NEWNAL DATA ARCHITECTURE

Newnal
Personal Data Security
Architecture

Put to use, never pooled.

Stolen, but unusable.

PERSONAL DATA OPEN SQUARE

PERMISSIONED DATA SPHERE

NEWNAL Inc.

CONTENTS

> **“Why do you say a minor fan's data is safe with you? What makes you different from us?” — This document answers that question.**

The short version. Every data service until now has required trusting the company that ran it — the roster, the permissions and the logs all sat inside that company's database, where only the company could see them. Newnal has moved this layer of trust outside the company, onto a public blockchain. Because the records of ownership, permission and access sit on an open ledger that anyone can inspect and verify, the architecture asks for trust in no company at all, Newnal included.

Newnal's systems store no fan's legal name, national ID number, account, phone number or email address. Encryption keys exist only on the fan's own device. So data stolen from outside cannot be used, no one on the inside holds the authority to open it, and Newnal itself cannot make use of a fan's data without that fan's permission. And yet every feature of the service, and every partner's use of data, still works. Section 9 explains why only one blockchain can serve as this layer of trust; Section 10 shows that it has already been proven at national scale. What follows is how the architecture works.

01 What Turns a Leak Into a Disaster

> For a leak to become a bomb, three things have to meet in one place: **content × identity × reach**.

- **Content** — the sensitive material itself (personal stories, feelings, spending, health).
- **Identity** — whose it is (legal name, account, national ID number).
- **Reach** — the means of getting to that person (phone number, email, address).

Conventional services bring all three together because of the account: signing up binds content, identity and reach into a single row from day one. Spending on security does not solve this. Defenses are breached eventually, and they do nothing about the actions of an insider who legitimately holds the authority to open the data — a substantial share of real breaches begin inside.

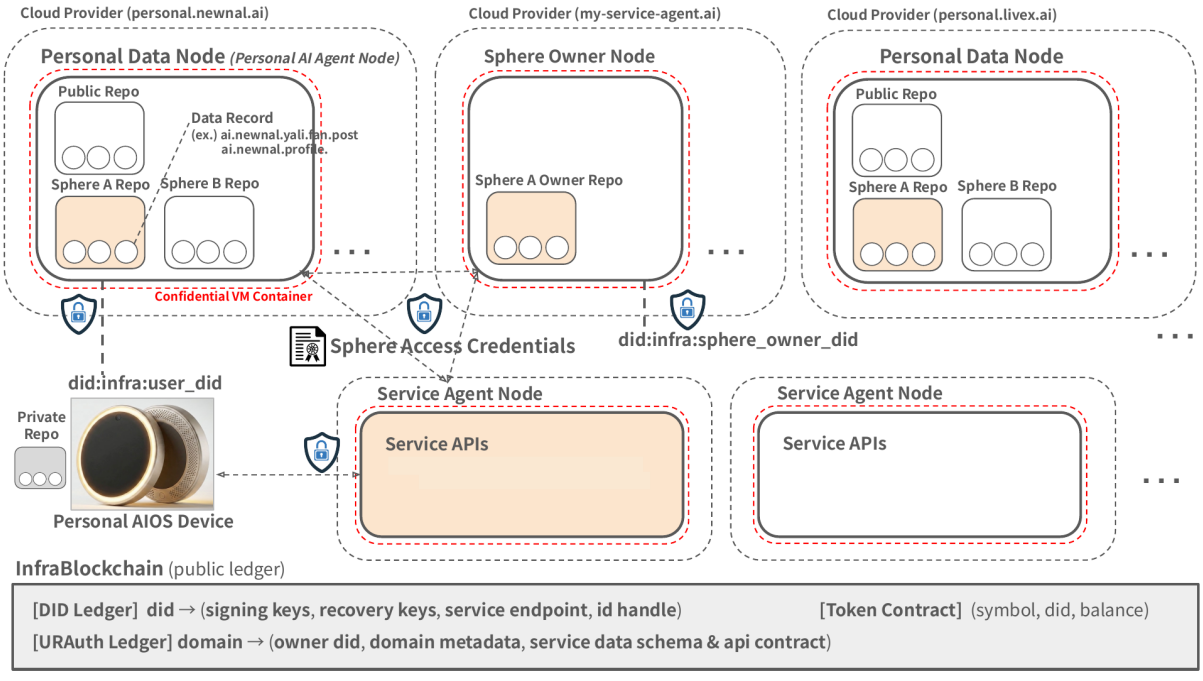
Newnal's systems contain no identity and no reach. Steal from them and what is left in your hands are records whose owner you cannot name and whom you cannot contact. To get anything usable you would have to break into fans' devices one at a time, and one device yields one person, so the economics never work.

