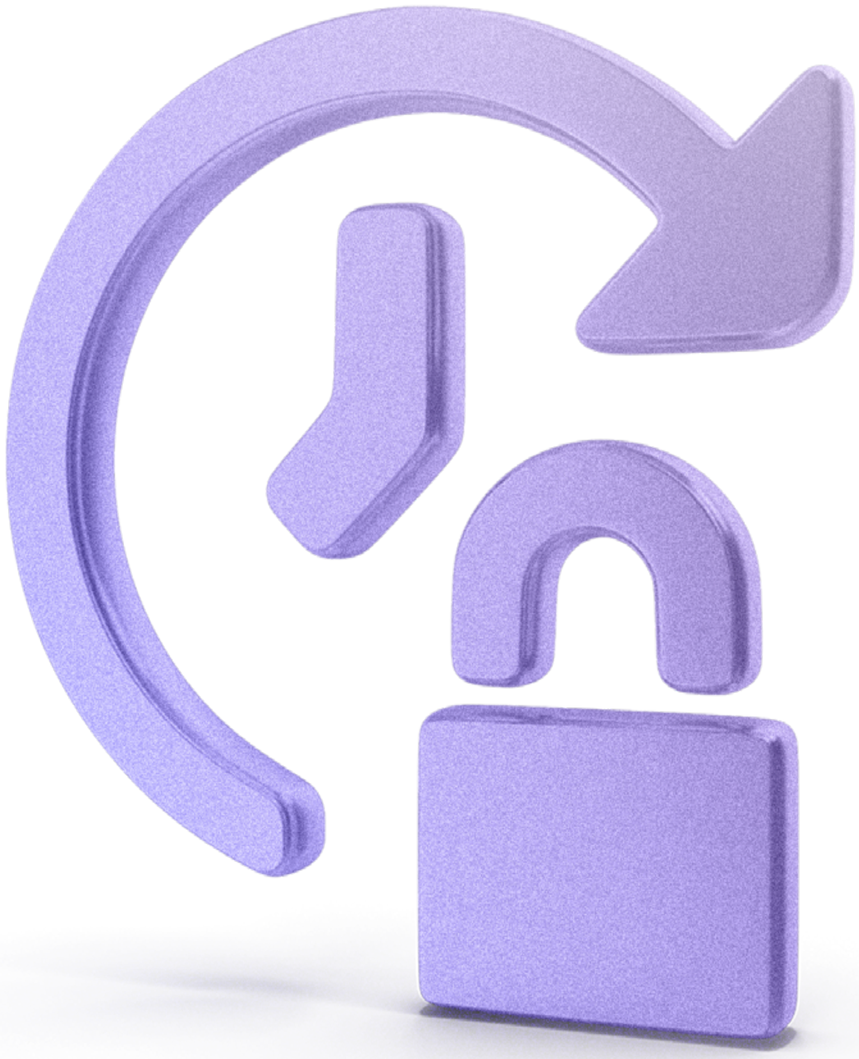
FS-ISAC

Building Cryptographic Agility in the Financial Sector

Effective, Efficient Change in a Post Quantum World



October 2024

Contents

Executive Summary

The concept of cryptographic agility has been known for at least two decades.[1] The FS-ISAC Post-Quantum Cryptography Working Group's definition is:

> ***Cryptographic Agility*** *is a measure of an organization's ability to adapt cryptographic solutions or algorithms (including their parameters and keys) quickly and efficiently in response to developments in cryptanalysis, emerging threats, technological advances, and/or vulnerabilities.*
>
> *It is also a design principle for implementing, updating, replacing, running, and adapting cryptography and related business processes and policies with no significant architectural changes, minimal disruption to business operations, and short transition time.*

The goal of cryptographic agility – or crypto agility – is simple: to enable business continuity if/when existing cryptography is compromised or weakened.

The move to crypto agility must begin immediately because quantum computing is likely to make a commonly used class of cryptography algorithms insecure in the next few years. The financial services sector cannot risk insecure data transmission or storage – it would break the way we conduct business today. And as the number of systems, dependencies between systems, and overall technical complexity grow, the effort to update cryptographic assets has intensified.

The sector has been through several cryptographic transitions in the past and each was more complex and time-consuming than the last. It is an endless cycle and becoming unsustainable. By becoming crypto agile, practitioners will be able to quickly replace cryptographic algorithms using a repeatable process with minimal impact or downtime and provide sufficient confidentiality, integrity, and/or non-repudiation guarantees. Importantly, firms will avoid the vulnerabilities of insecure algorithms.

This paper, written by the FS-ISAC Post-Quantum Cryptography Working Group (PQC Working Group), explains the process in detail and provides guidance and advice to financial services firms – or those of any industry – to help them become crypto agile. The document provides these key concepts:

- A framework for implementing crypto agility
- Challenges and how to overcome them
- Insights on transition governance and architecture

We leverage and expand on crypto agility concepts coined at Google,[2] IBM,[3] Hochschule Darmstadt (h_da)[4] and others and provide new information as well. As such, this is the first comprehensive document to detail precisely what is needed to implement crypto agility in financial services institutions.

The first part of the paper is an executive brief that describes what's new and distinctive about crypto agility. It will help executives, business leaders, board members, and other stakeholders understand how crypto agility impacts decisions for the business.

The second part offers more detailed technical definitions of crypto agility. It is meant for technologists and practitioners who manage applications, systems, infrastructure, operations, security, risk, and other issues.

Definitions and an API service schema are available in the Appendix.

The FS-ISAC PQC Working Group produced this document to help financial institutions prepare for a post-quantum world. Our intention is to help the sector manage change for the next likely transitions in information security. The sector must demonstrate to our customers, boards, and regulators that we're taking the challenge head on, working together, and preparing for the long term. Embracing crypto agility is a necessary element of that.

Crypto Agility Transition Prerequisites

While quantum computing increases the urgency of transitioning to crypto agility, the industry has undergone many cryptographic transitions over the years. It's never easy but it is essential: outdated algorithms lower the barrier to breaches.

Cryptographic agility is the most effective, efficient way forward. Implementing it can be facilitated with a few key prerequisites, specifically:

- Understanding the current state of cryptographic operations in your firm, such as key rotations, issuing certificates, etc.
  - The PQC Working Group's document Current State (Crypto Agility) Technical Paper can help you determine how your cryptography is managed.
- Knowing the current state of your firm's cryptographic inventory. Know where your keys are stored, what systems are using what versions of cryptographic algorithms, and the like. This is a crucial undertaking because you can't transition what you don't know. You'll have to keep this up to date as you become more agile and change algorithms more often than you do now.
  - The PQC Working Group's Infrastructure Inventory Technical Paper will help.
- Prioritizing your efforts. You may want to tackle the riskiest assets first, and FS-ISAC's summary of leading risk models can help you ascertain the most important assets to protect from a cryptographic viewpoint:

This approach will give you a holistic lens and help you prepare for your transition to crypto agility, as well as the infrastructural and operational changes to expect in moving to a "crypto-as-a-service" approach and migrating individual systems and infrastructure.

Part 1: Why a Crypto Agile Approach to Infrastructure Change is a Security and Business Necessity

Introduction: The Impact of Quantum Computing

The financial services sector has undergone many cryptographic transitions over the years. Each of those transitions was more complex and time-consuming than the last. And each was treated as a one-off because technologists believed that cryptographic algorithms, such as one called RSA, would suffice for our lifetimes. It probably won't.

The Goals of Crypto Agility

> Allow migration to new algorithms as needed -- not a one-time move to new cryptographic algorithms

> Design architectures that allow the replacement of algorithms quickly, easily, and holistically

> Preserve the confidentiality and integrity of the data between algorithms

> Reduce disruption when switching architectures

> Insulate the cryptographic change from the "application"

> Provide disintermediation between the consumers and providers

But because we thought these algorithms would last for decades, they're installed everywhere – deployed across applications and the technology ecosystem, deep in software applications, embedded in hardware devices, and many other places, too. Such was our confidence in those algorithms that few institutions kept track of them.

The next radical change will come from quantum computing. The technology's arrival is expected in the next few years,[5] and when it's fully functional, financial services institutions will be able to harness the power of quantum mechanics – hence "quantum computing" – to solve many complex problems (such as stochastic modeling, trade optimization, and much more) exponentially faster than today's computers can.

