Applicability of a Cryptographic Metric Taxonomy in Cryptosystem Procurement Process and in Evaluation of Open Standards

Outi-Marja Latvala*, Jani Suomalainen†, Kimmo Halunen*,
Markku Kylänpää†, Reijo Savola*
VTT Technical Research Centre of Finland
*Oulu and †Espoo, Finland
{firstname.lastname}@vtt.fi

Mikko Kiviharju
Finnish Defence Research Agency
Finland
mikko.kiviharju@mil.fi

***Abstract*—Measuring the security of cryptographic systems in a simple and effective way is a difficult problem. There are several metrics that need to be taken into account. Earlier studies have produced one taxonomy of these different metrics, but the applicability of the taxonomy and the different metrics have not been tested. In this paper, we present a revised taxonomy of metrics for cryptographic systems and show results of applying it in two different scenarios: a procurement process for cryptosystems and in evaluation of open standards, namely the TLS 1.2 and TLS 1.3 standards. Applicability and meaningfulness of a taxonomy depends on its ability to differentiate cryptosystems and thus enable comparisons. Our results show that the revised taxonomy can help in differentiating systems and standards, especially when examining implementation related metrics. Future work should streamline the overly complex evaluation process.**

Keywords–cryptography; metrics; taxonomy; evaluation

I. INTRODUCTION

Measuring the security of systems against adversarial attacks is a very difficult problem. In cryptography, there exist some measures for the security of cryptosystems, but a comprehensive measure is still lacking. In [1] we presented a first version of a taxonomy of different metrics of cryptographic systems. In this paper, we extend that work and provide a revised taxonomy. This work is based on applications of the metric in some use cases by measuring the cryptosystems with the help of the metric taxonomy.

A good test for a metric is its applicability in real world use cases. Cryptosystems are used in many different products and protocols in modern society. Some areas, where the security of cryptosystems is crucial, include government communications (military and diplomatic use) and banking. In order for a cryptographic product to be used in sensitive government applications, cryptosystems and applications need to be certified. The certification is usually a fairly lengthy process especially when higher confidentiality/classification levels are aspired for.

Cryptosystems are needed and used also in our everyday communications and digital services. Without the many advances in cryptography, it would be extremely difficult to build digital services at the scale we are seeing now. Especially public key cryptography has played a crucial role in this development [2]. Thus, measuring also the security of the cryptosystems that secure these communications and services is important both to assure their trustworthiness as well as to enable their continuous improvement.

In order to better understand different metrics and the attributes that they measure, we need to have a taxonomy of these. Existing efforts for providing taxonomies for cryptographic metrics include e.g., Benenson et al. [3] who explored metrics from attackers' point of view and Jorstad et al. [4] who studied metrics in algorithms. Several standardization efforts, e.g., [5]–[8], have also provided guidelines or criteria for implementations. To the best of our knowledge, a comprehensive view of cryptographic metrics has been lacking until recently. The previous work of [1] provided a comprehensive taxonomy of metrics that gave concrete and generic measures both for algorithms as well as for implementations. This paper revises the taxonomy. The applicability of the metrics is increased as standards and products can be distinguished from each other in more accuracy. The revised taxonomy also provides new details and clarifies definitions.

We provide results from applying this metric to six real world use cases related to cryptographic products offering communications confidentiality services. We also apply our taxonomy in evaluating two different versions of the Transmission Layer Protocol (TLS). The results of these case studies are described and analysed. The revision of the taxonomy is based on lessons learned from these case studies.

The paper is organised in the following way. The next section describes the background for our work including an overview of the previous version of the taxonomy. The third section is dedicated to our case studies that provide the rationale for further improving the taxonomy and metrics. The fourth section describes our revised taxonomy and fifth section discusses our findings. Finally, we give conclusions from our research and some future directions for further study.

II. BACKGROUND

In this section, we present the relevant background on measuring cryptography and on the certification of cryptosystems for classified communications.

This research has been funded by Defence Forces Research Program 2017 (PVTO 2017).

A. Measuring Security

It is noteworthy, that there are many measures and metrics for security in general and also for cybersecurity specifically. These can take many forms and have a number of different dimensions. These also work on several levels of abstraction ranging from a cybersecurity index measuring nations [9] to a measurement of a single device or a piece of software (e.g., IoT labels [10]). However, measuring the security of cryptosystems does not have that many good metrics.

The terms security performance and level are commonly used in practice to refer to the *effectiveness* of security countermeasures, the main objective of security work and solutions. In addition, *efficiency* is essential because personnel and time resources, and costs have always constraints. In addition to effectiveness and efficiency, *correctness* is a fundamental objective in security measurements. Correctness is a necessary but not sufficient requirement for effectiveness; i.e., it is a side effect in effectiveness, not its driver. Sufficiently effective security countermeasures are based on sufficient risk awareness of the target system in focus. There are various factors which enable effectiveness in practice, such as risk-driven design of security controls, configuration correctness, sufficiently rigorous implementation and deployment of security controls and proper security assurance activities. In practice, complexity, limited observability, a lack of common definitions and the difficulty of predicting security risks make it impossible to measure security as a universal property. Therefore, the term security metrics is misleading, yet widely used [11].

In practice, there are various gaps and biases between security effectiveness measurement objectives and practical security measurements. In risk analyses, it is not possible to identify and prioritize all actual risks. Difficulties in understanding well the target system or risk situation can cause bias. Further gaps are introduced when developing security requirements and the actual system. Different phases of R&D cause easily additional bias. Due to the gaps and biases, security effectiveness can be achieved only asymptotically. In security metrics modelling, an important goal is to minimize gaps and biases, making the more practical security correctness objectives as close as possible to security effectiveness objectives [11].

According to the results of the expert opinion survey reported in [12], *correctness, measurability* and *meaningfulness* are the core quality criteria for security metrics. Moreover, usability is considered to be important, but not so essential as the three before-mentioned dimensions. In the following, we discuss these quality dimensions and their relationships to more detailed quality criteria.

The term accuracy is often used instead of correctness to emphasize the fact that all-inclusive correctness is impossible. Using the term correctness makes it possible to differentiate between accuracy and precision. The time dependability of metrics can be seen partly as a sub-criterion of prediction accuracy and partly as an independent sub-criterion of 'general' correctness. The representativeness of security metrics is crucial to their correctness in a security context. The granularity of a metric and associated measurement should be at least at a level where adequate decision-making based on them is possible. Contextual specificity (or, inversely, contextual independence) is a special case of granularity. Completeness of security metrics is related to representativeness, addressing one or a collection of metrics. Non-intrusiveness is an important criterion related to correctness. Security measurements and the associated metrics should not overly affect or hinder the actual functionality of software systems or the functions of an organization [12].

Measurability is a prerequisite for the meaningfulness and usability of security metrics. Attainability of measurable information is related to measurability, while availability is a sub-goal of attainability. Together, reproducibility and repeatability form the precision of the measurement. Scale reliability is a sub-criterion of reproducibility. In addition to measurability, reproducibility, repeatability, and scale reliability are related to correctness.

To be meaningful, the metric should answer the original essential question that reflects the need for evidence. Clarity is closely related to meaningfulness: the clearer the formulation of the metric, the easier it is to understand provided that the person interpreting it has enough knowledge about the underlying context. Succinctness increases clarity and thereby meaningfulness. Good succinctness also increases the efficiency of the metric, a sub-goal of usability. In order for the security metric to be meaningful, they should incorporate applicability to decision-making. Comparability of different measurement results is desirable when making selection decisions among different security controls. The ability to show progression, is a special case of comparability [12].

Usability of security metrics is important, yet not as critical as correctness, measurability and meaningfulness, because poor usability is not fatal for security metrics. The criteria related to usability include the following: efficiency, cost effectiveness and controllability, scalability, and portability [12].

B. Measuring Cryptography

For many years, the most prevalent measure used in describing the security of cryptographic systems has been the algorithm-specific key length, with recommendations from governmental authorities as well as standardization bodies [13], [14]. There are, however, a number of shortcomings when using only this one metric to evaluate cryptographic systems: firstly, the update cycle of the recommendations may be years, and the coverage of standards only includes the most commonly used systems. Secondly, the key length indicates resilience in the most simplistic adversarial models only: the so-called "brute-force" attack, where the attacker only tries to guess (or compute) the key. This leaves out multiple implementation-level issues [15], protocol vulnerabilities, some more niche use cases [16] of regular algorithms and many more.

