in principle for any quantum-safe encryption algorithm to be used in the hybrid approach.

Since the IETF has not yet designated a single algorithm for use to provide quantum-safety, and since the quantum-safe algorithm used is intended to enhance security rather than being the only source of security, it is appropriate for there to be multiple algorithms that may be used in a quantum-safe hybrid setting. This provides an opportunity for implementers to compare different quantum-safe algorithms before the choice of a single one becomes vital. However, an algorithm should clearly satisfy some baseline set of criteria before it is approved for use in the quantum-safe hybrid setting, even if those criteria are more relaxed than they would be for selecting a single algorithm to rely on.

This document specifies what those criteria are.

The remainder of this document is organized as follows. Section 2 provides necessary background of the modular design of quantum-safe handshake for TLS. Section 3 specifies selection criteria. Section 4 provides a preliminary list of recommended encryption algorithms. Section 5 describes IANA considerations.

This is followed by the lists of normative and informative references cited in this document, the authors' contact information, and

statements on intellectual property rights and copyrights.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Well-known abbreviations and acronyms can be found at RFC Editor Abbreviations List [REAL].

2. Background

2.1. Quantum Attacks on Cryptosystems

If there exists a general purpose quantum computer, any cryptosystem that is built on top of the mathematical hard problems of integer factorization, finite field discrete logarithm (DL), or elliptic curve discrete logarithm (ECDL) will be vulnerable. This includes RSA, DSA, DH, ECDH, ECDSA and other variants of these ciphers, including variants currently under consideration for standardization by CFRG and all public key cryptosystems used in TLS. A quantum computer may allow a real-time attack on the authentication within a handshake protocol, or may allow an attacker to decrypt previously recorded network traffic.

It is not clear when quantum computers will become available. The EU has expressed in their Horizon 2020 project a desire for systems to be "quantum-ready" by 2020 [H2020]. Research groups have optimistically predicted practical and powerful quantum computer could become available by the same date [TPM15], which may be large enough to solve some instances of the elliptic curve discrete log problem that are currently secure. It is, however, clear that data exchanged today may be vulnerable to the harvest-then-decrypt attack described below.

2.1.1. Shor's algorithm

Many of the hard problems used in public key cryptography can be reduced to the Hidden Subgroup Problem over a finite cyclic group. For an finite cyclic group G and finite set X, a function f : G -> X is said to hide a subgroup H if f(a) = f(b) iff a - b is in H. The Hidden Subgroup Problem (HSP) is to determine the hidden subgroup H given black box access to f. Shor's algorithm [SHOR97] is a probabilistic quantum algorithm that solves the HSP over any finite cyclic group in polynomial time. Among the problems that reduce to the HSP are the integer factorization and discrete logarithm problems that underly RSA, DSA, DH, ECDH, and ECDSA, hence all of these systems are vulnerable to quantum attacks.

2.1.2. Grover's algorithm

Grover's algorithm [GROV96] is a probabilistic quantum algorithm that finds the unique input to a black box function that produces a particular output value. Compared with classical algorithms, Grover's algorithm finds the solution with a quadratic boost, i.e., within O(N^(1/2)) evaluations of the function, where N is the size of the function's domain.

While an exact cost analysis of Grover's algorithm will depend crucially on architecture dependent parameters that are not currently available, it is a common belief among cryptographers that Grover's algorithm is effective against symmetric primitives [BRA98]. To be conservative we ignore constant factors and simply assume that Grover's algorithm finds preimages quadratically faster than classical brute force search. Likewise, we assume that Grover's algorithm reduces the time required to recover the key of a symmetric cipher by a quadratic factor. As an example, AES-256 provides 256 bits of security against classical computers, but is assumed to provide only 128 bits of security against quantum computers.

2.2. Harvest-then-decrypt attack

The harvest-then-decrypt attack is a straightforward yet effective attack. In such an attack, the attacker stores encrypted data for long periods of time until legal, technological, or cryptanalytic means become available for revealing keys.

Under the context of quantum computing, this attack becomes extremely powerful. TLS relies on RSA and ECC, both will be broken when quantum computer becomes available. Hence, any data encrypted now will be vulnerable to this attack. It is likely that it will be some time before breaks become so cheap that all harvested traffic can be decrypted. However, this is little consolation to the people whose traffic is initially targeted for decryption. It seems prudent to provide protection against the harvest-then-decrypt attack natively to secure data exchange protocols as soon as possible.