02 Same Cloud, Different Filling

Newnal does not use exotic storage. Data sits on servers and in the cloud, as it does for any other service. What differs is the way it is filled.

- **Conventional** — one large database. Every user's identifiers, contact details and records pile up in the same tables, keyed by account.
- **Newnal** — one **personal data node** per fan. Each node holds only that person's records, bound not to a legal name but to a decentralized identifier (DID). On no server do two people's data sit in one table.

- **A node is a logical unit, not a physical place.** It exists on the cloud, any cloud will do, and it can be moved at any time. What creates ownership is not where it is kept but the address (bound to the identity), the key (only on the device) and the title record (on the ledger).
- **Being able to move it is what ownership actually amounts to.** A fan can choose which cloud their node runs on, and changing the address registered on the ledger takes their data somewhere else. Running their own server is possible too. Data lodged somewhere it cannot leave is, however well protected, not the individual's asset but that company's asset. That is why two different cloud addresses are drawn side by side in the architecture diagram.
- **A node has two compartments.** Records that are fine to publish — social data such as posts and comments — go into the **public repository**; records that require per-service permission go into a **sphere repository**. And every record has a schema registered publicly in advance: which service may hold what structure is defined up front, and anyone outside can read it.
- **Encryption keys, and records that never leave the device, exist only on the fan's device.** The node runs inside a **confidential execution environment** with memory encrypted at the hardware level, so neither the cloud operator nor Newnal can see the contents. This is unlike the ordinary cloud operating model, in which administrator privilege amounts to read access.
- **And that fact is verifiable from outside.** A confidential execution environment provides **remote attestation** — meaning a third party can confirm that "the code that was published is the code actually running right now." There is no scenario in which Newnal quietly loads something else.



Same cloud, different filling — a personal data node per fan (a confidential execution environment), the private repository inside the device, the nodes of sphere owners and service agents, and the public ledger (InfraBlockchain) that notarizes all of it.

03 Service Without Accounts — A Substitute for Every Function

Why the service still works without storing identity information: everything the account used to do has a substitute.

What the service needs	Conventional method (identity required)	Newnal's method (no identity required)
Login and identity check	ID and password, phone verification	DID plus device binding — holding the device is the proof
Contact and notification	Phone number, email, app push (open address networks)	DID closed network — only permitted parties connect (Private Phone patent)
Subscription and entitlement	Payment history tied to an account	Credential — verifies only that it is valid
Loss and recovery	Identity check by national ID or phone number	Owner-driven recovery (guardian takes part for minors)
Customer support and refunds	Account lookup	The sales channel's job (partner commerce) — the party already doing it today
Personalization and recommendation	Joined against the user profile on the server	Joined inside the fan's device, with the fan's key

Why contact details are unnecessary — the network itself is different. Conventional contact happens over open address networks. A phone number or an email address is a public address, so anyone who knows it can send to it. That is why spam and voice phishing are possible, and why a leaked contact list becomes a weapon. Contact at Newnal happens on a DID-based closed network — a Private Phone on which calls and messages work over DIDs with no phone number involved, and Newnal's patented technology. This network has no public addresses, and a connection exists only with parties the fan has permitted. The authority over reach is inverted: on an open address network the sender who knows the address decides reach; on this closed network the recipient's permission decides it. Nothing changes if a DID becomes known. Unpermitted sending simply does not exist on the network.

Read the schema instead of believing the promise. In the age of accounts there was nothing to do but take the sentence “we collect only the minimum” on faith. At Newnal, what each service may read and write is registered on the public ledger like a contract. And those schemas have no field for a legal name, national ID number, phone number or account in the first place. The object of trust shifts from a company's sentence to a published blueprint.

The account's last function is notarization: the company's database recorded who was who and what was permitted, and that is why the company needed a roster. In Newnal's architecture the public ledger takes on that role (Section 8). The reason for any company to hold a roster disappears.

04 Personal Data Open Square — Open Use, Divided Custody

> The **Personal Data Open Square** is the central component of Newnal Agent Place. It is an open space in which many companies open services on a single personal-data backend instead of each building their own, and use data only within the scope each fan has permitted.

The Square does three things.

