MICKAI™

MICKAI EBOOK SERIES · No. 15

Signing for 2035.

The post-quantum record that outlives the algorithm, and why crypto-agility, not a single scheme, is the only honest migration.

AUTHOR

Micky Irons

Founder and named inventor, Mickai LTD.

19 June 2026 · v1 · mickai.co.uk

Mickai LTD · Companies House 17166618 · press@mickai.co.uk · mickai.co.uk
UK IPO register, named inventor Mickarle Wagstaff-Irons · Trade mark UK00004373277

TABLE OF CONTENTS

Contents

Foreword

A note from the author

Part I · The Problem

1. The shelf life of trust
2. Harvest now, decrypt later
3. The myth of the final algorithm

Part II · The Floor

4. Reading FIPS 204 honestly
5. The cost of the lattice
6. Belief, not proof

Part III · The Agility

7. Crypto-agility as a property of records
8. The record that carries its own history
9. Rotation without repudiation

Part IV · The Practice

10. Designing the Open Audit Record to last
11. Migrating without breaking the past
12. The honest migration

Appendix

About the author

FOREWORD

A note from the author

I have spent the last few years building a system whose entire promise is that a record made today can still be trusted decades from now. That is a strange thing to promise in cryptography, because the one constant of the field is that schemes break. RSA was sound until factoring got cheaper. Elliptic curves were elegant until we learned what a sufficiently large quantum computer would do to them. So when I tell you the Mickai SIOS seals every consequential action into a post-quantum Open Audit Record, the honest follow-up question is not which algorithm did you choose, it is what happens when that algorithm falls. This book is my answer.

Let me be plain about my position. We chose FIPS 204 ML-DSA-65 as our signing floor, and I will defend that choice carefully in these pages. But I have never told an investor or a defence reviewer that picking the right lattice scheme is the migration, because it is not. The scheme is a snapshot. The migration is the capacity to change schemes without breaking the chain of trust that came before. A record signed in 2025 has to remain verifiable in 2035 even if ML-DSA-65 is deprecated in 2030, and that only holds if the record carries its own algorithm history with it. That is the thread running through everything here.

Underneath all of this sits a quieter urgency that rarely gets said outside specialist rooms. The adversary does not need a quantum computer today to hurt you tomorrow. Encrypted traffic and signed records that matter are already being copied and stored, waiting for the day the maths becomes cheap. Harvest now, decrypt later is not a thought experiment, it is a procurement strategy for anyone with patience and storage. If your data has a shelf life longer than the time it takes to build a cryptographically relevant quantum computer, you are already exposed, and you were exposed the moment you sent it.

I write in the first person because I am not reporting on someone else's research, I am accountable for a system that makes these promises to real users on their own hardware. Mickai is the Sovereign Intelligence Operating System, fifty specialised brains running offline on the operator's own machine, and every record it produces is meant to outlive the fashions of cryptography. If I get this wrong, people who trusted a sovereign record will be left holding an artefact they cannot prove. So this is not a survey. It is the engineering argument I had to win with myself before I was willing to ship the seal.

Micky Irons

Founder and named inventor, Mickai LTD · 19 June 2026

PART I · THE PROBLEM

Why a signature you trust today is a liability you inherit tomorrow.

1. The shelf life of trust

Most discussions of cryptography quietly assume that the moment of signing and the moment of verification are close together. You sign a transaction, it clears, the matter is settled. Under that assumption the only thing that matters is whether the scheme is sound right now, and right now RSA and elliptic curves are still doing useful work. But the records I care about are not like that. A sovereign audit record, a deed of assignment, a chain of custody for evidence, a medical decision, these are artefacts whose whole value is that someone can come back in fifteen years and prove what happened. The gap between signing and verification is the entire point, and that gap is where the danger lives.

Once you accept a long gap, you have to reason about the algorithm not as it is today but as it will be over the lifetime of the record. A signature is a bet that the underlying hard problem stays hard for the whole period you need the proof to hold. That bet has been lost before. It will be lost again. The question is never whether a scheme will eventually weaken, it is whether your records are built to survive the day it does, and most are not, because they were designed as if signing and verifying were the same instant.

