On the State of Crypto-Agility

Nouri Alnahawi, Nicolai Schmitt, Prof. Dr. Alexander Wiesmaier, Prof. Dr. Andreas Heinemann, Tobias Grasmeyer[1]

Abstract:
The demand for crypto-agility, although dating back for more than two decades, recently started to increase in the light of the expected post-quantum cryptography (PQC) migration. Nevertheless, it started to evolve into a science on its own. Therefore, it is important to establish a unified definition of the notion, as well as its related aspects, scope, and practical applications. This paper presents a literature survey on crypto-agility and discusses respective development efforts categorized into different areas, including requirements, characteristics, and possible challenges. We explore the need for crypto-agility beyond PQC algorithms and security protocols and shed some light on current solutions, existing automation mechanisms, and best practices in this field. We evaluate the state of readiness for crypto-agility, and offer a discussion on the identified open issues. The results of our survey indicate a need for a comprehensive understanding. Further, more agile design paradigms are required in developing new IT systems, and in refactoring existing ones, in order to realize crypto-agility on a broad scale.

Keywords: Cryptographic Agility, Crypto-Agility

1. Introduction and Related Work

Cryptographic primitives and protocols require constant modifications in order to maintain the security of IT-systems. Many researchers argue that applying the notion of crypto-agility provides more feasible and practical adaptation of cryptographic systems [45], especially in the light of the expected transition to PQC [14, 18]. However, there is no unified definition for this notion, nor a common understanding of the requirements that can enable it. Moreover, it is not entirely clear what measures need to be taken in order to apply crypto-agility in practice, and which aspects and challenges exist towards this endeavor. This paper surveys the state of the art of crypto-agility, as well as related works dealing with general challenges and recommendations in this regard. Additionally, we present a graphical categorization scheme of the aforementioned aspects, and provide a comprehensive list of found and surveyed work. We use our findings as starting point to initiate an open community project in the form of a website[2] to keep track of the ongoing efforts and the state-of-the-art cryptographic migration and agility research. Thereby, we offer a single entry-point into the subject reflecting the current state in a timely manner.

We choose Ott et al. [45], Macaulay and Henderson [42], and Johnson and Millett [24] as a starting point for our survey. To the best of our knowledge, these works provide the most comprehensive and thorough overviews and discussions regarding the notion of crypto-agility, as for its definition, requirements, characteristics, use-cases, benefits, drawbacks, and possible realization approaches. We trace back the multiple definitions

[1] Hochschule Darmstadt, Fachbereich Informatik
[2] https://fbi.h-da.de/cma, last accessed 2021-12-23

of crypto-agility to some of the earlier works in this regard and address its emergence in the world of cryptography. We build on that, correlating with further literature that deals with individual issues in more detail. This is done through a systematic literature review[3], using aspects addressed in the selected sources as an initial seed for a keyword-based search for the following terms and notions: *Cryptographic Agility, Crypto-Agility, Agile Cryptography, Algorithm Agility, Protocol Agility, Implementation Agility, Compliance Agility, Security Strength Agility, Migration Agility, Retirement Agility, Composability Agility, Platform agility, Context Agility, and Agile Post-Quantum Cryptography*. The keyword-based search is followed by careful selection of agility related literature, forward/backward citation chasing, and categorization of identified aspects.

The categorization scheme is presented in Fig. 2 and is also reflected in the structure of this. Sec. 2 offers a brief introduction to the notion of crypto-agility, describes the demand for crypto-agility and its scope, and sheds some light on the requirements and varying facets of crypto-agility. Sec. 3 presents several design approaches and efforts towards developing crypto-agile IT-systems. In Sec. 4 we survey the use of automation tools and (testing) frameworks in realizing and supporting crypto-agility. Sec. 5 introduces some incentives and best practices in regard to their role in encouraging and enabling crypto-agility. Sec. 6 demonstrates the need for crypto-agility in a cryptography migration process based on a current example from the expected PQC migration in electronic identification documents (eID). In Sec. 7 we present open issues and challenges, as well as respective possible solutions. Finally, we discuss our final thoughts and offer ideas for possible future work in Sec. 8. The corresponding overview in Tables 1, 2, 3, and 4 highlights the current state and provides brief commentary on the respective literature findings.

2. A Comprehensive View on Crypto-Agility

Although crypto-agility recently started to attract more attention due to the growing focus on PQC, the notion exists for more than twenty years. This term includes several facets and addresses many aspects, which makes it difficult to summarize its clear meaning in one simple definition. Multiple works provide different interpretations depending on the angle from which the notion is viewed. The term was initially introduced in the context of algorithms and encryption [4, 5]. Vasic and Mikuc later generalized the definition to include protocol security agility [8]. Ott et al. [45] address the need to broaden the notion of crypto-agility and emphasize the importance of establishing a unified understanding. This is also true for the IT-components that need to be cryptographically agile, in order to clearly define the scope of crypto-agility. In the context of requirements, we differentiate between prerequisites for realizing crypto-agility, and certain characteristic properties it needs to fulfill, including the modalities introduced in [45]. Lastly, the challenges accompanying this extended view are also addressed.

[3] Using online platforms such as Google Scholar, IEEE, Research Gate, and ACM Digital Library

2.1 Definitions

At the first glance, crypto-agility describes the feasibility of replacing and adapting cryptographic schemes in software, hardware and infrastructures [24, 39], and should enable such procedures without interrupting the flow of a running system [7]. The ISARA corporation [55] also propose the definition as the ability to adopt and integrate new cryptographic algorithms with no significant changes to the infrastructure, and without disruptions to running systems. Similarly, Ott et al. [45] suggest that crypto-agility implies the ability to apply repeated cryptographic changes (migrations) over time within a stable (non-changing) IT-architecture. Another close interpretation states that an IT-system is then considered crypto-agile, when it is able to maintain its stability towards other systems, even after adapting its cryptographic measures [77]. Last but not least, [42] states that crypto-agility can be summarized as the ability to implement, update, and replace cryptographic components within IT-systems, without affecting its functionality.

2.2 Need

Some literature claims that crypto-agility is currently most relevant for the migration from classical to PQ cryptography [39], while others state that it should apply to all cryptographic components including algorithms and protocols [13]. Most calls for agile cryptographic schemes and protocols stem from the recent focus on the expected migration towards PQC, such as in [25, 50, 28, 64, 73, 71, 53, 55, 30], where the suggested realization of crypto-agility is centered around algorithm agility, hybrid solutions and backward compatibility. Nevertheless, PQC is not the only reason to think about the necessity of applying crypto-agility to existing and future IT-systems. Macaulay and Henderson [42] state that cryptography in IT, being a discipline on its own, also requires management as any other IT-system. This encompasses the selection, implementation, maintenance, and retirement of used cryptography. In other words, it needs to be installed, updated, and deprecated instead of being static.

2.3 Scope

Ott et al. [45] name the cryptographic units, over which crypto-agility should be defined. These are algorithms, program code, protocols, applications, services, systems, distributed infrastructures, cloud services, and complex domains. Similarly, according to [24, 7], crypto-agility is deemed essential for protecting against future threats in general, and should not be restricted to PQC algorithms. Several other risk factors, besides the quantum threat, emphasizing the need for crypto-agility are addressed in [42]. These include new cryptanalysis methods, implementation flaws, side-channel attacks, and custom or sovereign cryptography. Various use-cases of crypto-agility are highlighted in [42] as well, including mitigating new cryptographic vulnerabilities, compliance to laws and standards, amortization of internet of things (IoT), platform design, cloud services and interoperability. Moreover, crypto-agility should also account for non-technical issues, such as internationalization (different standards in different countries or regions), user experience (developers as users of cryptography), and transparency [24].