- **Tenancy** — a partner registers the data schema and API contract it will handle on the public ledger, and opens its service. It builds no member database, no identity management and no personal-data storage.
- **Use** — rather than pulling originals into its own servers, it calls the data that stays in the fan's node, within the permitted scope. It gains the value of use without the responsibility of custody.
- **Settlement** — a DID-based ledger executes settlement automatically. Tenants receive service fees; they receive neither the customer's identity nor the original data.

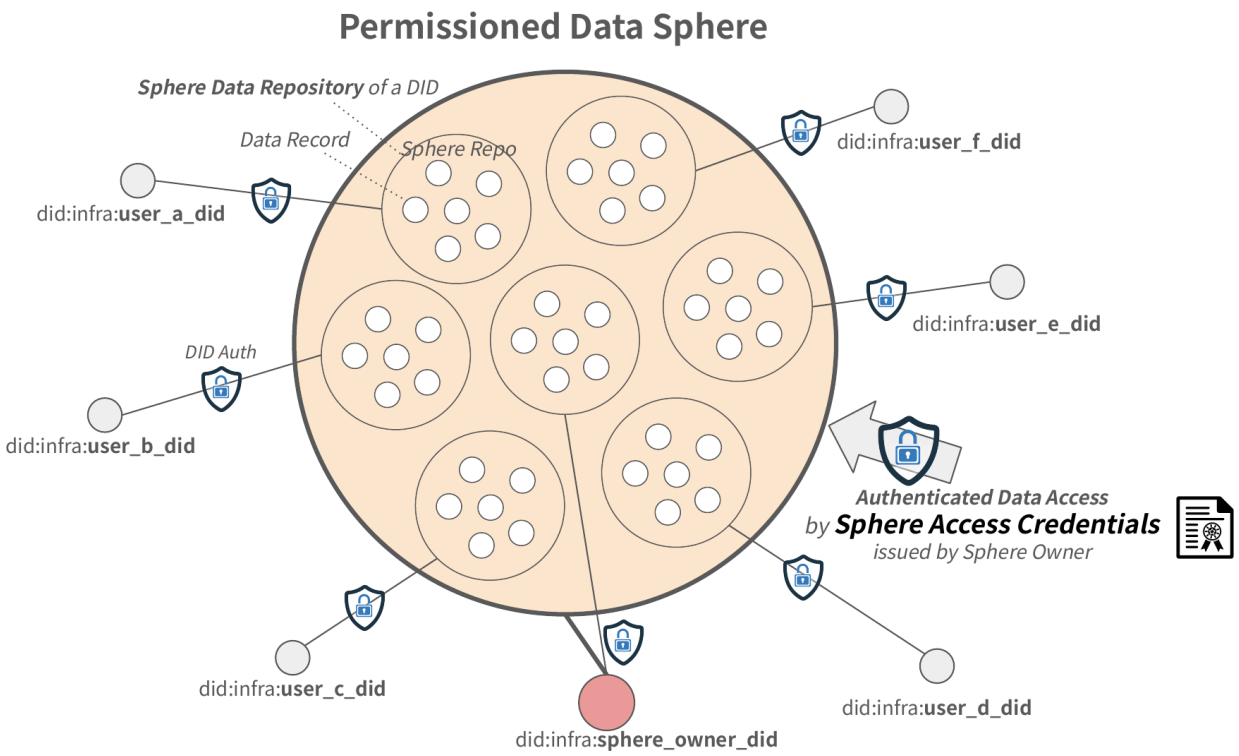
Being open is itself a risk. As services multiply, one person's data across several domains — fan activity, family media, spending, health — accumulates in the same personal data node. If there were no partitions between services, risk that had been scattered would end up concentrated in one place. No single service should be able to reach all of one person's data; it should reach only what has been permitted to it.

The technique inside the Square that solves this is the **Data Sphere**.

05 Data Sphere — Partitions Inside the Square

> A **data sphere** is not a physical space. It is the data space that a single service domain opens; the boundary is drawn by the service domain (the sphere owner), and what fills it is the permission of each individual fan.

- **Inside a sphere there is no shared table.** Participants' data is not gathered into one table; each participant DID has its own independent repository, placed in parallel. Being in the same sphere does not mean putting data into the same room — it means sharing the same conditions of permission.
- **Data does not move.** When a fan grants permission, a compartment for that sphere simply comes into being inside the fan's node. What a sphere consists of is not data but a list of permissions.
- **Change the permission and the boundary is redrawn at once.** Revocation is not a request that waits on a company to act; it is a redefinition of the boundary.
- **A cohort forms without a roster.** Every fan who has drawn permission into a given sphere is a participant in it — you obtain the cohort “fans currently permitting” with no overall database, and even grouped together what is visible is DID-level records, not a roster.
- The space called “all of it,” the sum of the whole, exists nowhere in the system — not even at Newnal, the operator.