The downside is that threat actors will have the same abilities. Malicious actors know that quantum computers will be the key that opens many locks. They're already buying or stealing data they can't possibly decrypt, knowing that quantum computing will do it eventually (it's called "harvest now, decrypt later").[6]

Meanwhile, more PQC algorithms are expected in cycles of staggered development,[7] and it is increasingly clear that new architectures will probably have shorter operational lives. They'll need replacing and updating more often, while some of the existing infrastructure cannot even make the jump to more nascent cryptographic algorithms.

Crypto Agility is a Long-Term Strategy, Not a One-Off Implementation

It is possible to migrate to post-quantum cryptography without the complexity of re-architecting for agility, but one-off transitions don't set the industry up for safety in the long run. We just won't have enough time to transition to the next algorithm when it is necessary.

The sector needs to boost its capacity for implementing, updating, replacing, and adapting software, hardware and infrastructure cryptography, such that no significant architectural changes are incurred and without disruption to running systems. Actual systems need to be upgraded to accept new cryptographic algorithms in an agile manner. In short, we need to embrace crypto agility to keep financial services firms more secure and compliant for the long term.

The Problem in Sum

> Today's algorithms are a potential attack vector, pervasive across the stack, and uncatalogued.

> The complexities and size of transition efforts are growing exponentially as systems and hardware proliferate.

> Viewing cryptographic change as a "once and done" operation is not effective.

The transition will take significant resources from each firm and collaboration across the industry – the sector is too interdependent across networks, exchanges, ecosystems, tooling, and more to do it alone. However, crypto agility gives us the tools we need to safeguard our consumers, our operations, and the trust that runs our businesses. Those qualities are vital in a risk environment that is always changing.

We've Been Here Before: The History of Cryptographic Transitions

The financial services industry is no stranger to cryptographic transitions. To name a few:

- **1970s:** The sector transitioned from proprietary algorithms to the NIST Data Encryption Standard (DES).
- **Early 1990s:** The RSA algorithm was widely used with a 1024-bit public key with several hash algorithms, including MD5 and SHA1. (RSA was often described using the number of digits, e.g., RSA-309, the same as 1024-bit keys, so the history gets confusing.)
- **Mid-1990s:** The industry transitioned from DES to triple DES (or 3DES) as NIST's call for the next-generation Advanced Encryption Standard (AES) was still underway. However, the potential of breaking DES was very high (the DES III Challenge in 1999 determined the key in less than 24 hours).
- **Early 2000s:** MD5, published in 1992, was found to be susceptible to hash collision. The industry transitioned to SHA1 published in 1995 by NIST.
- **2001:** AES was published but the use of 3DES was so entrenched that the transition from 3DES to AES is ongoing.
- **2002:** NIST published the SHA2 suite due to growing concerns about SHA1.
- **Around 2012:** The sector transitioned from RSA 1024- to 2048-bits.
- **2017:** Google announced a SHA1 collision that triggered another hash transition.
- **2020:** RSA-250 (829-bits) was factored, forcing those who had not transitioned to -2048 to do so.
- **2022:** NIST announced its Round 3 selections for post-quantum algorithms. NIST continues to evaluate its Round 4 candidates and called for additional digital signature algorithm submissions.
- **2024:** NIST published its first tranche of quantum-resistant encryption standards and algorithms.

When a cryptographic relevant quantum computer (CRQC) becomes available, Shor's Algorithm, published in 1994, will become a reality that triggers yet another cryptographic transition from the legacy asymmetric algorithms (RSA, Diffie-Hellman, ECC) to the NIST Post Quantum Cryptography (PQC) algorithms.

We expect multiple waves of PQC algorithms over the next decade or so. Their staggered development highlights the necessity for crypto agility soon, not later.

Getting Started: Testing and Measuring Your Crypto Agility Capacity

Crypto agility is a process, not a binary outcome. Part of the process is the change management of key distribution, cryptographic algorithm and architecture transitions, workforce preparation, and more. Gauging your readiness to migrate helps you prioritize the changes. (For more insight on crypto agility maturity, see the Maturity Model for Crypto Agility section.)

Before beginning the transition, analyze how well your institution can implement, update, replace, and adapt cryptographic schemes today in software, hardware, and infrastructures and what you need to move in order to embrace crypto agility concepts. The following questions will help.

1 *To what extent is crypto agility a design feature?*

The answer determines how thoroughly implementing, updating, replacing, and adapting cryptographic schemes is already incorporated into the design of the software, hardware, and infrastructures involved. While technology is the primary focus, people and processes must also be considered during design. More mature organizations will have a greater understanding and documentation of the changes that need to be made, as well as the pre-defined structures to implement them.

2 *What's the likelihood that architectures that embrace crypto agility will incur no significant architectural changes to adapt to a new algorithm as needed?*

This issue focuses on the **end state** of the software, hardware, and infrastructures in scope. More mature organizations will have a greater idea of what the systems will look like when the changes have been completed.

3 *What's the probability that crypto agility will not disrupt running systems?*

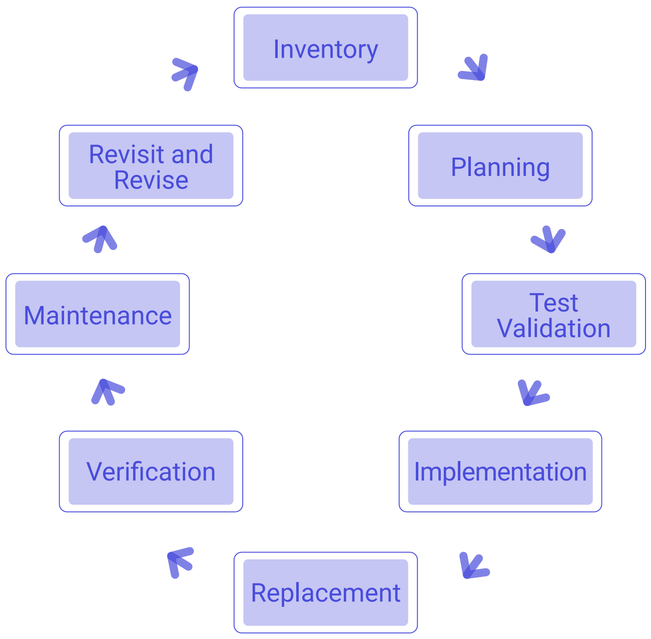
This question reflects on the **quality of the change process** when you're implementing, updating, replacing, and adapting crypto agility plans. More mature organizations will have more detailed inventories, plans, and appropriate contingencies in place.

Framework for Replacing an Insecure Algorithm

The foundation of crypto agility is the ability to replace an individual algorithm with another easily. That often requires iteration and building core capabilities, among other efforts. Begin the transition with an orderly plan – you'll be more efficient and overlook fewer issues. With that in mind, consider drafting:

- A shared/centralized approach (with libraries) to implement, switch, pivot, and deploy new cryptographic architectures.
- An exit plan that incorporates business objectives.
- A risk assessment regarding how the business adapts to changes in cryptography/algorithms, e.g., what do you do with the contracts with RSA or Dilithium if those signature(s) become at risk?
- A framework to evaluate your risk from using less-crypto-agile third parties and to update your third-party risk assessment forms regularly.
- A plan for managing crypto agility. Consider why, where, and how crypto agility needs to go over time.

The following eight-phase framework provides a detailed roadmap for replacing algorithms. This use case refers to upgrading from traditional cryptography to new quantum-resistant cryptography algorithms as an example, but note that the framework applies to any upgrade in cryptographic algorithms.



1 *Inventory Phase: Assess your algorithmic assets, dependencies, and risks*

A. Evaluate your current cryptographic landscape, including the types and usage of cryptographic algorithms across all systems. Consider grouping your usage by endpoint, infrastructure, and back-end/server components.

B. Inventory cryptographic assets, identify dependencies on industry-specific requirements (PCI, etc.) and legacy cryptography that must evolve, and assess risk levels associated with quantum threats.

> Identify third-party dependencies (e.g., vendor software libraries, SaaS services, etc.).