> **A signature is a promise made now that has to be kept by a stranger in a future you cannot see.**

This is why I keep returning to shelf life as the first design input, before any choice of scheme. If a record needs to be trusted for two years, your risk window is short and your options are wide. If it needs to be trusted for twenty, you are signing into a period that almost certainly contains at least one cryptographic regime change. You do not get to know in advance which scheme breaks or when. You only get to decide, today, whether the record you produce can absorb that break without becoming worthless.

In the Mickai SIOS I treat the audit record as the long-lived object and the signing scheme as the perishable one. That inversion is deliberate. The record is canon, it must outlive the algorithm. The algorithm is a current best choice that I fully expect to replace at least once within the useful life of the records it protects. Designing the other way round, treating the algorithm as permanent and the record as disposable, is the most common and most expensive mistake in this whole field.

2. Harvest now, decrypt later

The phrase sounds like a slogan, and it is, but the behaviour underneath it is mundane and already happening. An adversary with storage and patience copies your encrypted traffic and your signed

records today, not because they can read or forge them now, but because they expect the cost of doing so to collapse later. For confidentiality the threat is decryption. For signatures and records the threat is subtler, it is the eventual ability to forge a signature that looks original, or to manufacture a record indistinguishable from one you genuinely produced. Both share the same shape, today's protection, tomorrow's exposure.

What makes this rational for an adversary is the asymmetry of cost over time. Storing ciphertext is cheap and getting cheaper. Building a cryptographically relevant quantum computer is expensive and getting cheaper too, just on a different curve. The attacker does not need the two curves to cross today. They only need to believe they will cross before the value of your harvested data expires. For anything with a long shelf life that belief is entirely reasonable, which is precisely why the harvesting is already under way against traffic that will still matter in a decade.

The exposure is retroactive

The cruelty of harvest now, decrypt later is that the decision to protect yourself had to be made in the past. You cannot retroactively make 2025 traffic quantum-safe in 2031, because the copy the adversary holds was made under the old scheme. This is what separates the quantum migration from ordinary security patching. A patched server protects you going forward. It does nothing for the conversations already sitting in someone else's archive. For long-lived secrets and long-lived records, the only honest position is that the clock started some time ago and the only variable left is how much more you keep exposing.

I labour this because it reframes urgency correctly. People hear quantum, picture a distant machine and a distant deadline, and they relax. The deadline that actually binds you is not the day the quantum computer is switched on, it is the day you stop producing harvestable material under a vulnerable scheme. If your records need to survive to 2035, the migration was never a 2030 problem, it was a today problem, and treating it as a future line item is itself the exposure.

The Mickai pantheon.

3. The myth of the final algorithm

Every generation of cryptographers has a scheme it quietly believes is the destination, the one solid enough that the migration story can finally end. For a long time that was RSA, then it was elliptic-curve cryptography, and there is a real temptation now to let the post-quantum lattice schemes inherit that mantle. They are good. They survived a long and adversarial standardisation process. But the belief that any single scheme is the final algorithm is a category error, and it is the specific error this book exists to refute.

There are two reasons the final algorithm does not exist. The first is mathematical, our confidence in a scheme is confidence that nobody has yet found a fast attack, not proof that none exists. Lattice problems are believed hard, and the believing is well founded, but believed is not the same word as proven, and history is unkind to anyone who forgets the difference. The second is practical, even a scheme that never breaks mathematically gets deprecated for performance, for implementation flaws, for side-channel weaknesses, for changes in what we are willing to assume about hardware. Schemes retire for reasons that have nothing to do with their core security.

> **The destination was never a scheme. The destination is the ability to leave one.**

So when I say ML-DSA-65 is a floor and not a destination, I mean it as a structural claim, not a hedge. The floor is the level of protection I am unwilling to go below today. The destination, if the word even applies, is a system that can raise that floor whenever the ground shifts, without invalidating anything built on the old level. A system designed around a final algorithm is brittle by construction, because its entire trust model assumes the one thing cryptographic history guarantees will not stay true. The honest design assumes the opposite from the first line.

PART II · THE FLOOR

What ML-DSA-65 actually buys you, and exactly where it stops.

4. Reading FIPS 204 honestly

FIPS 204 is the standard that specifies ML-DSA, the Module-Lattice-Based Digital Signature Algorithm, the standardised form of what the research community knew as CRYSTALS-Dilithium. It is a digital signature scheme, which means it does one job, it lets the holder of a private key produce a signature that anyone holding the matching public key can verify, with security believed to hold against both classical and quantum adversaries. ML-DSA-65 is the middle parameter set, sized to a security level that maps roughly to the strength people expect from a strong conventional scheme, and it is the one I chose as the Mickai signing floor.

