Performance Comparisons of Post-Quantum Cryptography: A Case Study of IPsec

Alice Ekblad
Linköping University
Linköping, Sweden
aliek496@student.liu.se

Casper Jensen
Linköping University
Linköping, Sweden
casje896@student.liu.se

ABSTRACT

This report is inspired by a published article about the transition to post-quantum cryptography in the context of DNSSEC by Müller et al. (2020). However, the focus of this report is the IPsec protocol instead of the DNSSEC. This report use the tests used in the earlier mentioned article for two PQC algorithms and rewrites them to test for different message sizes. This is to test for the PQC algorithms potential adaptiation to IPsec, that uses different kinds of message sizes. The PQC algorithms that are used in this report are Falcon-512 and Rainbow-I$_a$.

1 INTRODUCTION

In the future, quantum computers have the potential to crack cryptographic algorithms much faster than is possible to this day. This means that some of our today considered safe and unbreakable cryptographic algorithms used in Internet protocols may be at risk. The National Institute of Standards and Technology[1] (NIST) has initiated a process to develop, test, and standardize so called quantum-safe algorithms, that is post-quantum cryptographic (PQC) algorithms, and there is currently research taking place to prepare for and ease the transition to these new algorithms.

This report is inspired by an earlier published article about the transition to post-quantum cryptography in the context of DNSSEC by Müller et al. [1]. With their article, Müller et al. published performance tests for PQC in DNSSEC, which we in this report have tried to modify to be applied for IPsec instead. The already existing performance tests published for DNSSEC is testing three different PQC algorithms; Rainbow-I$_a$, Falcon-512 and RedGeMSS128, these algorithms will also be used for the IPsec performance tests. Beyond this, the report will research about the future of PQC for IPsec.

The choice was made to focus on PQC for IPsec because there is not currently much research available on the topic, and because IPsec is a widely used protocol suite to ensure data sent over public networks is secure. As an example, IPsec is often used to set up secure virtual private networks (VPNs), which are an essential tool for many.

Research questions. In this report the following set of problems and questions are addressed:

(1) What requirements and prerequisites exist for the IPsec protocol regarding transitioning to post-quantum cryptography algorithms?
(2) Can the published PQC performance test programs for DNSSEC from Müller et al. [1] be compiled, run, and modified in a useful way?
(3) If the answer to the second question is yes, what PQC algorithms could be suitable for IPsec based on the results of performance tests?

Delimitations. In regards to the limited time to conclude this report, delimitations has been made. Firstly, only one part of what IPsec does as a protocol suite is treated in this report; that is ensuring data integrity and origin authentication (digital signatures). The other part of IPsec, data confidentiality, is not touched on in this report (public-key encryption and key-establishment).

Further, only a small subset (two to be precise) of the PQC algorithms for digital signatures available from NIST[2] is used in the performance tests in this report. This is because the ones used here already had basic performance test programs available through the published article by Müller et al. [1], which was mentioned earlier.

2 RELATED WORK

As mentioned in the introduction, this report is based on a paper by Müller et al. [1] that is discussing about PQC performance test programs for DNSSEC. The paper is performing a case study that analyzes the impact of PQC on DNSSEC.

Another paper that has a similar approach is wtritten by van Heesch et al. [2]. In that paper they have implemented and evaluated PQC algorithms in OpenVPN and over HTTPS by using an adapted version of OpenSSL. Both OpenVPN and HTTPS uses TLS to set up secure channels and therefore the work they have done can be relevant for other software solutions that uses TLS. The performance tests were focused on CPU and network overhead.

[1] https://www.nist.gov/

[2] https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions

There is another paper by Eric Crockett et al. [3] that is discussing the implementation of PQC for two major internet security protocols. Namely the Transport Layer Security (TLS) and Secure Shell (SSH) protocols. The article explores the possibility of integrating post-quantum and hybrid key exchange and authentication into communication protocols generally and especially into TLS and SSH.

3 BACKGROUND

In this section, relevant topics are briefly described to aid understanding of the results of this report. First, the basics of PQC is explained, followed by a brief description of the IPsec protocol (suite). Lastly, the most significant concepts of cryptography in the context of this report is explained.

