1337

ADVISORS

The Lifecycle of a Future Compute Right

A Design Analysis of How Future Compute Access Can Be Specified, Transferred, and Redeemed as a Formal Market Instrument

Mike Ossowski
1337 Advisors

Contents

Executive Summary

WP-01 established that compute is shifting from a consumed resource to a reserved one, and that this shift creates an allocation problem that bilateral procurement cannot solve. Organizations are committing to future capacity under uncertainty. Some will reserve too much. Others too little. The mechanisms to move excess reserved capacity toward the participants who need it do not yet exist at the scale the market requires.

WP-02 defines the instrument through which that reallocation becomes possible.

The starting observation is simple but consequential: when an organization reserves future compute capacity, the underlying hardware does not change hands. The GPUs, the racks, the network fabric, the datacenter itself, all of it stays exactly where it is. What changes, or should be able to change, is the entitlement to access that hardware. Compute futures, like the cash-settled contracts ICE and Ornn announced in May 2026, address one half of this problem: they let a participant hedge the price of future compute. They do not let a participant move the access itself from one holder to another. That gap, between pricing future access and transferring it, is the subject of this paper.

This paper introduces the Future Compute Right: a transferable, precisely specified entitlement to future compute access. It argues that the specification is the right itself, not a description attached to it, and that the categories of information required to make that specification meaningful are more numerous and more interdependent than in any prior commodity instrument. It compares the Future Compute Right to the markets that have already solved adjacent problems, leases, spectrum licenses, power purchase agreements, warehouse receipts, and financial transmission rights, and shows that no single analog is sufficient on its own. It then walks through the full lifecycle of the instrument: how a right is created and certified, how ownership and transfer work once the underlying compute never moves, and what actually happens at redemption.

The paper closes by arguing that the digital nature of both the underlying resource and the entitlement itself makes a tokenized implementation the natural, not merely fashionable, medium for this instrument. Structured metadata, programmable transfer, and automated redemption are not benefits incidental to tokenization. They are requirements the instrument already has, with or without a blockchain underneath it. And because that metadata is machine-readable by construction, the same instrument that lets a CFO transfer unused capacity to another business

unit also lets a software system, including an autonomous agent acting on a model's own behalf, evaluate, acquire, and redeem compute programmatically. That is not a speculative footnote. It is one of the strongest arguments for why this instrument should be digital-native from the start.

WP-03 takes up the market that forms around it.

Part I · The Right

What a Future Compute Right is, what it is made of, and why the specification is the instrument itself.

---

1. Why Compute Needs a Transferable Instrument

The evolution of compute markets has fundamentally changed the nature of what organizations purchase. Historically, compute was consumed on demand. An organization identified a need, procured the necessary resources, and consumed them within a relatively short period of time. The transaction was immediate, the planning horizon was limited, and the economic value of future access was modest.

As artificial intelligence systems have grown in sophistication and importance, that model has changed. Organizations increasingly compete not only for compute available today, but for compute that will be available months or years into the future. Hyperscalers and infrastructure providers have responded by expanding reservation programs, long-term commitments, and dedicated capacity agreements designed to secure future access to increasingly scarce resources.

This shift represents more than a change in procurement practice. It represents a change in the object being purchased. Organizations are no longer simply acquiring compute. They are acquiring future access to compute.

The distinction matters. Compute consumed today has immediate utility and immediate value. Future compute access derives its value from the ability to support objectives whose requirements remain uncertain. Model architectures evolve. Product roadmaps change. Customer demand fluctuates. Competitive dynamics shift. The further an organization forecasts its future compute requirements, the greater the probability that the forecast will diverge from reality.

This creates a structural challenge, one already documented in WP-01. Organizations that reserve too little future capacity may find themselves unable to secure resources necessary to support strategic initiatives. Organizations that reserve too much may discover that portions of their commitments are no longer required. In both cases, the original reservation decision may have been entirely rational given the information available at the time. The problem is not poor decision making. The problem is uncertainty, and uncertainty compounds as commitments grow larger and extend further into the future.

Historically, markets facing this exact challenge have responded the same way: by developing transferable instruments that let rights and obligations move between participants as circumstances change. Leases can be assigned. Spectrum licenses can be transferred. Power contracts can be bought and sold. In each case, transferability provides flexibility in environments where future needs cannot be predicted perfectly.

Compute markets have made a first move in this direction, but only a partial one. In May 2026, Intercontinental Exchange and Ornn announced cash-settled GPU compute futures, referencing the Ornn Compute Price Index and covering H100, H200, B200, and RTX 5090 compute. The arrival of a futures contract is a meaningful signal: it means the market now agrees that future compute access has a price worth hedging. But a cash-settled futures contract solves only the financial dimension of the problem. It lets a participant manage the cost of future compute. It does not let that participant move the access itself to someone who needs it more.

The more fundamental question is how future access itself can be reallocated.

> **KEY INSIGHT**
>
> A compute futures contract and a Future Compute Right solve different problems. Compute futures transfer price risk. Future Compute Rights transfer access. A participant can hedge the cost of compute without ever being able to move the underlying access, and can hold an enforceable claim on future access while remaining fully exposed to price risk. The market has built the first instrument. It has not yet built the second.