It is worth being precise about what the standard does and does not promise, because vendors blur this constantly. FIPS 204 gives you signatures that resist forgery under the lattice assumptions, with parameter sets chosen so that breaking them is believed to require resources no foreseeable quantum computer will have. What it does not give you is confidentiality, key management, identity binding, or any answer to the question of what happens when the scheme is superseded. It is a signing primitive, an excellent one, and reading it as anything larger than a signing primitive is the first step toward overclaiming.

Why the middle parameter set

There is always pressure to reach for the highest parameter set on the theory that more is safer. I resisted that for ML-DSA-65 deliberately. The higher set buys a larger security margin at a real cost in signature size and verification time, and for an audit record produced thousands of times on an operator's own hardware those costs compound. ML-DSA-65 sits where the security margin is generous against the threat we are actually defending against, while the records stay a size and speed a sovereign machine can produce and verify without strain. The floor is a deliberate balance, not a reflexive maximum.

Choosing ML-DSA-65 as a floor also means committing explicitly to the standardised parameters and the standardised encoding, not a private variant. That matters for the long game, because an independent verifier in 2035 needs to recognise the scheme from a public standard, not from my documentation. Standardisation is part of what makes the floor a floor. A bespoke scheme, however clever, is a record nobody else can confidently verify once you are no longer in the room, and a sovereign record that only its author can validate is not sovereign, it is private.

The Mickai pantheon.

5. The cost of the lattice

Post-quantum signatures are not free, and pretending otherwise sets up the disappointment that derails migrations. Compared with the elliptic-curve signatures most systems use today, ML-DSA signatures and public keys are substantially larger, and the verification work is heavier. On a single transaction this is invisible. Across a high-volume audit log, an embedded device, a constrained channel, or a record format with tight size budgets, it is the first thing that bites. Anyone who tells you the lattice migration is a drop-in replacement has not run it at volume.

These costs change architecture, not just performance counters. A larger signature changes how you store records, how you transmit them, how you batch them, and how you think about the overhead of sealing a small action. In the Mickai SIOS, where the ambition is to seal every consequential action, the per-record overhead is not a footnote, it is a budget I have to manage actively. The honest engineering response is not to wish the cost away but to design the record format and the sealing cadence around it from the start, so the floor is affordable at the scale you actually intend to run.

There is a subtler cost too, in implementation. Lattice schemes have their own side-channel surface, their own sensitivity to how sampling and arithmetic are implemented, their own ways of leaking through timing if you are careless. A correct ML-DSA on paper can still be a vulnerable ML-DSA in silicon if the implementation is naive. Treating the standard as the end of the security work, rather than the start of an implementation discipline, is how a sound scheme becomes an unsound product. The floor is only as solid as the implementation that pours it.

I dwell on cost because cost is where good intentions go to die. Migrations stall not because people disagree that post-quantum is necessary, but because the bill arrives in a form they did not budget for, in storage, in latency, in engineering time. Naming the cost up front, and designing the floor to be affordable rather than maximal, is what keeps the migration from becoming a slide deck that never

ships. A floor you cannot afford to stand on is not protection, it is an aspiration.

6. Belief, not proof

The security of ML-DSA rests on the believed hardness of certain structured lattice problems. That word, believed, is not a weakness in the standard, it is the honest state of essentially all practical cryptography. We do not have proofs that the problems underlying our schemes are hard in the absolute sense. We have decades of the best people trying and failing to break them, which is a strong reason for confidence and a poor reason for certainty. A responsible system treats that distinction as load-bearing rather than pedantic.

What follows from belief rather than proof is a design obligation, not a panic. If my confidence in the floor is grounded in nobody having broken it yet, then my system has to be ready for the day someone does, because that day is consistent with everything I currently know. This is not pessimism about ML-DSA specifically, I think it is an excellent scheme and I expect it to last. It is realism about the epistemic status of all such schemes, and realism is the only foundation a long-lived record can safely sit on.