There are also cases, when new algorithms are evaluated for completely new applications that are not covered by current standards; in these cases, it would be preferable to have more general set of metrics to use. Finally, there are many different levels of abstraction and each level combines methods from lower levels to reach the security goal of that level. Even when a security proof exists and gives a good guideline on how to choose the security parameter, this one parameter - the key length - oversimplifies the complexities of implementing

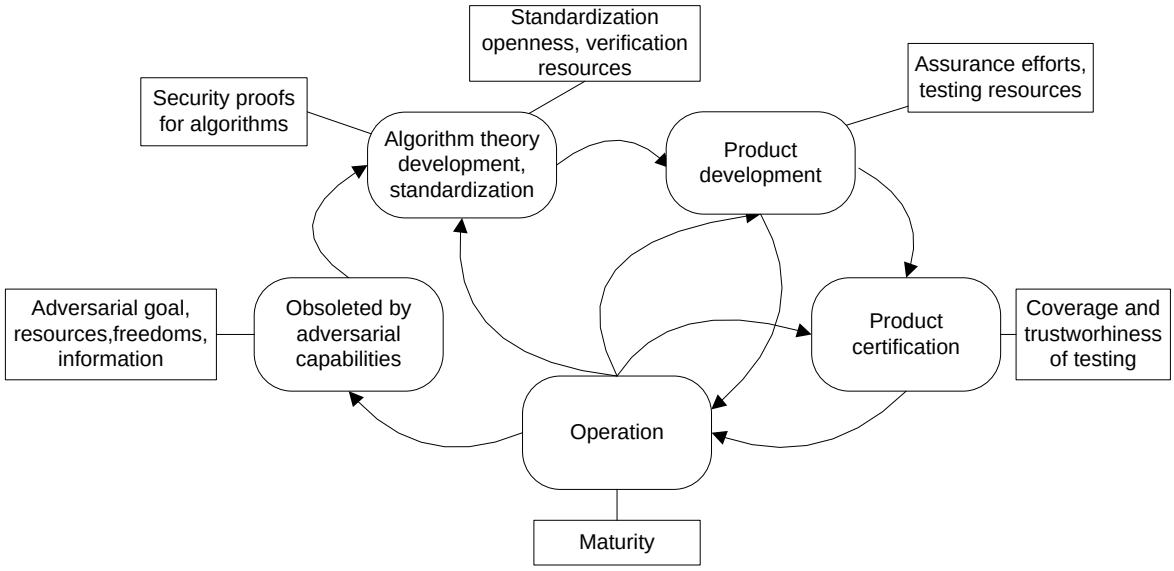

Figure 1. A life cycle for cryptographic systems and its relation to metrics.

cryptosystems. In mathematical terms, a certain key length can be seen as a necessary, but not sufficient condition for security of some cryptosystem in some given context, e.g., a digital signature protocol.

One of the main uses of metrics is to compare candidate cryptosystems for a specific purpose. In this case, it may happen that the security features are all approximately on the same line, but due to some conventions in the detailed implementation, they do not perform equally well. In this case, performance-related metrics become relevant, and we claim that there are certain performance-related metrics that are fairly common across the use of cryptographic primitives.

Metrics can be identified from different phases of the life cycle of cryptographic systems (illustrated in Figure 1). Initiation of development of new cryptography is typically motivated by expanding adversarial capabilities, which are measured through resources, goals, freedoms of action and information. Acceptance of new algorithms is achieved with theoretical security proofs as well as through standardization by openness and verification. During the algorithm development and standardization, developers and academic reviewers verify and analyze the theoretical strength of algorithms using common metrics and methodologies. These may be specific to the use case or more generic measures accepted in the community for measuring the cryptosystem.

For example, with hash functions there exist many different results on potential attacks on the Merkle-Damgård type iterative hash functions. However, not all these have been considered serious enough to be explicitly defended against for example in the NIST call for candidates for the SHA-3 hash function standard. This shows that there exists a some sort of a consensus on what is considered a serious threat to the security and what does not among the community. In some cases, the results that show an attack can be derivative of more fundamental attacks or properties of the system. In the case of hash functions for example multicollision attacks result from the underlying property of iteration irregardless of the implementation [17], [18]. Some of these have been generalised also to the sponge construction employed in the current SHA-3 standard [19].

During the implementation of software or hardware products, the vendors test products to detect implementation failures. Developers also often expose products for certification testing that verifies that the product fulfils the intended security criteria. Users may also require certification before accepting products for procurement and operational use. Independent or national accredited testing laboratories have product certification frameworks and programs for assuring that implementations have required functionality and behave as expected with different inputs. During deployment and operation, the product matures and its parameters must be adjusted to withstand threats from evolving environment. Technologies and implementations mature within time until new disruptive adversarial capabilities or new security requirements necessitate new algorithms and products. The legacy systems and algorithms are phased out over time. Changes in operational context - new performance requirements or needs to certify products against emerging threats - may also result in changing the phase of the cryptographic system in the life cycle (illustrated with arrows in Figure 1).

C. Certification of Cryptosystems

An important area, where metrics for cryptosystems are needed, is the certification (or approval) of cryptographic products. The need for certification exists in many different areas of official use. The process of certification can be very cumbersome and is based on evaluations. These evaluations can be very lengthy and be based on some standards (e.g., Common Criteria [20]) or some more heuristic criteria set by the certifying authority.

A comprehensive, applicable and possibly even simple measure for the strength of a cryptosystem would make these evaluations easier to conduct and faster. It could also help in comparing cryptosystems designed for different purposes and

systems certified under differing and proprietary standards. A good metric could also be used as a guideline or threshold for cryptosystems to be used for certain levels of classification.

Governments routinely use a well-defined evaluation and approval process to ascertain that systems used for protecting resources on a certain sensitivity level are actually fit for the purpose. Due to the nature and use of cryptologic information (i.e., for intelligence gathering), the details of these processes are often hidden and proprietary. The usual outcome of such a process merely answers *"yes"/"no"* for questions of the type: *"Is product X fit for protecting assets on sensitivity level Y?"*. Thus, it is meaningless to compare two products on the same level Y, which result from different certification bodies. This also implies the need for standard metrics.

In high-security environments there may also be a disconnect between procurement and assurance processes. Then it is not even expected that the Request for Information (RFI) or Request for Quotation (RFQ) phases produce information related to security metrics (other than what can be gleaned from open sources and standards). Security metrics are then evaluated with a pass/fail-grade just before actual procurement. This type of operation is deemed necessary in environments where there are contradicting requirements of secrecy and information needs, and may result in sub-optimal outcomes for the procurer, when compared to a completely open process.

There are several standards that have been developed to support certification of cryptosystems. For example, several ISO/IEC standards [5], [6], [21] consider the requirements of cryptographic systems, the verification and testing of such systems etc. These give good guidelines for certification, but do not in general define specific metrics and values that need to be met and measured.

D. Previous Taxonomy for Measuring Cryptosystems

The first comprehensive taxonomy for measuring cryptosystems in [1] classified metrics into four main categories. The first two categories - adversarial model and security proof framework metrics - addressed metrics that are related to security of the algorithms. The latter two categories - verification and maturity as well as cost and performance metrics - addressed feasibility and security of cryptographic implementations.

The authors of the previous taxonomy defined a cryptosystem to be any algorithm, protocol or method for providing cryptographic guarantees with respect to some security goal. They also defined *a metric* as a way to measure some part or the totality of the security of a cryptosystem. A metric can have numerical values or it can be a qualitative description depending on the use case. A metric is *measurable* if there is a standard convention on how the metric is measured and this is uniform across all applications of the metric (e.g., kilograms for weight). A metric is *semi-measurable* if there are several different conventions on how to measure the metric and some of these are not readily comparable with each other. The third category of metrics is *non-measurable*, which means that a standard for measurement does not exist or that the different values that the metric can have are not comparable in meaningful ways. Note the difference to the mathematical concept of a metric, which always has a numerical value. We also distinguish between *quantitative* and *qualitative* metrics. Quantitative metrics give a numerical or several numerical values to the cryptosystem and qualitative metrics give a description of the state of the cryptosystem.

In the original taxonomy, also the practical relevance was represented by stating whether an attribute was theoretical or practical. In our revised taxonomy, we have chosen not to use this attribute. We think that this distinction is not meaningful, as the metrics need to be actionable at all levels of abstraction. However, it is possible that for a certain use case and scenario, some metrics can have greater practical impact than others.

One of the major shortcomings of the previous taxonomy is that in practical applications, many cryptosystems utilise certain standards (e.g., TLS) and thus the values of metrics of many implementations do not vary in meaningful ways. For example, the adversarial model metrics are usually fixed in any standard. Thus, these metrics cannot differentiate between different cryptosystems against *the same adversarial model*. Thus, there is a need for metrics that can provide the differentiating factors.

On the other hand, it is important to measure also the adversarial model metrics so that we can distinguish different levels of security between different adversarial models. This can be useful not only for practical implementations of cryptosystems but also in research. By finding differences in the metrics, one can find room for improvement and new research problems in trying to address these.

III. Revised Taxonomy

Our taxonomy, which is presented in Table I, has three main categories. The first two relate to algorithmic metrics that involve cryptosystems independently of their realization in code or hardware. Algorithmic metrics are here divided to adversarial model metrics (Subsection III-A) and proof framework metrics (Subsection III-B). The last category combines different metrics that are relevant for evaluating realizations and implementations of cryptosystems, their feasibility, maturity, and costs (Subsection III-C).