3.1 Post-quantum cryptography

Post-quantum cryptography refers to cryptographic algorithms that are thought to be secure against future quantum computers. It is believed that an ample quantum computer running Shor's algorithm [4] will be able to solve several of the complex mathematical problems that some of today's algorithms take advantage of to ensure their security. PQC algorithms can be grouped into four groups; lattice-based, multivariate, hash-based, and code-based cryptography [5].

Lattice-based cryptography is a term used for cryptographic algorithms that involves lattices. Many lattice based algorithms have their security based on the assumption that lattice problems cannot be solved efficiently [6, 7].

Multivariable polynomial cryptography is a set of algortihms that relies on the difficulty of solving the multivariable polynomial algorithm over finite fields [7].

Hash-based cryptography is used for digital signatures and can resist quantum computer attacks becuase they are based on the security properties of crypto hash functions. More exaclty, the collision resistance and pre-image resistance [7].

Code-based cryptography is a set of algorithms that are based on error-corecting to construct a one-way function. It is relying on the hardness of decoding a message that contains random errors in it and still recover the code structure [8].

3.2 IPsec

Internet Protocol Security (IPsec) is a protocol suite that in a secure way authenticates and encrypts packets of data that is communicated between two clients in an Internet network. The Internet Protocol (IP) is not a part of this suite, but IPsec runs directly on top of IP. IPsec has two modes it can operate in; transport mode and tunnel mode.

Transport mode provides host-to-host encrypted traffic. If two hosts has established a connection, they can securely send data between each other. However, in transport mode the IP header is visible, which means that the routing will be intact, and the final destination of a packet will be known to intermediary routers.

Tunnel mode provides both secure connections for host-to-host, host-to-network, and network-to-network communications. Just as in transport mode the data between two entities will be secured on the way, but in addition to this the original IP header will be encrypted. This means that intermediary routers will not have access to final destination of packets, only temporary addresses placed in an outer header is used by them to know where to route the packet forward. Tunnel mode is used for VPNs [9].

IPsec uses several protocols to perform various actions. They are responsible for different parts of the process of providing a secure connection between two entities, and together they make up IPsec.

Authentication Headers (AH) ensures data integrity and data origin authentication, by the usage of a hash function and a shared secret key in its algorithm. However, nothing is encrypted by the AH [9].

Encapsulating Security Payload (ESP) also provides data integrity and data origin authentication, but foremost it assures data confidentiality through encryption. In transport mode, ESP only encrypts the payload of the IP packet, while the header remains visible. However, in tunnel mode ESP encapsulates the whole IP packet, including the (inner) header, and adds an (outer) header which is visible [9].

Security Association (SA) is the mechanism that negotiates and establishes cryptographic algorithm, encryption keys, and hash function for integrity. This process takes place before any data can be sent securely over the network [9].

3.3 Signing and Verification

Signing in short means that something is proven to be owned by someone. For example that a document has been owned or written by someone and then they sign it to make sure that it is proven that they signed it [10]. It works in such a way that the document also has a hash digest that is hashed using a hash algorithm with the senders private key, then the recipient receives the document, hash digest and the senders public key. The receiver then hash the message using the public key and if the resulting digest matches the one received then it is proven that it is the sender who signed it, this is what **verification** means.

4 METHOD

To answer Question 1, a literature study was conducted. How it was performed is described in this section. Also described is the performance tests that were created to answer Question 2 and Question 3.

4.1 Literature study

To derive requirements for transitioning IPsec to PQC, material on the subject was gathered and worked through. The focus has been to read published articles from acknowledged conferences within the area of security, networking, and cryptography, as well as official publications from large authorities and IETF[3] documentation (RFCs). Much material have been taken from official publications from NIST[4], along with RFCs for IPsec, protocols within the IPsec suite, and cryptographic algorithms used by IPsec.

4.2 Performance tests

In the NIST PQC project[5] there are two main groups of algorithm candidates; public-key encryption and key-establishment algorithms, and algorithms for digital signatures. For this report, the three PQC algorithms that Müller et al. [1] published tests for was initially considered; Rainbow-I_a[6], Falcon-512[7] and RedGeMSS128[8]. After trying to compile and run the original tests, it was decided that RedGeMSS128 would not be considered further due to complications of compiling that test. Both Rainbow-I_a and Falcon-512 are third round finalists of the NIST standardization process [11], and both are for digital signatures.