> **Confidence is knowing nobody has broken it. Certainty is forgetting that confidence is all you have.**

This is also why I distrust any architecture that bets everything on a single scheme, however good. Concentrate all your trust in one mathematical assumption and a single advance, a single clever attack, a single overlooked structural weakness takes the whole edifice down at once. The belief-not-proof framing pushes you naturally toward agility and toward composition, toward systems that can survive being wrong about one assumption. A floor is something you stand on while you keep an eye on the door. It is not something you weld yourself to.

None of this should read as a lack of conviction about the choice. I chose ML-DSA-65 because, weighing the standardisation, the parameter balance, the implementability and the threat, it is the most defensible floor available to me today. Conviction about the floor and humility about its permanence are not in tension. They are the same engineering posture, applied to the present and the future of the same decision. You commit fully to the floor and you refuse, just as fully, to mistake it for the ground.

The Mickai pantheon.

PART III · THE AGILITY

How a record carries its own algorithm history and survives the schemes it was signed with.

7. Crypto-agility as a property of records

Crypto-agility is usually described as a property of systems, the ability to swap one algorithm for another without rewriting everything around it. That is necessary but it is not sufficient, and the insufficiency is exactly where long-lived records get hurt. A system can be perfectly agile going forward, happily signing new records with a new scheme, while every record it already produced stays stranded under the old one. Forward agility protects tomorrow's records. It does nothing for the archive, and the archive is usually the thing that matters most.

The reframing I insist on is that agility has to be a property of the record itself, not only of the system that produced it. A record is agile if it can be re-secured under a new scheme without losing the proof of what it was and when, and if a future verifier can understand which scheme protected it at each point in its life. That capacity has to travel inside the record, because the system that made it may be long gone by the time someone needs to verify it. A record that depends on its original system still existing is not a record, it is a session.

> **Agility you cannot carry into the archive is agility that protects only the records you have not made yet.**

In the Mickai SIOS this is why the Open Audit Record is the central object rather than the signing call. The seal is not a single signature stamped on once and forgotten. It is a structure designed so that the record can accumulate cryptographic protection over time, so that a verifier reads not just a signature but a history of how this record was secured. The signing scheme is one entry in that history. When the floor rises, a new entry is added, and the old one is not erased, it is contextualised. That is what it means to make agility a property of the record.

8. The record that carries its own history

Concretely, an audit record that intends to outlive its algorithm has to carry, as first-class data, the answer to which scheme signed me, under which parameters, at what time, and according to which policy. None of that can be implicit. A verifier in 2035 cannot be expected to infer that a record was signed with ML-DSA-65 because that was the default in 2025, because defaults are exactly the institutional memory that gets lost. The algorithm identity has to sit inside the sealed data, bound to the signature it describes, so that it cannot be quietly changed after the fact.

Once the algorithm identity is part of the record, you can do the thing that makes long-term survival possible, which is to add new protection without destroying the old. When the floor moves from one scheme to its successor, the record is re-sealed under the new scheme in a way that also covers the original seal and its history. The result is a layered object, where each layer attests to everything beneath it, including which algorithm secured each previous layer. A verifier can walk that history and decide, for the period that concerns them, whether the protection in force at that time was sound.

Why old layers are kept, not replaced

There is a tempting shortcut, when a scheme is superseded, to simply re-sign everything under the new scheme and discard the old signatures. Resist it. The old signature is evidence about the past, and discarding it destroys your ability to prove what protected the record before the migration. If a dispute concerns an event from before the upgrade, you need to show what the protection was at that time, not what you replaced it with afterwards. Keeping the old layers, clearly marked as superseded, is what lets the record speak truthfully about its whole life rather than only its current state.

This layered structure is also what makes the migration auditable rather than a leap of faith. Because every transition is itself recorded inside the object, the act of upgrading the cryptography becomes part of the audit trail it protects. You can prove not only what a record says, but when and how its protection was strengthened, and under whose authority. The record does not merely survive the algorithm change, it documents it. That self-documentation is the difference between a record that outlives its scheme and a record that merely outlasts it by luck.

The Mickai pantheon.

9. Rotation without repudiation

Keys have to rotate. They are compromised, they expire, people leave, hardware is retired, policy mandates it. The hard part is not rotating the key, it is rotating it without giving anyone a reason to doubt the records the old key signed. A naive rotation throws the old key away and starts fresh, and in doing so it quietly undermines every record that depended on that key, because now there is no living key to anchor the trust, and an old signature with no clear, recorded provenance is exactly what a forger wishes existed.