2.4 Prerequisites

Macaulay and Henderson [42] suggest that managing cryptography in an agile manner requires identifying all related aspects and components such as the protected information, the expected protection time-frame, platforms and environments of operation, and known (old and new) vulnerabilities. Ott et al. [45] state that frameworks and interfaces are essential for achieving crypto-agility, which requires elaborate attention in early design phases. Paul and Niethammer [46] similarly suggest that interfaces, update mechanisms, and proper documentation are needed to provide crypto-agility in industrial automation. Also, according to [28], secure update strategies, primitive-agnostic cryptographic application programming interfaces (API), and agile cryptographic protocols are necessary to realize crypto-agility. In RFC 7696, R. Housley [13] identifies several other preconditions regarding crypto-agility. These cover a wide range of aspects focusing on agile design and implementation considerations for cryptographic protocols. Comparably, crypto-visibility and system awareness is considered an important requirement [55]. Similar prerequisites are proposed in [48], including testing, inventory creation and migration processes.

2.5 Characteristics

Ott et al. [45] formulate properties upon crypto-agility, in order to utilize its full expected potential. These identified characteristics are effectiveness, measurability, interpretability, enforceability, security, and performance. Other features presented by D. B. Nelson [7] include backward-compatibility and interoperability. Mehrez and El Omri [83] define some properties of crypto-agility naming aspects such as extendibility, flexibility, compatibility, and upgradeability. Macaulay and Henderson [42] propose similar and further properties for crypto-agility such as heterogeneity, automation, scalability, policy-awareness, interoperability, and real-time capability. In a way, most of these aspects can be seen as requirements of a well-designed agile cryptographic IT-infrastructure. This applies especially to aspects such as interoperability, backward-compatibility, and scalability.

2.6 Modalities

Ott et al. [45] refer to different manifestation forms of crypto-agility as modalities, which extend the scope of crypto-agility and display it in broader manner. The core characteristic of agility and also of these modalities, is the ability to adjust to a certain environment without any or with as little as possible human intervention. Whereas Algorithmic Agility as in algorithm compatibility issues [19] and algorithm agility models [9], is restricted to cryptographic primitives and schemes, Implementation Agility addresses the question of how the process of implementation can be supported so that applications can be crypto-agile. For example, O'Neill et al. propose a TLS API [31] to enable adapting to new cryptographic schemes and negotiate their usage with other parties. The ability to combine cryptographic keys or signatures in a secure manner, such as hybrid and composite schemes [79, 36, 27, 60, 51, 59, 21], is called Composability Agility. An example is given in a draft for composite keys and signatures for public key infrastructures (PKI) [58], where a sequence of public keys and signatures can fit in the dedicated spots for simple public keys and signature in various cryptographic structures.

Security Strength Agility is seen in enabling dynamic scaling of the algorithm security levels based on the provided configuration. This aspect applies to new vulnerabilities according to [42], and allows for fast adaptation in the case of a sudden breaking of a specific algorithm or security level. Hybrid PQC scheme / cipher-suite negotiation for TLS [60, 51], and hybrid X.509 certificates [36] can be seen as an example for enabling this property. Cryptographic algorithms also need to run on across different platform types, Platform agility should allow for independent and seamless usage on and between different types of devices, regardless of the underlying hardware and software platforms. This is especially the case for IoT and embedded systems [42], which exist in a very wide variety and in many shapes and forms. A rather similar property is Context Agility, which refers to the flexibility of algorithms and their respective security levels, so that they can be dynamically configured according to the related system attributes. Through better use of cryptographic metadata in protocols and applications such as in TLS [60] and IKEv2 [78] cipher-suite identifiers, Migration Agility can be established. This enables automatic transitions from one scheme to another if necessary. On the contrary, Retirement Agility enables enforcing the exclusion (retirement) of obsolete, insecure, or broken cryptographic algorithms. Compliance Agility is defined as the ability to reconfigure cryptographic infrastructures so that they suit different international regulations and frameworks. This aspect is also addressed in [24] with special emphasis on the various security standards in different parts of the world. In [42], compliance is also referred to this term in relation to local laws and standards, and how service providers can manage the applied cryptography to ensure their products are available for customers in all regions.

2.7 Challenges

However, broadening the spectrum of aspects covered by crypto-agility comes with a price. There are several challenges that accompany applying crypto-agility on a such scale. A first major issue is the average time-frame of a successful cryptographic update, which ranges between twelve to fifteen years [24]. This issue becomes even more complex considering embedded systems and IoT, which may not support a normal update process, and are usually used for a pre-defined life span [24]. Other hardware related challenges also exist, as Paul and Niethammer [46] also state that current protocol and application standards face additional challenges in the special case of PQC, due to size and length limitations, as well as physical limitations of the underlying communication channels. Security and complexity trade-offs are also of great concern, as having many cryptographic options opens up an unknown space for attack surfaces, such as downgrade attacks [45, 24]. Another challenge is seen in defining the right areas to insert agility within a complex infrastructure [45]. Other challenges include testing and validation, as well as refactoring legacy systems that are sometimes impossible to reconfigure [45].

3. Development Efforts

The development efforts towards agile cryptography take place on different levels of the IT-stack. Existing approaches and solutions can be divided into the following categories: Algorithm and protocol agility, design agility, hardware agility, and API agility.

3.1 Algorithm and Protocol Agility

The notion of agility and its applicability to cryptographic primitives is addressed in [5]. They provide a formal analysis of different primitives showing when they are considered agile, and when not. One core idea is that some primitives are not agile in their nature, but could be if combined with others forming a set of schemes. Thus, the support of multiple cryptographic algorithms can be interpreted as an implementation of crypto-agility. When using hybrid schemes, the keys consist of different components to be used with different algorithms. Examples for existing protocols and formats designed with algorithmic crypto-agility in mind are JCA/JCE, TLS, SSH, IKEv2 and X.509 v3 digital certificates. While TLS, SSH, and IKEv2 utilize a cipher negotiation mechanism, X.509 certificates contain flexible algorithm identifiers. For instance, Heider [41] proposed a design and implementation of the IKEv2 protocol, that can be instantiated with cryptographic hybrid schemes. A backward compatible PQC-hybrid-extension of X.509 Certificates using the qTESLA-algorithm, was proposed and implemented in [36]. A structural system to manage composite keys in a PKI is described in [58]. An extension of TLS, which uses OpenSSL together with the PQC library LibOQS to feature hybrid, post-quantum-secure TLS is described in [37]. [21] describes a hybrid scheme, that integrates NTRUEncrypt into the TLS handshake. Bindel et al. [35] propose multiple hybrid schemes, which offer more security, since they stay secure as long as at least one of the combined schemes remains secure. Hybrid schemes might be used in a transition time, in which the old algorithms types are being decommissioned, while the new algorithms are still under development. Moreover, work on enhancing existing protocols like TLS [31] and PKINIT [34] has been made. Vasic et al. [8, 22] propose a layer independent negotiation protocol similar to TLS. Heesch et al. [40] integrated a modified Version of OpenSSL, that is capable of using PQC-Algorithms, into OpenVPN and evaluated its performance. Richter at al. [77] demonstrates the usefulness of algorithm-agility through running two different protocols (Quantum Digital Signature (QDS) and Quantum Secret Sharing (QSS)) on the same hardware platform as a proof of principle. A design for an algorithm independent hybrid handshake for the Noise-Protocol is proposed in [59]. Furthermore, there are approaches and implementations of making the Signal-Protocol, which is used in the Messenger App Signal as well as other Messengers, Crypto-Agile [72, 32, 65]. These works are highly driven by the advent of Post-Quantum Cryptography.