Before future access can be transferred, it must first be defined. A participant cannot acquire, value, transfer, or redeem an entitlement that lacks a clear specification. The market therefore requires an instrument capable of describing future compute access in a standardized, transferable form.

This paper introduces that instrument: the Future Compute Right, a transferable entitlement to future compute access defined by a structured set of specifications governing the resources, environment, timing, ownership, and redemption characteristics associated with that entitlement.

The purpose of this paper is not to prescribe a final implementation standard. It is to establish a foundational framework for thinking about future compute access as a transferable right, and to examine the information required to define, transfer, and ultimately redeem that right within emerging compute markets.

Section 2 begins with the most basic question this framework must answer: what, precisely, is a Future Compute Right?

---

2. Access Without Ownership

The concept of a Future Compute Right begins with a simple observation: the underlying compute resource itself is not what changes ownership. GPUs do not move between participants. Servers do not move. Racks, networks, storage systems, and datacenters remain exactly where they are. What changes ownership is the entitlement to access those resources.

This distinction shifts the discussion away from compute as infrastructure and toward compute as an access right. When an organization enters into a reservation agreement with a hyperscaler or infrastructure provider, it is not purchasing ownership of physical assets. It is acquiring the right to access those assets under specified conditions at a future point in time.

In many respects, this resembles a lease. A tenant does not purchase the building. The tenant acquires the right to use the building for a specified period under defined terms and conditions. The building remains in place while the access right may be assigned, transferred, exercised, or allowed to expire. A Future Compute Right applies the same concept to compute infrastructure. The underlying assets remain unchanged while the entitlement to access them becomes the object of ownership and transfer.

A Future Compute Right can therefore be defined as a transferable, precisely specified entitlement to future compute access.

Each element of this definition carries weight.

The entitlement must be transferable. If ownership cannot change, the instrument cannot facilitate reallocation when compute requirements change. Transferability is the mechanism through which future access moves from participants who no longer require it to those who do.

The entitlement must be future-oriented. The purpose of the instrument is not to govern compute being consumed today. It is to describe access that becomes available at a future date, and that may need to be reallocated before that date arrives.

Most importantly, the entitlement must be precisely specified. An access right that simply grants the holder the ability to access "compute" is insufficient, because compute is not a homogeneous resource. The economic value of future compute access depends on the characteristics of the resources being accessed, the quantity of those resources, the environment in which they operate, the time period during which access is available, and the conditions governing ownership, transfer, and redemption.

This raises a natural question: access to what, exactly?

The answer is that a Future Compute Right represents access to a specifically defined compute entitlement. The holder is not acquiring a generic claim on compute. The holder is acquiring a claim on compute that conforms to a particular set of characteristics and requirements, and those characteristics ultimately define the right itself.

This is the critical distinction. In most markets, a specification describes an asset that exists independently of it. A car has a VIN number, but the VIN does not make the car. A Future Compute Right has no such independent existence: strip away the specification and there is nothing left to hold, value, or transfer.

> KEY INSIGHT
>
> In many markets, specifications are attached to an asset. In a Future Compute Right, the specification is the right. Without specification, there is no meaningful entitlement to transfer, value, or redeem. The specification is what transforms an abstract access right into a transferable instrument.

The importance of specification becomes concrete when considering capacity. A right providing access to a single GPU for one hundred hours represents the same aggregate number of GPU-hours as a right providing access to one hundred GPUs for a single hour. Both describe identical aggregate capacity. They do not describe identical economic value. A distributed training run that requires one hundred accelerators operating simultaneously gains almost nothing from a right that delivers a single accelerator at a time, no matter how many cumulative hours that accelerator remains available. Aggregate GPU-hours is a convenient comparison metric. It is not a sufficient

description of what the holder can actually do with the right. Concurrency, the number of accelerators available to the holder at any single moment, has to be specified independently of total duration.

For this reason, a Future Compute Right should not be viewed merely as a contract describing future compute access. It should be viewed as a standardized instrument capable of carrying a defined entitlement between market participants. Ownership of the instrument conveys ownership of the entitlement. Transfer of the instrument transfers the entitlement. Redemption of the instrument converts the entitlement into access.

Once future compute access can be defined this precisely, it becomes possible to separate the entitlement from the original reservation agreement and let ownership move independently of the underlying infrastructure. That separation is the foundation on which secondary markets, benchmark methodologies, tokenized representations of compute access, and eventually both cash-settled and physically settled compute futures can be built.

Before any of that, the entitlement itself has to be defined in enough detail to survive contact with a market. The next section maps what that detail actually has to include.

---

3. The Anatomy of a Future Compute Right

The previous section established a foundational principle: the specification is the right. A Future Compute Right derives its meaning not from the existence of an entitlement alone, but from the information that defines what that entitlement actually provides. Without specification, there is no meaningful way to value, transfer, compare, or redeem future compute access.

This raises a natural question: what information must a Future Compute Right actually contain?

The answer that follows should not be mistaken for a final industry standard. Compute infrastructure continues to evolve, new accelerator types and interconnect generations arrive faster than any taxonomy can be finalized, and the precise fields required to define future access will evolve along with it. What follows is a proposed framework, drawn from how compute reservations actually specify, or fail to specify, the resources they govern today. It is offered for scrutiny and refinement by providers, buyers, and legal and technical teams, not as a closed standard. The more durable claim is narrower: a Future Compute Right must answer a small set of foundational questions before it can be valued, transferred, or redeemed.