Non-repudiation is the property that the signer cannot later credibly deny having signed. Rotation threatens it from both directions. Handle rotation badly and a legitimate signer can repudiate old records by pointing at the murky key history, while an illegitimate party can exploit the same murk to assert that a forged record is genuine. The defence is the same as for algorithm agility, the record and the key history have to be explicit, sealed, and continuous, so that the retirement of a key is itself a recorded, attested event rather than a silent gap.

> **A key that simply vanishes takes the trustworthiness of everything it signed with it.**

So in the model I build toward, rotation is an event that gets sealed, not a maintenance task that happens off the record. When a key is retired, the act of retiring it is signed, dated, and bound into the same audit structure as everything else, ideally under the new key and the old one together, so the handover is provably continuous. A verifier can then see an unbroken chain, this key signed these records during this period, then it was retired in this recorded event, and this successor took over. There is no moment where trust simply has to be assumed across a gap.

The same discipline applies when rotation is forced by an algorithm break rather than a key compromise. If ML-DSA-65 is deprecated, every key under it is effectively rotated at once, and the migration to its successor has to preserve non-repudiation across the whole archive. This is the convergence point of the whole book, key rotation and algorithm agility are the same problem viewed from two angles, and a record format that handles one cleanly is most of the way to handling the other. Solve them together or you will solve neither, because the day the algorithm falls is the day every key under it rotates whether you planned for it or not.

PART IV · THE PRACTICE

Migrating real records to last, and the discipline that keeps the promise honest.

10. Designing the Open Audit Record to last

Everything in the earlier parts converges on one artefact, a record format deliberately built to outlive its cryptography. In the Mickai SIOS that artefact is the Open Audit Record, and its design rules follow directly from the argument so far. It must carry its own algorithm identity, it must support layered re-sealing without discarding history, it must treat key and scheme transitions as recorded events, and it must be verifiable by a stranger who has only the public standards and the record itself. Each rule exists because some plausible future would otherwise leave a record unprovable.

The floor for that record today is FIPS 204 ML-DSA-65, sealing every consequential action the system takes. But the format is explicitly designed so that the floor is a field, not a fixed assumption. The day a successor scheme becomes the responsible floor, the record can adopt it, layer it over the existing seals, and continue, with the whole transition captured in the same audit structure. The record was never about ML-DSA-65 specifically. It was about being able to name, prove, and change its own protection over a span longer than any single scheme will credibly last.

Open by necessity, not ideology

I call it an Open Audit Record because openness is a survival requirement, not a branding choice. A record whose verification depends on proprietary tooling is a record that dies with the tooling. For something meant to be checked in 2035 by a party I will never meet, verification has to rest on published standards and an inspectable format, so that anyone with the public key and the specification can confirm what happened, independent of me, my company, or any product still being maintained. Sovereignty that cannot be independently verified is just a stronger word for trust me, and trust me is precisely what an audit record exists to replace.

This is also where the sovereign posture and the cryptographic posture reinforce each other. The records live on the operator's own hardware, fifty specialised brains running offline, and the seal is meant to let that operator prove their own history without depending on any external authority. A record format that carries its own algorithm history is exactly what a sovereign operator needs, because there is no central service to fall back on when the scheme changes. The operator's record has to be self-sufficient across decades, and self-sufficiency across decades is the whole specification in a single phrase.

The Mickai pantheon.

11. Migrating without breaking the past

A migration plan that only describes how new records will be signed is not a migration plan, it is a launch plan. The genuinely hard work is the existing archive, the records already signed under the current floor that must remain verifiable through and beyond the change. The first principle of doing this without breaking the past is the one established earlier, never discard the old protection. Re-seal forward, layering new signatures over old ones, so that at every moment the archive can prove both what it says and how it was protected at each stage of its life.

The second principle is that the migration itself must be sealed under the regime it is leaving as well as the one it is entering. Wait until the old scheme is fully broken before you migrate, and the migration events you record are themselves signed with a broken scheme and inherit its weakness. The window to migrate cleanly is while the current floor is still sound, which is precisely when the pressure to migrate feels lowest. This is the discipline that separates a real post-quantum posture from a stated intention, you move before you are forced, so the bridge between regimes is built on solid ground at both ends.