> Identify cryptographic anti-patterns in your source code that will hurt the transition to new cryptography.

C. Identify specific vulnerabilities to classic cryptography. Conduct detailed discovery of potential weaknesses in current cryptographic implementations.

Example: A financial institution conducts a comprehensive review of its encryption protocols for online transactions and discovers that some older systems still rely on outdated RSA algorithms vulnerable to quantum attacks. This highlights the need to prioritize these systems in the transition plan.

2 *Planning Phase: Draft a transition and integration plan, select algorithms*

A. Develop a detailed transition plan that aligns with organizational goals and the projected timeline for quantum and other cryptographic risks. Coordinate with vendors and/or partner timelines. Remember to account for regular maintenance, service life, and replacement cycles.

B. Select appropriate quantum-resistant algorithms, design a roadmap for integration, and set clear milestones and budgeting. Note: NIST has not only selected stateless algorithms for PQC but also some stateful ones, such as XMSS. For example, when considering a smart card use case, stateful algorithms are also valid.

Example: After identifying vulnerabilities, a credit union decides to integrate quantum-resistant algorithms. It starts with its mobile banking platform, which handles bulk customer transactions. The plan includes a detailed timeline, budget, and resource allocation for training developers and IT staff on the new cryptography standards.

3 *Test Validation Phase: Analyze algorithms' performance in your system*

A. Validate the new cryptographic solutions in a controlled environment to ensure they meet security and performance standards.

B. Conduct rigorous testing of quantum-resistant algorithms to ensure they function correctly and securely in the existing infrastructure.

C. Analyze the impact of new cryptographic implementations to prevent introducing new vulnerabilities or degrading system performance.

D. Perform regression testing and analysis to verify that updates do not negatively affect other system components.

Example: A bank sets up a sandbox environment to test the integration of lattice-based cryptographic algorithms. The tests include performance benchmarks, stress testing, and security evaluations to ensure the new algorithms work seamlessly.

4 *Implementation Phase: Begin the migration with non-critical systems*

A. Execute the transition in a controlled, phased approach to minimize disruptions.

B. Begin with non-critical systems to test quantum-resistant algorithms, followed by gradual deployment across critical assets. Use risk models to guide deployment priorities.

Example: An exchange tests the waters by rolling out post-quantum cryptography in its internal communications and data storage solutions. By starting with less critical data, it can monitor the impact on system performance and security before expanding to client-facing applications.

5 *Replacement Phase: Methodically update systems and components*

Algorithm replacement isn't often instantaneous. Some aspects may involve handling several algorithms at once until every caller can upgrade to the latest algorithm.

A. Systematically replace old cryptographic systems with new quantum-resistant solutions.

B. Methodically replace outdated cryptographic components with updated solutions as validated by previous phases.

Example: A financial institution systematically replaces its existing Transport Layer Security (TLS) 1.2 encryption with TLS 1.3, which supports post-quantum cryptographic algorithms. The replacement is done in stages, starting with internal systems and gradually extending to customer-facing applications to minimize disruption.

6 *Verification Phase: Audit and validate cryptography internally and externally*

A. Ensure that the new cryptographic systems function as intended and provide the necessary security enhancements.

B. Conduct thorough testing and auditing and involve internal and external stakeholders in validating security performance.

Example: Following the implementation, a trading firm uses both internal audit teams and external cybersecurity firms to perform rigorous testing of the new cryptographic measures. It simulates various attack scenarios to evaluate the resilience of quantum-safe algorithms in safeguarding trading and investment data.

7 *Maintenance Phase: Update cryptography, databases, and risk assessments*

A. Continuously monitor, update, and improve cryptographic measures to adapt to new quantum developments and emerging threats.

B. Ensure that changes to system/network architecture and new vendor software are identified and added to the database of known locations for cryptographic algorithms.

C. Implement regular updates, conduct training, and perform ongoing risk assessments.

Example: An insurance company establishes a quarterly review process for its cryptographic practices. This includes updates based on the latest quantum computing advancements and ongoing employee training sessions. The company also engages with industry working groups to stay aligned with emerging best practices in quantum cryptography.

8 *Ongoing: Revisit and revise algorithms and performance*

A. Use an iterative process for continuous improvement and adaptation to new cryptographic threats and standards.

B. Update related processes to provide information in an ongoing fashion.

C. Regularly repeat the discovery, testing, analysis, and replacement phases to ensure ongoing robustness against emerging quantum threats.

Example: An exchange sets up a bi-annual review process to assess the effectiveness of its cryptographic measures. This includes repeating the discovery phase to identify new vulnerabilities, testing the latest quantum-resistant algorithms, analyzing any impacts, and replacing outdated components as necessary. This iterative approach keeps the organization ahead of emerging threats.

Technological Challenges to Crypto Agility Migration

Becoming crypto agile is a big job, and you'll face obstacles from a high level, such as decentralized cryptography within a firm, and barriers at the operational and user level, such as disruptions caused by replacing outdated hardware.

Below is a non-exhaustive list of challenges that firms can expect in their transition to crypto agility.

- **Cryptography is not part of your software lifecycle:** Cryptography code is often deep and spread across the code of applications that directly call upon cryptographic protocols and primitives. Some cryptography even hard codes specific versions, key length, and other cryptographic parameters. Consequently, when needing to change, those applications must be entirely refactored and tested.
- **Managing cryptographic third-party risk:** Cryptography is also woven into third-party components. Their cryptography dependency is not always apparent, leaving their customers blind to related cryptographic vulnerabilities. Updating cryptography in these components often translates into adopting a completely new release of third-party components involving much more testing and integration, thereby slowing down adoption.
- **Developers lack specialized skills:** Cryptographic changes are infrequent, so application developers don't always have the skills to correctly implement and use cryptography in code (it's not uncommon to find wrongly configured cipher suites even in protocols as widely used as TLS).
- **Application code is out of date:** A cryptographic roadmap is often well-defined centrally, but its execution relies on each development team updating its application code. If that code isn't kept current, you'll have to bring it up to date, possibly across the entire code base.
- **Old systems:** It can be much harder to update older or legacy systems, especially those that cannot be replaced without significant effort.
- **Lack of cryptographic standards, languages, and definitions:** Organizations often use a variety of cryptographic methods. Inconsistency makes transitioning to crypto agility more complex. Following the current state processes paper will help.
- **Lack of overall business agility:** Firms that don't use agile principles in general – or not very well or very often – may have a slower route to crypto agility.
- **Risk-averse developers:** Considering the deep integration of cryptography in code and the specialized skills required to change it, developers might feel reluctant to update cryptographic implementations or postpone needed work for fear of introducing a breaking change or incompatibility. In addition, they might maintain support of weak cryptography in their server applications longer than necessary out of concern that some client applications may still depend on this compatibility for use of the service.
- **Manual key management practices:** Managing cryptographic keys manually is inefficient and introduces errors. Automating key management processes as part of a holistic review of cryptography enhances security and efficiency.

Operational and End User Challenges of Crypto Agile Migration

The following points describe how certain changes, though seemingly minor, can have significant implications for both back-end operations and the end user experience.

- **Overworked IT Department:** Many critical updates in cryptography are invisible to end users, but require significant adjustments within the organization's infrastructure. Deploying quantum-resistant algorithms in data encryption for internal databases can be a massive undertaking for IT departments and cryptography often involves substantial changes in the enterprise's technology stack. Updating customer-facing applications to incorporate new encryption protocols, for example, requires extensive testing and deployment phases to prevent disruptions to the user interface or performance. Yet these changes must go unnoticed by customers, who should continue to experience the same quality and speed.

- **Operational and customer service disruptions caused by updating hardware and software:** Legacy hardware and software systems can require significant overhauls to support modern cryptographic methods. Replacing outdated hardware that supports only older cryptographic standards can cause temporary disruptions but is necessary for long-term security.
- **Employee upheaval:** Updating processes and organizational structures – especially entrenched ones – can be difficult. You might need to implement new reporting structures, automate some practices, re-train staff, and much more. That can cause friction, but modernizing cryptographic practices and adopting a flexible architecture that supports plug-and-play cryptographic modules helps you to quickly adapt to new standards as they emerge.
- **The limitations of third parties:** In a rapidly evolving technological landscape, certain aspects of cryptography may be outside direct organizational control, such as those managed by third-party vendors or regulatory bodies. A vendor's timeline and compliance with industry regulations may dictate the financial institution's updates and changes to cryptographic standards.

