Cryptographic Validation Beyond Implementation Correctness

Manuel Barbosa, François Dupressoir, Andreas Hülsing, Vincent Laporte, Pierre-Yves Strub

Cryptography is at the core of the security of digital systems, and the privacy of their users. Yet, evaluating cryptographic designs and implementations still rely mostly on expertise. Evidence provided during expert evaluation is often kept private, and remains mostly useless for further review even when published, often requiring as much expertise to read and understand as it does to produce and write it in the first place.

Formal methods, as deployed—for example—in many safety-critical settings such as nuclear energy and software for aerospace, can help. Not in making the evidence less obscure, but in making it independently verifiable. This is of course useful in assessing the correctness and safety of applications, but formal methods for safety-critical systems are applied at all stages of the engineering process, from design, through successive refinements down to implementation, and through verification and validation in the successive of high-assurance and high-integrity system development processes.

Our claim is that the evaluation of cryptography similarly needs to be carried out from top-to-bottom.

Cryptographic Security is Not Safety

Unlike the complex systems that high-integrity development processes typically target, cryptography consists of small artefacts. With few exceptions, the mathematics involved are simple and well-studied (for good reason!). We expect no emergent behaviours from cryptography, and although the composition of cryptographic systems can sometimes give rise to complex behaviours that are not always well understood, cryptographers do not want to keep any of those emergent behaviours, and generally understand how to avoid them.

Yet this lack of complexity should not be confused for simplicity. Cryptography is not complex, but is complicated. It is untestable by definition; its desirable properties are often specified by equivalence

to an unimplementable ideal system; the abstractions and safety margins that typically enable safe engineering all enable a malicious party—one cryptography means to ward against—to act in breach of the abstraction.

Those properties matter; often a lot more than the correctness of an implementation. Even when assessing the security of an implementation (for example against side-channels), and even when using automated or semi-automated techniques, the analysis *must* be informed by the desired high-level security property if efficiency is to be retained—and indeed it must be retained if the cryptographic algorithm is to be used.

The Cryptographic Properties of Specifications Matter

For this reason, the specification for a cryptographic algorithm *should* include a specification of its desired properties—those it is expected to meet. For a KEM, this might include IND-CCA, but also its expected binding property, and notions of resilience against invalid public keys. For an AEAD, this might include the precise authentication modes (inline, all at once) it is meant to provide security in, and the kind of nonce misuse it is expected to withstand.

Those expected security properties inform how implementations should be evaluated. They also inform the construction of systems that use the specified building blocks. An AEAD that is only secure against a nonce-respecting adversary, for example, must be used in a stateful way, or paired up with a strong and robust source of randomness.

Specifying such properties in a machine-readable way will then be useful in verifying a match between assumed and guaranteed properties for the usage that is made of cryptographic primitives in a lightweight way, as well as enable the easier verification and validation of their implementation.

High-Assurance Specifications: Machine-Checked Cryptographic Properties

But more importantly, specifying such properties in a machine-readable manner will also enable the deployment of high-integrity-style techniques for the verification of designs as they are being refined into high-assurance implementations. The security or privacy-preserving properties of the specification itself will also then be measurable and independently verifiable.

There exist a number of techniques and tools for the production and independent verification of cryptographic security and privacy properties on algorithmic specifications. Some are aimed at interactive protocols, others are aimed more generally at cryptography but most readily applicable to the kind of constructions standardised by NIST in FIPS documents.

The specifications those tools target must, by necessity, be higher-level than those that might be used in verifying an implementation. In proving the security of ML-KEM, it matters that the Number Theoretic Transform is an invertible linear operation. In verifying an implementation correct, it does not—and the precise instantiation of the NTT matters a lot more.

Therefore, for cryptography, a good specification should include:

- an algorithmic description useful for the study of the specification's security properties;
- a precise specification of the desired and expected security properties;
- an implementation-oriented specification, suitable for the verification of implementations, including using lightweight methods; and
- independently-verifiable evidence that the implementation-oriented specification securely refines the algorithmic description, and, in particular, achieves the claimed security properties.

Proposal For NIST Workshop on Formal Methods within Certification Programs

The authors propose a presentation on the above topic, followed by a panel discussion on the role of formally verified proofs in the assessment of algorithms and implementations for ongoing and future NIST selection/standardization processes.

The presentation will motivate the inclusion of clear cryptographic security objectives in specifications of cryptography. This will be discussed in the context of the machine-checked verification of cryptographic properties for international standards, covering the ML-KEM, ML-DSA, and SLH-DSA use cases. Finally, the presentation will discuss the opportunity such an inclusion may present in the design-time evaluation of applications that use cryptography, and in their practical deployment. This will then give way to panel discussions.