PQ-WireGuard: we did it again.

NIST Third PQC Standardization Conference

Mathilde Raynal[1,2] Aymeric Genêt[1,2] Yolan Romailler[1]

[1]Kudelski Security

[2]EPFL

Introduction

WireGuard

WireGuard is a fast and secure VPN solution[1] using “modern” but **not quantum-resistant** cryptography. It features:

- authentication,
- identity hiding,
- perfect forward secrecy,
- high-speed.

WireGuard’s protocol establishes a secure tunnel between client and server using a symmetric session key derived from a handshake.

As doubling the symmetric key size is enough to provide quantum security, the research focus is put on quantum resistance of the **handshake protocol**.

---

[1] Donenfeld, “WireGuard: Next Generation Kernel Network Tunnel”.

State of the art PQ-VPN

Microsoft's team proposed a PQ variant[2] of the OpenVPN protocol, using FrodoKEM or SIDH.

Hülsing *et al.*[3] added PQ security to WireGuard's handshake using McEliece and Saber. The PQ-WireGuard software integrates AVX optimizations and is implemented directly in the Linux kernel space.

Protocol	Traffic in bytes	# of IP packets	Time in ms	
			Client	Server
PQ-OpenVPN	8996	23	1277	1269
PQ-WireGuard	2532	2	0.92	0.30

Figure 1: Performance of handshake protocols.

---

[2] Paquin, Easterbrook, and Kane, *Post-quantum Cryptography VPN*.

[3] Hülsing et al., "Post-quantum WireGuard".

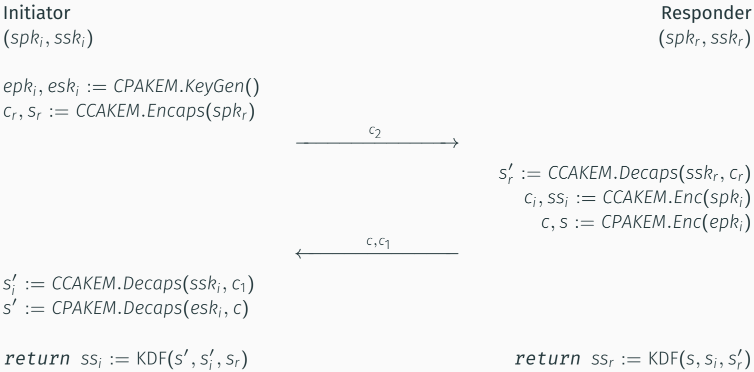
Possible trade-offs

While the communication and computational costs are both important, one can prevail according to the setting:

- low execution time is preferred to save on computational resources (relevant for both client and server, but most especially for the client, which performs concurrent tasks, compared to the dedicated server),
- a bandwidth-restrained scenario requires lighter traffic.

Following the steps of Hülsing *et al.*

Fujioka transform



Initiator
(spk_i, ssk_i)

Responder
(spk_r, ssk_r)

$epk_i, esk_i := CPAKEM.KeyGen()$
$c_r, s_r := CCAKEM.Encaps(spk_r)$

$\xrightarrow{c_2}$

$s'_r := CCAKEM.Decaps(ssk_r, c_r)$
$c_i, ss_i := CCAKEM.Enc(spk_i)$
$c, s := CPAKEM.Enc(epk_i)$

$\xleftarrow{c, c_1}$

$s'_i := CCAKEM.Decaps(ssk_i, c_1)$
$s' := CPAKEM.Decaps(esk_i, c)$

$\texttt{return}\ ss_i := \mathrm{KDF}(s', s'_i, s_r)$

$\texttt{return}\ ss_r := \mathrm{KDF}(s, s_i, s'_r)$

Figure 2: Fujioka's transform[4] combining two KEMs to obtain an AKE.

---

[4] Fujioka et al., "Strongly Secure Authenticated Key Exchange from Factoring, Codes, and Lattices".