3.2 Design Agility

Other approaches expand existing infrastructure to be able to exchange cryptographic algorithms [15, 19, 2]. Crypto-agility can also be established through design, as [15] show that key lengths and signature algorithms can be adapted. [52] follow the same design principle in being algorithm independent. In [38] a software-update mechanism with a hybrid code-signing mechanism is presented. It uses the post-quantum secure hash-based signature scheme XMSS together with classical ECDSA to ensure post-quantum security without sacrificing the security of well-known classical algorithms. A secure update mechanism enables updating other software components in order to provide algorithm-agility. As recommended in [17], a deep integration of security and cryptography into the software and development life-cycle minimizes the potential of security flaws due to misconfiguration and bugs, as well as increase the maintainability of the cryptographic components of the software.

3.3 Hardware Agility

Another field of research is the hardware side of cryptography. Cryptographic algorithms are often accelerated with the help of dedicated hardware-components, which are often algorithm specific. A mechanism to repurpose hardware designed for the use with RSA together with lattice-based algorithms is described here [82]. An FPGA based cryptographic accelerator, designed with algorithm-agility in mind is described in [1]. Also, a PQ-secure hybrid solution based on the SIKE scheme together with the ECDH based x434 based classic scheme was implemented on an artix-7 FPGA, as described in [63]. Mert et al. [74] propose a hardware implementation using a crypto-processor for lattice-based signatures and key encapsulation mechanisms (KEM), where a compact, unified instruction-set architecture leverages the synergies between similar PQC schemes.

3.4 API Agility

Another research area focuses on developing usable and agile cryptographic libraries and APIs. Lee et al. [10] implement a plug-in structure in the cryptography API: Next Generation from Microsoft to exchange cryptographic algorithms without any change to the code of the program. Michael et. al. [75] describe the Open Quantum Safe project, that contains the library libOQS, as well as a PQ-Secure version of OpenSSL. LibOQS is a library that provides many Post-Quantum-Cryptographic algorithms and makes it easy to select and interchange them in different security strength levels. Acar et al. [23] examine the usability of cryptographic libraries and found poor documentation and missing code examples to be an issue. They call for simple interfaces and code examples for common tasks. As a solution, Mindermann and Wagner [29] present a web platform for cryptographic code examples with an experiment where the participants were more effective in solving the task and the code more secure. In [56], the same authors found that code examples need to be more concrete to have a meaningful impact. Similarly [54] examine a prototypical implementation of a documentation system for two crypto APIs, showing that good documentation is essential for efficient and correct usage of an API. Zeier et al. [49] discuss the challenges of implementing multiple cryptographic schemes and integrating them into IT-systems, and present a highly abstract cryptographic API (eUCRITE) [81] to prevent implementation errors. [61] introduce a framework that makes exchange the underlying cryptographic algorithm resource-efficient. Another way to prevent mistakes on the usage of cryptography is to use very specific crypto-aware programming languages as cPCL, CAO or Cryptol, as described in [17]. Compilers of such languages can cross compile to other programming languages, for further usage of the code.

4. Quality Management

To ensure crypto-agile software is used correctly, it needs to be checked, tested and validated. Testing and validation can happen on many levels, from checking the mathematical models, over source code audits, up to checking the entire infrastructure, whether every component works in its assigned role.

4.1 Testing and Validation

Just as other software features, crypto-agility should be subject to testing and validation [45]. Open questions around this topic include the safe replacement of deprecated cryptography and the need for respective testing frameworks. As described in [17], for some programming languages, formal verification tools can be used, to prevent application defects early in the development cycle. For the programming language Cryptol a such tool exists, that can also verify the implementation. Similar tools also exist for hardware implementations. The NIST proposed a testing suite in NISTIR 800-22 [84] that tests particular building blocks of cryptographic algorithms. For this particular test-suite, Simion and Burciu [85] already proposed a simplification through a reduction of needed tests. Another approach for implementation testing is proposed in [11]. This approach is based on metamorphic testing and works without oracle based random Testing. Another metamorphic testing approach is proposed in [47].

4.2 Automation Frameworks

Another field of research is the automation of testing. NIST demonstrates a testing infrastructure that is able to automatically download test vectors and executes them on the target device under test, to validate cryptographic modules in the context of the cryptographic module validation program (CMVP), as described in [86]. In [57] an extension to NIST's CMVP is proposed, that extends the test suite with a Large Data Test, that tests the algorithms with input sizes larger than 4 GB, because vulnerabilities were detected on some implementations in regard to large input sizes. A further testing framework, that includes output-randomness, pre-defined test vectors, as well as the performance of the algorithm, is demonstrated in [12].

5. Incentives and Best Practices

Already ten years ago, RFC 7696 [7] provided a guideline on crypto-agility and algorithm selection for IETF protocols. Advantages of specific design decisions are discussed and considerations for individual implementations are introduced, often resulting in trade-offs between security on the one hand, and usability, interoperability or agility on the other. Armknecht et al. [33] also address this issue as one of the biggest failures in IT-security. They came to the conclusion, that a meaningful certification is needed. Since by today there is no common understanding of the meaning and the scope of certification, the discussions have to go on. Further, they propose an app-store-like ranking system, where adherence to security best practices results in higher rankings. Moreover, according to Ott et al. [45], it is important to provide proper incentives to implement agility into software. Recognizing that crypto-agility presents a variety of challenges, from hardware issues to processes and organizational aspects within enterprises and entire industries, Paul and Niethammer [46] identify important building blocks towards crypto-agility, namely, APIs, update mechanisms, and documentation. The use of static code analysis is by today a standard in software quality assurance. As mentioned in [17], it is quite common in security software and should be considered a best practice. Static code analyses tools may warn in case of using outdated cryptographic algorithms. In [17], it is also concluded, that a well-crafted set of tools helps on developing secure software.

6. Case Study: Crypto-Agility in the context of PQC Integration for eCards

In a former survey paper on the state of PQC [62], we discovered that the very important topic of securing electronic personal documents (eCards) such as electronic identity (eID) cards [67] or machine-readable travel documents (MRTD) [66] is an almost overlooked matter in the ongoing PQC integration efforts. At the same time, several governmental projects are conducted aiming at even broadening the usage of eCards, such as the German ID Wallet-App[4] and the German eID for EU citizens[5]. Considering the sensitivity of the eCard data, and PQC being the only available security alternative at this point, eCards need to soon adapt to the new PQC schemes thereby enabling protocol- or crypto-agility. This becomes even more pressing considering the lead time to introduce new eCards (e.g., up to ten years for eID cards [80] and the long exchange intervals (e.g., ten years for MRTDs [76]) in which these documents are issued.

As defined in the specification documents [69], currently used standard security protocols in eCards include Passive Authentication (PA), Password Authenticated Connection Establishment (PACE) and Extended Access Control (EAC), including its sub-protocols Terminal Authentication (TA) and Chip Authentication (CA) [68]. These protocols rely on different security mechanisms. Mainly, they utilize asymmetric schemes such as Diffie-Hellmann (DH) and elliptic curve (EC)DH key agreement, and Rivest–Shamir–Adleman (RSA) and EC digital signature algorithm (ECDSA) signatures, all of which, besides the threat of conventional attacks, are considered vulnerable to attacks leveraging quantum computers. This led to the need of proper cryptographic alternatives. In addition, these protocols are implemented on hardware with strict resource constraints.

Based on the initial analysis on the current state of eCards, we identify various components that require adaptation, should a transition in the used cryptography take place. These components reside on several levels of abstraction. Fig.1 shows the layers that comprise the development architecture for eCards, which can be applied to any other

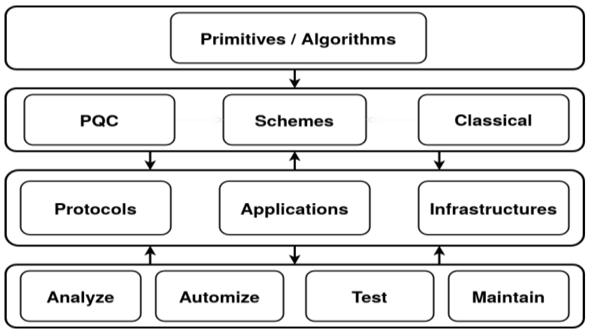

Figure 1: Crypto-Agility in PQC Migration

[4] https://www.bundesregierung.de/breg-de/suche/e-id-1962112 (2021-12-18)

[5] https://www.bundesdruckerei.de/en/Newsroom/Latest-News/Launch-eID-card-EU-citizens (2021-12-18)

IT-system utilizing cryptographic security measures. All of these layers require proper mechanisms that enable crypto-agility, starting with algorithm agility, all the way to the high-level applications.