What is being accessed? How much is being accessed? In what environment does it operate? When is access available? Who is obligated to provide it? Under what conditions may ownership change? How is access claimed? When does the entitlement terminate?

Eight categories answer those eight questions.

Resource Specification. This category establishes the identity of the underlying compute resource: accelerator type, hardware generation, memory configuration, and related characteristics. The granularity required here is easy to underestimate. An H100 SXM and an H100 PCIe share a product name and a generation. They do not share an interconnect architecture or a memory bandwidth profile: the SXM variant delivers 3.35 TB/s of memory bandwidth, the PCIe variant roughly 2.0 TB/s. For a training workload that is memory-bandwidth bound, which most large model training runs are, that is not a cosmetic difference. It is the difference between two instruments that happen to share a name. Access to "compute" is not a meaningful description of a Resource Specification. Access to a specifically defined compute resource is.

The same gap shows up at the inference layer, where the workload name alone is even less informative.

Accelerator	Per-GPU Throughput
H100 SXM	3,913 tok/s
H200 SXM	4,432 tok/s
B200	12,357 tok/s

Exhibit 1: Per-GPU inference throughput, llama2-70b, identical scenario.
Source: 1337 Advisors Platform · platform.1337advisors.com/intelligence/llm

Generation alone moves the number by more than three times. A specification that names the model and confirms it ran is not a specification of what was actually delivered.

Capacity Specification. Knowing what resource is being accessed is insufficient without knowing how much of it is being provided, and how. This category covers accelerator count, duration, aggregate GPU-hours, and concurrency. As established in Section 2, aggregate GPU-hours alone cannot describe economic utility: a single accelerator for one hundred hours and one hundred

accelerators for one hour are the same number on a ledger and different instruments in practice. Capacity Specification has to preserve concurrency as an independent field, not derive it from duration after the fact.

Environment Specification. Compute does not operate in isolation, and the environment around it frequently determines as much of its usefulness as the hardware itself. This category includes network topology, interconnect, storage architecture, orchestration framework, geographic location, and compliance posture. Two clusters with identical accelerator counts can deliver materially different training throughput depending on interconnect alone: NVLink 4.0 delivers roughly 900 GB/s of bidirectional bandwidth per GPU, while PCIe Gen5 delivers around 128 GB/s. For distributed training across hundreds of accelerators, that gap is often the actual binding constraint on throughput, not the accelerators themselves. Software environment carries similar weight in practice: a mismatch between the CUDA version a holder's training code requires and the version a provider's driver stack supports is one of the more common, and most avoidable, causes of training failure in current compute procurement, and it happens precisely because software environment is rarely specified in reservation agreements today.

Temporal Specification. A Future Compute Right is fundamentally a time-based instrument. This category defines when access becomes available, how long it remains available, and the conditions governing that window: start date, duration, notice requirements, and substitution terms. Substitution terms deserve particular attention because they are the field a legal reader will ask about first. Hardware generations turn over. A right specifying an H100 SXM reserved eighteen months in advance may not be deliverable against that exact specification if the provider has transitioned its fleet to a newer generation by the time the window opens. Without substitution terms defined in advance, that equivalence question gets negotiated bilaterally at the exact moment the holder most needs certainty, which defeats much of the purpose of having a standardized instrument in the first place.

Provider Specification. Every Future Compute Right ultimately represents an obligation, and this category identifies who bears it: legal entity, service commitments, reliability history, certifications, and the nature of the delivery commitment itself. That last point is the most consequential distinction in the entire category. Current cloud contracts commonly use "commercially reasonable efforts" language to describe capacity availability. That is not a guarantee of delivery. It is a commitment to try. A Future Compute Right that aspires to function as a transferable instrument needs an actual delivery guarantee, because a right whose underlying obligation is merely best effort cannot be valued or transferred with any confidence. This is an upgrade over current market practice, not a restatement of it.

Ownership and Transfer Specification. The defining characteristic of a Future Compute Right is transferability, and this category defines the conditions under which the entitlement may change hands: restrictions, divisibility, assignment rules, and holder eligibility. The goal is not to establish who owns the entitlement today. It is to define the circumstances under which ownership may change tomorrow, a question Section 6 takes up directly.

Redemption Specification. This category defines what is required to convert the entitlement into access: verification requirements, redemption deadlines, notice periods, and credential delivery. The precise operational workflow will vary by provider and platform. What has to be invariant is that the holder has a clearly defined, unambiguous path from holding the right to using the compute it describes.

Lifecycle Specification. Every Future Compute Right eventually reaches a terminal state, and this category defines the rules governing that finality. In the framework developed in this paper, there are exactly two terminal outcomes: the holder redeems the entitlement and receives access, or the access window closes without redemption and the entitlement expires.

Category	Question Answered	Representative Fields
Resource	What is being accessed?	Accelerator type and generation, memory per accelerator, hardware class
Capacity	How much is being accessed?	Accelerator count, duration, aggregate GPU-hours, concurrency
Environment	In what environment does it operate?	Network topology, interconnect bandwidth, storage architecture, region, compliance certifications
Temporal	When is access available?	Start date, duration, notice period, substitution terms
Provider	Who is obligated to provide it?	Legal entity, reliability history, support tier, delivery guarantee vs. commercially reasonable efforts
Ownership and Transfer	Can ownership change?	Transferability, divisibility, assignment rules, holder eligibility
Redemption	How is access claimed?	Verification requirements, redemption deadline, credential delivery
Lifecycle	When does the entitlement terminate?	Redeemed status, expired status, retirement

Exhibit 2: Specification fields by category, illustrative and not exhaustive.
Source: 1337 Advisors.