We revised the original taxonomy from [1]. The main difference is that in the highest categorization level, we now have combined all cryptosystem implementation related metrics under the same main category - verification and feasibility - as the methodologies for verifying various metrics are similar. Refined metric taxonomy contains also some new metrics and definitions for old metrics have been clarified based on feedback and experience on applying the taxonomy.

One issue in the old taxonomy was the disconnect between theoretical adversarial models and the implementation level differences in actual threats and capabilities of adversaries. For example, one could have a disk encryption application (e.g., Bitlocker [22]) and a TLS implementation using AES with the same keylength. In a theoretical sense and in the metric of [1] these would yield the same result in the metric. On the practical and implementation level, there are many differences on the information that an attacker can get on the data and the overall threat model. These are not reflected well in the original taxonomy. Thus, we have added new metrics to the taxonomy. These are *Side-channels* and *Metadata*.

TABLE I. CATEGORIES AND PROPERTIES OF DIFFERENT METRICS OF CRYPTOGRAPHIC SYSTEMS.

Main category	Subcategory	Metric	Measurable	Type
Adversarial model	Degrees of freedom	Observe/choose/choose adaptively	yes	Qualitative
		Corruption power - num. of principals	yes	Quantitative
		Corruption power - degree of corruption	semi	Qualitative
		Security game compliance	semi	Qualitative
	Adversarial available information	Pre-crypto	yes	Quantitative
		Post-crypto	yes	Quantitative
		Secret key material	yes	Quantitative
		Protocol runs	semi	Quantitative
		Setup parameters	semi	Qualitative
		Simulation environment	semi	Qualitative
	Adversarial goal	Goal	semi	Qualitative
	Adversarial resources	Computation power Instantiated	yes	Quantitative
		Computation power Non-instantiated	semi	Qualitative
		Memory Instantiated	yes	Quantitative
		Memory Non-instantiated	semi	Qualitative
Proof framework	Security assumptions	Mathematical complexity	semi	Qualitative
	Abstraction assumptions	Type	semi	Qualitative
		Num. of assumptions	yes	Quantitative
		Maturity of assumptions	no	Qualitative
	Methodology	Tightness	yes	Quantitative
	Rigor	Rigor	semi	Qualitative
Verification and feasibility	Assurance levels	Assurance standard or profile	no	Qualitative
		Evaluation assurance level	yes	Quantitative
	Test coverage	Percentage of covered areas	yes	Quantitative
	Vulnerabilities	Number	yes	Quantitative
		Number of classified	semi	Quantitative
	Side-channels	Existence	semi	Qualitative
	Metadata	Leaked amount and type	no	Qualitative
	Evaluator acceptance and reputation	Reviews	yes	Qualitative
	Evaluator experience	Academic publications	semi	Quantitative
		Experience in years	yes	Quantitative
	Verification time	Time since released for evaluation	yes	Quantitative
		Size and efforts of eval. community	yes	Quantitative
	Openness of target	Software/design	semi	Qualitative
	Readiness level	Technology readiness level	yes	Quantitative
		Integration readiness level	yes	Quantitative
		System readiness level	yes	Quantitative
		PETS maturity model	yes	Qualitative
	Key length	Bits for criteria compliance	yes	Quantitative
	Time costs	Execution overhead	yes	Quantitative
		Communication overhead	yes	Quantitative
	Memory and transmission costs	Run-time memory	yes	Quantitative
		Storage capacity	yes	Quantitative
		Communication bandwidth	yes	Quantitative
	Implementation complexity	Size of software	semi	Qualitative
		Dedicated hardware requirements	semi	Qualitative
	Energy efficiency	Algorithm complexity dependent joules	yes	Quantitative
		Hardware platform dependent joules	yes	Quantitative

A. Adversarial Model Metrics

Cryptology and especially cryptographic theory aims to formalize how cryptographic algorithms work and withstand cryptanalysis. Due to the need for rigorous formalisms in cryptographic theory, the models used need to be very detailed, and yet general with respect to adversarial behaviour.

Algorithmic metrics are sometimes difficult to define and may be difficult to compare, due to the close association to actual schemes. Many qualities that are essential to one type of cryptographic algorithm may make no sense with another. However, there are still metrics that can be measured and that can be used to measure cryptosystems of similar nature (e.g., block ciphers or digital signatures). Some of these can then also generalise to metrics that can be used to wider varieties of cryptosystems.

For this purpose, even in the abstract world of algorithms and cryptography theory, it is beneficial to be able to state that algorithm A is (X times) more secure than algorithm B. This in turn requires some metrics, and to distinguish them from the other metrics used in this paper.

This results in the following:

- Algorithmic metrics are used in generally accepted combinations, rather than picked independently. In general, there exists good consensus which metrics form a good and reasonable combination. On the other hand, it is entirely possible that new cryptosystems will need to combine these metrics in new ways. Also, new metrics can be brought about through research.
- The exact definitions tend to be scheme specific, resulting in different understanding of which constructs or postulated goals are actually more secure than others, making some metrics effectively incomparable with each other.

This is one of the main difficulties in measuring and comparing cryptosystems.
- Metrics themselves may be scheme and usage-specific: protocol security metrics may not apply to primitives, and asymmetric primitive security metrics not to symmetric primitives.
- Just collecting all the possible values of the metrics ever used, from thousands of papers is a daunting task, which is why we only give examples of the values. This means that a measurement using these metrics needs to be aware of the context and also the research on the topic.

As an example, consider the combination of the metrics in the following common concept: INDistinguishability under Chosen Ciphertext Attack or IND-CCA [23] and existential unforgeability under chosen-message attack, or EUF-CMA [24]. We observe here the following independent metrics:

- Adversarial goal: distinguish between random strings and actual ciphertext.
- Adversarially available information: a polynomial amount of information, before and after the cryptographic transformation.
- Adversarial degrees of freedom of actions include choosing the ciphertext-plaintext pairs adaptively (excluding the keys).

In addition to the three metrics above, we consider *adversarial resources*, which the designer of the cryptosystem expects the attackers to be able to wield. For existing attacks, the required adversarial resources may prove to be smaller than expected by the designers, which may break the system's security even in practice. The four above metrics together are related to the *adversarial model*. The metrics in the category are selected to be as independent of each other as possible. As mentioned already earlier, these dependencies are currently inescapable in the taxonomy work.

The adversarial resources consist of *computing power* and available *memory*. They are mostly well-defined and accessible metrics, with practical relevance.

Computing power is addressed here in both of its forms: exact attack complexities, and approximate or asymptotic complexities. Cryptographic theory rarely elaborates the adversarial models down to the detail of exact number of operations required to break the system. Instead, asymptotic estimates are given, and often even they are described only on the level of computational complexity classes.

In the case of exact complexities, values can be given, e.g., as the amount of floating point operations per second (FLOPS). In quantum computing, the unit can be based on, e.g., the amount of universal qubits and gates in the quantum computer or the quantum volume of the system [25]. As adversarial time is limited, the computing power is typically presented in relation to time, for example as CPU years (work done by 1 GFLOP machine in a year). The exact actual amount of computing required to break a cryptographic algorithm is known for demonstrated attacks. For example, finding SHA-1 collisions with 6500 CPU years or 100 GPU years [26], factoring of RSA-512 with 1.1 TFLOPS capacity in 4 hours [27], breaking of 7.45 DES keys within a nanosecond by custom ASICS hardware [28].

However, the exact work and time needed to break previously unbroken algorithms can be only estimated. Affecting adversarial factors include both the sophistication of breaking algorithms as well the type of processor {`CPU, GPU, ASIC, quantum`}. This metric is semi-measurable because for a given algorithm known attacks can be measured, but between different algorithms these might not be comparable. It is also important to note that for more complex cryptosystems breaking a single component of the protocol (e.g., a hash function) may not constitute a full breach of the system. This metric is also quantitative.

In the latter case, where the complexity class border is crossed, literature usually refers to different "computational models", the most common being probabilistic polynomial time (PPT) adversary corresponding to the complexity class bounded-error probabilistic polynomial time (BPP), where polynomially bound, probabilistic Turing machines are expected. Other notable models include bounded-error quantum polynomial-time (BQP) for quantum computers; and statistical, or unconditional security model, where the adversary is given limitless computational power. This metric is semi-measurable (as the exact relations between complexity classes are not known) and qualitative.

Because the exact running time estimates can only be fixed once a cryptosystem is fully instantiated and parametrized, we consider this measure to consist of two subclasses of the whole: instantiated and non-instantiated computing power (asymptotic notations can be computed to exact metrics once the parameters, such as key size, are fixed).