CRYSTALS-Kyber

Kyber[5] is a Key Encapsulation Mechanism (KEM) among NIST's finalists.

It is defined by a tuple: KeyGen; Encaps; Decaps.

Kyber is constructed over a weakly secure LWE-based encryption scheme using the Fujisaki–Okamoto (FO) transform.

The security of Kyber can be reduced to a lattice problem, and the simple underlying arithmetic structure offers great performance.

---

[5] Bos et al., "CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM".

Results

We build on the Go implementation of the classical WireGuard software, and use our non-optimized Go Kyber implementation, with a *recommended* security level.

Protocol	**Traffic in bytes**	**# of IP packets**	**Time in ms**	
			Client	**Server**
PQ-OpenVPN	8996	23	1277	1269
PQ-WireGuard	2532	2	0.92	0.30
Our work (K+K)	4816	4	0.64	0.43

Figure 3: Performance of handshake protocols.

Optimizing the execution time

Tweaked Kyber

We start by removing the FO transform and build a CPA-secure KEM instance over Kyber's encryption scheme directly.

For the *recommended* security level, Kyber has a decapsulation failure rate of 1 out of 10^{47} which, compared to a standard packet drop rate of 1 out of 10^{6}, can be disproportionate.

Higher outputs compression for CPA secure Kyber instance:

- reduces the size of the ciphertext and/or public key,
- boosts performance,
- increases security.

Results

Protocol	Traffic in bytes	# of IP packets	Time in ms	
			Client	Server
PQ-OpenVPN	8996	23	1277	1269
PQ-WireGuard	2532	2	0.92	0.30
Our work (K+K)	4816	4	0.64	0.43
Our work (K+tK)	3760	4	0.50	0.29

Figure 4: Performance of handshake protocols.

Limitations

We noticed that there is no way to tweak Kyber's parameters to use only two packets without compromising on the security or getting too many decapsulation failures.

As the main bottleneck comes from the size of ciphertexts in the Kyber CCA-secure instance, we deviate from Hülsing *et al.* approach and use another AKE construction.

Optimizing the traffic

Del Pino transform

Initiator
(spk_i, ssk_i)

Responder
(spk_r, ssk_r)

$epk_i, esk_i := CPAKEM.KeyGen()$
$\sigma_i := DSA.Sign(ssk_i, epk_i)$

$$\xrightarrow{\sigma_i, epk_i}$$

$\texttt{if}$ $DSA.Verify(spk_i, \sigma_i, epk_i)$ $\texttt{== false: abort}$
$c, s := CPAKEM.Enc(epk_i)$
$\sigma_r := DSA.Sign(ssk_r, c)$

$$\xleftarrow{c, \sigma_r}$$

$\texttt{if}$ $DSA.Verify(spk_r, \sigma_r, c)$ $\texttt{== false: abort}$
$s' := CPAKEM.Decaps(esk_i, c)$
$\texttt{return}$ $ss_i := \mathrm{KDF}(s', \sigma_i, \sigma_r)$

$\texttt{return}$ $ss_r := \mathrm{KDF}(s, \sigma_i, \sigma_r)$

Figure 5: del Pino's transform[6] combining a KEM and a DSA to obtain an AKE.

---

[6] del Pino, Lyubashevsky, and Pointcheval, "The Whole is Less Than the Sum of Its Parts: Constructing More Efficient Lattice-Based AKEs".

Rainbow

Rainbow[7] is a Digital Signature Algorithm (DSA) among NIST's finalists.

It is defined by a tuple: KeyGen; Sign; Verify.

Rainbow is based on multivariate cryptography and stands out because of its very small signature size.

---

[7] Ding and Schmidt, "Rainbow, a New Multivariable Polynomial Signature Scheme".

Results

Protocol	Traffic in bytes	# of IP packets	Time in ms	
			Client	Server
PQ-OpenVPN	8996	23	1277	1269
PQ-WireGuard	2532	2	0.92	0.30
Our work (K+K)	4816	4	0.64	0.43
Our work (K+tK)	3760	4	0.50	0.29
Our work (R+tK)	1816	2	5.7	5.4

Figure 6: Performance of handshake protocols.