The Nine Core Elements of a Successful Crypto Agility Transition

Certain approaches can smooth your path to crypto agility. As a general rule, balance security and performance. That may lighten the "performance overhead" cost – cryptographic processes can be resource-intensive and impact system performance.

Moreover, consider whether your hardware is ready for the migration to the new algorithm in the short term. Hardware is often insufficiently sized or not capable of running PQC algorithms, especially PQC algorithms in IoT devices, and migrating those devices too early may cause more stress in the environment than desired.

We also recommend staying updated on regulations to ensure your cryptographic practices are compliant and audit-ready; DORA (Article 9, 4(d)), PCI (DSS v4.0 requirement 3.6.1), and other standards mention the need to protect keys and data.

With that in mind, the PQC Working Group notes the following nine elements that will help your migration process.

1	**Align**	With Goals
2	**Assess**	Infrastructure
3	**Create**	Governance Structures
4	**Teach**	Crypto Skills
5	**Collaborate**	With Vendors
6	**Evaluate**	Algorithms
7	**Monitor**	Measures
8	**Inform**	Stakeholders
9	**Communicate**	With Callers

1 **Align the crypto agility transition with your organization's broader strategic goals and plans.** You can smooth the migration process by aligning it to or combining it with other processes, such as modernization and cloud migration, digital transformation, or enhancing customer trust. If your organization is moving services to the cloud, for instance, integrate crypto agility into the process to leverage cloud-based quantum-safe solutions.

2 **Assess and upgrade your technological infrastructure to support new cryptographic**

standards. This involves evaluating current cryptographic uses and identifying areas that need quantum-resistant algorithms. If you use encryption for customer data in transit and at rest, plan how you'll transition to quantum-safe encryption methods without disrupting services.

3 **Establish clear policies and governance structures to oversee the transition.** Besides creating efficiency and oversight, policies and governance ensure compliance with industry regulations and standards. Create a PQC steering committee responsible for overseeing the transition, setting timelines, and ensuring alignment with regulatory requirements.

4 **Teach your team PQC skills.** This includes training for technical staff and awareness for non-technical stakeholders. Consider conducting workshops and training sessions on cryptography for your IT and security teams to equip them with the basics of quantum computing and its implications.

5 **Collaborate with software vendors, service providers, and third parties.** Ensure that their products and services will support your transition to crypto agility. For instance, learn your software providers' roadmap for incorporating PQC algorithms and how it aligns with your transition plans. When acquiring new hardware or software, be sure to advise your procurement department on the considerations around new forms of cryptography.

6 **Test your quantum-resistant algorithms in a controlled environment to identify potential issues.** Before full-scale implementation, set up a test environment to evaluate the performance and compatibility of new quantum-safe encryption for consumer-facing solutions. Also consider the possibility of side-channel vulnerabilities, especially with newer and less stress-tested algorithms.

7 **Monitor the new cryptographic measures post-implementation.** Stay adaptable to evolving quantum technologies and threats. Monitoring tools will help you detect any security weaknesses introduced by the new cryptographic standards.

8 **Keep stakeholders informed.** Many people in your institution need to stay current on the transition process, its impact, and any actions on their part – including customers. Send them regular updates about how the transition to quantum-safe cryptography keeps financial transactions and data safe.

9 **Deprecate older algorithms on a schedule.** It may take some time for all callers/customers of an application to adapt to the new algorithms and/or protocols. A clear timeline and deprecation schedule should be given to callers to ensure orderly transitions. If the callers become agile, this window can be greatly reduced.

Part 2. Technical Aspects of Implementing Crypto Agility

Part 1 of this document explained the need for crypto agility, why a transition to post-quantum cryptographic algorithms ought to be an ongoing effort, and how to design a migration plan for the long term.

Now we look at more detailed technical definitions and explanations. The guidance in Part 2 is drafted for technologists and practitioners who manage and build applications, systems, infrastructures, operations, and security, and who manage risk.

Here you'll see what the FS-ISAC community believes our sector's vision should be for implementing, updating, replacing, and adapting cryptographic schemes in software, hardware, and infrastructures such that no significant architectural changes are incurred, and without disruption to running systems.

The application modernization steps needed to implement crypto agility are out of scope for this paper. When deciding whether to replace outdated systems or refactor them to support new cryptography, you must first assess the financial implications and potential security risks, and develop plans to introduce methods to protect critical systems that could not support quantum-safe algorithms.

Those decisions are unique to each firm, and we recommend considering the Gartner 5 Rs, "Rehost, Refactor, Revise, Rebuild, or Replace" as a starting point.

Maturity Model: The Characteristics of Crypto Agility

The Crypto Agility Maturity Model (CAMM)[8] from Hochschule Darmstadt (h_da, or Darmstadt University of Applied Sciences) proposes four levels of maturity, from Possible to Sophisticated.

PQC Working Group Framework Step One: Inventory Phase

1	Evaluate your current cryptographic landscape
2	Inventory crypto assets, identify dependencies
3	Identify vulnerabilities to quantum-resistant algorithms

This paper expands on that model (v1.1) by incorporating the technical aspects described in the model – note that CAMM level 2 is related to our framework's step 1 – and extends it by including process and other organizational aspects that clarify the relationship with the framework in section 1.

L0: Initial/ Not Possible	> No progression to crypto agility
L1: Possible	> Basic awareness > Limited inventory > Crypto discovery tools
L2: Prepared	> Starting policy > Formal roles > Modular design > Can start to exclude algos
L3: Practiced	> Established management > Dedicated teams > Regular testing > Agility enforcement
L4: Sophisticated/ Adaptive	> Dynamic controls > Highly responsive org > Automation > Cross-system interop

Level 0: Initial/Not Possible

Key Characteristics: The institution and its systems have not started progression towards crypto agility.

Level 1: Possible

Key Characteristics: Basic recognition but minimal documentation

- Basic recognition: The concept of crypto agility begins to gain traction among some employees, but it's not yet a formal part of organizational strategy or operations.
- Minimal documentation: There aren't any formalized processes or policies related to cryptography and practices are inconsistent across the organization. The firm has potential vulnerabilities.

Roles and Responsibilities

At this initial stage, awareness of crypto agility is limited to a few forward-thinking staff members who recognize the need for adaptable cryptographic practices. These individuals may start to informally advocate for better crypto management practices within their teams or departments.

Technical (CAMM v1.1): Knowledge, Updateability, Extensibility, Reversibility, and Documented Inventory

- System knowledge: Teams have the detailed knowledge of the affected system and its environment necessary to effectively evaluate crypto agility requirements.
- Updateability: Maintainers can modify the system and provide updates to new software versions.
- Extensibility: The system can be extended with new cryptographic algorithms and parameters.
- Reversibility: The system can be rolled back to a previous state.
- Inventory: The cryptographic functions used are documented and their current security level is known.

Level 2: Prepared

Key Characteristics: Policy development and inventory management

Policy development: The organization starts to develop specific policies that guide the management of cryptographic tools and procedures.

Essential crypto inventory management: A systematic approach is initiated to keep track of all cryptographic assets – including keys, certificates, and algorithms – to maintain control over these critical components.

Roles and Responsibilities

A formal role, such as a Crypto Manager, is established to lead the development of cryptographic policies. The Crypto Manager's team begins to formulate and document standardized procedures for managing cryptographic keys and software.

Technical (CAMM v1.1): Modularity and Exclusion, Algorithm IDs and Intersection, Opportunistic Security, and Usability

- Cryptographic modularity and exclusion: The system is modularly designed in such a way that changes to the cryptographic components do not affect the functionality of the other system components coupled to it.
- Algorithm IDs: The algorithms used are uniquely identifiable.
- Algorithm intersection: All subsystems share a common set of cryptographic algorithms.
- Opportunistic security: The system always uses the strongest available algorithm.
- Usability: The system's crypto agility module is easy to use.