6.1 Algorithm / Hardware Agility

This aspect is identified in the context of eCards in the ability to support different security strength levels for the implemented cryptographic schemes. This requires precise decisions regarding the mandatory and optional schemes supported by the eCard hardware, as well as by the software protocols that utilize them. However, this may prove extremely challenging, due to the hardware constraints of regular chip-cards on the one hand, and of special eCards on the other. According to the BSI [70], both ePA and ePass contain an NFC-complaint RFID chip, which must fulfill certain technical and cryptographic criteria [6] (e.g., NXP P60D145 [20] and Infineon SLC52 [16]; NFC type-4 tag [3]). Such chips contain built-in crypto co-processors that support standard cryptographic operations, which may, or may not be able to support other types of cryptography. For instance, Albrecht et al. [26] were able to implement R-LWE based lattice PQC schemes using an RSA co-processor, which does not necessarily apply to other PQC families. Possible PQC alternatives require additional hardware resources in order to be efficiently integrated into the established protocols. Therefore, modern hardware suitable for eCards is needed, which is capable of running PQC properly. The question here remains, whether the new hardware can be built with algorithm agility in mind (cf. Sec. 3).

6.2 Protocol Agility

The security protocols are designed in a way that provides security proofs based on many aspects such as the strength of the utilized cryptographic algorithms, key sharing schemes, digital signatures, as well as on communication mechanisms. For example, PACE relies on a small entropy pin for its password authenticated key agreement, which in turn uses the password to establish the DH key agreement domain parameters. This design intertwines a scheme dependent component with a higher-level protocol rendering PACE infeasible for usage with new PQC KEMs, since DH-like key exchange mechanisms (KEX) and PQC KEMs are not seamlessly interchangeable. Morgner and von der Hyden offer a concise analysis of the requirements of PQC for MRTDs [76], and identify the aforementioned protocol as one of the components that need to be adapted to make MRTDs PQ-secure. They suggest adapting PACE to implement PQC KEMs instead of DH. However, a new adapted version of the protocol does not solve the problem in regard to future agility, should PQC KEMs become insecure (cf. Sec. 3).

6.3 Infrastructure Agility

A very important component in the security of eCards is the connected PKI. Vogt and Funke [80] stress the importance of post-quantum (PQ) secure PKIs for eIDs, and shed some light on their current situation and future requirements. They offer several possible approaches for migrating certificate authorities and integrating quantum-secure certificates. They suggest using quantum-safe certificates, hybrid certificates using X.509 extensions, hybrid certificates using signature concatenation, and parallel certificate hierarchies. Whereas the second and third approach may partially solve the agility problems on the algorithm level (cf. Sec. 3), only the latter proves promising on the application level. However, it requires a major update to the entire PKI, which is not previously accounted for in the initial design.

6.4 Application Agility

Assuming that crypto-agility can be provided on the eCard side, the service terminals that communicate with them also need to enable such agility on their side. This leads back to the issue of protocol agility, since protocols on resource constrained devices are not usually designed with many security establishment options, as in high-level protocols such as TLS or IKEv2 (cf. Sec. 3). Moreover, it may not be feasible to aim for application agility in highly specific authentication use cases such as in eCard applications.

6.5 Conclusion

The aforementioned works raise several open questions and issues that need to be solved, and identify them as necessary steps required to realize the next generation of eIDs and MRTDs. These issues include finding the best candidate schemes for the resource constrained eCards; adapting the security protocols using DH to PQ key encapsulation mechanisms (KEM); a practical evaluation of the performance and security of PQC under realistic settings; the migration to stateful PQC signatures in certain eID certificates; the use of mobile and virtual eIDs; security certification of open source PQC libraries and adapting back-end PKIs of eIDs. We pose additional questions regarding the practicality of striving towards a crypto-agile approach to be applied in the design of the next generation eCards.

7. Discussion and Open Issues

In the following we discuss the findings of our literature survey and address identified challenges and open issues.

7.1 On the Notion of Crypto-Agility

Our findings in Sec. 2 indicate a great focus in the ongoing research on the definition, requirements, and application areas of crypto-agility. However, it is not possible to provide one unified definition, considering the different interpretations presented in the literature. In some sense, crypto-agility can be viewed as a generalization of migration, which formalizes the ability for repetitive change and transition. Connected with this generalization is the wish to make migration as a whole more approachable and simpler.

Several requirements and characteristics proposed for and upon crypto-agility, may or may not apply depending on the context. Moreover, it is no entirely clear, how the suggested characteristics of crypto-agility are meant to be realized. So far, concrete ideas and solutions are mostly offered for algorithms, schemes, and protocols. This can be traced back to the fact that this aspect is most interesting for the PQC development issues, but no necessarily for cryptography as a whole. This can be recognized in the missing prerequisites in aspects such as interfaces, frameworks, and industrial automation tools. Modalities such as implementation agility can be solved through frameworks. However, ensuring the desired cryptographic agility within the relevant IT-systems cannot be considered a practical endeavor. Currently there are too few automated tools, and dedicated frameworks capable of managing the configuration of cryptographic components, let alone sustaining any cryptographic agility on a large scale of algorithmic agility. Such automation should preferably be able to identify, analyze, deploy or replace cryptographic components within IT systems according to system and context attributes, following well studied and planned strategies and guidelines. Algorithm and composability agility realized through combining sets of keys and signatures for cryptographic schemes, does not suffice if the implementations are not able to handle the given diversity. Additionally, this property is still restricted to hybrid schemes in current implementations. The same issues apply to security strength agility, as it relies on algorithm parameters and protocol configuration. Other modalities did not get enough attention so far. Overall, most modalities can be summarized under the general term of context agility, the ability to adapt to a different context. And while as a future goal this ability seems to be a promising research candidate that might solve many current issues, it is still a long way to reach it. It should be mentioned that all of these abilities and modalities could introduce severe vulnerabilities but at least will introduce more complexity which should only carefully be increased.

To sum up, there is currently no coherent definition available. Even more, there are various motivations, requirements, benefits, challenges, and characteristics related to crypto-agility. We propose establishing a clear holistic definition of crypto-agility, regardless of the various facets it should apply to. This definition needs to remain separate from the related properties, prerequisites, and modalities. This proposal is based on the fact that it is practically infeasible to fulfill all of the aforementioned aspects in all known types of IT-systems.

7.2 Development Considerations

As shown in Sec. 3, we still need cryptographic libraries following crypto-agile design principles. Current approaches are cipher negotiations or utilizing hybrid cryptography. Research for different architectural ideas is thin and widely distributed between fields and mainly concentrates on improving cipher negotiation schemes, working with plug-in structures and adaptable key lengths hash and signature algorithm. We need a change in direction for developing current or new cryptography libraries. If these do not follow crypto-agile principles it will bring us back to the issue we have now, that constant adjustment is needed and development / maintenance costs will occur. Nonetheless, new crypto-agile schemes will introduce new complexity and therefore attack surfaces, that need to be researched and understood. In terms of user interfaces for different end users, questions asked in [45] are still unanswered. Should software development principles not be changed, and other general purpose crypto-agile schemes be presented, questions about attack surfaces and user interfaces will be on hold and will reemerge again in the near future.