2.3. Quantum-safe hybrid approach

The quantum safe hybrid approach defeats the quantum harvest-then-decrypt attack by introducing a second quantum-safe cryptographic primitive running in parallel with existing handshake approaches. This measure assures that when the classical cryptography fails, the attacker still need to break the corresponding quantum-safe encryption algorithm.

It is easy to see that this approach is at least as strong as the

stronger primitive of classical cryptography and the quantum-safe cryptography in the pre-quantum world [QSHTOR]. Therefore, it is an ideal approach to migrate into quantum-safe cryptography for TLS as it does not reduce the security guarantees that TLS is already delivering; in the meantime, it allows for trial usage of quantum-safe algorithms and protects data against the aforementioned harvest-the-decrypt attack.

2.4. Symmetric algorithm

For 128 bit security, implementations of a quantum-safe hybrid approach SHOULD use a symmetric algorithm with a 256-bit key, but MAY use a symmetric algorithm with a 128-bit key for interoperability or performance reasons.

2.5. Random bit generation

For 128 bit security, implementations of a quantum-safe hybrid approach SHOULD ensure that any Deterministic Random Bit Generator (DRBG) used in key generation or encryption for a quantum-safe primitives is instantiated with at least 256 bits of entropy from a secure random source.

3. Selection Criteria

The hybrid approach is a modular design, which, in order to support various quantum-safe algorithms, does not recommend any specific quantum-safe encryption algorithm. In this section we give guidelines for selecting encryption algorithms that are suitable for experimental use in the hybrid approach.

3.1. Similar work

To date, multiple groups have been involved in the work of evaluating quantum-safe encryption algorithms.

o   The National Security Agency of the United States has announced a plan to migrate to quantum-safe cryptography [NSA15].

o   The ETSI Quantum-Safe Cryptography (QSC) Industry Specification Group (ISG) aims to assess and make recommendations for quantum-safe cryptographic primitives and protocols, taking into consideration both the current state of academic cryptology and quantum algorithm research, as well as industrial requirements for real-world deployment [ETSIQ].

o   The Secure Architectures of Future Emerging Cryptography (SAFEcrypto) project [SAFEC], supported by H2020 project, focuses

on practical implementation of quantum-safe encryptions algorithms, particularly lattice-based public key cryptography.

- The Post-quantum cryptography for long-term security (PQCRYPTO) group, also supported by H2020 project, has made their initial recommendations of long-term secure post-quantum systems [PQCRY].

Note that PQCRYPTO is the only group that has made initial recommendations on quantum-safe cryptography.

This document describes criteria for quantum-safe encryption algorithms, with a focus on those algorithms existing today and suitable for transitional use until the quantum era. The intent is ultimately to align with other industry groups while enabling earlier deployment of algorithms that can reasonably be expected to make things better, not worse.

3.2. Mandatory aspects

Algorithms to be considered by quantum-safe hybrid approach MUST meet the following criteria.

3.2.1. Security levels

The candidate algorithm MUST provide 128 bit security in the quantum-safe setting.

If the candidate algorithm is subject to decryption failures, these MUST happen with a probability of less than 2^-74 (such that 128 billion devices (2^7 * 2^30) each initiating 128 billion connections (2^7 * 2^30) will with high probability encounter no decryption failures). Note that an attacker will be able to create invalid messages that do not decrypt correctly, so an implementation will have to correctly handle this failure case even when the chance of a decryption failure is negligible on a valid message (or even when this chance is zero).

3.2.2. Freely available specifications of the algorithm

The candidate algorithm MUST have a set of publicly accessible documents specifying common techniques and implementation choices. The documents MAY be Internet Drafts. Topics MUST include:

- Cryptographic primitives: the building blocks for a secure cryptographic scheme;

- Cryptographic schemes: complete sequences of operations for performing secure cryptographic functions;

- Supported parameter choices: specific selections of approved sets of values for cryptographic parameters;

- Classical security levels for the proposed parameter sets;

- Argument for post-quantum security levels for the proposed parameter sets;

- Encoding of cryptographic data items: specifies encoding/decoding of public keys and ciphertexts.

In addition, it MAY include relevant information to assist in the development and interoperable implementation, including:

- Security considerations;

- Open issues.

3.2.3. Freely available source code for a reference implementation

It is important to have a stable reference implementation available for the candidate algorithm. The code needs to be rigorously tested and reviewable. A poor implementation of a good cryptosystem can be as harmful as a broken cryptosystem.