Level 3: Practiced

Key Characteristics: Established management processes and regular training and awareness

Established management processes: There are transparent processes for regularly updating and replacing cryptographic modules, along with well-defined metrics to measure the effectiveness and efficiency of cryptographic practices.

Regular training and awareness: Continuous training programs are in place to ensure that employees who manage cryptographic processes understand and can effectively implement the latest cryptographic protocols and tools.

Roles and Responsibilities

The organization builds a dedicated team – including IT security personnel, privacy, compliance, risk, and other stakeholders – tasked with implementing and maintaining cryptographic policies. This dedicated team trains other staff on crypto-related issues and updates.

Technical (CAMM v1.1): Policies, Performance Awareness, Hardware Modularity, Testing, Enforceability, Security, Backward Compatibility, Transition Mechanism, and Effectiveness

- Policies: Policies are used to restrict the allowed algorithms and their parameters.
- Performance awareness: The additional effort and impact of crypto agility are known and accepted.
- Hardware modularity: Hardware and software can be improved or exchanged independently of each other in a compatible manner.
- Testing: The system is regularly tested for compliance with crypto agility requirements.
- Enforceability: Crypto agility can be made mandatory for certain contexts.
- Security: The crypto agility mechanism is secure against attacks.
- Backwards compatibility: New versions are compatible with older states of the system.
- Transition mechanism: A strategy that ensures the functionality of the overall system for the transition period of performing an update.
- Effectiveness: Migration between cryptographic algorithms is feasible in a reasonable amount of time.

Level 4: Sophisticated/Adaptive

Key Characteristics: Advanced crypto agility program and responsiveness to cryptographic standards and threats

Advanced crypto agility program: A mature and dynamic crypto agility program is firmly established, which is continuously monitored, assessed, and updated to meet the changing landscape of cybersecurity threats, especially those posed by quantum computing. Strong cryptographic management control measures and regular audits are performed.

Further, the organization's mature cryptographic management controls can quickly identify, assess, and mitigate any cryptographic weaknesses as they arise. This involves establishing protocols, tools, and trained personnel to ensure swift and effective action to protect data integrity and security against emerging threats.

Responsiveness to cryptographic standards and threats: The organization is highly responsive to emerging cryptographic standards and threats. It integrates new standards into its operational practices and has a structured approach for rapid threat assessment and response.

Roles and Responsibilities

At this stage, the organization's leadership teams take a proactive and strategic approach to cryptographic management. They actively forecast and assess future cryptographic needs, integrating insights from industry trends, advancements in quantum computing, and emerging security threats.

This proactive approach ensures the organization's readiness for future challenges and inspires a culture of continuous improvement, emphasizing the importance of staying ahead of the curve in cryptographic practices.

The institution employs specialized personnel such as cryptography analysts and specialists responsible for evaluating and implementing secure cryptographic practices. These experts are also integrated with IT governance to ensure that cryptographic strategies align with broader organizational IT policies and objectives.

Technical (CAMM v1.1): Automation, Context Independence, Scalability, Speed, and Cross-System Interoperability

- Automation: Modifications to crypto agile modules do not require manual interaction.
- Context independence: The requirements and techniques used to implement crypto agility can be used for other scenarios as well.
- Scalability: The crypto agility implementation can be deployed in additional systems.
- Speed: Modifications to cryptographic functions become active in the production system as immediately as possible.
- Cross-system interoperability: Different cryptographically agile systems are interoperable with each other.

Implementation Issues and Considerations

The financial services sector has had its fair share of transitions – EMV in cards, PCI in conducting payments, among others. Prior to the introduction of crypto agility, each was treated as a singular update or transition, which took considerable time, effort, and planning to produce from scratch. Some of these transitions have been going on for years and still aren't done.

Clearly, we need to be more agile because we need to expect transitions. But there's more than one way to implement crypto agility – the most appropriate approach depends on your institution's needs and infrastructure. However, most financial firms will face similar processes, beginning with the transition itself.

Cryptographic transitions[9] can be described as managing the passage from one security architecture to another in a methodical approach that is consistent with prudent business practices and security guidelines.[10] To successfully transition a cryptographic architecture, an organization needs to review and potentially update its security policies, cryptography standards, key management practices, and certainly its implementations.[11, 12]

Quantum-resistant cryptography is new, and there will require many transitions within very short timeframes.

We won't have decades between cryptographic transitions; we'll have years or possibly months.

Historically, these transitions occur when a new threat impacts an existing solution, and cryptographic architecture needs to be upgraded in response. The inevitable threat from a cryptographically relevant quantum computer (CRQC) changes the paradigm. Because all legacy public key cryptography (PKC) is impacted, all the cryptographic architectures need updating within the same timeframe.

The accelerated timeframe makes crypto agility a necessity and adaptable, expandable, and interoperable cryptographic architectures vital. Our crypto agile approach must give us the capability to manage multiple cryptographic transitions, including the ability to begin the next transition while the previous transition is still in progress.

This section describes the process and outcomes of implementing crypto agility under those conditions.

Process Guidance 1 is a recommended topology for a crypto agile central design. Instead of leaving each individual application to handle the complexities of crypto agility, this design and scheme allows for a more central management approach that facilitates agility in swapping out algorithms at the app level.

Process Guidance 2 presents the technical details of replacing one algorithm with another. Calling and implementing different algorithms requires a level of abstraction and agility probably not considered in a pre-crypto agile design, and this part describes what may vary between them.

Process Guidance 1: Topology for Crypto Agile Central Design

Cryptographic Architectures, Cryptographic Metadata, Third Parties, Cloud Environments, Vendor Black Box Applications

Cryptographic architectures

Cryptographic architectures show where keys are stored, how the algorithms are used, and the key management lifecycles.

Like network architectures that show the physical connections between routers, switches, firewalls, servers, etc., cryptographic architectures provide information about cryptographic assets. And like application architectures that show data flows and transactions between clients, web portals, services, microservices, databases, etc., cryptographic architectures provide information about cryptographic systems.

It is important to recognize that financial services firms may support many discrete cryptographic architectures. While the underlying cryptographic functions are the same – e.g., encryption, digital signatures, key management, etc. – the reasons for using cryptography will differ.

For example, key management for an ATM to protect the PIN for cardholder authentication is not the same as electronic signatures for protecting 30-year mortgage documents. Similarly, using public key certificates with TLS to protect online banking is not the same as relying on certificates for entity authentication of individuals or network devices.

Accordingly, the variety of cryptographic architectures re-emphasizes the importance of crypto agility.

The need for supporting multiple algorithms simultaneously is a likely use case until such time that the calling applications can use the newer or desired algorithms. In the interim, systems establishing a secure channel will need to negotiate supported algorithms and select the most secure algorithm supported between them.

Cryptographic Metadata

Cryptographic metadata is the data that describes and provides information about the configuration, usage, and lifecycle of cryptographic keys and algorithms. Financial services firms must be capable of tracking and reporting on cryptographic metadata to properly transition to new algorithms. The following metadata issues should be considered in a crypto agile transition.

- **Key attributes:** Information about cryptographic keys, including essential types (e.g., symmetric, asymmetric), size, algorithm use, purpose (e.g., encryption, signing), and the dates of crucial generation, expiry, and renewal. Tracking these attributes helps ensure keys are updated or replaced by security policies and advancements in cryptography.

- **Algorithm specifications:** Details about the cryptographic algorithms in use, including algorithm types (e.g., RSA, ECC) and configuration settings. You need this metadata to assess each algorithm's quantum resistance and plan upgrades to quantum-resistant algorithms.
- **Usage parameters:** Data on where and how cryptographic keys and algorithms are used, such as the systems, applications, and processes that utilize them. This information helps you understand your cryptographic landscape and ensure that all areas are covered in the transition to PQC. (Also, where/how the cryptography is configured, e.g., a configuration file, and if so, where is that file? How is it updated? Centralized?)
- **Compliance information:** Records related to compliance with relevant regulations and standards, such as GDPR, PCI DSS, or HIPAA, which may dictate specific requirements for cryptographic practices. This metadata is crucial for maintaining compliance during and after transitioning to new cryptographic standards.
- **Performance metrics:** Information regarding the performance of cryptographic systems, such as processing times for encryption and decryption operations and the impact on system performance. This helps evaluate the efficiency of current cryptographic implementations and the potential implications of migrating to different cryptographic methods.
- **Access and authorization data:** Details about who can access cryptographic keys and under what circumstances, including policies for key custodianship and the audit trails of crucial usage. This is critical for maintaining the security and integrity of cryptographic keys, especially during the transition phase to PQC.
- **Cryptographic dependencies:** Information about various systems and components' dependencies on specific cryptographic algorithms. This is important for assessing the impact of changing or upgrading cryptographic algorithms as part of PQC readiness.