Memory is the amount of memory that the attack requires. Analogously to the computing power, we divide this into two subclasses: instantiated (measurable and quantitative) and non-instantiated (semi-measurable and qualitative). While not so common in cryptographic scheme design (with the exception of memory-hard password hashing), the attacks may require large memories expressible in complexity classes, e.g., with time-memory trade-offs. Also, the type of the memory and memory access can have effect on the time that the attack takes. Memory can also have some effect on the computing power needed for the attack. Some example memory complexity classes could be logarithmic space (LOGSPACE) and polynomial amount of space (PSPACE).

Adversarially available information is the amount and type of data that the attack needs or is allowed for the adversary. This can mean plaintext-ciphertext pairs, number of connections or interactions with a server, related keys, bits from a random generator, cryptographic parameters etc. Metric is quantitative within one type of data, but not necessarily across types, as this is scheme-dependent. Thus, the metric is only semi-measurable. We distinguish here at least six different types: *pre-crypto* (data before encryption, signing or other cryptographic transformation), *post-crypto, secret key-material* (symmetric or private asymmetric), *protocol runs, setup parameters* and *simulation environment master*. The four first ones are measured in bits, bytes or messages/keys/runs, the last two are discussed as follows:

- The access to setup parameters becomes relevant in cryptographic protocols, giving rise to, e.g., variants of Universal Composability (UC): Joint UC [29] and Global UC

[30]. Possible value space could be {`local/global, per protocol/several runs`}.

- The control of the master process, which in cryptographic protocol security proofs models to what degree the adversary is able to control the (unspecified) protocol environment (resulting in yet other UC variants [31]). Possible value space could be {`Sim+Adv, Advonly, Env, *`}.

Technically, the scheme or protocol description and parameters of the system also belong to the measured adversarial information. However, modern systems very often assume all of them to be public, or constantly available (see Kerckhoffs's principle [32]), so we do not consider them here.

Adversarial goals are a complement to the security goals: if the security goal is indistinguishability, the corresponding adversarial goal is to distinguish (an output of cryptographic transformation from a random string); if the adversarial goal is a total break of the system, it suffices as a security goal to be able to keep even one message confidential. Thus, these two metrics are actually two sides of the same metric. Unfortunately, to be comparable, the goals need to be rigorously formalized, which usually results in case-specific definitions, and almost all values for the metric are incomparable, making it both qualitative and semi-measurable only. An example value space for typical goals is {`Semantic deduction, Information Leak, Local deduction, Global deduction, Total Break`}.

Adversarial degrees of freedom of action refer here to what the adversarial model is expecting the adversary to do. (Note: "anything" is not a rigorous enough answer to this). We propose to divide the degrees of freedom into three: *General, Corruption power* and *Game compliance*.

Corruption power. In interactive protocols, the adversary is also assumed to be able to access and/or modify the private information of some of the principals. This is called corruption, and depending on the scheme, only a certain number of principals are allowed to be corrupted. Sometimes even more fine-grained "corruptive power" is allowed [33]. The example values of this metric could include a (quantitative) percentage of corrupted principals and a (qualitative) description of the degree of corruption within one principal (see [33] for subprotocol-level detail).

For the *general metrics*, cryptographic formalisms differ in the amount of principals: Single-party settings (conventional encryption and signatures) and multiparty settings (protocols). As we show below, the multi-party setting does not bring that many new metrics per sé.

In the single-party setting, only one or fixed, integral set of cryptographic transformations (a black box) are usually considered. Thus, there is a set of black boxes (e.g., encryption and decryption, or signing and verification), with a number of inputs and outputs. In this case, the adversary may be able to *observe* some or all of the inputs, or to *choose* (possibly adaptively) some or all of them. Note that we consider modification of inputs and other adversarially available information to belong to the "choosing" process. Some of the possible values in the single-party setting would then be `'Observe'`, `'Choose'` and `'Choose adaptively'`, in increasing order.

In the multi-party setting, i.e., protocols, the situation with adversarial behaviour appears at first sight to be more complex, as the security models are more varied. (Dolev-Yao model, Inexhaustible Interactive Turing Machines [34], Reactive Simulatability [35] and others [36]). In the Dolev-Yao model [37], the principle is that the "attacker carries the message", or that the adversary is free to read, modify, add and delete protocol messages and corrupt protocol principals (in effect stealing their private key material). Relaxations to this model include [38], where corruption does not include divulging private key material, which makes the corruption probabilistic.

However, the convention we made in the single-party setting already covers the deletion, modification and adding of protocol messages, global setup parameters modification and protocol environment control, since the ability to choose message (/parameters/environment properties) for a single party translates to all of the above. Furthermore, corruption of principals itself (note: this does not include the number of principals) can be thought of as a combination of adversarially available (keying) information and the ability to choose keying inputs to cryptographic transformations. We thus conclude that we have not identified more metrics from the multi-party setting.

Game compliance. Many of the formalisms in cryptographic security can be divided into two: game-based approaches and simulation-based approaches. Game-based approaches are basically a protocol, which try to model the adversarial behaviour in some commonly thought scenarios. These approaches result in efficient schemes that are relatively easy to prove secure. Simulation-based approaches try to enable showing security irrespective of the adversarial behaviour. The best the attacker can do, is to perform the idealized, non-cryptographic tasks assigned to replace cryptography in the simulation (SIM) model, since all of the cryptographic tasks are idealized to be functionally equivalent non-cryptographic ones, but performing the tasks by other means than cryptography. This is a very strong model, and not many constructions can be proven secure in this.

In the metrics, we consider this distinction to be an adversarial degree of freedom in the sense that the adversary is either constrained to follow some security game protocol, or not. Loosely speaking, the simulation-based security proofs also capture the adversary's actions within a Turing Machine, making the proof technique similar to that of reductionist proofs with complexity assumptions. The main difference in the proof technique is that the degrees of freedom are larger with the Turing Machine (basically Turing-Complete) than with a dedicated security game. The values could be, for example {`Game-App, Game-Gen, SIM`}, making a further distinction between general security games and very application-specific games.

B. Security Proof Framework Metrics

Proof framework is the framework in which the security proof is conducted. This includes multiple assumptions (for abstractions of certain functions and for the complexity of

several mathematical problems), the rigor used (ranging from heuristics to verified full proofs) and the proof methodology.

A metric clearly tied to the proof methodology is *tightness* of the proof. This concept indicates how exactly the resource needs for different phases of the proof are estimated. In "loose" proofs, polynomial reductions (without stating the actual degree of the reduction polynomial) and upper limits are common, whereas tighter proofs uses "less margin", resulting in more optimistic adversarial resources and ultimately in more efficient parameters for a scheme. This metric is measurable and quantitative, as typical asymptotical $O(f(n))$ expressions are used here.

Complexity assumptions are the foundation of many types of cryptographic proofs especially in the realm of computational security. They are assumptions on the hardness of different mathematical problems, usually that their time-complexity is superpolynomial in the security parameter. These assumptions can have several metrics:

- Assumption's time-complexity. This has relevance in many asymmetric schemes, since the complexity is not always exponential. The metric is measurable, quantitative and practical, as it directly affects key size. The metric is expressed with the Big-Oh-notation (e.g., $O(f(n))$).
- Assumption maturity (we consider this to be in the verification category and not elaborated more here)
- Type, if the assumption belongs to a known sequence of implications (e.g., Decisional Diffie-Hellman (DDH) $\Leftarrow$ Computational Diffie-Hellman (CDH) $\Leftarrow$ Discrete Log (DL) problem.) A possible common labelling borrows from the general ordering for several problems, where decisional problems (DDH) are usually easier than computational problems (CDH), and finally the primitive inversion problem (DL): {`decisional, computational/search, inversion`}. This metric is semi-measurable and qualitative.

Abstraction assumptions cover how much the proof methodology uses abstractions, what kind of type they present and their maturity. Typical abstractions give different functions as ideal oracles, the most famous probably being the Random Oracle Model (ROM, [39] with variations in [40] and [41]). Many other oracles exist as well, e.g., the Generic Group Model (GGM) [42], the Ideal Cipher Model (ICM) [43], and the Common reference string model [44]. Sometimes the oracles are implicit, such as the Dolev-Yao modelling on encryption operations, which are assumed to be secure. If no abstractions are used, the proof is said to be conducted in the Standard Model.

The actual metrics are proposed as follows:

- The number of abstractions used. For a proof in the standard model this would be zero. Different abstractions would be weighed differently depending on their maturity and suitability for the cryptosystem
- Assumption maturity (we consider this to be in the verification category and not elaborated more here)
- Type. Not all of the abstraction are equal, as there are some known relations among them (e.g., ICM and ROM have been proven equal in some cases [45]). We then postulate that like with the complexity assumptions, there is a common metric able to classify abstraction assumptions as well, but we leave it for future study.

Rigor refers to the level of detail of the proof, its compliance to commonly used proof techniques and the assurance in the validity of the proof. Many schemes outside the cryptologic community often rely on pure heuristics, others merely state that the scheme is essentially similar to an earlier scheme and overlook the security proof completely. Many other systems are too complex to contain fully rigorous proofs in single conference papers, making the authors only outline the proofs. Ideally, proofs should be fully detailed, and externally verified. The value space for this metric would then be {`Heuristic, Referenced, Outlined, Full, Verified`}.