The implementation SHOULD also be open source to allow for public auditing. In particular, any default choice of parameters MUST be justified.

3.3 Desirable aspects

The following aspects are desirable. Algorithms that meet those criteria are preferred.

3.3.1. SUPERCOP implementation

System for Unified Performance Evaluation Related to Cryptographic Operations and Primitives (SUPERCOP) [SUPEC] is a toolkit developed by the Virtual Application and Implementation Research (VAMPIRE) Lab for measuring the performance of cryptographic software. The latest release of SUPERCOP measures the performance of hash functions, secret key stream ciphers, public key encryption systems, public key signature systems, and public key secret sharing systems.

The candidate algorithm MAY have a reference implementation for

SUPERCOP. Performance of the implementation on SUPERCOP MAY be taken into consideration when selections are made.

3.3.2. Constant-time implementation

An implementation of a cryptosystem is constant-time means that the time for encryption/decryption functions is constant, regardless of the input and the output of the functions. As an example, the time to decrypt any valid ciphertext should use a same time as decrypting an invalid ciphertext and producing a decryption error. Constant time implementation is important for cryptography as it makes side-channel attacks substantially harder.

Algorithms with provable constant time implementations SHOULD be preferred. However, this is not an absolute requirement as the QSH setting uses ephemeral keys and an implementation of QSH SHOULD only decrypt once with any key, so an attacker is unlikely to gain sufficient information from the time of a single decryption to recover the plaintext.

3.3.3. Standardization

The candidate algorithm MAY be standardized by another standards body, such as ANSI X.9, IEEE, or ETSI. Algorithms that maintain creditability among multiple standards bodies SHOULD be preferred.

3.3.4. Patent and IP related issues

The candidate algorithm MAY be either non-patented or patented but with FRAND (Free or Reasonable and Non-Discriminatory) licensing statement made and all relevant IETF IP declarations provided.

4. Recommendations, justifications and considerations

4.1. Preliminary list of recommendations

The following list is an (incomplete) list of recommended quantum-safe encryption algorithms and parameters for 128 bits security that MAY be considered in the hybrid approach.

- NTRUEncrypt lattice-based encryption scheme with parameter sets ees443ep1, ees587ep1, and ees743ep1 as defined in [EESS1];
    - Specification: [EESS1] provides a concrete specification of NTRUEncrypt with parameter set ees443ep1, ees587ep1, and ees743ep1, including primitives, syntax, reference to classical security analysis in [HOF15], quantum security analysis, and encode/decode mechanisms

- Open-sourced reference implementation: Available from [NTRU-GIT]

- SUPERCOP implementation: Available for related parameter sets, not yet available for the recommended parameter sets

- Constant-time implementation: Not yet available

- Standardization: The recommended parameter sets have not been published in a standard, but the encryption scheme and other parameter sets have been standardized in IEEE 1363.1 and ANSI X9.98.

- Patent and IP Issues: NTRUEncrypt is subject to patents held by Security Innovation. Security Innovation has provided IPR Declaration 2588 to IETF.

4.2. Schemes under consideration

The following schemes are under consideration. They are well known quantum safe encryption algorithms in the literature. However, due to the lack of specifications, implementation of those schemes are non-trivial. For this reason we list those schemes as "under consideration". They will be promoted to the recommendation list once detailed specifications are provided.

- Learning with error lattice-based encryption scheme [REG05], with parameter set form [LIN11];

- NTRUEncrypt lattice-based encryption scheme instantiated with learning with error problem [STE11];

- McEliece code-based encryption scheme [MCELI] with parameter set for McBits [MCBIT];

- McEliece code-based encryption scheme with Quasi-cyclic Moderate Density Parity-Check (MDPC) codes [MDPC].

5. IANA Considerations

This document does not establish any new IANA registries, nor does it add any entries to existing registries.

6. References

6.1. Normative References

[BER09]   Bernstein, D., "Cost analysis of hash collisions: Will quantum computers make SHARCS obsolete?", SHARCS'09. <http://cr.yp.to/hash/collisioncost-20090823.pdf>

[BRA98]   Brassard, G., Hoyer, P., and Tapp, A., "Quantum cryptanalysis of hash and claw-free functions", LATIN'98: Theoretical Informatics.