> **You have to cross the bridge while both banks are still standing.**

The third principle is to treat verification, not signing, as the thing that must never regress. It is acceptable, sometimes unavoidable, for a verifier to need multiple algorithm implementations to check an old archive, the old scheme to validate historical layers, the new scheme to validate current ones. What is not acceptable is for a record to reach a state where no available verifier can confirm any layer of its history. Designing for that means keeping verification paths for retired schemes alive long after

you stop signing with them, because a record you can no longer verify is a record you have effectively destroyed, however carefully you stored the bytes.

Put together, these principles describe a migration that is continuous rather than a cutover. There is no flag day when the old world ends and the new begins. There is a long overlap in which records accumulate layers, transitions are sealed, and verification spans multiple regimes, and that overlap is the normal, healthy state of a long-lived archive, not a temporary inconvenience to be rushed through. A system that expects a clean cutover will break its past at the seam. A system that expects perpetual overlap will carry its past intact.

12. The honest migration

I called this book Signing for 2035 because that year is far enough away to make the problem real and close enough to make it urgent. The records I seal today have to be verifiable then, through at least one probable change of cryptographic regime, without anyone having to take my word for what happened in between. Everything I have argued reduces to a single test, can a stranger in 2035, holding only the record and the public standards, confirm what was sealed, when, under which scheme, and prove the protection was sound for the period that concerns them. If the answer is yes, the migration was honest. If it depends on me still being around to vouch, it was not.

The honest migration, then, is not the adoption of a scheme. It is the adoption of a posture. The posture says, I will choose the best floor available and commit to it fully, I will assume that floor is temporary and design every record to survive its replacement, I will seal not only my actions but my algorithm and key transitions, and I will keep verification alive for everything I have ever signed. That posture does not end. There is no version of it where the work is finished and the algorithm is final, because the whole point is that finality is the thing we stopped believing in.

> **The migration is never done. That is not the failure of the plan, it is the plan.**

I want to close where I began, with accountability. Mickai is the Sovereign Intelligence Operating System, and the promise at its core is that a record made on your own hardware today will still be yours to prove decades from now. I will not make that promise on the strength of a single algorithm, however good, because I have read enough history to know how that sentence ends. I make it on the strength of a record that carries its own history, raises its own floor, and documents its own migration. ML-DSA-65 is where that record stands today. It is a good place to stand. It is not where the record will still be standing in 2035, and the entire architecture exists so that this is a feature and not a failure.

If you take one thing from this book, take the inversion. Stop asking which scheme will last and start asking whether your records can survive the scheme not lasting. The first question has no honest answer. The second has an engineering answer, and it is the one I have tried to give you here. Sign for 2035 by refusing to bet on 2035's cryptography, and by building records honest enough to admit, in their own structure, that you never could.

The Mickai pantheon.

APPENDIX · ABOUT THE AUTHOR

Micky Irons

Founder and chief executive of Mickai LTD (Companies House 17166618, registered office 20 Wenlock Road, London, N1 7GU) and named inventor on the Mickai SIOS patent corpus: 101 filed UK patent applications, around 2,234 claims. Trade mark Mickai registered at UK00004373277.

Profiles

mickai.co.uk

crunchbase.com/person/micky-irons

linkedin.com/in/mickyirons

References and further reading

· NIST FIPS 204, Module-Lattice-Based Digital Signature Standard (ML-DSA), National Institute of Standards and Technology, 2024. The normative specification for the signing floor discussed throughout this book.

· NIST FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM), 2024. The companion confidentiality standard, relevant to the harvest-now-decrypt-later threat to encrypted data.

· NIST IR 8547, Transition to Post-Quantum Cryptography Standards (draft), National Institute of Standards and Technology. Guidance on deprecating and disallowing classical schemes and sequencing a migration.

· CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation, Ducas, Lyubashevsky, Schwabe et al. The research lineage behind ML-DSA and the basis for confidence in the lattice assumptions.

· Quantum Computing and the Financial System: Spring Cleaning Ahead, and related central-bank and standards-body assessments of harvest-now-decrypt-later risk to long-lived records.

· ETSI and IETF working materials on cryptographic agility and algorithm identifiers in signed data formats, as background for designing records that carry their own algorithm history.