The original test programs were modified to test performance of signing and verification for varying message sizes. An overview of the modified tests that was used for this report is shown in Figure 1. The test programs for Rainbow-I_a and Falcon-512 had the same structure (the difference between them being the algorithms for creating key pair, signing, and verification). The performance tests was run on a machine with four cores equipped with Intel Core i5-7500T CPU (2.70GHz), 16GB RAM, running Ubuntu 20.04.4 LTS.

The actual test part of the program, the darker grey section of Figure 1, started with creating a key pair, followed by generation of a random message with a given length. The message is a char array of randomized chars. Then signing of the message is performed for ten seconds, where only signings per second is the value pasted forward. Lastly, the same message is prepared for verification before being verified for ten seconds. Again, only verifications per second is passed forward to the main part of the test program. This process was executed eleven times with a different message size each time, that started on 21 bytes with an increment of about 6550 bytes each round up to 65520 bytes for the final round.

[3]https://www.ietf.org/
[4]https://www.nist.gov
[5]https://csrc.nist.gov/Projects/Post-Quantum-Cryptography
[6]https://www.pqcrainbow.org/
[7]https://falcon-sign.info/
[8]https://www-polsys.lip6.fr/Links/NIST/GeMSS.html

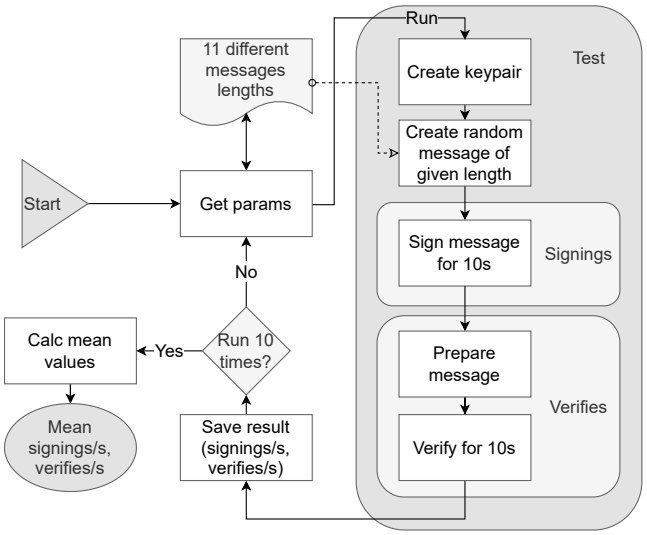

Figure 1: High-level overview of the modified test programs.

5 IPSEC REQUIREMENTS

It has been difficult to derive exact and definite requirements that exist on IPsec as a suite, regarding signing and validation speed. The requirements described next should perhaps best be seen as estimations that we discuss rather than hard requirements.

Key size. For IPsec encryption, NIST[9] currently recommends four algorithms to be used; AES-GCM, AES-CTR, AES-CBC, and AESCCM [12]. Those allow for 128-bit, 192-bit, and 256-bit keys. 128-bit seems to be the default as of now, but because IPsec sessions can have a long lifetime (recommended eight hours [12, 13]) and carry multiple packets, it could be considered favourable to move towards 256-bit keys as standard in the long run [14].

Signature size. NIST currently recommends HMAC-SHA256, HMAC-SHA384, HMAC-SHA512, and AES-GMAC as algorithms to ensure integrity with IPsec. HMAC-SHA256/384/512 generates signatures of size 32 bytes, 48 bytes, and 64 bytes respectively [15].

Data unit size. The recommended and typical maximum transmission unit (MTU) for IPsec is 1500 bytes [13, 16]. However, there exist recommendations from entities to set it to a value no more than 1360 bytes [13], to leave space for a potential IPsec encapsulation overhead. Given that, it should be noted that IPv6 requires a MTU of 1280 bytes or more [16].

[9]https://www.nist.gov/

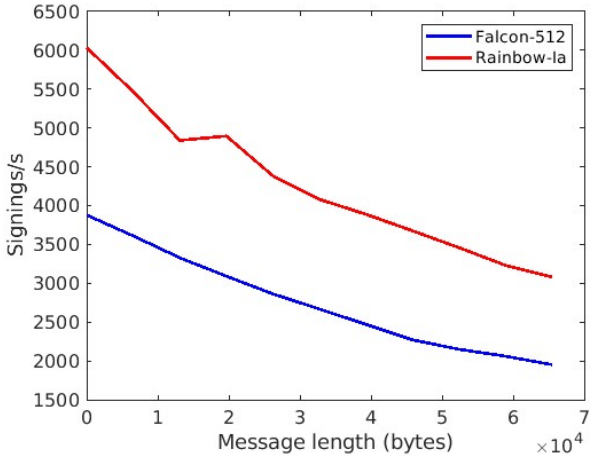

Figure 2: Number of signings/s for different message sizes.