Conclusion

Results summary

Protocol	Traffic in bytes	# of IP packets	Time in ms	
			Client	Server
PQ-OpenVPN	8996	23	1277	1269
PQ-WireGuard	2532	**2**	0.92	0.30
Our work (K+K)	4816	4	0.64	0.43
Our work (K+tK)	3760	4	**0.50**	**0.29**
Our work (R+tK)	**1816**	**2**	5.7	5.4

Figure 7: Performance of handshake protocols.

Conclusion

The performance of a VPN application can benefit from exploring its various dimensions and proposing relevant trade-offs.

We show that using different cryptographic tools allows for competitive results for all settings.We highlight the fact that our primary goal is not to directly compare the performances of different software or platforms, but to experimentally demonstrate the practicality of post-quantum cryptography.

For future work, we want to integrate optimizations and eventually port our implementations to the Linux kernel space to further increase performance.

Questions?

References i

References

- Bos, Joppe W. et al. “CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM”. In: *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018*. IEEE, 2018, pp. 353–367. URL: `https://doi.org/10.1109/EuroSP.2018.00032`.
- del Pino, Rafaël, Vadim Lyubashevsky, and David Pointcheval. “The Whole is Less Than the Sum of Its Parts: Constructing More Efficient Lattice-Based AKEs”. In: *Security and Cryptography for Networks - 10th International Conference, SCN 2016, Amalfi, Italy, August 31 - September 2, 2016, Proceedings*. Ed. by Vassilis Zikas and Roberto De Prisco. Vol. 9841. Lecture Notes in Computer Science. Springer, 2016, pp. 273–291. URL: `https://doi.org/10.1007/978-3-319-44618-9%5C_15`.
- Ding, Jintai and Dieter Schmidt. “Rainbow, a New Multivariable Polynomial Signature Scheme”. In: *Applied Cryptography and Network Security*. Ed. by John Ioannidis, Angelos Keromytis, and Moti Yung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 164–175.

References ii

Donenfeld, Jason A. "WireGuard: Next Generation Kernel Network Tunnel". In: *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. The Internet Society, 2017. URL: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/wireguard-next-generation-kernel-network-tunnel/.

Fujioka, Atsushi et al. "Strongly Secure Authenticated Key Exchange from Factoring, Codes, and Lattices". In: *Public Key Cryptography – PKC 2012*. Ed. by Marc Fischlin, Johannes Buchmann, and Mark Manulis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 467–484.

Hülsing, Andreas et al. "Post-quantum WireGuard". In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 379. URL: https://eprint.iacr.org/2020/379.

Paquin, Christian, Karen Easterbrook, and Kevin Kane. *Post-quantum Cryptography VPN*. URL: https://github.com/microsoft/PQCrypto-VPN.

Additional slides

Extended Results

Protocol	Traffic in bytes	# of IP packets	Time in ms	
			Client	Server
PQ-OpenVPN	8996	23	1277	1269
PQ-WireGuard	2532	**2**	0.92	0.30
Security Level > 1				
Our work (K+tK)	3120	4	**0.40**	**0.26**
Our work (R+tK)	**1620**	**2**	4.6	4.7
Hybrid				
Our work (K+tK)	3760	4	**0.50**	**0.29**
Our work (R+tK)	**1816**	**2**	5.7	5.4
Security Level > 3				
Our work (K+tK)	4304	4	**0.59**	0.34
Our work (R+tK)	**2360**	**2**	6.2	5.9

Figure 8: Performance of handshake protocols.

Tweaked Instances

	d_u	d_v	d_{pk}	Failure Rate	Public key Size	Ciphertext Size
t-Kyber512	8	3	9	2^{-17}	608	608
t-Kyber768	9	3	8	$2^{-17.5}$	800	960

Figure 9: Internals.