These categories are not intended to represent a final industry standard. They are intended to illustrate the types of information required to transform future compute access into a transferable instrument.

Importantly, a detailed specification does not require every participant to engage with every field. A buyer who cares only about accelerator type and price can transact on that basis alone. A buyer running a large distributed training workload may treat network topology as decisive. A third buyer may care primarily about geography, compliance, or provider identity. The purpose of the specification is not to force uniform priorities onto the market. It is to preserve information, so that whatever a given participant cares about is available, standardized, and attached to the entitlement itself.

> KEY INSIGHT
>
> Specification enables abstraction. A complete specification lets the market decide, transaction by transaction, how much detail any given participant needs to engage with. An incomplete specification removes that choice for everyone, because detail that was never captured cannot be recovered later, no matter how much a buyer or seller might want it.

A barrel of WTI crude oil needs roughly four or five fields to describe it: commodity type, quantity, grade, delivery location, delivery date. A Future Compute Right, even under a deliberately preliminary and still-evolving taxonomy, needs closer to thirty once every category above is filled in with its full set of fields. That gap is not complexity for its own sake. It reflects what compute actually is: a layered system in which the hardware, the network, the storage, the software, the location, and the provider relationship each carry their own independent claim on the specification. Strip any load-bearing field and the right becomes ambiguous. An ambiguous right is unenforceable. An unenforceable right cannot be transferred, and a right that cannot be transferred cannot participate in a secondary market.

Before future access can be transferred, it has to survive being written down completely. Part II turns to how this entitlement compares with the instruments markets have already built to solve adjacent problems, and where each comparison breaks down.

Part II · The Complexity

Why no single existing market instrument fully captures what a Future Compute Right has to do, and what that gap actually costs.

---

4. Future Compute Rights in Context

A Future Compute Right may look novel at first glance. It is better understood as the latest entry in a long, well-documented pattern: markets have repeatedly built transferable rights whenever access to a scarce, economically significant resource became valuable enough to outgrow simple spot transactions. No existing instrument mirrors the characteristics of future compute access exactly. Several mature markets nonetheless provide useful, and instructive, analogs.

The pattern is worth naming directly, because it shows up across every market examined in this section. Each time a commodity resisted simple spot trading, the market built its forward infrastructure in the same order: a financial layer first, instruments that priced the underlying and let participants hedge, followed later by a physical delivery layer, instruments that actually transferred possession or access. NYMEX launched the Henry Hub natural gas futures contract in 1990. It priced gas to a single benchmark delivery point. It did not solve the problem that a buyer in Chicago or Boston faced a different delivery cost than the benchmark implied, because pipeline capacity between delivery points was constrained. That gap, known as basis, was eventually addressed by a second layer of instruments: locational swaps and the physical scheduling systems that govern actual gas delivery today. Electricity markets followed the same order. Financial Transmission Rights were invented to hedge the congestion cost between grid nodes; Power Purchase Agreements carry the actual delivery obligation. Grain futures priced the commodity from the 1850s onward; the warehouse receipt, a negotiable document of title formalized under UCC Article 7 and extended to electronic form in a 2004 revision, is what actually lets a holder claim delivery.

Compute now has its financial layer. The ICE/Ornn cash-settled futures discussed in Section 1, priced off an index built from printed transactions rather than surveyed quotes, are that layer. They price compute. They do not deliver it. Compute is, by this reading, at exactly the stage in its market development where every comparable commodity has previously needed to build its physical delivery layer. This paper is an attempt at that layer, and the rest of this section examines the instruments each prior market built along the way.

Leases and Subleases. The most intuitive analogy for a Future Compute Right is a lease. A tenant leasing office space does not acquire ownership of the building. Ownership remains with the landlord. What the tenant acquires is the right to access and use a defined portion of the property for a defined period, under defined terms. The economic value of the lease comes from access, not ownership. Leases can be assigned or subleased: when that happens, the building does not move. What changes is the identity of the party entitled to use it. A Future Compute Right works the same way, with the infrastructure provider as landlord and the entitlement, not the underlying hardware, as the asset that actually changes hands.

Spectrum Licenses. Where leases explain transferability, spectrum licenses demonstrate the importance of specification. Spectrum holders do not own radio frequencies. They hold the right to use a specific portion of the electromagnetic spectrum under a detailed technical specification: frequency range, geographic restriction, power limit, duration. The FCC went further than simply granting these rights; in 2003 it created a formal secondary market framework for spectrum leasing, demonstrating in practice that a right defined almost entirely by

technical metadata could support an active, regulated secondary market. The value of a spectrum license is inseparable from that specification. Two licenses can look similar and carry dramatically different economic value depending on the metadata attached to each. A Future Compute Right shares this property directly, which is the entire argument of Section 3: the specification does not describe the entitlement, it defines it.