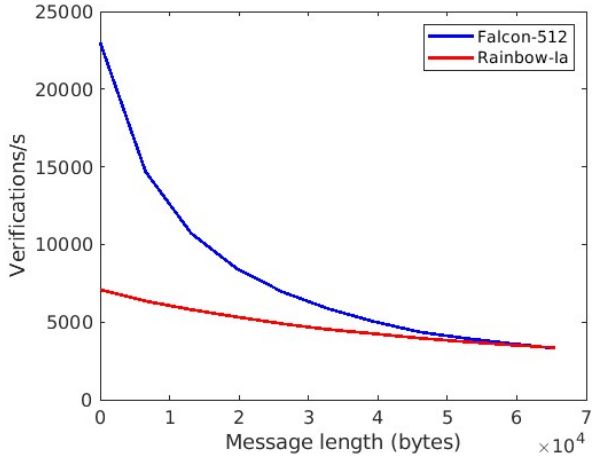

Figure 3: Number of verifications/s for different message sizes.

6 PERFORMANCE TEST RESULTS

In this section, the performance test results are presented. First the Falcon-512-specific results are described, followed by results specific for Rainbow-I_a.

6.1 Falcon-512

In the Falcon-512 test there is a clear indication that both the signing speed and the verification speed slows down with the increase of message size. As seen in Figure 2 the speed of signings (signings/s) decreases linearly when the message size increases. The signings/s decreased 49.7% when the message size went from 21 bytes to 65520 bytes.

For the number of verifications/s it is obvious from Figure 3 that the number of verifications per second decreases exponentially. There is a steep decrease of verifications/s in the beginning and then the decrease slows down as the message size increases. The verifications/s decreased 85.6% when the message size increased.

6.2 Rainbow-I_a

The Rainbow-I_a test differed some from the Falcon-512 test with the Rainbow-I_a algorithm having a higher number of signings/s overall but the number of verifications/s is lower than for the Falcon-512 algorithm. As seen in Figure 2 the number of signings/s decreases linearly from 6000 to 3080, meaning a decrease of 49.0% when the message size increased from 21 bytes to 65520 bytes.

When studying the verifications/s for the Rainbow-I_a algorithm it is seen in Figure 3 that the decrease in speed is almost linear. The decrease is from 7089 to 3352, meaning a decrease of 52.7% when the message size increased.

7 DISCUSSION

When comparing the results from Subsection 6.1 and Subsection 6.2 the first thing to notice is that the decrease in signings/s is linear for both the Falcon-512 and Rainbow-I_a algorithms. The difference is that Rainbow-I_a have a higher starting speed than Falcon-512, almost 2000 signings/s higher. Then the decrease is linear for both of the algorithms with the exception that around 15 000 to 20 000 byte message size, the signings/s increases a little bit to then later continue to decrease linearly. Since the data the plot is made of is the mean values for 10 different runs it is significant that this "bump" is showing since it means that it appears regularly. Why this happens is hard to know.

However, when looking at the starting speed of verifications/s it is much higher for the Falcon-512 algorithm with it beginning at 23000 verifications/s in comparison to the 7000 that Rainbow-I_a start at. The large difference is how the speeds decrease when the message size increases. For the Rainbow-I_a algorithm, the speed of the verifications decreases linearly to around 50% of the starting value. For the Falcon-512 algorithm on the other hand, its speed decreases exponentially with a total decrease of 85.6%, but still is faster than the Rainbow-I_a algorithm for large message sizes.

The first main reason for the large difference in performance for the two different algorithms is that Falcon-512 is a lattice-based cryptographic system and Rainbow-I_a is a multivariable public key crypto system (same category as multivariable polynomial cryptography as described in Subsection 3.1).

8 CONCLUSION

To answer the research questions stated in Section 1, the conclusions are as follows.

Question 1: It was difficult to derive assertive requirements and prerequisites for transitioning IPsec as a protocol suite to PQC. Currently used algorithms in IPsec can be looked at in order to reason about what is acceptable, at this time, regarding key, signature, and data unit size.