Third-Party Product Considerations

It is possible that each of your vendors has a different approach to crypto agility. As a consumer and user, you need to consider how the (potentially) disparate approaches need to mesh. Your suppliers' strategies are likely to be a key component of your own strategy, so they need to be reviewed and understood thoroughly.

When assessing your third party for post-quantum crypto agility, focus on:

A. The vendor's understanding of PQC and their plans to switch to new, more robust encryption methods when needed.

B. Their encryption practices, specifically whether they can quickly adopt new standards. Confirm that they regularly update security measures and follow industry guidelines. This way, you can be confident that your partners will protect your data against future threats from quantum computers. The following questions from the CFDIR vendor questionnaire (Appendix E) will help:

> Will the product or service require my firm to replace existing hardware or make system architecture changes to support the PQC migration?

> How will the product or service support cryptographic agility to allow flexible administration of configurations for planned cryptographic migration, or an unplanned and immediate migration to remediate a weakness in an algorithm?

> What operational/configuration guidance will you provide to migrate the product or service to utilize PQC?

C. Your legal obligations and business commitments, such as NDAs, contracts, etc.

Building Crypto Agility Within a Public Cloud Environment

Designing crypto agility in environments you don't directly control, such as the cloud, involves certain considerations. First, recognize that the apps maintained external to your data center are still in scope for crypto agility. SaaS apps are likely – but not guaranteed to be – responsible for managing their own cryptography. SaaS apps need to be considered for each service consumed.

Secondly, the overall CSP or external host may have its own approach to managing crypto agility and keys in general, and that design needs to be considered when integrating its crypto agility into your own. It is conceivable that some CSP configurations may not have options to change/adjust. Identify that well in advance, then either work with the vendors for future enhancements or note the configuration gaps in your crypto agility design.

Vendor Black Box Applications

Your crypto agility will be determined by your vendors (until there is a universally recognized crypto agility level the vendors support). To manage crypto agility in vendors, define or request minimum changes in API or user interface of their products and get an agreed window of upgrade to a new crypto algorithm.

Vendors should provide their policy in crypto agility. Their policy should include:

- How they minimize customer impact when switching cryptographic algorithms
- Their trigger to switch to cryptographic algorithms
- Their expected time to respond to the deprecation of an algorithm
- How changes to APIs or other consumption mechanisms will be communicated. Consider the security changes as well as changes to performance, speed, etc.

Process Guidance 2: Technical Details of Algorithm Replacement

Abstraction, Crypto-as-a-Service, Crypto Libraries, Service Mesh for Encryption in Transit

Abstraction – or updating one system without changing the host system or application logic – can be achieved in multiple ways but a central principle of abstraction is that cryptographic systems and processes should be the only source of cryptographic activity. Similarly, non-cryptographic systems and processes should have no cryptographic capabilities. That way a single change can be consumed in multiple places and a cryptographic estate that is known can be updated completely.

To undertake this sort of crypto agility, a modularity must be used such that the need for a cryptographic operation is realized as a requirement and executed as a separate system (which may be a code library, application, hardware appliance, or some other technology external to the application's own logic) rather than hardcoded in the application itself.

Note that abstraction in third-party services or public cloud SaaS hosts may be impossible – they control the service, thus the cryptography. You may need to consider other methods for dealing with those services in your approach to applying the crypto agility framework, such as learning vendors' schedules and adaptability, or working with vendors to identify the algorithms their pipelines should prioritize.

Crypto-as-a-Service allows you to host cryptography within a separate application/system for the focused administration of cryptography and key management, a level of abstraction that facilitates agility. This separation puts a 'magic curtain' between the application and the cryptography. As a result, operations like key rotation are completely transparent while others, like key or cipher changes, may be of limited impact and developers won't have any new implementation tasks.

Separation helps ensure proper oversight and gives you some user guardrails but requires the use – and potentially the software development – of an API to undertake cryptographic operations separate from the application. Still, the developer will only need to understand the API syntax and the expected return payload.

Cryptographic operations serviceable in this manner include:

> Encryption/decryption of data at rest

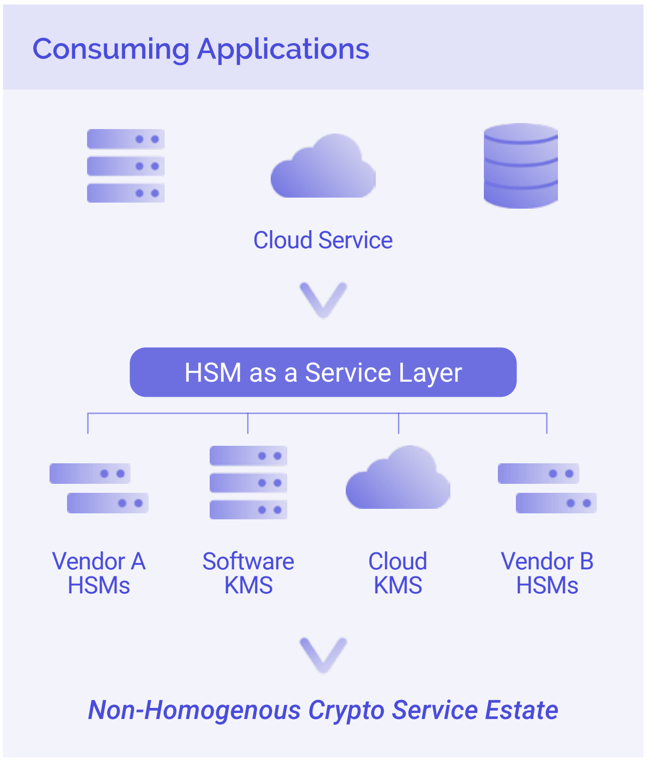
> Signing/signature verification

> Cryptographic hashing

> Key/Certificate management

Ideally, the service has a suitable cryptographic system for asset management and cryptographic operations, a front-end user interface for administration, a full capability API server, an endpoint, and on-demand worker nodes for executing API requests. It's best if each crypto algorithm is implemented in its own source code module that can be loaded and unloaded on the API server. Each module should present methods that hide all algorithm-specific requirements.

The security administrator would, via the user interface, select a cryptographic algorithm for a cryptographic service. The API would call these modules via their methods upon application request to a particular cryptographic service. The underlying encryption algorithm, key size, and other parameters should be pre-selected in the front-end user interface by the security administrator.



Pros:

> Complete segregation of duties between cryptographic key managers and users.

> Processes must follow the guardrails in place.

> Can be amended by the correct administrators as usage or standards change.

Cons:

> Can add inefficiencies and processing time compared to hard coded alternatives.

> Adds cost.

> The fixed nature of an API means that changes have to be communicated and tested well ahead of time.

> Changes in a suite of apps using the same cryptographic subsystem must be holistic. It's not possible to update the cryptography in some apps but leave others in the older system.

Cryptographic libraries are similar to crypto-as-a-service, except that the capability resides adjacent to the code it supports. That helps you ensure maintainable application logic while cryptography logic and ciphers sit outside and are easily updateable.

Pros:

> A trusted library can be more secure and well-stocked than coding built from scratch.

> New ciphers can be implemented with a minimum of changes to existing code and functionality.

> Crypto can be more performant than latency introduced traversing the network.

> "Write once, deploy many" changes are highly automatable.

> Obsolete algorithms can be easily identified and remediated.

Cons:

> Libraries don't do key management work.

> Change management and migration would need to be more carefully undertaken to ensure successful transitions.

Automated Public Key Infrastructures (PKI) and Certificate Authorities (CAs) can be used to update and change ciphers and capabilities as keys are either naturally renewed or, if an urgent change is required, in bulk.