Power Purchase Agreements. PPAs demonstrate a different principle: future access itself can become the object of a transaction, well before the underlying resource is consumed. A PPA buyer commits today to electricity that will be produced and delivered later, and the value of the agreement comes from the future availability of the resource, not from immediate consumption. The forward-commitment behavior documented throughout WP-01, hyperscalers signing multi-year GPU contracts before workloads are fully defined, is the compute-market equivalent of PPA behavior. Organizations have already proven they will commit to future resource access years in advance, under enforceable obligations, at a scale that makes the cost of being wrong substantial. What current compute contracts lack, and what a PPA does not need to solve because electricity is fungible at the delivery point, is a way to make that future commitment transferable to someone other than the original buyer.

Warehouse Receipts. A warehouse receipt is historically the clearest example of separating a claim from the asset it represents. The receipt is evidence that goods were deposited with a custodian; ownership of the receipt conveys a claim on those goods without the goods themselves moving, and the receipt itself can be transferred, pledged, or redeemed for delivery. The relevance to compute is not the physical nature of the goods. It is the separation of entitlement from possession, formalized under law well before any of this was imagined for compute. A Future Compute Right represents a similar claim: transferable independent of the underlying resource, redeemable later for access. Its limit is temporal. A warehouse receipt covers goods that already exist in storage. A Future Compute Right covers a resource available at a future date, a forward warehouse receipt for a service rather than a stored commodity.

Financial Transmission Rights. FTRs illustrate how infrastructure constraints, on their own, can be valuable enough to justify an entirely new instrument. Electricity markets are shaped not just by generation capacity but by the limits of the transmission network connecting producers to consumers; a megawatt in Chicago is not a megawatt in Boston when the lines between them are congested. FTRs were created specifically to let participants manage the economic consequence of that constraint. Compute exhibits the same kind of constraint without yet having an instrument for it: a cluster in Virginia is worth something different than a nominally identical cluster in

Singapore, because of latency, power cost, compliance posture, and proximity to data. A Future Compute Right addresses this directly, by carrying location and the rest of the environment as explicit specification fields, rather than through a separate hedging instrument layered on top.

Where the Market Already Shows the Strain. The clearest evidence that compute needs this instrument is not theoretical. It is already visible in how providers handle transfer requests today. AWS opened a Reserved Instance Marketplace in 2012, an early and reasonably successful attempt at letting customers resell unused reserved capacity. In January 2024, AWS banned resale of discounted Reserved Instances specifically, closing the part of the secondary market that let buyers capture the spread between list price and discounted reservation price. Google Cloud reservations remain zone-specific and non-transferable. These are not arbitrary restrictions. They reflect a real, unsolved problem: providers cannot extend the trust, compliance posture, and pricing terms of an original buyer to an unknown secondary holder without a mechanism to verify that holder independently. That is precisely the gap a properly specified, properly verified Future Compute Right is built to close, a question Section 6 returns to directly.

A Synthesis of Established Concepts. No single analogy captures a Future Compute Right completely. Leases explain how access rights separate from ownership and transfer between participants. Spectrum licenses demonstrate how a detailed specification can define an economically valuable entitlement on its own. Power Purchase Agreements show how future access becomes the object of a transaction well before consumption. Warehouse receipts show how a claim can move independently of the asset it represents. Financial Transmission Rights show how an infrastructure constraint, by itself, can justify a new instrument class. Each one solved a piece of this problem for a different resource, at a different point in market history.

> KEY INSIGHT
>
> A Future Compute Right is not a deviation from how markets normally solve this kind of problem. It is the next entry in a pattern with a century and a half of precedent: financial layer first, physical delivery layer second, and a new instrument built each time an existing one proved insufficient for a resource's specific constraints. Compute has its financial layer. What it does not yet have is what every comparable market eventually built.

Future Compute Rights combine elements of all five instruments, but the combination is not arbitrary. It is a response to a resource that is simultaneously standardized and highly differentiated. Two buyers may both want H100 access while assigning entirely different value to topology, network, storage, location, or provider identity. In some contexts compute looks like a commodity. In others, the differences are economically decisive. That duality is why a Future

Compute Right needs a richer specification than any of its individual ancestors, and why Part III now turns to what happens to that specification once it starts moving through a real market: how the right is created, how ownership changes hands, and what happens when a holder redeems it.

Part III · The Lifecycle and the Medium

How a Future Compute Right is created, owned, transferred, and redeemed, and why its digital nature makes it a natural candidate for tokenization, not as a trend, but as a structural fit.

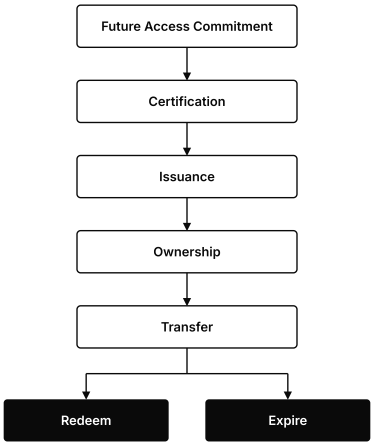

Exhibit 3: The lifecycle of a Future Compute Right, from initial commitment to terminal outcome.

Source: 1337 Advisors.

5. Creation and Certification

A Future Compute Right does not create compute capacity. It creates a transferable entitlement to future compute access.