Question 2: Yes, two out of the three published PQC performance tests could be recreated and even modified in a way to explore different message sizes for signing and verification.

Question 3: It is hard to answer. Both algorithms could work for the IPsec protocol, they are a little bit opposite to each other with one being faster at signings and the other at verifications. However, for other parts of the IPsec protocol, such as encrypting information or key-exchange, these algorithms are not made for it.

An important note is that according to a very recent article in the Cryptology Eprint Archive [17], the Rainbow PQC algorithm is breakable. However, this is very recent news and there still needs to be more research done.

8.1 Future work

As mentioned as delimitations in Section 1, only the data integrity and origin authentication (digital signatures) part of IPsec has been treated in this report. Therefore, performing performance tests for public-key encryption and key-establishment PQC algorithms is of high interest. Additionally, since only a subset of the available digital signature PQC algorithms was tested in this report, doing so on the rest of them is also of interest.

The performance tests conducted in this report only tests how efficient the used PQC algorithms is on signing and verifying digital signatures for different message sizes in a certain range, so expanding the test surface is of great interest. Concerning transitioning the IPsec protocol suite to PQC, clear and distinct requirements on its cryptographic features (e.g. signature and key size) has to be determined in order to make an educated and successful choice of PQC algorithms when the time comes to transition.

REFERENCES

[1] Moritz Müller et al. "Retrofitting post-quantum cryptography in Internet protocols: A case study of DNSSEC". In: *ACM SIGCOMM Computer Communication Review* 50.4 (2020), pp. 49–57.

[2] Maran van Heesch et al. "Towards Quantum-Safe VPNs and Internet". In: *Cryptology ePrint Archive* 1277 (2019).

[3] Eric Crocket et al. "Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH". In: *NIST 2nd Post-Quantum Cryptography Standardization Conference 2019* (2019).

[4] Peter W Shor. "Polynomial time algorithms for discrete logarithms and factoring on a quantum computer". In: *International Algorithmic Number Theory Symposium*. Springer. 1994, pp. 289–289.

[5] Daniel J Bernstein. "Introduction to post-quantum cryptography". In: *Post-quantum cryptography*. Springer, 2009, pp. 1–14.

[6] *Lattice-based cryptography*. 2022. URL: https://en.wikipedia.org/wiki/Lattice-based_cryptography.

[7] "NIST Post-Quantum Cryptography, A Hardware Evaluation Study". In: *Journal of Cryptography, International Association for Cryptologic Research (IACR)* 047 (2019). URL: https://eprint.iacr.org/2019/047.pdf.

[8] G Ganesan C Balamurugan K Singh and M Rajarajan. "Post-Quantum and Code-Based Cryptography—Some Prospective Research Directions". In: *MDPI, Cryptography* 5 (2021). URL: https://doi.org/10.3390/cryptography5040038.

[9] Steven Bellovin. *Guidelines for Specifying the Use of IPsec Version 2*. RFC 5406. 2009. URL: https://www.rfc-editor.org/info/rfc5406.

[10] *What is the difference between Encryption and Signing? Why should you use digital signatures?* 2022. URL: https://www.encryptionconsulting.com/education-center/encryption-and-signing/.

[11] Gorjan Alagic et al. "Status report on the second round of the NIST post-quantum cryptography standardization process". In: *US Department of Commerce, NIST* (2020).

[12] Elaine Barker et al. "Guide to IPsec VPNs". In: *NIST Special Publication* (2020).

[13] Forcepoint. *Forcepoint IPsec Advanced: Configuration Guide*. 2022. URL: https://www.websense.com/content/support/library/web/hosted/ipsec_advanced/ipsec_advanced.pdf.

[14] Paul Wouters et al. *Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)*. RFC 8221. 2017. URL: https://www.rfc-editor.org/info/rfc8221.

[15] Sheila Frankel and Scott G. Kelly. *Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec*. RFC 4868. 2007. URL: https://www.rfc-editor.org/info/rfc4868.

[16] Bob Hinden and Dr. Steve E. Deering. *Internet Protocol, Version 6 (IPv6) Specification*. RFC 2460. 1998. URL: https://www.rfc-editor.org/info/rfc2460.

[17] Ward Beullens. "Breaking Rainbow Takes a Weekend on a Laptop". In: 214 (2022).