"PKI" is a term that covers all aspects of the systems needed for public/private key pair production, storage, distribution, and management.

The two most prevalent PKI use cases are TLS certificates – sometimes known by the legacy term SSL – and SSH certificates. Centralizing the issuing and management of these keys to produce metrics showing the transition will highlight the long tail of your ciphers and key lengths.

The shorter the life span, the more crypto agile an organization can be in this respect. (It is worth considering some of the root and intermediate certificates that have a longer life and are harder to update quickly.)

There is some debate over the use of hybrid/composite multiple key certificates. Stay abreast of developments to see how they might fit into your environment.

Certificates are intrinsically time restricted at the point of issuing. This means that updating standards can be forward looking, such that extensions like hybrid certificates – containing both RSA and quantum safe keys – can be deployed in advance without affecting current operations. You can start using them when enough are installed or when it is decided that quantum safe keys should be used widely instead.

Pros:

> Certificates usually don't have a legacy dependency, they're specific for that algorithm/scheme.

Cons:

> PKIs can be slow to update, which is a problem in an emergency.

> Changes in key length need application support.

> Cipher changes need support at both ends of the connection.

> Requires change in the supporting applications to ensure full crypto agility.

> Industry ratification and adoption of new ciphers can be slow, which hinders roll-out.

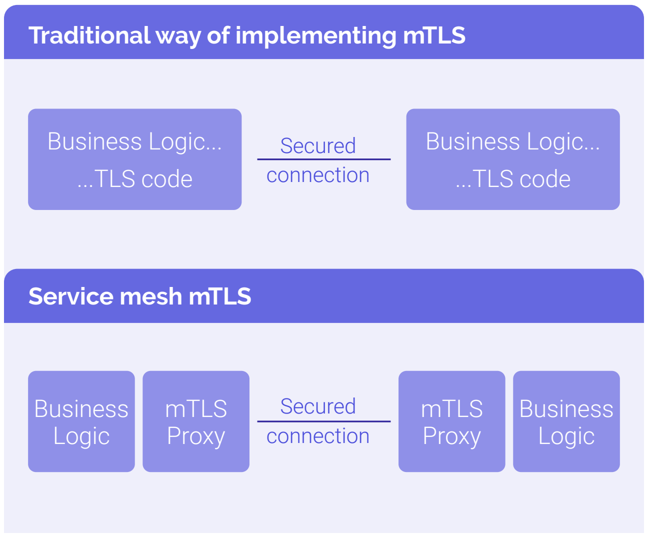
> You have to remove old ciphers to boost crypto agility without legacy vulnerabilities.

In an emergency, a new CA could be stood up quickly and TLS/SSH certificates issued and deployed automatically at speed, but this would involve technical support if some devices were offline, or if other pieces haven't been fully tested for new algorithms and ciphers.

A **service mesh for encryption in transit** can operate as an encrypted, mutually authenticated (mTLS) virtual network such that only authenticated nodes can talk to each other. Traffic into and out of the mesh is controlled through specific nodes.

To facilitate agility for encryption in transit, the cryptographic operations (Certificate Signing Requests, Issuance, and more) should be separated out as a service that can be used by the applications that are going to establish a secured connection.

This encryption in transit service could be implemented as a service mesh, which is a dedicated infrastructure layer for facilitating service-to-service communications between services or microservices using a proxy.



By using service mesh for mTLS, application developers can focus on the implementation of the business logic and let the mTLS proxy handle all the details. By providing TLS service centrally, a new cryptographic algorithm would be updated with minimum disruptions to applications.

Pros:

> Easy for applications to consume.

Cons:

> Updates may be required across the entire mesh as a dependency to use the strongest levels of encryption.

The appendix has API Schemas and further details.

Transition Governance and Considerations

Transition governance and considerations for PQC involve upgrading programming languages, enforcing the discontinuation of outdated cryptographic algorithms, and conducting thorough crypto discovery. These strategies are vital for preparing financial institutions to seamlessly integrate quantum-resistant technologies and ensure their cryptographic systems are robust against emerging quantum threats and other cryptographic threats.

Embedding the six transition strategies below into more expansive IT modernization plans helps maintain operational efficiency, comply with evolving regulatory requirements, and better secure your future.

1. **Handling different programming languages and schemes:** Ensuring systems can handle different programming languages and cryptographic schemes (libraries, protocols, algorithms, etc.) allows your institution to maintain control. This involves adapting systems to be compatible with multiple programming environments and the cryptographic methods they support. It enables the agile integration of new quantum-safe algorithms as they become standardized and practical, ensuring system interoperability and security during the transition phase.

2. **Updating classical programming languages (e.g., C, C++, Java, .NET, etc.):** Classical programming languages are the backbone of many existing cryptographic implementations and will play a significant role in the transition to PQC. However, these languages need updates or extensions to support new cryptographic standards to minimize disruptions in existing processes and ensure the efficient implementation of quantum-safe algorithms.

3. **Sunsetting classical practices and algorithms:** A firm deadline to phase out older cryptographic algorithms – such as RSA, 3DES, DSA, or SHA-1 – and eliminate backward compatibility forces financial services institutions to adopt newer, quantum-resistant algorithms, accelerates the adoption of secure cryptographic practices, and reduces the risk of quantum attacks. Jettisoning outdated algorithms also helps you streamline your security protocols and focus resources on future-proof cryptography.

4. **Running a cryptography discovery:** Cryptography discovery involves identifying and cataloging cryptographic assets (algorithms, keys, protocols). The process makes all cryptographic assets known, managed, and updated appropriately. It is fundamental for transitioning to PQC – it helps your firm assess your current cryptographic landscape and plan necessary upgrades or replacements with minimal disruption.

5. **Modernizing your algorithms:** The shift from classical cryptographic algorithms to PQC is a crucial component of broader IT modernization efforts. It's also a technical update – such as larger key sizes and different encryption methods -- and a strategic overhaul. Integrating PQC readiness into the IT modernization strategy aligns crypto agility with other technological upgrades, optimizes resources, minimizes conflicts, and addresses architectural challenges by building new systems flexibly and adopting emerging technologies and standards.

6. **Algorithm negotiation:** Make sure systems have a way to default to the most secure algorithm shared between them. Not all systems will embrace the latest and most secure algorithms at once. Have ways to negotiate and determine the highest and most secure algorithm that each side can utilize, to ensure the highest level of security possible for that session.

Conclusion

This document is the first to define crypto agility holistically. Engineered for both a business and technical audience in the financial services sector, we believe it will help stakeholders across organizations understand the problem space, grasp the necessity of crypto agility, and define an approach that works for their institutions.

FS-ISAC designated this document TLP WHITE specifically to encourage sharing across the entire ecosystem, and we ask you to do just that.

The contributors to this document welcome feedback and suggestions for expansion.

Contributors

Peter Bordow, Wells Fargo, and FS-ISAC PQC Working Group Chair

Andrew Mulvenna, HSBC, and FS-ISAC PQC Working Group Vice Chair

Dr. Bassem Nasser, HSBC

Dr. Bingrong He, Fidelity Investments

Bojan Spasic, Swift

Carl Mehner, USAA

Carlos Recalde, Sheltered Harbor

Dr. Kenneth Giuliani, CIBC

Chris Ratcliff, HSBC

Don Aliberti, Valley National Bancorp

Eric Chu, Morgan Stanley

Erwin Carrow, USBank

Hala Shakra, Principal Financial

Dr. Hubert Le Van Gong, JPMC

Isabelle Noblesse, Swift

Jaime Gómez García, Santander

Oscar Covers, Dutch Banking Association

Jeff Stapleton, Wells Fargo

Joel Van Dyk, State Street

Jörg Schneider, Deutsche Bank

Lawrence Belton Jr., Ally

Dr. Maria Christofi

Martin Lemyre, BNC

Omkhar Arasaratnam, formerly at OpenSSF/Linux Foundation

Philip Vero, RBC

Robby Burko, Scotiabank

Dr. Thibaud Ecarot, BNC

Tim Schneider, Deutsche Bank

Dr. Carrie Gates, FS-ISAC

Mike Silverman, FS-ISAC

Appendix

Definitions

Abstraction: Clear separation of immutable logical cryptographic concepts (encryption, digital signatures, key exchange, digests etc.) from their implementation.