C. Verification and Feasibility of Implementations

The strength and correctness as well as maturity and feasibility of cryptographic implementations can be verified with different verification methods and testing tools.

1) Verification Metrics: Verification is a process establishing security, correctness, compliance, or validation of cryptographic implementation. Verification metrics describe the coverage and effectiveness of the verification and testing actions that the cryptographic product has passed.

Assurance Levels are measurements indicating system's security compliance when compared against common or standard evaluation and testing requirements. For instance, Evaluation Assurance Level (EAL) is a seven point-scale metric used by the Common Criteria (CC) [46] security evaluation framework for implementations; Common Criteria's Protection Profile is simpler two point-scale (compliant/non-compliant) metric for specific product categories; Cryptographic Algorithm Validation Program (CAVP) [7] defines functional and statistical tests for algorithms with a two-point (pass-fail) scale; Cryptographic Module Validation Program (CMVP) [47] defines validation tests for hardware implementations in four point scale (i.e., FIPS 140-2 security levels); and ISO 29128 [48] Protocol Assurance Levels define requirements for the scope and automation of formal modelling and verification of cryptographic protocols.

In addition to the generic frameworks, there also exist frameworks that are specific for industry field or for an area of cryptography. For instance, the Payment Card Industry [8] has defined its own test requirements for two point scale evaluation of cryptographic hardware modules. National Institute of Standards and Technology (NIST) has specified [49] a large suite for *randomness* testing. Randomness sources are important for many cryptosystems as the keying usually requires some randomness to provide security. This attribute can have values like user-supplied, system supplied, user + system supplied (e.g., password and a salt) and verified randomness. Verified randomness can also be quantum randomness, but the extent to which this improves security of a system is not necessarily easy to quantify. Assurance levels are semi-measurable and quantitative metrics.

Testing coverage refers to the percentage of potentially vulnerable areas that are verified. Coverage is complete if every area, as specified, e.g., by a particular certification criteria or test set, is verified. The areas that can be tested include,

e.g., symmetric algorithms, asymmetric algorithms, key management, functionality, interfaces and protocols, randomness, susceptibility to side-channel and fault injection attacks, lifecycle, as well as susceptibility to physical tampering and to reversing of obfuscated functionality attacks. Existing test suites, validation program requirements or common criteria profiles can be utilized when estimating whether all relevant areas are included to verification and whether all tests for the relevant areas are executed. This metric of coverage is measurable and quantitative.

Existing systems have often known *vulnerabilities* of different impact and consequences. A most serious vulnerabilities are typically patched but due to costs or backward compatibility vulnerabilities with lower risks may remain. A straightforward quantitative metric is the amount of known all kinds of vulnerabilities. A little more sophisticated metric is to classify vulnerabilities according to their seriousness and count instances in each category, e.g., through the CVE [50] and CVSS systems [51].

One important implementation consideration, which is often covered in verification, is the existence of *side-channels*. These are possible information sources that come from outside the adversarial model. It is noteworthy that some adversarial models can include some side-channels and show that a system is resistant to these. However, not many systems can be shown to be resistant to all possible side channels.

Most common side-channels are timing and power consumption, but depending on the cryptosystem there might be others such as errors and faults, sound, heat etc. We distinguish between the different side channels and each type of side channel is its own attribute. This leads to a situation, where we need to have an attribute for ”other” side channels that are not listed in our taxonomy. This is because new side channels can be found later on.

Side-channels are semi-measurable and qualitative. Possible values for the side channel metric are:

- known side-channel attack,
- no known attacks,
- covered by the adversarial model
- not applicable

However, it is important to note that in many cases the applicability of different side-channels varies greatly. The not applicable value can also be used in case the evaluation is done on a theoretical algorithm without a specific implementation.

Another implementation consideration is the amount and type of *metadata* that the system allows to the potential attacker. This may be the length of the message, number of recipients, the algorithms used in the cryptosystem, programming languages and/or libraries used by the system. Also, timestamps, user names, location etc. fall into this category. This is a non-measurable and qualitative metric. This metric can contain a lot of information about the context, where the cryptosystem is applied. Thus, it is important to have an understanding what is relevant to the current analysis. This also makes it difficult to measure the different values against each other as the context matters greatly.

Capabilities of evaluating laboratories, communities, and individuals - skills, experience, and methods - are an indirect measure also for the evaluated systems. The more skilled and scrutinized reviews the product has passed, the more less likely the system is to contain unknown and hidden vulnerabilities. These capabilities can be measured, e.g., by looking at the *experience and education* of evaluators. Quantitative and measurable metrics for human verification include count of verifiers, verifiers' experience in years, number of performed evaluations, as well as the scientific author metrics (number of fresh related publications of the evaluated cryptographic system).

Similarly, other verifiable and relevant public demonstrations of expertise can be included. For example, the number of relevant CVEs (Common vulnerabilities and exposures) [50] submitted by the evaluators should be considered. The evaluators' effectiveness depends on the available software and hardware facilities, as well as on the quality of the processes in the evaluating community or laboratory. Qualitative metrics for effectiveness include *maturity and acceptance of evaluating* laboratory or method by scientific community and organizations. Similarly, qualitative metrics for human effectiveness includes *evaluator's reputation*, which is based on past performance. Effectiveness does not necessarily ensure accuracy. Even though an evaluating person, laboratory, or method detects large amount of known vulnerabilities it may also miss many.

The *openness* of the source code or design (in case of a hardware system) is a particular metric, which affects to the effectiveness of verification. This metric is semi-measurable and qualitative. The possible values are {`open source, closed source, auditor access`}. The last one means that the code is closed to the general public, but is available for auditing by some parties outside the implementors (e.g., officials or potential clients).

Verification time refers to the hours, months, or years that have been spend on exploring the cryptographic solution against vulnerabilities. Time accumulates from intensive product evaluations as well as from the verification and testing by scientific and user community during the system lifetime. The older and more dispersed the system is, the less unknown weaknesses it is likely to have. This is a quantitative and measurable metric.

2) Maturity Metrics: The maturity metrics measure how ready and suitable a cryptosystem is. This can be a metric for a specific component as in Technology Readiness Level (TRL) [52], its derivates such as Systems Readiness Level (SRL) [53] and Integration Readiness Level (IRL) [54], or a more comprehensive metric of a whole systems, such as the Privacy Enhancing Technologies (PETs) maturity metric [55].

TRL measures the readiness of a single component. This metric is measurable and quantitative. *IRL* measures the readiness of components to be integrated to form a more complex system. This metric is measurable and quantitative. *SRL* measures the readiness of a complete system based on the TRLs and IRLs of the different components. The metric is measurable (if normalized) and quantitative. *PETs maturity metric* is a measure for the quality and readiness of privacy enhancing technologies. The PETs measurement is carried out with both measurable indicators (such as the number of papers/patents and lines of code) and a more heuristic

evaluation by experts. There is a defined procedure on how to reach consensus on possibly differing evaluations by experts. In some sense, this is similar to the jury evaluation used in some cryptographic standardisation competitions. In the PETs maturity metric, the evaluation is open and transparent, whereas in some cryptographic competitions this is not the case. The PETs maturity metric is measurable and qualitative.

As already mentioned in the introduction, *key length* is one of the most used metrics for cryptosystem security. In our taxonomy, key length considers the maturity and verification level that a cryptosystem has. It is an indicator that shows if the security parameters of a cryptosystem are up to date and provide wanted security against known threats. Key length defines the upper-bound of algorithm security. As different algorithms do not provide equivalent security, estimates on the corresponding strength are made to enable comparisons (e.g., 128 bit symmetric key corresponds to 2000 bit factoring modules and 250 bit elliptic curve keys [14]). The metric is measurable and quantitative.

3) Cost and Performance Metrics: The feasibility of cryptographic products depends not only of their security strength, but also on cost and performance. These metrics can be calculated for the whole system in total or separately for an individual role (e.g., decrypter, encrypter, signer, verifier). Typically, costs are divided unevenly between different roles. This asymmetric cost division may be beneficial, e.g., in cloud or Internet of Things scenarios where another party has more resources available for cryptographic operations. Costs for attackers are estimated with the adversarial resource metrics in Subsection III-A; this subsection focuses to defenders' costs.

Time costs originate from the computations, such as key generation, encryption and decryption, as well as public and private key operations, and from communications, where cryptography causes additional overhead, expands communication and negotiations. Time costs can be estimated by counting elementary operations that a cryptographic solution implies or by experimenting and benchmarking, see e.g., [56]. Typical units for measurement are computing cycles per encrypted block or throughput (bits/second) within particular CPU frequency. Time cost is a quantitative and measurable metric.