Sec. 3 also shows that the whole tooling for developers plays a huge role in preventing and finding software defects early. A critical point is the static code analysis, which should be crypto-aware. Also, there is a need for proper and more documentation of crypto-libraries, as well as easy to use APIs, to minimize the risk of creating software-defects through wrong usage. Furthermore, there do exist specific crypto-aware programming languages which are designed to help developers to prevent bugs in the field of cryptography. In our opinion, developers should be encouraged to setup and use tools which help on developing secure software. Crypto-agile libraries and APIs should minimize the manual maintenance of any developer adjusting source code when new cryptographic algorithms need to be used. In an ideal world, a developer would only update the used libraries. The idea is to have widely used cryptographic libraries with a crypto-agile design.

The development of Hardware based crypto-accelerators is also an active field of development. Current hardware often uses accelerators for standard cryptographic functions. There is ongoing work in repurposing this hardware for PQC Algorithms. This could reduce the need to change already deployed hardware. Also, there is new hardware under development explicitly for PQC algorithms, which could be critical if these algorithms have to be changed again. Even two decades ago an FPGA based general purpose cryptographic accelerator was developed, which was based on the idea, that FPGAs can be reprogrammed. Further, hardware/software hybrid accelerator approaches are developed, as described in [73], which have lower-level building blocks in hardware and organize these in software. This is considered a huge step towards crypto-agility, since there is no need for workarounds to keep the hardware in use, when cryptographic algorithms change. Another challenge is that many implementations and protocols are dependent on specific key lengths. In such cases, these have to be adapted or made independent of key-lengths, however this work has to be only done once, when implementing algorithm-agility. Since the complexity of software increases with crypt-agility, the potential for bugs and therewith for security issues also increases. Moreover, especially in embedded applications resources like RAM, clock speed, storage and power might be constrained, where additional complexity might bring additional resource-consumption or new algorithms might not be suited anymore. Also, the hardware might need more generic crypto-accelerators, which could be an additional effort in development and product cost.

Given the task of constant change, the question is left whether emerging cryptographic schemes and applications, that can differ considerably in structure and use case from current ones, are able to adapt to migration and crypto-agility. It is not clear, how certain categories of cryptography like fully homomorphic encryption (FHE), password authenticated key agreement (PAKE), blockchain or threshold cryptography are going to react to migration and cryptographic agility. Some of these put constraints on the amount of agility they are able to support. Blockchain technologies for example seem prepared for PQC but cannot be agile because of their protocols [44]. Other areas that face heavy constraints to their ability to implement agile solutions are satellite-based communication systems. These systems can adapt to PQC but can only be updated in very limited circumstances if at all [43].

7.3 Testing

As Sec. 4 shows, advanced testing suites to test the security and quality of implementations are existing and still extended by new attack vectors. Also test methods are continuously refined and optimized. Moreover, test automation frameworks for automated testing of implementations against various test-vectors exist. The existing test-suites are supposed to detect bugs as well as common attack vectors. But they don't detect existing or missing agility, as it might be hard to measure agility. Also, as described in Sec. 3, static code analysis tools do exist. As these tools have access to the source code, they could check for abstraction patterns or fixed sizes and warn about potential missing agility. As research in the crypto-agility field is still in an early stage, testing and validation has to be examined more.

7.4 Incentives & Best Practices

If looking at the effort and time it took for historic cryptographic updates (e.g., DES, RC4, MD5 or SHA-1) to be executed is not incentive enough, the nowadays very strict data protection laws in many countries around the world (e.g., GDPR in Europe) along with their respective legal (i.e., monetary) consequences should be sufficiently convincing. The question which incentives for introducing crypto-agility could or should be given is still open. Besides the possible competitive advantage mentioned in 5, further answers could include adherence to existing laws (e.g., GDPR in Europe) and reduction of cost (e.g., for switching algorithms). It is also thinkable that new regulations will appear, demanding crypto-agility for applications in certain contexts, e.g., for critical infrastructures. All said apart, we need to find out how to motivate the developers to want to weave crypto-agility into their code. Equipping the very cryptographic libraries with (semi-)automatic agile features will surely lower the bound. As described in 5, potential ways to motivate developers could be certifications or rankings of products, to give developers incentives to make software agile. These rankings and certificates are not jet specified and could be a future field of study.

Regarding best practices, RFC 7696 [7] (cf. Sec. 5) only covers a small fraction of the vast field of possible application areas of crypto-agility. Further research and hands-on development are needed here, too. The community must develop a common understanding including definitions, terms and language in general in order to support the discussion on crypto-agility and place the various contributions on a common ground. In our view, a widely accepted reference model should be developed for this purpose. It is also thinkable to reuse some of the best practices in general software design, as crypto-agility is now seen as an independent discipline in the field of software engineering.

8. Discussion and Open Issues

In this paper we presented a literature survey on the state of the art in crypto-agility. We addressed its definition, scope, characteristics and requirements, as well as current development efforts and challenges towards realizing crypto-agility in existing and future IT-systems. This includes algorithm and protocol design, deployment strategies, legacy systems, testing frameworks and process automation. Much attention is given to algorithm and protocol agility, especially in relation to development efforts in PQC. However, other aspects in the wider scope of crypto-agility need to be sufficiently addressed, such as testing and automation. The research community has to make sure that new classes of applications that rely on cryptography are examined for their suitability to make use of crypto-agility, as some categories of cryptography put constraints on the amount of agility, they are able to support. Last but not least, the community needs to develop a common understanding including definitions, goals, and terms in order to support the discussion on crypto-agility and place the various contributions on common ground. In our view, a widely accepted reference model should be developed for this purpose, as well as automated tools and managed processes, which are still limited in availability.

The overview and findings at hand provide a starting point for giving answers to the yet untouched issues. We identified challenges on the way towards large-scale crypto-agility, such as hardware limitations, security and complexity trade-offs, and legacy systems. Identifying open issues is, however, only the first step in our research. As this can only be a snapshot of the current state, we initiated a website (https://fbi.h-da.de/cma) that we keep updating and invite the community to support us in keeping track of the current state over time. We are also working on the development of solutions aiding the community in establishing crypto-agility and in transitioning to PQC. For example, a first prototype of an easy-to-use cryptographic interface called eUCRITE (https://fbi.h-da.de/eucrite), providing minimal knowledge abstractions for both conventional and PQC functionalities is evolved. Moreover, we are developing a reference model for evaluating the maturity of crypto-agility in a given system, as well as an automated crypto-detection tool to support inventory creation cryptographic components in IT-infrastructures.

Acknowledgment: This research work has been funded by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cyber-Security ATHENE.