Automation: Generation of cryptographic artifacts (configuration files, code, etc.) from a single, managed source.

Governance: Understanding where and how cryptography is used, as well as the ability to initiate and monitor changes.

Hybrid: Operating in a classical/non-PQC and PQC mode simultaneously. You can do hybrid without being agile, but if you are agile, you are able to easily be hybrid.

Composite quantum-safe cryptographic algorithm: A security method that uses various techniques to protect data from current and future threats, including powerful quantum computers.

> Think of it like a security system with multiple locks, each designed to resist attacks. Even if a future quantum computer breaks one lock, the others will keep the data safe. This approach ensures that information remains private and secure, even as technology advances.

Several methods can achieve composite algorithms, but creating them can get complex through the alignment of underlying keys, numerous categories of PQC (e.g., hybrid, native, composite, etc.), and non-PQC certificates.

Quantum resistant: Algorithms that withstand code-breaking efforts from quantum computers.

Basics of Cryptography Types

Symmetric-key cryptography: This type of cryptography uses the same key for both encryption and decryption, such as AES. Symmetric-key cryptography is fast and efficient but requires secure key distribution.

Asymmetric-key cryptography (public key cryptography): This type of cryptography uses a pair of keys (public and private). Examples include RSA and ECC. It facilitates secure key exchange and digital signatures but is computationally heavier.

Hash functions: These produce a fixed-size hash value from arbitrary input data; they are irreversible. Examples include SHA-256. They are primarily used for integrity checks.

Quantum cryptography: Uses principles of quantum mechanics (like quantum key distribution) to secure data. It is considered resistant to quantum computing attacks. Quantum-safe cryptography uses traditional infrastructure to build and execute algorithms supposed to be resistant to quantum computers.

Keys

Asymmetric public/private key pair:

> **Public key:** A cryptographic variable used with an asymmetric (public key) cryptographic algorithm and is associated with a private key. The public key is associated with an owner and may be made public. For example, in the case of asymmetric encryption, this is used to encrypt data (which can be decrypted by using the corresponding private key only). In the case of digital signatures, the public key is used to verify a digital signature that was generated using the corresponding private key.

> **Private key:** A variable in cryptographic algorithms used to decrypt data or create a digital signature. Unlike public keys, which are disseminated widely, private keys are known only to the owner and kept secret. Ensuring the security of private keys is paramount, as unauthorized access can compromise the associated encryption system or digital identity.

Symmetric secret key: A variable in cryptographic algorithms used to both encrypt and decrypt data. Unlike asymmetric encryption, symmetric encryption uses only one secret key for both encryption and decryption functions. Keeping the secret key secret is the dominant consideration, as unauthorized access can compromise any data that has been encrypted with this secret key.

High-Level Overview of Shor and Grover Algorithms

The relevance of the Shor and Grover algorithms to quantum computing and post-quantum cryptography is crucial.

Shor's algorithm: Developed by Peter Shor in 1994, it is a quantum algorithm capable of factoring large integers and computing discrete logarithms in polynomial time. This capability means it can potentially break many of the cryptographic systems currently in use, such as RSA and ECC, which rely on the difficulty of these problems as the basis for their security. This demonstrates a critical vulnerability in current cryptographic practices, underscoring the need for crypto agility.

Grover's algorithm: Formulated by Lov Grover in 1996, this quantum algorithm provides a quadratic speedup for database searching problems and can be adapted to attack symmetric cryptographic algorithms. While not as devastating as Shor's algorithm to current cryptography, it implies that symmetric key lengths might need to be doubled to maintain current security levels against future quantum computers.

API Service Schema

APIs may evolve as the underlying cryptographic algorithms change over time. API version management is a common development exercise and is not discussed specifically in this paper.

When designing an API to undertake cryptographic operations, the schema can be inspected to see if the elements within it are suitable for crypto agility. Developers should take note that sometimes the default options in cryptography are at lower levels of security. The recommendation is to question the defaults and ensure you are operating at an appropriate level for your use cases and regulatory requirements; then for agility to be able to update those options.

What follows is a sample API outlined by the elements that would make them very agile (strong), limited agile (medium), or no agile (weak).

Crypto Agility Type	API
Crypto Service Provider	True
Criteria	> API names are algorithm independent > All arguments are algorithm independent > Cryptographic algorithms are selected for API on an administration platform
Agility Level	**Strong**
Examples	> encrypt (plaintext, signed_token) > decrypt (ciphertext, signed_token) > tokenize (plaintext, signed_token) signed_token = {application identity and other meta data} signed by calling application

Crypto Agility Type	API
Crypto Service Provider	True
Criteria	> API names are algorithm independent > Some argument values are algorithm dependent
Agility Level	**Medium**
Examples	> encrypt (algo=AES256, plaintext, signed_token) > decrypt (algo=AES256, ciphertext, signed_token) > tokenize (algo=FF1, plaintext, signed_token) signed_token = {application identity and other meta data} signed by calling application

Crypto Agility Type	API
Crypto Service Provider	True
Criteria	> API names are algorithm specific > Arguments are algorithm specific
Agility Level	**Weak**
Examples	> AES_encrypt (plaintext, keysize, signed_token) > RSA_sign (plaintext, public_key) signed_token = {application identity and other meta data} signed by calling application

Lower levels crypto agility API can be used to implement higher levels crypto agility API.

Examples	Encrypt (plaintext, signed_token)
	{
	    encrypt(algo=AES256, plaintext, signed_token)
	}
	
	Encrypt (algo=AES256, plaintext, signed_token)
	{
	    AES_encrypt(plaintext, keysize, signed_token)
	}

API Schema with Vendors

When reviewing a vendor product where technical details are limited, it is still possible to enumerate the crypto surface of a product to assess its potential agility.

What follows are sample descriptions outlined by the elements that would make them very agile (strong), limited agile (medium) or no agile (weak).

Crypto Agility Type	Vendor Blackbox System
Crypto Service Provider	False
Criteria	When cryptographic algorithms are updated: > No change on user interface > No change on external system interface
Agility Level	**Strong**

Crypto Agility Type	Vendor Blackbox System
Crypto Service Provider	False
Criteria	When cryptographic algorithms are updated: > Change is required on user interface > No change on external system interface
Agility Level	**Medium**

Crypto Agility Type	Vendor Blackbox System
Crypto Service Provider	False
Criteria	When cryptographic algorithms are updated: > Change is required on user interface > Change is required on external system interface
Agility Level	**Weak**

Endnotes

1 Alnahawi, N., Schmitt, N., Wiesmaier, A., Heinemann, A. and Grasmeyer, T., 2023. On the State of Crypto-Agility. *Cryptology ePrint Archive*. https://eprint.iacr.org/2023/487.pdf

2 Cryptographic Agility and Key Rotation - Google Bug Hunters

3 Crypto-agility and quantum-safe readiness | IBM Quantum Computing Blog

4 Hochschule Darmstadt University of Applied Sciences

5 Quantum Threat Timeline Research Report 2023 - Publication (evolutionq.com)

6 Harvest now, decrypt later: https://www.industrialcybersecuritypulse.com/threats-vulnerabilities/harvest-now-decrypt-later-encrypted-data-is-under-hackers-radar/

7 Announcing PQC Candidates to be Standardized, Plus Fourth Round Candidates | CSRC (nist.gov)

8 CAMM https://camm.h-da.io/

9 Cryptographic Transitions, Jeff Stapleton, Ralph Spencer Poore, Proceedings of the 2006 IEEE Region 5 Annual Technical Conference, April 2006

10 Cryptographic Transitions: Historical Considerations, Jeff Stapleton, Peter Bordow, Ralph Spencer Poore, ISSA Journal, Volume 20, Issue 9, September 2022

11 Cryptographic Architectures, Missing in Action, Jeff Stapleton ISSA Journal, Volume 15, Issue 7, July 2017

12 Security without Obscurity: A Guide to Cryptographic Architectures, Jeff Stapleton, CRC Press, Tayor & Francis Group, an Auerbach Book, ISBN 978-0-8153-9641-3, July 2018