This distinction matters because the value of a Future Compute Right is ultimately derived from an underlying commitment. The right itself does not generate additional GPUs, expand datacenter capacity, or increase the supply of compute available to the market. It represents access to capacity that already exists, or is committed to exist, for future use. A Future Compute Right

should therefore be understood as a representation of a future access commitment, not as a synthetic instrument created independently of any underlying resource. The integrity of the right depends entirely on the integrity of the commitment it represents.

How does a Future Compute Right actually come into existence?

It begins with a future access commitment: a reservation agreement, a dedicated capacity arrangement, a long-term infrastructure contract, or some other mechanism through which future access is secured. Whatever form it takes, the arrangement establishes an entitlement to future compute access under a defined set of conditions. At this stage, however, the entitlement remains embedded in the original agreement. It has not yet become a standardized, transferable instrument.

The next step is certification: the process through which the underlying entitlement is verified and prepared for issuance as a Future Compute Right. Certification adds nothing to the entitlement that was not already there at the moment of commitment. What it adds is confidence: that the entitlement is real, identifiable, and capable of being fulfilled. The entity responsible for certification may vary by market structure. The provider may perform this function directly. An authorized issuer, marketplace operator, or independent custodian might perform it instead. The more important principle is that a Future Compute Right has to be linked to a verifiable future access commitment, not merely an asserted one.

This is not a hypothetical operational gap. Providers are already building exactly this kind of verification infrastructure, for reasons that have nothing to do with secondary markets. CoreWeave's MLPerf Training v6.0 submission in June 2026 was built on what the company calls Mission Control, a system that continuously performs health checks across its fleet, validating hardware, firmware, network, and thermal status before and during large-scale training jobs, specifically to ensure that customer workloads run on infrastructure that has been verified rather than merely provisioned. That is certification in practice, built for an entirely different reason: making sure the capacity a customer is promised is the capacity a customer actually gets. A Future Compute Right needs the same discipline applied to the reservation itself, before it ever changes hands.

> KEY INSIGHT
>
> Certification is not an administrative formality layered on top of a Future Compute Right. It is the mechanism that converts an asserted future commitment into a verifiable one, which is the only kind of commitment a secondary market can safely price.

Once certified, the entitlement may be issued as a Future Compute Right. Conceptually: a future access commitment leads to certification, which leads to issuance, which produces the Future Compute Right. Ownership of the right then conveys ownership of the entitlement. Transfer of the right transfers the entitlement. Redemption of the right converts the entitlement into access.

Issuance does not imply standardized capacity. A Future Compute Right may represent a single accelerator, a cluster, a defined quantity of compute-hours, or a more complex entitlement built from the specification categories described in Section 3. The specification determines the nature of the entitlement; issuance simply transforms that entitlement into something transferable. Two rights from different providers, representing different quantities, operating in different environments, are not meant to be made identical by issuance. They are meant to be made transferable while remaining exactly as different as their specifications say they are.

Trust is therefore the foundational requirement of the entire system. Market participants need confidence that the underlying entitlement exists, that the specification describes it accurately, and that the obligated party can actually fulfill the commitment at redemption. Certification is what makes that confidence possible, and it is what makes the rest of the lifecycle, ownership, transfer, and redemption, function as more than a paper exercise.

---

6. Ownership and Transfer

The purpose of a Future Compute Right is not merely to define future compute access. It is to let that access change hands.

This matters because the original holder of a future compute entitlement will not necessarily be its ultimate consumer. As Section 1 established, every reservation is a forecast, and forecasts diverge from reality for reasons that have nothing to do with poor decision making.

The result is a market in which some participants hold future compute access they no longer need, while others need more capacity than they originally secured. The compute may exist. The problem is that access to it has already been allocated to someone who no longer values it as highly as someone else would.

A Future Compute Right addresses this by separating future access from the original reservation agreement and letting ownership of the entitlement move independently of the underlying infrastructure, the same separation established in Section 2. What changes is the identity of the

party entitled to access those resources during the window the instrument defines, not the infrastructure itself. This mirrors how transfer already works in markets with a similar structure: a sublease does not relocate a building, and a spectrum transfer does not relocate spectrum.

Transferability does not mean every right transfers whole. Depending on the specification and the terms set at issuance, some rights may be divisible and others may not. A right representing access to one hundred GPUs might be used entirely by a single organization, or the holder might require only a portion and transfer the remainder. Whether that is permitted depends on the instrument's terms, not on some inherent property of compute itself. Transferability also does not mean unrestricted transfer. Providers may require qualifications before allowing a right to change hands: regulatory standing, jurisdictional eligibility, security posture, creditworthiness, or technical certification. None of this is unusual. Similar conditions already govern transfer in markets involving regulated assets and long-term contractual rights.

This is the problem current compute contracts already run into, and why most of them remain non-transferable. Section 4 noted where this plays out in practice: AWS's Reserved Instance Marketplace and its 2024 restriction on reselling discounted reservations, Google Cloud's flatly non-transferable zone-specific reservations. In both cases the underlying objection is the same: the provider has no reliable way to know whether a secondary holder meets the same standing, compliance posture, and creditworthiness as the original buyer, and absent that knowledge, restricting transfer is the only available risk control.

> KEY INSIGHT
>
> Ownership and consumption are not the same thing, and treating them as if they were is the reason most compute reservations today are locked to whoever signed the original contract. The organization that secures future access is not always the organization that should ultimately consume it. A Future Compute Right exists specifically to let that mismatch resolve itself without requiring the original infrastructure commitment to be renegotiated.