Data sphere — records (the small circles) stay in each person's own node while permissions define a single virtual space (the large circle). Access happens only through an access credential issued by the sphere owner, and its limit is the permission each person has drawn.

There are two keys

Access rights to data come in two layers.

- **Sphere access credential** — proves “are you eligible to take part in this space?” Issued by the sphere owner.
- **The data owner's delegation of access** — proves “are you authorized to open this range of my records?” It requires the fan's own DID signature.

Neither one alone opens any record. Even when a service operator is the owner of a sphere, that ownership confers no right to read participants' data. The authority to open the space and the authority to see the contents are separate.

Isolation runs in three directions

- **Owner versus participant** — the owner opens the space and manages eligibility but is not the custodian of the contents. Service operation rights and data reading rights are separate.
- **Participant versus participant** — being in the same sphere does not make another participant's repository visible. Every access must pass DID authentication and the delegation of that repository's owner.
- **Sphere versus sphere** — two different spheres have no path by which to query each other. **The same holds for two spheres belonging to the same person.** As services multiply, the unintended surface of exposure does not grow with them.

So from the user's point of view, the more services there are, the more data accumulates in their own node rather than scattering across companies — and it accumulates in rooms that stay separate. The sum of personalization builds up on the user's side, and on no company's side.

06 Agents and Payments — Two Separations

A tenant of Agent Place implements its service on a **service agent node**. Instead of pulling original data into its own database, it calls the fan's node through the API contract published on the ledger. That call is valid only if it passes both the access credential issued by the sphere owner and the fan's own delegation. What the tenant receives is the opportunity to serve and the settlement; what it does not receive is identity, originals or contact details. Newnal bears the burden of custody, partners do the using, and ownership stays with the fan.

Payment identity stays where it already lives today — with partner commerce and the payment gateway. Payment and service are joined only by a credential stating that this device's subscription is valid. For a minor fan the payer is the guardian, so “a minor's payment information” never comes into existence in the system at all.

07 Every Access Is Logged and Reported

- No activity can take place without permission — not one read, not one search.
- Every access is logged and reported to the owner of the data.
- Records of ownership, permission and access sit on the ledger, where an operator cannot alter them after the fact, and they are verified by external audit.
- Insider leaks under the conventional architecture were an abuse of privilege. In this architecture the privilege to open a fan's data does not exist in the first place.

08 Blockchain — The Trust Layer, Moved Out of the Company

Until now, a service's trust lived inside the operating company's database: the company recorded who the users were, what was permitted and what had happened, and only the company could see it. Users had no option other than to trust the company. Newnal has moved this layer onto a public ledger that anyone can inspect and verify.

Four things go onto the ledger, and three ledgers divide the work between them.

- **The title of identity** — the DID, the signing and recovery keys, the node's location. → **DID Ledger**
- **The record of permission** — who opened what, how far, and when they closed it.
- **The publication of contract** — the range of data each service may read and write (the data schema and API contract), and the owner of that domain. → **URAuth Ledger**
- **The book of settlement** — automatic settlement on a DID basis. → **Token Contract**

None of these is an internal Newnal database, and once a record is on the ledger no one can alter it at will.

Original data does not go onto the ledger. Stories and records stay in the individual's node; keys stay on the device. The blockchain's role is notarization, not storage — a ledger outside the company takes over the work that the account, which is to say the company's registry office, used to do. And so no company needs a roster.

At the center of this layer stands the owner of the data. Whatever physical space the data occupies, no company — Newnal included — can make use of it without that individual's permission, and anyone can confirm as much on the ledger.

09 Why No Other Blockchain Will Do

To serve as a layer of trust, a chain has conditions to meet: anyone must be able to verify it (it must be public), costs must be stable, and it must carry the load of a real service. Existing public blockchains are coupled to cryptocurrency issuance and fail these conditions. Fees ride the coin price, speculation and regulatory risk come along with it, and the processing architecture cannot carry real service volume. They were unusable for enterprises and governments, and in fact no public chain has ever carried an everyday service used daily by tens of millions of people.