Memory and transmission costs relate to the need for run-time and storage memory, as well as to the communication bandwidth. They depend on the sizes of keying material, ciphertexts, and signatures, as well as on run-time memory requirements of algorithms. This is a quantitative and measurable metric, which is typically presented as bit as bytes.

Implementation complexity relates to the size and costs of software or hardware implementations. A key attribute is whether the solution is suitable for standard computing platforms (e.g., Intel x86 or ARM-based) or whether it requires specialized hardware. An important attribute is also whether the performance of the algorithm can be improved with special hardware, such as parallel platforms or extended instruction sets. Complexity can be estimated either by counting lines of code or by counting required hardware resources like gate counts. This is a semi-measurable (as there are many ways to measure complexity) and qualitative metric.

Energy efficiency depends on the use of computing, memory, and communication resources and their costs in different platforms. Hardware computing factors affecting to energy efficiency include, e.g., area size, amount of gates, bit transitions per clock cycle, cycles per algorithm execution, as well as block size [57]. Energy efficiency is a quantitative metric that can measured using joules/bits, watts, or in some cases through the environmental emissions.

IV. Case Studies for Applying Taxonomy

We applied the new taxonomy to evaluate both cryptographic implementations as well as standards. First, we applied the taxonomy to six use cases, where information about a cryptosystem was available through a formal requisition process. All of these systems represent closed-source products, from which all of the technical details may not be available.

Second, we evaluated three different settings and versions from a readily available standard, TLS, with the help of the metric. As a simple test to the metric, we chose to limit our sources to the TLS standards themselves and obvious immediate references such as AES and RSA standards. It can be argued that this is a limiting factor of our analysis. However, it is also important to notice that these metrics should be fairly simple to use and that having a thorough expert analysis through all the relevant research literature on the subject is not possible in most cases, where these metrics should be applied.

A. Measuring Closed-Source Products

We applied both the versions of the taxonomy to six closed-source cryptographic products performing communications security and requiring separate certification before approval for use. Our aim was to see to which extent the metrics could differentiate products and how readily they were available. The results can be seen in Table II.

All of the products aim to provide the same type of security service, and represent the newest versions of their vendors. For this reason, it should be expected that certain baselines and standards will be identical for these products, including their metrics. Furthermore, the metrics were originally designed to cover many types of use cases and algorithms, but in this case study, the products only concentrate on one or two types. Thus, not all metrics can be considered valid for all products.

In addition, due to their closed-source nature, measurement could only be performed with varying degrees of success, as not all information was available, or even asked for in the first place. We have then presented the average availability of the metrics with six levels (1 to 6) as follows:

- Level 1: Metric explicitly stated
- Level 2: Metric implied by an open standard (including independent research on the standard)
- Level 3: Metric implied by a closed standard
- Level 4: Metric implied otherwise
- Level 5: Metric released to evaluators only, via formalized process
- Level 6: Metric withheld

In some cases the metric concerned such features that were not supported, or were not at all available (not even for the manufacturer) or requested in any part of either the procurement or assurance process.

TABLE II. RESULTS OF TESTING THE METRIC TAXONOMY IN A PROCUREMENT PROCESS

Main category	Subcategory	Metric	Diff.[a]	Avail.[b]	Avg. diff.	Avg. avail.
Adversarial model	Degrees of freedom	Observe/choose/choose adaptively	2	3.3		
		Corruption power - num. of principals	0	6.0		
		Corruption power - degree of corruption	1	3.4		
		Security game compliance	2	3.7		
	Adv.avail. Inform.	Pre-crypto	0	2.8		
		Post-crypto	0	2.8		
		Secret key material	0	2.5		
		Protocol runs	1	3.2		
		Setup parameters	3	3.8		
		Simulation environment	0	N/a		
	Adversarial goal	Goal	1	3.3		
	Adv. resources	Computation power Instantiated	0	4.5		
		Computation power Non-instantiated	0	4.0		
		Memory Instantiated	0	4.5		
		Memory Non-instantiated	0	4.0	0.7	3.7
Proof framework	Sec. assumptions	Mathematical complexity	0	3.0		
	Abstr. assumptions	Type	0	3.0		
		Num. of assumptions	0	3.0		
		Maturity of assumptions	0	3.0		
	Methodology	Tightness	0	3.0		
	Rigor	Rigor	0	3.0	0.0	3.0
Verif.&feas.	Assurance levels	Assurance standard or profile	6	1.0		
		Evaluation assurance level	6	1.0		
	Test coverage	Percentage of covered areas	5	5.2		
	Vulnerabilities	Number of detected vulnerabilities	2	2.7		
	Side-channels	Existence	3	5.3		
	Metadata	Leaked amount and type	4	2.8		
		Openness	3	1.0		
	Human efficiency	Academic publications	3	5.4		
		Verifier experience in years	3	5.4		
	Verification time	Time since released for evaluation	4	2.2		
		Size and efforts of eval. community	5	4.5		
	Readiness level	Technology readiness level	0	-		
		Integration readiness level	0	-		
		System readiness level	0	-		
		PETS maturity model	0	-		
	Key length	Bits for criteria compliance	6	1.8		
	Time costs	Execution overhead	6	1.0		
		Communication overhead	5	3.7		
	Mem. & BW	Run-time memory	5	5.2		
		Storage capacity	5	5.2		
		Communication bandwidth	6	1.0		
	Impl. compl.	Size of software	5	5.2		
		Dedicated hardware requirements	3	1.5		
	Energy efficiency	Algorithm complexity dependent joules	0	6.0		
		Hardware platform dependent joules	6	1.0	3.6	3.2

[a] How many products this metric can differentiate.[b] Availability of the metric (average of the values for the availability levels as specified in Section IV-A).

The first version of the taxonomy proved to be too vaguely described in the adversarial model and security framework parts, and could not be readily applied to product level. Thus we applied the taxonomy fully only to the current revised version. The three categories did not, surprisingly, have significant differences in the availability of the metrics on behalf of the vendors. This may be partly due to standardization, but also due to the fact that many performance and security evaluation details are actually company secrets. The algorithmic metrics were slightly more difficult to obtain, due to some closed standards prevalent in the field.

The second "metric of a metric" we measured was how many products (of the six) a particular metric was able to differentiate (technically "five" is redundant, since the sixth could be identified by being the remaining one, but we consider the applicability only).

According to this differentiation ability, there is a striking difference between performance and verification metrics compared to others:

- None of the proof framework metrics could differentiate between products (either due to standardization or to complete unavailability)
- The adversarial model metrics could make a difference between products only on protocol level (i.e., key exchange) details, not primitives (like block cipher) themselves.
- The verification and feasibility metrics, however, were able to distinguish 3.6 or closer to 4 products out of 6. This may reflect the very strict control on security features and a stable set of use cases, forcing the vendors to compete on the performance rather than security.

Some notes for individual metrics include:

- The assumption for non-instantiated computer power and memory for the adversary are implicitly "PPT"

(/"PSPACE"), unless quantum security is considered. At this point, no off-the-shelf product we looked at offered quantum-secure constructions, thus these metrics would not be expected to provide any difference.
- If some metric is promoted by authorities and standards, it will usually eventually become part of public sales brochures or public knowledge in any case. One good example of this is the cryptographic key length: it is probably one of the most exact and widespread metrics in cryptography, required to be specified very ubiquitously. This has led even closed standard algorithms to divulge or leak their key lengths (e.g., BATON [58], and Libelle [59]). This is also evidenced by our study: key length is one of the few metrics we could pinpoint in all of the test subjects.

The adversarial model metrics mostly concern algorithm level issues, which are common to standardized products. The models are developed independently and publicly inside the cryptologic community, and although they may not be part of a standard, they are implicitly understood to follow from cryptographic research. Many algorithmic metrics of standards may be difficult to locate in the cryptographic literature for a non-expert in cryptographic theory. Thus it is advisable that the most important standards be readily measured in a table (for example) by a group of experts beforehand. Examples include the UC-security of (a portion of a version of) TLS [60] and IND-CCA security of IPSec IKEv2 [61]. Sometimes it is not easy to see, whether a metric actually is included in another, e.g., whether UC-security includes adaptive adversaries [62].

The main outcome of this case study is that algorithmic best practices are usually incorporated into various standards, which many products will follow. Thus, if the standards used are very uniform, such as the case with AES, algorithmic metrics are unlikely to make a difference between products. However, in cases where a standard allows multiple solutions (such as the planned PQC-standard), there are no standards (many closed-market products and new innovations such as blockchains and homomorphic encryption) or the standards are vague (e.g., implementation details, parametrization, protocol-level solutions), all the metrics are likely to serve a definite purpose also from end-user perspective. This case study was not able to weigh the proposed metrics across product types or across standards.

B. Measuring TLS 1.2 and TLS 1.3

For TLS 1.2 [63] we evaluated one setting, which was the ciphersuite `TLS_RSA_WITH_AES_128_CBC_SHA`. All TLS 1.2 implementations must support this ciphersuite according to the standard.