This is precisely where certification, established in Section 5, does its real work. A provider's objection to transfer is not really about the right changing hands. It is about not knowing who is now holding it. A Future Compute Right whose holder eligibility has already been verified, the same certification discipline that established the right's legitimacy at issuance, gives the provider exactly the assurance that today's bilateral contracts cannot provide: that whoever presents the right at redemption has already been vetted, regardless of how many times the right has changed hands since issuance. Transferability and provider trust are not in tension. A properly certified instrument is what reconciles them.

Without transferability, forecast errors stay locked into the original allocation. Organizations that over-reserve cannot recover value from unused commitments. Organizations that under-reserve cannot obtain what they need without renegotiating from scratch. With transferability, future access can migrate toward whoever values it most, providers benefit from more efficient utilization of already-committed capacity, and the market gains a mechanism for absorbing forecast error that does not require unwinding the underlying infrastructure relationship at all.

7. Redemption and Delivery

Transferability addresses the allocation problem. Redemption addresses the access problem.

A Future Compute Right derives its value from the entitlement it represents. Ownership lets the holder possess, transfer, or retain that entitlement. Redemption is the act of exercising it: converting the right into actual access to compute resources, by activating an entitlement that already exists rather than creating a new one. That entitlement was established through the original commitment, defined through the specification, and carried by the instrument through however many transfers it has been through.

What is actually delivered at redemption is not a GPU, a server, a rack, or a datacenter. The physical infrastructure stays exactly where it has always been. A lease does not deliver a building; it grants access to one. A spectrum license does not deliver spectrum; it grants the right to use it. Redemption of a Future Compute Right works the same way: it grants access to infrastructure conforming to the specification, not ownership of the infrastructure itself.

This is where Section 3's Provider Specification, and specifically the distinction between a delivery guarantee and "commercially reasonable efforts," stops being an abstract design choice and becomes the entire question at redemption. A right backed only by reasonable efforts gives the holder no enforceable claim if the provider simply cannot deliver on schedule, which is exactly the failure mode a transferable instrument cannot tolerate: a secondary buyer has no relationship with the provider to fall back on, no history of goodwill to invoke, nothing but the instrument itself. A Future Compute Right that intends to function in a secondary market has to carry an actual delivery guarantee, verified at certification and enforceable at redemption, or it is not really a transferable instrument at all. It is a best-effort contract wearing a different label.

The specific operational mechanics of redemption, how credentials are issued, how access is provisioned, what verification steps occur, will vary across providers and platforms, and this paper does not attempt to prescribe a universal workflow. What has to hold regardless of

implementation is that the provider verifies ownership of the entitlement, confirms the right remains valid and within its access window, and grants access consistent with the specification, in that order, every time.

Redemption is a terminal state. Once access has been granted, the holder possesses compute access rather than a transferable right, and the instrument itself is retired. Not every right reaches that state through redemption. A holder may choose not to exercise the entitlement, or circumstances may render the underlying compute unnecessary before the window closes. When the access window closes without redemption, the right expires, which is the second terminal state.

> KEY INSIGHT
>
> Disputes over a Future Compute Right will almost never originate at redemption itself. They will trace back to ambiguity introduced upstream, in a specification field that was left vague, incomplete, or open to interpretation back in Section 3. Redemption is simply where that ambiguity finally gets priced.

These two outcomes complete the lifecycle this paper has built section by section, from a future access commitment to its final resolution.

What the holder actually possesses changes at every stage along the way: a contract at reservation, a verified entitlement once certified, the Future Compute Right itself from issuance through however many transfers it goes through, and then exactly one of two things at the end. Either it becomes compute access, or it becomes nothing. There is no third outcome and no partial credit. That is what makes the instrument enforceable rather than aspirational.

What that lifecycle makes possible, once it can run end to end on a standardized instrument, is the subject of the final section.

---

8. From Rights to Markets

The preceding sections focused on a single objective: defining a transferable instrument capable of representing future compute access. The discussion deliberately stayed close to the instrument itself rather than the markets that might eventually form around it.

That restraint matters, but it should not be mistaken for modesty about the implications. A Future Compute Right is not valuable because it might create new markets. It is valuable because it lets future compute access be defined, owned, transferred, and redeemed at all. Markets are a consequence of that capability, not the reason for building it. Once future access can be represented this way, several things become possible that are not possible today.

The most immediate is a secondary market for future compute access itself. As future commitments grow larger and extend further out, forecast divergence becomes more likely, not less. A Future Compute Right gives organizations a way to accommodate that divergence without renegotiating the underlying infrastructure commitment. Providers continue to allocate future capacity through reservation agreements exactly as they do today. A Future Compute Right creates a mechanism for that allocation to evolve afterward, as circumstances change.

A second consequence is better information for benchmarking and pricing. Compute benchmarking today relies on provider pricing, bilateral transaction data, and reservation agreements, methodologies that represent real progress but that are necessarily backward-looking and dependent on whoever is willing to disclose. Firms like Ornn are already building index infrastructure for the financial layer, and firms building independent reference and authentication layers for compute as a financeable asset, the kind of work Setara Financial Corp has been doing around GPU-backed debt benchmarking, are building the analogous infrastructure for the credit side. A mature secondary market in Future Compute Rights would not replace either. It would add a third source of market-derived pricing information: what participants actually pay to acquire or offload specific, fully specified future access, under real transfer conditions rather than survey responses. That data is complementary to existing benchmark work, not competitive with it.