Appendix

Categorization Scheme

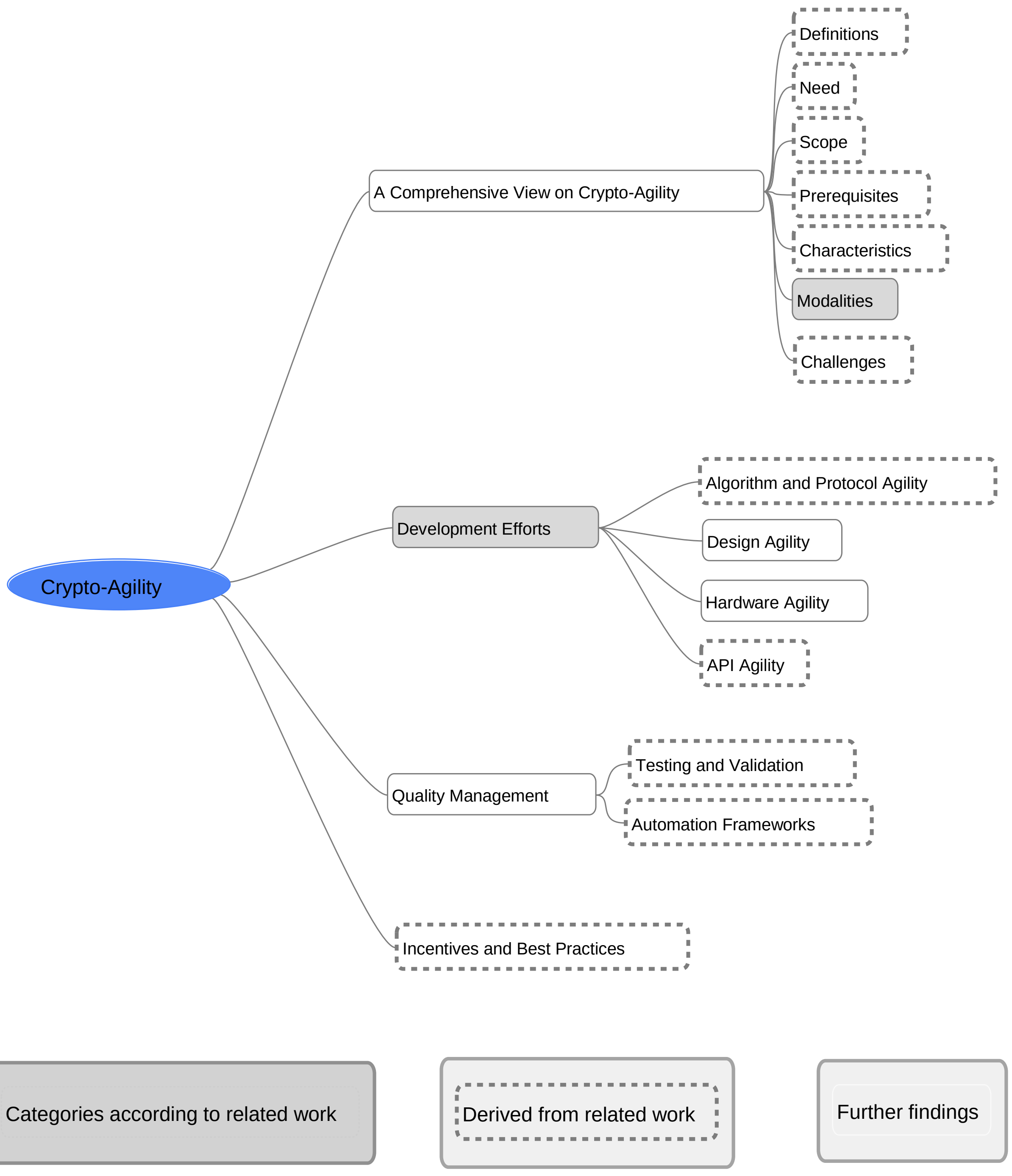

Figure 2: Categories of Crypto-Agility

Survey Overview

Tab. 1: Overview: The Notion of Crypto-Agility

Definitions	
Crypto-Agility in algorithms, encryption schemes, and protocols	[3, 4, 6]
Ability to migrate crypto repeatedly without changing IT-architecture	[41, 51]
Feasibility of adapting crypto schemes in HW, SW, and infrastructures without interrupting running systems	[20, 35, 5]
Maintain stability after adapting crypto measures	[68]
Ability to implement, update, and replace crypto without affecting functionality	[38]
Need	
Migration to PQC	[35, 21, 46, 24, 60, 64, 62, 49, 26, 51]
Management of all crypto components	[11, 38, 20, 41]
Scope	
Data encryption standards	[26]
Vulnerabilities, compliance, IoT, platform design, cloud, and interoperability	[38, 41]
Cryptanalysis methods, implementation flaws, and side-channel attacks	[38]
Crypto units: algorithms, code, protocols, apps, services, systems etc.	[41]
Protecting against all future threats and not only PQC	[20, 5]
Internationalization, user experience, and transparency	[20]
Prerequisites	
Information to protected data, protection time, platform, and vulnerabilities	[38]
APIs, Interfaces, frameworks, update mechanisms, and documentation	[41, 42, 24]
Agile design, crypto-visibility, awareness, testing, inventory, and processes	[11, 51, 44]
Characteristics	
Heterogeneity, automation, scalability, policy-awareness, and interoperability	[38]

Effectiveness, measurability, security, performance etc.	[41]
Backward-compatibility, interoperability	[5]
Extendibility, flexibility, compatibility, and upgradeability	[83]
Modalities	
Algorithmic Agility	[41, 16, 7]
Implementation Agility	[41, 27]
Composability, Context, Migration, and Retirement Agility	[41]
Security Strength Agility	[41, 38]
Platform agility	[41, 38]
Compliance Agility	[41, 38, 20]
Challenges	
Security and complexity trade-offs (new attack surfaces)	[41, 20]
Average time-frame of crypto updates	[20]
Hardware constraints on embedded and IoT devices (key sizes and PQC)	[42, 20]
Testing, validation, and refactoring legacy systems	[41]

Tab. 2: Overview: Development Efforts

Algorithm and Protocol Agility	
Notion of Crypto-Agility, Applicability to crypt. primitives	[4]
Implementation of crypto-agility in IKEv2	[37]
Extending X.509-Compliant Certificates for Hybrid Schemes	[32]
Proposals for structuring hybrid keys	[54]
Integrating hybrid keys into protocols	[33, 17, 31, 55]
Proposal to integrate SSL into Operating systems	[27]
Making PKINIT Algorithm-Agile	[30]
Alternative to TLS	[6, 18]
Integration of PQC in VPN Protocols	[36]
Demonstration of the usefulness of Algorithm-Agility	[68]

Making Signal Protocol algorithm-agile	[63, 28, 61]
PKI Crypto-agility	[13, 2]
Hardware Agility	
Algorithm Agility of Trusted Platform Modules	[16]
Algorithm-Agility	[48]
Update Mechanisms	[34]
Development Lifecycle	[14]
Algorithm-Agile hardware acceleration	[75, 1, 59, 65]
API Agility	
Algorithm-agility of libraries	[8, 57]
PQC Libraries	[66]
Documentation Issues	[19, 25, 52, 50]
Usability of APIs	[45, 72]
Crypto-aware programming languages	[14]

Tab. 3: Overview: Quality Management

Testing and Validation	
Need for testing and validation	[41]
Formal verification tools to prevent early defects	[14]
Statistical test-suite for number generators in crypto applications	[84, 85]
Property-based testing framework for encryption programs	[9]
Systematic testing of PQC implementations using metamorphic Testing	[43]
Automation and Frameworks	
NIST Cryptographic module validation program (CMVP), and extensions	[86, 53]
Framework for security and resource consumption of crypto algorithms	[10]

Tab. 4: Overview: Incentives and Best Practices

Guidelines for algorithm selection in IETTF protocols	[5]
App-store-like ranking system	[29]
Need for proper incentives and best practices	[41]
APIs, update mechanisms, and documentation	[42]
Static code analysis in quality assurance	[14]

References

[1] Christof Paar et al. “Algorithm-agile cryptographic coprocessor based on FPGAs”. In: Reconfigurable Technology: FPGAs for Computing and Applications. SPIE, 1999, pp. 11–16.

[2] Jan Sönke Maseberg. “Fail-Safe-Konzept für Public-Key-Infrastrukturen”. PhD thesis. Technische Universität, Aug. 2002.¨

[3] Bryan Sullivan. “Cryptographic Agility”. In: MSDN Magazine (2009).

[4] Tolga Acar et al. “Cryptographic Agility and Its Relation to Circular Encryption”. In: Advances in Cryptology – EUROCRYPT 2010. Springer Berlin Heidelberg, 2010, pp. 403–422.

[5] David B. Nelson. Crypto-Agility Requirements for Remote Authentication Dial-In User Service (RADIUS). Nov. 2011. DOI: 10.17487/RFC6421.

[6] Valter Vasic and Miljenko Mikuc. “Security Agility Solution Independent of the Underlaying Protocol Architecture.” In: AT. 2012, pp. 128–137.