We used the taxonomy in [1] and searched through the TLS 1.2 standard for valuations of the different metrics in the taxonomy. For many metrics the standard did not contain suitable answers. Then also the references in the standard were studied. In several cases we also needed to do larger searches for example to find out about the number of vulnerabilities.

Finding the correct framing for measuring all the metrics was not an easy task. In some cases the valuation was left empty or Not applicable was used. For TLS this was the case for the different readiness levels as there is no authoritative source for this type of information.

For TLS 1.3 [64] we measured two settings: `TLS_AES_128_CCM_SHA256` and `TLS_AES_128_CCM_SHA256`. The former was chosen as the one closest to the TLS 1.2 ciphersuite `TLS_RSA_WITH_AES_128_CBC_SHA`, although the differences between the two versions are so vast that a perfect match does not exist. The latter was the mandatory setting for TLS 1.3.

The first thing we noticed when applying the metric to the TLS cases was that the adversarial model metrics are essentially the same for all three cases, since they are about the freedom and powers granted to the adversary when constructing the proofs for the underlying cryptographic primitives. The documentation we were using did not provide direct answers to these metrics, instead prior knowledge, or expert opinion, needed to be used (or further research if we were not artificially restricting our sources). For example, we could say that the adversary would have "substantial resources" to get to the goal of "information leak", but that would not come from the TLS documentation, or any of the other specifications we were using.

Differences between standard versions are in the performance and in the security and feasibility of implementations. Metrics related to the performance and new metrics related to the side-channel and metadata can capture these differences. Further, TLS 1.3 specification mitigates some previous attacks that were related, e.g., to renegotiation, protocol version downgrading, and compression [64]. These advances are visible in the verification metric category, where number of known vulnerabilities are counted and indirectly also in the readiness level metric where knowledge of existing vulnerabilities should be visible as the lowered maturity evaluations. When considering the evaluation efforts, both protocols have passed similar large scale reviews by global security community and industry. TLS 1.2 may however seen more evaluations as it was released for evaluation 2008, about ten years before release of TLS 1.3. There are also some formal verification efforts related to TLS, e.g,, in [65].

V. Discussion

Measuring the security of systems is a very difficult task even though the area has been researched for a long time and the interest has been increasing in recent years. Measuring the security and strength of cryptosystems seems to be even harder, because there are so many different metrics and variables involved and these also interact in many ways. Furthermore, the notion of security is very much context dependent. Even a secure cryptographic primitive used in a wrong context provides very little security. An insecure primitive in a wrong context provides even less security, e.g., [66].

The reason for revising the earlier taxonomy of [1] was the difficulty in using that taxonomy in gaining actionable information of real cryptosystems. Our new taxonomy should alleviate this situation, but there is still room for improvement and future work. We have focused more on metrics, which are relevant for the development and operational phases in the life cycle of cryptographic systems. We have given more concrete

metrics for addressing implementation specific threats, side-channels and leaked metadata.

As stated before, correctness, measurability and meaningfulness are core quality criteria for security metrics. We investigated the metrics in use cases, giving enough contextual specificity and same time dependability, both contributing to correctness. Two other dimensions of correctness, granularity and completeness are difficult to be analyzed based on the use cases, and would require more experimentation. Most of the metrics presented in the revised taxonomy are measurable, or at least semi-measurable. Investigating meaningfulness of the taxonomy's metrics would also require more extensive studies. However, clarity and applicability to decision-making were driving the taxonomy development.

One key issue already pointed out in [1] is that of dependencies between different metrics. For instance, feasibility metrics, costs and performance, depend on the algorithmic model. The revised taxonomy is still open for new inter-metric derivatives. For instance, there are some measures that we have chosen, due to simplicity, not to present in our taxonomy as their own separate metrics. One is the total (monetary) cost of an attack as a resource metric. It could be argued that this is one of the most relevant metrics there is. On the other hand, it is also derivative of the adversarial resource metrics that have been included in our taxonomy. A major direction of future improvement of these metrics will be to quantify the relationships as well as to find more independent measures. With independent metrics, it could be easier to produce information on cryptosystems that is understandable and also usable to the different stakeholders measuring cryptosystems.

As mentioned already in the beginning of this paper, the utility of any metric is in its applicability to real use cases and the ability to discern between different cryptosystems. Thus, the new taxonomy needs to be applied in some use case to determine its usefulness. Then also new improvements can be made.

It is also important to note, that for example in the verification metrics category there are many standards that can be applied to cryptosystem evaluation. However, having for example good coverage of a given testing standard, e.g., FIPS 140-2 [67], does not necessarily mean that the randomness generation can withstand adversarial attacks [68]. Furthermore, there are cases such as the Dual-EC, where a backdoor was included in the standard and remained there for quite some time [69]. Thus, even good values in this type of coverage metric can not guarantee good security without other information or further metrics.

One clear shortcoming of both the original metrics from [1] and our revised version is the results of many metrics being incomparable. This means that it is hard to get an absolute ordering of cryptosystems and their security. With truly independent metrics that are also all quantitatively measurable, one could compute (a possibly weighed) average of the different values and use that as the score for the system. These could then be ordered easily. The current ensemble of different metrics gives only a vector in a space, where distances are not well defined or do not even exist in any meaningful way.

VI. Conclusion

In this paper, we have revised the taxonomy of metrics for cryptographic systems from [1] to a new taxonomy. We have used this new taxonomy to evaluate both closed source implementations and the open TLS 1.2 and TLS 1.3 standards. Although there are some metrics where differentiation can be achieved, the taxonomy and the different metrics in it do not provide yet a usable tool for evaluations. The process of evaluating the metrics and differentiating between different implementations is fairly complex and the results are not necessarily decisive.

For future work, there is still a great need to improve the metrics and to find ways to simplify the measurement process. Whether this can be achieved through revision of the taxonomy presented here or through some different approach, is an important venue for further study.

References

[1] K. Halunen, J. Suomalainen, O.-M. Latvala, M. Kylänpää, V. Vallivaara, and M. Kiviharju, "A taxonomy of metrics for cryptographic systems," in Thirteenth International Conference on Emerging Security Information, Systems and Technologies, SECURWARE, 2019.

[2] P. C. van Oorschot, "Public key cryptography's impact on society: How diffie and hellman changed the world."

[3] Z. Benenson, U. Kühn, and S. Lucks, "Cryptographic Attack Metrics," in Dependability Metrics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 133–156.

[4] N. Jorstad and T. S. Landgrave, "Cryptographic algorithm metrics," 20th National Information Systems Security, 1997, http://csrc.nist.gov/nissc/1997/proceedings/128.pdf, (accessed 9.12.2020).

[5] EN ISO/IEC 19790:2020, "Information technology. Security techniques. Security requirements for cryptographic modules (ISO/IEC 19790:2012)," International Organization for Standardization, Geneva, CH, Standard, 2020.

[6] ISO/IEC 29128:2011(E), "Information technology — Security techniques — Verification of cryptographic protocols," International Organization for Standardization, Geneva, CH, Standard, 2011.

[7] NIST, "Cryptographic Algorithm Validation Program," https://csrc.nist.gov/Projects/Cryptographic-Algorithm-Validation-Program, (accessed 9.12.2020), 2018.

[8] PCI Security Standards Council, "Payment card industry data security standard v3.2.1," 2018, https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2-1.pdf, (accessed 9.12.2020).

[9] International Telecommunication Union, "Global Cybersecurity Index (GCI) 2018," 2018.

[10] P. Emami-Naeini, H. Dixon, Y. Agarwal, and L. F. Cranor, "Exploring how privacy and security factor into iot device purchase behavior," in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019, pp. 1–12.

[11] R. Savola, C. Frühwirth, and A. Pietikäinen, "Risk-driven security metrics in agile software development: An industrial pilot study," Universal Computer Science, vol. 18, 2012, pp. 1679–1702.

[12] R. M. Savola, "Quality of security metrics and measurements," Computers & Security, vol. 37, 2013, pp. 78–90.

[13] N. Smart, "Algorithms, Key Size and Protocols Report (2018)," Tech. Rep., 2018.

[14] BSI, "Cryptographic mechanisms: Recommendations and key lengths," https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf, (accessed 9.12.2020), BSI TR-02102-1, Tech. Rep., 2018.

[15] M. Carvalho, J. DeMott, R. Ford, and D. A. Wheeler, "Heartbleed 101," IEEE Security Privacy, vol. 12, no. 4, 2014, pp. 63–67.

[16] A. Biryukov and D. Khovratovich, "Related-key cryptanalysis of the full aes-192 and aes-256," in International Conference on the Theory and Application of Cryptology and Information Security. Springer, 2009, pp. 1–18.

[17] A. Joux, "Multicollisions in iterated hash functions. application to cascaded constructions," in Annual International Cryptology Conference. Springer, 2004, pp. 306–316.