The market is already moving in this direction from the financing side, independent of anything proposed in this paper. CoreWeave priced a combined $3.25 billion in senior notes in 2026 and was added to the Nasdaq-100 index, and GPU-backed debt facilities have seen financing spreads compress from low-teen yields in early transactions to roughly 2.25% over SOFR in more recent ones, with at least one facility receiving an A3 rating from Moody's. None of that happens unless institutional capital already believes compute is a durable, financeable asset class. A Future Compute Right is the instrument that lets that belief extend from the balance sheet of the original buyer to an actual secondary market, rather than staying locked inside whichever company happened to sign the original reservation.

This is also where the distinction from Section 1 matters most. Compute futures, including the cash-settled contracts ICE and Ornn announced in May 2026, transfer price risk. A Future Compute Right transfers access. An organization can fully hedge the price of future compute and still be unable to obtain the resources it actually needs. Another organization can hold real access rights while remaining exposed to price movement. These are genuinely different problems, and solving one does not solve the other. Looking further out, this distinction is also what a physically settled compute future would actually require: a mechanism for specifying exactly what gets delivered against the contract. A Future Compute Right, or a defined basket of them meeting a contract's criteria, is a plausible answer to that question, though the market structure required to support it belongs to WP-03, not this paper.

The strongest implication, however, is not about futures contracts at all. It is about what kind of object a Future Compute Right actually is.

Because both the underlying resource and the entitlement itself are inherently digital, a Future Compute Right is a natural candidate for tokenization. This is not enthusiasm for tokenization as a trend; the instrument's value comes entirely from the transferable entitlement it defines, full stop, with or without a token underneath it. But the specific characteristics this paper has built into the instrument, identifiable ownership, structured metadata, programmable transfer conditions, automated redemption, defined expiration, map almost exactly onto the capabilities tokenized assets were originally designed to provide. A tokenized Future Compute Right is therefore less an implementation choice than the closest fit available: the form that lets the specification travel with the right rather than living in a separate document that has to be checked against it every time the right changes hands.

This is the argument that matters most, and it is worth stating directly rather than as a hedge at the end of a long list of possibilities: a digital, metadata-rich, machine-readable Future Compute Right is not only easier for a human buyer or a CFO's procurement system to transfer and redeem. It is the form of the instrument that an autonomous software agent could evaluate, acquire, transfer, and redeem on its own, without a human approving each step. That is not a distant or speculative use case bolted onto the paper's conclusion. It is one of the strongest possible arguments for why this instrument needs to be digital-native from the start, rather than digitized after the fact. AI systems already procure services, optimize workloads, and make resource decisions with increasing autonomy. As that autonomy extends to compute itself, including a model acquiring access to the compute it needs to run, the entity transacting is not always going to be a person clicking through a procurement portal. It is going to be code, reading a specification, checking it against a need, and executing a transfer. A Future Compute Right, built exactly as Sections 3 through 7 of this paper describe it, is the form that transaction

requires. Tokenization is not the innovation here. The Future Compute Right is the innovation. Tokenization is simply the medium that lets it function exactly as designed, for human and machine participants alike.

> KEY INSIGHT
>
> A Future Compute Right does not need a blockchain to be a Future Compute Right. It needs structured metadata, programmable transfer, and machine-readable redemption, which a tokenized instrument happens to provide more naturally than any paper contract ever could. That is the entire case for tokenization. It was never about the word. It was always about what the instrument has to be able to do.

WP-01 examined why future compute access may need to be reallocated. This paper has examined the instrument that makes reallocation possible. The next question is how a market actually forms around that instrument: pricing, liquidity, exchange structure, clearing, and the risk management a secondary market in compute access would require. That is the subject of WP-03.

Conclusion

WP-01 diagnosed a problem: compute is shifting from consumption to reservation, and reservation under uncertainty produces allocation mismatches that bilateral procurement cannot resolve on its own. This paper has proposed the instrument required to solve it.

A Future Compute Right is not a complicated idea dressed up in complicated language. It is the recognition that when an organization reserves future compute access, what it actually holds is an entitlement, not a piece of infrastructure, and that entitlement can be specified, certified, owned, transferred, and redeemed independently of the hardware it describes. The complexity in this paper comes entirely from what it takes to specify that entitlement completely enough to survive a transfer to someone other than the original buyer. Markets have solved versions of this problem before, in leases, spectrum, power, grain, and transmission rights. None of them solved it for an asset this layered, which is why compute needed its own instrument rather than a borrowed one.

This paper has not attempted to finalize that specification, certify a particular issuer, or design the market that would trade these rights. It has tried to establish, with enough rigor that a CEO, a legal team, an engineer, an SRE, and a provider's own infrastructure team could each find their

part of the problem addressed, that the instrument can be designed at all, and what its absence is currently costing the market.

WP-03 takes up what happens once it exists: how it is priced, how liquidity forms, what an exchange for these rights would actually require, and what risks that exchange would need to manage. The instrument comes first. The market and its microstructure are next.