[7] Roque Gagliano, Stephen Kent, and Sean Turner. Algorithm Agility Procedure for the Resource Public Key Infrastructure (RPKI). Request for Comments. RFC Editor, Apr. 2013. DOI: 10.17487/RFC6916.

[8] Kyungroul Lee et al. “Security Issues on the CNG Cryptography Library (Cryptography API: Next Generation)”. In: 2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. IEEE, July 2013, pp. 709–713. DOI: 10.1109/IMIS.2013.128.

[9] Chang-ai Sun, Zuoyi Wang, and Guan Wang. “A property-based testing framework for encryption programs”. In: Frontiers of Computer Science (June 1, 2014), pp. 478–489. DOI: 10.1007/s11704-014-3040-y.

[10] Cristina-Loredana Duta et al. “Evaluation Framework for Security and Resource Consumption of Cryptographic Algorithms”. In: Journal of Control Engineering and Applied Informatics (2015), pp. 77–87.

[11] R. Housley. Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms. RFC Editor, Nov. 2015, RFC7696. DOI: 10.17487/RFC7696.

[12] Michele Mosca. Cybersecurity in an era with quantum computers: will we be ready? 2015.

[13] Markus Ullmann, Christian Wieschebrink, and Dennis Kugler. “Public Key Infrastructure and Crypto Agility Concept for Intelligent Transportation Systems”. In: (2015), p. 6.

[14] Alexandre Braga and Ricardo Dahab. “Towards a Methodology for the Development of Secure Cryptographic Software”. In: Aug. 2016.

[15] Lily Chen et al. Report on Post-Quantum Cryptography. National Institute of Standards and Technology, Apr. 2016, NIST IR 8105. DOI: 10.6028/NIST.IR.8105.

[16] Liqun Chen and Rainer Urian. “Algorithm Agility – Discussion on TPM 2.0 ECC Functionalities”. In: Security Standardization Research. Springer International Publishing, 2016, pp. 141–159. DOI: 10.1007/978-3-319-49100-4_6.

[17] John M. Schanck, William Whyte, and Zhenfei Zhang. Quantum-Safe Hybrid (QSH) Ciphersuite for Transport Layer Security (TLS) version 1.2. Internet-Draft. Internet Engineering Task Force, July 2016.

[18] Valter Vasic, Miljenko Mikuc, and Marin Vukolić. “Lightweight and adaptable solution for security agility.”´ In: KSII Transactions on Internet & Information Systems (2016).

[19] Yasemin Acar et al. “Comparing the Usability of Cryptographic APIs”. In: 2017 IEEE Symposium on Security and Privacy (SP). IEEE, May 2017, pp. 154–171. DOI: 10.1109/SP.2017.52.

[20] Engineering National Academies of Sciences and Medicine. Cryptographic Agility and Interoperability: Proceedings of a Workshop. The National Academies Press, 2017. DOI: 10.17226/24636.

[21] Ruben Niederhagen and Michael Waidner. Practical Post-Quantum Cryptography. Fraunhofer SIT, Aug. 2017.

[22] Martin R. Albrecht et al. “Implementing RLWE-based Schemes Using an RSA Co-Processor”. In: IACR Transactions on Cryptographic Hardware and Embedded Systems (Nov. 9, 2018), pp. 169–208. DOI: 10.46586/tches.v2019.i1.169-208.

[23] Franziskus Kiefer and Kris Kwiatkowski. Hybrid ECDHE-SIDH Key Exchange for TLS. Internet-Draft. Internet Engineering Task Force, Nov. 2018.

[24] Michael Kreutzer, Ruben Niederhagen, and Michael Waidner. Eberbacher Gespräch: Next Generation¨ Crypto. Fraunhofer SIT. Jan. 2018.

[25] Kai Mindermann and Stefan Wagner. “Usability and Security Effects of Code Examples on Crypto APIs”. In: 2018 16th Annual Conference on Privacy, Security and Trust (PST). IEEE, Aug. 2018, pp. 1–2. DOI:10.1109/PST.2018.8514203.

[26] Nagy Moustafa and Vladimir Soukharev. Crypto Agility Is a Must-Have for Data Encryption Standards. 2018.

[27] Mark O’Neill et al. “The Secure Socket API: TLS as an Operating System Service”. In: 27th USENIX Security Symposium (USENIX Security 18). USENIX Association, 2018, pp. 799–816.

[28] Joel Alwen, Sandro Coretti, and Yevgeniy Dodis. “The Double Ratchet: Security Notions, Proofs, and Modularization for the Signal Protocol”. In: Advances in Cryptology – EUROCRYPT 2019. Lecture Notes in Computer Science. Springer International Publishing, 2019, pp. 129–158. DOI: 10.1007/978-3-030-17653-2_5.

[29] Frederik Armknecht et al. “Biggest Failures in Security”. In: (2019), p. 23.

[30] L. Hornquist Astrand et al. “Public Key Cryptography for Initial Authentication in Kerberos (PKINIT) Algorithm Agility”. In: (2019).

[31] Nina Bindel et al. “Hybrid Key Encapsulation Mechanisms and Authenticated Key Exchange”. In: PostQuantum Cryptography. Springer International Publishing, 2019, pp. 206–226. DOI: 10.1007/978-3-030-25510-7_12.

[32] Nina Bindel et al. “X.509-Compliant Hybrid Certificates for the Post-Quantum Transition”. In: Journal of Open Source Software (Aug. 12, 2019), p. 1606. DOI: 10.21105/joss.01606.

[33] Eric Crockett, Christian Paquin, and Douglas Stebila. Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH. 2019.

[34] Stefan-Lukas Gazdag, Markus Friedl, and Daniel Loebenberger. “Post-Quantum Software Updates”. In:(2019). DOI: 10.18420/INF2019_63.

[35] Olaf Grote, Andreas Ahrens, and Cesar Benavente-Peces. “Paradigm of Post-quantum Cryptography and´ Crypto-agility: Strategy Approach of Quantum-safe Techniques:” in: Proceedings of the 9th International Conference on Pervasive and Embedded Computing and Communication Systems. SCITEPRESS - Science and Technology Publications, 2019, pp. 91–98. DOI: 10.5220/0008162800910098.

[36] Maran van Heesch et al. Towards Quantum-Safe VPNs and Internet. 2019.

[37] Tobias Heider. “Towards a Verifiably Secure Quantum-Resistant Key Exchange in IKEv2”. PhD thesis. Ludwig-Maximilians-Universitat München, Oct. 28, 2019.¨

[38] Tyson Macaulay and Richard Henderson. Cryptographic Agility in practice: emerging use-cases. 2019.

[39] Andrew Neish, Todd Walter, and Per Enge. “Quantum-resistant authentication algorithms for satellite-based augmentation systems”. In: Navigation (2019), pp. 199–209. DOI: 10.1002/navi.287.

[40] M. D. Noel et al. “Stateful Hash-based Digital Signature Schemes for Bitcoin Cryptocurrency”. In: 201915th International Conference on Electronics, Computer and Computation (ICECCO). 2019, pp. 1–6. DOI:10.1109/ICECCO48375.2019.9043192.

[41] David Ott, Christopher Peikert, and other workshop participants. “Identifying Research Challenges in Post Quantum Cryptography Migration and Cryptographic Agility”. In: arXiv:1909.07353 [cs] (Sept. 16, 2019). arXiv: 1909.07353.

[42] Sebastian Paul and Melanie Niethammer. “On the importance of cryptographic agility for industrial automation: Preparing industrial systems for the quantum computing era”. In: at - Automatisierungstechnik (May 27, 2019), pp. 402–416. DOI: 10.1515/auto-2019-0019.

[43] Sydney Pugh et al. “Systematic Testing of Post-Quantum Cryptographic Implementations Using Metamorphic Testing”. In: 2019 IEEE/ACM 4th International Workshop on Metamorphic Testing (MET). IEEE, May 2019, pp. 2–8. DOI:10.1109/MET.2019.00009.

