QSC

The Quantum Secure Cryptographic library

A C language cryptographic tool-kit, used to develop the secure solutions of tomorrow

QSC

The Quantum Secure Cryptographic library

Security for the 21st century

QSC is a compact and self-contained post-quantum-secure cryptographic functions library written in the C programming language. It has been written to the highest standards of secure programming, using the MISRA rule-set, and has been designed to be easy to understand, verify, and implement. The code in this library is well structured, highly readable and well commented, and thoroughly documented throughout.

We felt that these are important qualities in a cryptographic library, qualities which are often overlooked. We wanted this library to be easy to navigate, the functions intuitive and well documented, and the code readable and well organized. Cryptography is not just about the strength and efficacy of the algorithms themselves, but great care must be taken to avoid implementation errors, and the more implementor-friendly, documented, and well organized a library is, the less likely there are to be mistakes made during implementation by developers.

A good case in point is OpenSSL, which has had much criticism for unnecessary complexity introduced into the library design. This has led to complete re-writes of the library, shedding huge amounts of unnecessary code, and efforts to make derived libraries like LibreSSL and BoringSSL more compact and well organized. OpenSSL has had many serious security issues in its history, and there have been several focused efforts to rewrite the library and reduce unnecessary complexity with limited success. But it still remains a bloated, and needlessly complex cryptographic library, and with the additions of post-quantum algorithms, that is likely to become even more problematic in the future.

What we have built, is a simplified set of tools, without all the unnecessary and irrelevant algorithms. We are not trying to be ‘everything to everyone’, maintaining code for algorithms that are insecure, and have outlived their usefulness. We have not implemented solely for backwards compatibility with insecure implementations, or retired hardware. What we have done instead is to ‘wipe the slate clean’, to start again, and build a library with ciphers and protocols that you actually need, and that work on today’s hardware. OpenSSL still contains many aging functions like DES, RC4, MD5, and SHA1, but we don’t implement these algorithms, because you shouldn’t be using them.

Likewise, we don't implement for irrelevant hardware; if it doesn't work on your 20-year-old SUN server, we don't think that is important from a design perspective, because from a systems security context, you shouldn't be using that hardware anyways. This is how we have eliminated so much of the bloat associated with popular cryptographic libraries, by being realistic, and not acting as an historical archive for retired cryptography and hardware, but as a streamlined, powerful, and modern set of tools.

We designed QSC to be a library of future cryptographic functions, functions that you will use. We believe that implementations of those functions in more complex forms like key-exchange protocols, should be separate from a cryptographic functions' library, delineated for ease of maintainability. There are many different uses for the cryptography, but implementations of those cryptographic functions as part of a larger protocol design, is more specific, and none of those protocols; TLS, SSH, SFPT, et. al., are going to be used in every implementation context. They are specific to an application of the cryptography itself, and lend too much variability and complexity to a cryptographic functions' library.

Many of these complex protocols are also constantly in flux, whereas the cryptography they use is far more constant; the AES specification hasn't changed in 20 years, but in the same time, SSL/TLS is about to undergo its 5th generation of specification changes with TLS 1.3. This constant flow of changes makes maintaining a cryptographic library that attempts to implement many different complex protocols, very challenging and error prone. Too often, such all-encompassing software constructs, incur subtle implementation mistakes, a result of the complex web of inter-connectivity of internal components, that change as they adapt from one use to another.

We have described what QSC is not, now let us tell you what it is. Our library represents the state-of-the-art of cryptographic technology. It contains only the most secure, proven, and powerful cryptographic primitives. It is focused on the future, not the past, and the future of the world's secure crypto-systems must be post-quantum secure. We have installed some algorithms for backwards compatibility; AES, ChaCha, SHA2, HMAC, HKDF, ECDH and ECDSA. But our primary design goal has been to create a set of modern tools, ones that can be used to build the security systems of the future.

We have implemented all of the NIST Post Quantum Competition's round 3 candidates; the asymmetric ciphers McEliece, Kyber, NTRU, and signature schemes, Dilithium, Falcon, and SPHINCS+. We have formatted them to MISRA secure coding rule-sets, added SIMD instructions using AVX/AVX2, and AVX512 instruction sets.

QSC has a powerful set of tools; an IPv4/IPv6 networking stack featuring synchronous and asynchronous sockets. Multi-threading tools, SIMD memory functions, integer conversion and evaluation functions, as well as a full set of cryptographic hashes, MAC functions, DRBGs, random providers, and random number generators. It is a complete set of tools, all vetted for secure operation, and written to stringent specifications, that can be applied in the most high-security of environments.

QSC also contains two of our symmetric cipher designs, RCS and CSX. These authenticated stream ciphers represent the two cryptographically strongest symmetric ciphers available in the world today, and were designed for true long-term security.

We have implemented the Keccak family of hash functions, and pseudo-random generators including SHA3, SHAKE, and KMAC. Keccak is probably the most well-studied set of cryptographic functions in modern times, having won the NIST SHA3 competition, after being the subject of intense academic scrutiny for more than three years.

The library contains many of the most currently used cryptographic functions, including HMAC, HKDF, SHA2, and AES, providing a window to backwards compatibility in your designs. What we haven't done, is install anything that we feel is insecure in the immediate context, while focusing on ciphers and protocols that provide the best guarantee of long-term security. We have made a very modular, compact and efficient library that future security products can be constructed with; our encrypted tunneling protocol QSMP, was built using this library, as well as the key distribution protocols SKDP and MPC system MPDC. Its small size makes it ideal for IOT devices and deployments with memory and storage constraints. QSC can just as easily be used in server/client software, or any implementation that requires a securely coded, fast and powerful set of cryptographic tools. The modular design, makes it ideal as a base tool-set implemented in more complex communications protocols. QSC was designed to be the future of cryptographic development, lean and mean, and containing only the most powerful tools available today.