InfraBlockchain is a public blockchain that operates without issuing cryptocurrency. Its consensus algorithm is patented in the United States, internationally, and in Korea.

10 Proof at Scale — 43 Million Users on COOV

COOV, the vaccine pass, is proof that this architecture ran at national scale.

- 43 million people in Korea used it every day, more than 30 million verifications ran on peak days, and more than 260 million DIDs were issued and put to use.
- What it handled was the most sensitive data there is — medical records, legal names, national ID numbers, passport numbers.
- Originals stayed on the servers of the issuing authorities (the Korea Disease Control and Prevention Agency, the Ministry of Foreign Affairs and others), individuals' devices held only credentials, and verification did not query a server. Nowhere was there a single place that yields everything if you break it.

No other blockchain data architecture in the world has a real-service proof point at this scale.

11 Alignment With Regulation

The two yardsticks of privacy regulation worldwide are identifiability and the act of collection. This architecture has neither.

What regulation requires	Conventional response	Newnal architecture
Data minimization (GDPR Art. 5)	Efforts to reduce the fields collected	Identifiers and contact details are never collected to begin with
Withdrawal as easy as consent (Art. 7)	Consent management systems and compliance procedures	Permitting and revoking are the same action

What regulation requires	Conventional response	Newnal architecture
Privacy by design (Art. 25)	Adding review steps to the development process	The architecture is itself the implementation of this article
Right to erasure (Art. 17)	Find the scattered copies, delete them and prove it	The company never held a copy
Right to portability (Art. 20)	Build a separate export feature	Data is bound to the person from the start and moves node and all
Right of access (Art. 15)	Receive requests and reply to them	Every access is logged and reported
Breach notification (Arts. 33, 34)	A 72-hour response program	Points toward there being no “identified victim” to speak of
Limits on international transfer (Ch. V)	Transfer assessments and standard contractual clauses	Node locations are placed per jurisdiction — no event in which a whole database crosses a border
Protection of children (Art. 8; US COPPA)	Age verification and guardian consent procedures	Children's identifiers and contact details are not collected, and the guardian is the subject of payment and consent

The core targets of US state law — the sale and sharing of personal information, data brokerage, unauthorized targeted advertising and the collection of children's information under California's CCPA/CPRA and its counterparts — are acts that cannot occur in this architecture. Detailed application by jurisdiction will be settled together with legal review.

12 Why This Is Hard to Copy

- **Big Tech** — their revenue comes from data asymmetry: targeted advertising and CRM. Adopting this architecture is not a feature addition but a replacement of the business model.
- **Device makers** — they can build the hardware, but they cannot give up the account. Accounts, push and app store payments are the spine of the business, and account-free service holds together only on a proprietary operating system and a dedicated device.
- **A record of trust** — a record of handling the most sensitive data at national scale without incident was made under conditions that a pandemic created and that no one can reproduce.
- **Startups** — the specific mechanisms are protected by some 90 patents registered or filed in the United States, Korea, Europe and internationally.

A Anticipated Questions (Q&A)

Q1. Why do you say it is safe for you to handle a minor fan's data?

Because Newnal's systems hold no information that identifies who that fan is, and no means of contacting them, so a leak yields nothing usable. Of the conditions that turn a leak into a disaster — content × identity × reach — identity and reach are structurally absent. The basis for safety is not a security rating but the fact that stolen data cannot become the raw material of a crime.

Q2. We invest heavily in security too. What is different?

The direction. Conventional security is the work of not being breached; defenses are breached eventually, and they do nothing about the actions of an insider who holds the authority to open the data. Newnal's is an architecture in which a breach does not become an incident, and in which no insider with that authority exists.

Q3. Separating user data from payment data — can't we do that ourselves?

Anyone can separate. What differs is the join key. Conventional separation is a matter of policy that leaves a common key (an account, a phone number) in both tables, so the two can always be rejoined. At Newnal the join key does not exist in the system at all, and the only place where content and identity meet is the fan's own device. It is the difference between a promise not to join them and a structure that cannot.

Q4. You run subscriptions and commerce — how can there be no identity information?

Payment identity does exist; it simply stays where it already does today, with partner commerce and the payment gateway, and Newnal does not collect it anew. Payment and service are joined only by a credential — “this device's subscription is valid” — and that single fact of validity is all that crosses into the Square. For a minor fan the payer is the legal guardian, which puts one more layer of separation in place.

Q5. If a fan writes their own name or school into a story, isn't that identity information?