[EESS1]   Consortium for Efficient Embedded Security, "Efficient Embedded Security Standard #1: Implementation Aspects of NTRUEncrypt", March 2015. <https://github.com/NTRUOpenSourceProject/ntru-crypto/raw/master/doc/EESS1-2015v3.0.pdf>

[ETSIQ]   ETSI White Paper No. 8, "Quantum Safe Cryptography and Security: An introduction, benefits, enablers and challenges", June 2015.

[GROV96]   Grover, L., "A fast quantum mechanical algorithm for database search", STOC 1996.

[H2020]   Lange, T., "PQCRYPTO project in the EU", April, 2015. <http://pqcrypto.eu.org/slides/20150403.pdf>

[HOF15]   Hoffstein, J., Pipher, J., Schanck, J., Silverman, J., Whyte, W., and Zhang, Z., "Choosing Parameters for NTRUEncrypt", 2015. <https://eprint.iacr.org/2015/708>

[LIN11]   Lindner, R., and Peikert, C., "Better Key Sizes (and Attacks) for LWE-Based Encryption", 2011.

[MCBIT]   Bernstein, D., Chou, T., and Schwabe, P., "McBits: Fast Constant-Time Code- Based Cryptography", 2013.

[MCELI]   McEliece, R., "A Public-Key Cryptosystem Based On Algebraic Coding Theory", 1978.

[MDPC]   Misoczki, R., Tillich, J., Sendrier, N., and Barreto, P., "MDPC-McEliece: New McEliece variants from Moderate Density Parity-Check codes", 2013.

[NSA15]   NSA, "NSA Suite B Cryptography", Aug 19, 2015. <https://www.nsa.gov/ia/programs/suiteb_cryptography/>

[NTRU-GIT] https://github.com/NTRUOpenSourceProject/NTRUEncrypt

[PKCS1]   RSA Laboratories, "PKCS#1: RSA Encryption Standard version 1.5", PKCS 1, November 1993

[PQCRY]   PQCRYPTO, "Initial recommendations of long-term secure post-quantum systems". <http://pqcrypto.eu.org/docs/initial-recommendations.pdf>

[QSHTLS]   Schanck, J., Whyte, W., and Zhang, Z., "Quantum-Safe Hybrid (QSH) Ciphersuite for Transport Layer Security (TLS) version 1.3", draft-whyte-qsh-tls13-00, July 2015.

[QSHTOR]   Schanck, J., Whyte, W., and Zhang, Z., "A quantum-safe circuit-extension handshake for Tor", March 2015. <https://eprint.iacr.org/2015/287>

[REAL]   "RFC Editor Abbreviations List", September 2013, <https://www.rfc-editor.org/rfc-style-guide/abbrev.expansion.txt/>.

[REG05]   Regev, O., "On lattices, learning with errors, random linear codes, and cryptography", 2005.

[RFC2119]   Bradner, S., "Key Words for Use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC2246]   Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[RFC2434]   Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 2434, October 1998.

[RFC4346]   Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[RFC4492]   Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.

[RFC5246]   Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[SAFEC]   Secure Architectures of Future Emerging Cryptography (SAFEcrypto). <http://www.safecrypto.eu/>

[SHOR97]   Shor, P., "Polynomial-time algorithms for prime factorization and discrete logarithm problems", SIAM J. Computing 26 (1997), 1484-1509. <http://www.research.att.com/~shor/papers/QCjournal.pdf>

[STE11]   Stehle, D., and Steinfield, R., "Making NTRUEncrypt and NTRUSign as secure as worst-case problems over ideal lattices", 2011.

[TLS1.3]   Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-05, March 2015.

[TPM15]   Morgan, T., "Google Sees Long, Expensive Road Ahead For Quantum Computing", July 2015. <http://www.theplatform.net/2015/07/22/google-sees-long-expensive-road-ahead-for-quantum-computing/>

6.2. Informative References

[RFC4366]   Blake-Wilson, S., Nysrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 4366, April 2006.

[RFC5990]   Randall, J., Kaliski, B., Brainard, J. and Turner S., "Use of the RSA-KEM Key Transport Algorithm in the Cryptographic Message Syntax (CMS)", RFC 5990, September 2010.

[RFC5859]   Krawczyk, H., Eronen, P., "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5859, May 2010.

Authors' Addresses

John M. Schanck
Security Innovation, US
and
University of Waterloo, Canada
jschanck@securityinnovation.com

William Whyte
Security Innovation, US
wwhyte@securityinnovation.com

Zhenfei Zhang
Security Innovation, US
zzhang@securityinnovation.com

Copyright Notice