[18] J. Kortelainen, K. Halunen, and T. Kortelainen, "Multicollision attacks and generalized iterated hash functions," Journal of Mathematical Cryptology, vol. 4, no. 3, 2010, pp. 239–270.

[19] M. A. AlAhmad, I. F. Alshaikhli, and M. Nandi, "Joux multicollisions attack in sponge construction," in Proceedings of the 6th International Conference on Security of Information and Networks, ser. SIN '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 292–296. [Online]. Available: https://doi.org/10.1145/2523514.2523551

[20] Common Criteria, "Common Criteria for Information Technology Security Evaluation - Part 1: Introduction and general model," Common Criteria, vol. 3.1, no. April, 2017, pp. 1–106.

[21] ISO/IEC 24759:2017(E), "Information technology — Security techniques — Test requirements for cryptographic modules," International Organization for Standardization, Geneva, CH, Standard, 2017.

[22] Microsoft, "Bitlocker," https://docs.microsoft.com/en-us/windows/security/information-protection/bitlocker/bitlocker-overview, (accessed 9.12.2020), 2018.

[23] M. Bellare and P. Rogaway, "Optimal asymmetric encryption," in Advances in Cryptology—EUROCRYPT'94. Springer, 1995, pp. 92–111.

[24] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," SIAM Journal on Computing, vol. 17, no. 2, 1988, pp. 281–308.

[25] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," Phys. Rev. A, vol. 100, Sep 2019, p. 032328. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.100.032328

[26] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, and Y. Markov, "The first collision for full SHA-1," in Annual International Cryptology Conference. Springer, 2017, pp. 570–596.

[27] L. Valenta, S. Cohney, A. Liao, J. Fried, S. Bodduluri, and N. Heninger, "Factoring as a service," in International Conference on Financial Cryptography and Data Security. Springer, 2016, pp. 321–338.

[28] J. Sugier, "Cracking the DES cipher with cost-optimized FPGA devices," in International Conference on Dependability and Complex Systems. Springer, 2019, pp. 478–487.

[29] R. Canetti and T. Rabin, "Universal composition with joint state," in Annual International Cryptology Conference. Springer, 2003, pp. 265–281.

[30] R. Canetti, Y. Dodis, R. Pass, and S. Walfish, "Universally composable security with global setup," in Theory of Cryptography Conference. Springer, 2007, pp. 61–85.

[31] A. Datta, R. Küsters, J. C. Mitchell, and A. Ramanathan, "On the relationships between notions of simulation-based security," in Theory of Cryptography Conference. Springer, 2005, pp. 476–494.

[32] K. Auguste, "La cryptographie militaire," Journal des sciences militaires, vol. 9, no. 538, 1883, p. 5.

[33] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in Proceedings 2001 IEEE International Conference on Cluster Computing. IEEE, 2001, pp. 136–145.

[34] R. Kusters, "Simulation-based security with inexhaustible interactive turing machines," in 19th IEEE Computer Security Foundations Workshop (CSFW'06). IEEE, 2006, pp. 12–pp.

[35] M. Backes, B. Pfitzmann, and M. Waidner, "Limits of the reactive simulatability/UC of Dolev-Yao models with hashes," 2006.

[36] M. Prabhakaran and A. Sahai, New notions of security. Princeton University, 2005.

[37] D. Dolev and A. Yao, "On the security of public key protocols," IEEE Transactions on information theory, vol. 29, no. 2, 1983, pp. 198–208.

[38] P. Syverson, C. Meadows, and I. Cervesato, "Dolev-Yao is no better than Machiavelli," Naval Research Lab, Washington DC, USA, Center for high assurance computing systems, Tech. Rep., 2000.

[39] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in Proceedings of the 1st ACM conference on Computer and communications security. ACM, 1993, pp. 62–73.

[40] P. Ananth and R. Bhaskar, "Non observability in the random oracle model," in International Conference on Provable Security. Springer, 2013, pp. 86–103.

[41] J. B. Nielsen, "Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case," in Annual International Cryptology Conference. Springer, 2002, pp. 111–126.

[42] V. Shoup, "Lower bounds for discrete logarithms and related problems," in International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 1997, pp. 256–266.

[43] C. E. Shannon, "Communication theory of secrecy systems," Bell system technical journal, vol. 28, no. 4, 1949, pp. 656–715.

[44] R. Canetti and M. Fischlin, "Universally composable commitments," in Annual International Cryptology Conference. Springer, 2001, pp. 19–40.

[45] J.-S. Coron, T. Holenstein, R. Künzler, J. Patarin, Y. Seurin, and S. Tessaro, "How to build an ideal cipher: the indifferentiability of the feistel construction," Journal of cryptology, vol. 29, no. 1, 2016, pp. 61–114.

[46] Common Criteria, "Common Criteria for Information Technology Security Evaluation Part 2 : Security functional components," Security, no. April, 2017, pp. 1–323.

[47] NIST, "Derived Test Requirements for FIPS PUB 140-2, Security Requirements for Cryptographic Modules," 2011.

[48] International Organization for Standarization, "ISO/IEC 29128:2011 - Information technology – Security techniques – Verification of cryptographic protocols," 2011.

[49] NIST, "Special publication 800-22. a statistical test suite for random and pseudorandom number generators for cryptographic applications," 2010.

[50] MITRE, "Common vulnerabilities and exposures," https://cve.mitre.org, (accessed 9.12.2020), 1999.

[51] FIRST, "Common vulnerability scoring system," https://www.first.org/cvss/, (accessed 9.12.2020), 2015.

[52] J. C. Mankins, "Technology readiness levels," White Paper, NASA, 1995.

[53] B. Sauser, D. Verma, J. Ramirez-Marquez, and R. Gove, "From TRL to SRL: The concept of systems readiness levels," in Conference on Systems Engineering Research, Los Angeles, CA, 2006, pp. 1–10.

[54] B. Sauser, R. Gove, E. Forbes, and J. E. Ramirez-Marquez, "Integration maturity metrics: Development of an integration readiness level," Information Knowledge Systems Management, vol. 9, no. 1, 2010, pp. 17–46.

[55] M. Hansen, J. Hoepman, M. Jensen, and S. Schiffner, "Readiness analysis for the adoption and evolution of privacy enhancing technologies: Methodology, pilot assessment, and continuity plan," Tech. rep., ENISA, Tech. Rep., 2015.

[56] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, "A survey of lightweight-cryptography implementations," IEEE Design & Test of Computers, vol. 24, no. 6, 2007, pp. 522–533.

[57] S. Kerckhof, F. Durvaux, C. Hocquet, D. Bol, and F.-X. Standaert, "Towards green cryptography: a comparison of lightweight ciphers from the energy viewpoint," in International Workshop on Cryptographic Hardware and Embedded Systems. Springer, 2012, pp. 390–407.

[58] Wikipedia, "BATON," en.wikipedia.org/wiki/BATON, (accessed 9.12.2020).

[59] ——, "Libelle (kryptographie)," de.wikipedia.org/wiki/Libelle_(Kryptographie), (accessed 9.12.2020).

[60] S. Gajek, M. Manulis, O. Pereira, A.-R. Sadeghi, and J. Schwenk, "Universally composable security analysis of tls—secure sessions with handshake and record layer protocols," Cryptology ePrint Archive, Report 2008/251, 2008, https://eprint.iacr.org/2008/251, (accessed 9.12.2020).

[61] A. Roy, A. Datta, and J. C. Mitchell, "Formal proofs of cryptographic security of diffie-hellman-based protocols," in International Symposium on Trustworthy Global Computing. Springer, 2007, pp. 312–329.

[62] D. Hofheinz, J. Mueller-Quade, and R. Steinwandt, "Initiator-resilient universally composable key exchange," Cryptology ePrint Archive, Report 2003/063, 2003, https://eprint.iacr.org/2003/063.

[63] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol version 1.2," 2008.

[64] E. Rescorla and T. Dierks, "The transport layer security (TLS) protocol version 1.3," 2018.

[65] C. Cremers, M. Horvat, J. Hoyland, S. Scott, and T. van der Merwe, "A comprehensive symbolic analysis of tls 1.3," in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1773–1788. [Online]. Available: https://doi.org/10.1145/3133956.3134063

[66] F. Y. Rashid, "Adobe's hacked passwords: They are terrible!" https://uk.pcmag.com/opinion/12060/adobes-hacked-passwords-they-are-terrible, November 2013.

[67] NIST, "Security Requirements for Cryptographic Modules. FIPS PUB 140-2," 2001.

[68] D. Hurley-Smith, C. Patsakis, and J. Hernandez-Castro, "On the unbearable lightness of fips 140-2 randomness tests," IEEE Transactions on Information Forensics and Security, 2020.

[69] D. J. Bernstein, T. Lange, and R. Niederhagen, "Dual ec: A standardized back door," in The New Codebreakers. Springer, 2016, pp. 256–281.