[44] Graham Steel. Blog - Achieving ’Crypto Agility’. Cryptosense. 2019.

[45] Alexander Zeier, Alexander Wiesmaier, and Andreas Heinemann. “API Usability of Stateful Signature Schemes”. In: Advances in Information and Computer Security. Springer International Publishing, 2019, pp. 221–240.

[46] BSI. “Migration zu Post-Quanten-Kryptografie - Handlungsempfehlungen des BSI”. In: (Aug. 2020), p. 9.

[47] Eric Crockett and Matt Campagna. Internet-Draft: Hybrid Post-Quantum Key Encapsulation Methods (PQ KEM) for Transport Layer Security 1.2 (TLS). Mar. 6, 2020.

[48] Bhargav Das et al. PQ-Fabric: A Permissioned Blockchain Secure from Both Classical and Quantum Attacks. Oct. 13, 2020.

[49] ETSI. Migration strategies and recommendations to Quantum Safe schemes. July 2020.

[50] Rolf Huesmann et al. “Zur Benutzbarkeit und Verwendung von API-Dokumentationen”. In: arXiv:2007.04983 [cs] (2020). DOI: 10.18420/muc2020-ws119-002. arXiv: 2007.04983.

[51] ISARA Corporation. Managing Cryptographic and Quantum Risk. July 24, 2020.

[52] K. Mindermann and S. Wagner. “Fluid Intelligence Doesn’t Matter! Effects of Code Examples on the Usability of Crypto APIs”. In: 2020 IEEE/ACM 42nd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion). Oct. 2020, pp. 306–307.

[53] Nicky Mouha and Christopher Celi. “Extending NIST’s CAVP Testing of Cryptographic Hash Function Implementations”. In: Topics in Cryptology – CT-RSA 2020. Springer International Publishing, 2020, pp. 129– 145.

[54] Mike Ounsworth and Massimiliano Pala. Composite Keys and Signatures For Use In Internet PKI. Internet Draft. Internet Engineering Task Force, July 2020.

[55] Trevor Perrin. KEM-based Hybrid Forward Secrecy for Noise. Noiseprotocol.org, Feb. 10, 2020.

[56] D. Stebila, S. Fluhrer, and S. Gueron. Internet-Draft: Hybrid key exchange in TLS 1.3. Feb. 12, 2020.

[57] Matthew Thornton. “Agile quantum cryptography and non-classical state generation: Agile quantum cryptography and non-classical state generation”. PhD thesis. University of St Andrews, 2020. DOI: 10.17630/sta/20.

[58] N. Alnahawi et al. “On the State of Post-Quantum Cryptography Migration”. In: INFORMATIK’21 – PQKP Workshop. GI-Edition: Lecture Notes in Informatics (LNI). 2021.

[59] Reza Azarderakhsh et al. Hardware Deployment of Hybrid PQC. 2021.

[60] William Barker, Murugiah Souppaya, and William Newhouse. [Project Description] Migration to PostQuantum Cryptography. NIST, Aug. 4, 2021, p. 16.

[61] Jacqueline Brendel et al. Post-quantum Asynchronous Deniable Key Exchange and the Signal Handshake. 2021.

[62] Dr. Heike Hagemeier, Dr. Stavros Kousidis, and Dr. Thomas Wunderer. “Standardisierung von Post-Quanten-Kryptografie und Empfehlungen des BSI”. In: Tagungsband zum 17. Deutschen IT-Sicherheitskongress. SecuMedia Verlag, Ingelheim, Germany, Feb. 2021, pp. 382–294.

[63] Keitaro Hashimoto et al. An Efficient and Generic Construction for Signal’s Handshake (X3DH): PostQuantum, State Leakage Secure, and Deniable. 2021.

[64] Dr. Tobias Hemmert et al. “Quantencomputerresistente Kryptografie: Aktuelle Aktivitäten und Fragestellungen”. In: Tagungsband zum 17. Deutschen IT-Sicherheitskongress. SecuMedia Verlag, Ingelheim, Germany, Feb. 2021, pp. 367–380.

[65] Ahmet Can Mert. A Unified Cryptoprocessor for Lattice-based Signature and Key-exchange. Nov. 7, 2021.

[66] Baentsch Michael et al. Updates from the Open Quantum Safe Project. Apr. 23, 2021.

[67] Frank Morgner and Jonas von der Heyden. “Analyzing Requirements for Post Quantum Secure Machine Readable Travel Documents”. In: Gesellschaft für Informatik e.V., 2021.¨

[68] Stefan Richter et al. “Agile and versatile quantum communication: Signatures and secrets”. In: Physical Review X (2021), p. 011038.

[69] Valery Smyslov. Intermediate Exchange in the IKEv2 Protocol. Internet-Draft. Internet Engineering Task Force, Mar. 2021.

[70] Sara Stadler et al. Hybrid Signal protocol for post-quantum email encryption. 2021.

[71] Sebastian Vogt and Holger Funke. “How Quantum Computers threat security of PKIs and thus eIDs”. In: 2021.

[72] Alexander Zeier, Alexander Wiesmaier, and Andreas Heinemann. “Zur Integration von Post-Quantum Verfahren in bestehende Softwareprodukte”. In: arXiv:2102.00157 [cs] (Jan. 30, 2021). arXiv: 2102.00157. [73] .

[74] Infineon Technologies AG. SLC52GDA448 Datasheet. https://www.infineon.com/cms/ en/product/security-smart-card-solutions/security-controllers/slc52/ slc52gda448/.

[75] Joppe W Bos, Joost Renes, and Christine van Vredendaal. “Post-Quantum Cryptography with Contemporary Co-Processors”. In: (), p. 15.

[76] German Federal Office for Information Security (BSI). Architecture electronic Identity Card and electronic Resident Permit. Technical Guideline TR-03127.

[77] German Federal Office for Information Security (BSI). Der elektronische Reisepass (e-Pass). https://www.bsi.bund.de/DE/Themen/Oeffentliche-Verwaltung/ElektronischeIdentitaeten/Elektronische-Ausweisdokumente/Elektronischer-Reisepass/ elektronischer-reisepass_node.html.

[78] German Federal Office for Information Security (BSI). Der Personalausweis. https://www.bsi. bund.de/DE/Themen/Oeffentliche-Verwaltung/Elektronische-Identitaeten/ Elektronische-Ausweisdokumente/Der-Personalausweis/der-personalausweis_ node.html.

[79] German Federal Office for Information Security (BSI). Security Mechanisms in Electronic ID Documents. https://www.bsi.bund.de/EN/Topics/ElectrIDDocuments/SecurityMechanisms/ securitymechanisms_node.html

[80] German Federal Office for Information Security (BSI). TR-03110 Technical Guideline – Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token. https://www.bsi.bund. de/EN/Service-Navi/Publications/TechnicalGuidelines/TR03110/BSITR03110. html.

[81] German Federal Office for Information Security (BSI). Zertifizierte Produkte – Elektronische Passe & Ausweise. https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Zertifizierung-und-Anerkennung/Listen/ Zertifizierte-Produkte-nach-TR/Paesse_Ausweise/Paesse_Ausweise.html.

[82] NXP Semiconductors Germany GmbH. P60D145 SDS Datasheet. https://www.nxp.com/docs/en/data-sheet/P60D145_SDS.pdf.

[83] Hassane Aissaoui Mehrez and EL Othmane. “The Crypto-Agility Properties”. In: ().

[84] Andrew Rukhin et al. “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”. In: (), p. 131.

[85] Emil Simion and Paul Burciu. “THE POWER OF NIST CRYPTOGRAPHIC STATISTICAL TESTSSUITE”. In: (), p. 10.

[86] Apostol Vassilev. “Automation of the Cryptographic Module Validation Program (CMVP)”. In: (), p. 13.