We cannot strip out what fans write themselves. So Newnal's precise claim is not that “identification is impossible” but that “the join keys to the outside world — legal identity, means of contact, financial information — are structurally absent.” Even if a story contains a name, no phone number, address or account is attached to that record. There is no link to make use of for a crime.

Q6. Isn't there a great deal of research showing that de-identified data can be re-identified once combined with other data?

Those studies assume two things: outside data to combine with and a common field to join on, and a whole dataset to serve as the object of combination. Newnal's architecture has neither. No join key is attached to the records, so there is nothing to link them to outside data; and because data exists only within spheres scoped by permission, there is no “whole” to obtain in one piece and attempt to combine. Even if someone succeeded in guessing whose a record is, the means of reaching that person still would not exist.

Q7. If the join happens on the fan's device, can't you just hack the device?

Hacking a server and hacking a device have different economics. Break a server once and tens of millions of people come out; break a device and one person comes out — and only by way of physical access or an individually targeted attack. Effort against reward does not add up, so it never becomes an industry. The keys and the private repository inside the device are protected at the hardware level.

Q8. What if a Newnal insider looks? What if Newnal is hacked?

They cannot open it, because the keys are not at Newnal. Insider leaks in the conventional architecture were the misuse of privilege by a person who held it; in this architecture no one inside Newnal holds the privilege to open a fan's data. Originals are encrypted, the keys are on the fan's device, and nodes run in a confidential execution environment where even the cloud operator cannot see the contents. Every access is logged and reported to the owner, and this is verified by external audit. If Newnal were hacked wholesale the result would be the same — stolen, but unusable.

Q9. If a fan loses their device, do they lose their data?

No. Originals are encrypted and stored redundantly, and keys are brought back through an owner-driven recovery procedure, with a guardian taking part for minors. The subject of recovery is always the owner, and by design there is no back door through which Newnal could open it on their behalf — a back door, even one meant for fans, ends up being the attacker's door.

Q10. What if law enforcement demands the data?

Newnal cannot hand over what it does not hold. What it can produce is encrypted originals and access logs, and reading them is a matter for the owner of the data to answer. This is the same structure that end-to-end encrypted services have established internationally, and that fact is exactly what protects both the fan and the agency.

Q11. What if Apple or Google builds the same thing?

It is not that they cannot build it; they cannot adopt it. Their businesses stand on accounts and identification — advertising, CRM, the app store — and the moment they adopt this architecture they negate their own business premise. Had it been a matter of technology, it would already exist.

Q12. Haven't there been attempts at personal data stores and data sovereignty before?

There have — and the point at which those attempts stalled is Newnal's starting point. They succeeded in moving custody to the individual but never created use. Individuals who got their data back had nothing to do with it, and companies had no way to use data that was scattered, so they never came. Newnal's architecture designs protection and use as one thing: the Open Square makes the place for use, spheres make usable cohorts without aggregation, and agents run services and settlement on top of them.

Q13. If protection is this strong, can we actually use the data?

That is the purpose of the architecture — protection is not the opposite of use but its precondition. Partners search the scope a fan has permitted in natural language, propose agent services and generate personalized content. They do not take the originals away; they work on top of permission, and so they gain the value without the liability of custody. Where protection is weak, the data worth using never arrives in the first place.

Q14. How does this settle legally?

In Newnal's architecture the direction is that the act of “collection” does not occur at all: data stays owned by the individual, does not become a company asset, and the records processed are unidentifiable from the company's point of view. Alignment article by article is in Section 11. The details will be settled together with legal review by jurisdiction.

Q15. So who takes responsibility if something goes wrong?

The answer has two layers. First, in this architecture the conventional kind of breach — sensitive information about identified individuals leaking together with the means of reaching them — does not come about. Second, the operation of the network and the maintenance of the architecture are nonetheless entirely Newnal's responsibility, and the discharge of that responsibility is proven by audit rather than promised. Partners bear no data liability, because they neither own the data nor hold it.

Q16. If we are the owner of a sphere, can't we see all the data of the fans who joined it?

No. What an owner holds is the authority to open the space and manage eligibility, not the authority to open the contents. Opening any record takes two keys: the sphere access credential the owner issues (“are you eligible to take part in this space?”) and the data owner's own delegation (“are you authorized to open this range of my records?”). The owner holds only the first. That operating authority and reading authority are separate is one of the core points of this architecture.

Q17. As services multiply, doesn't all of one person's data end up gathered in one node? Isn't breaking that one node enough?

It is true that data accumulates per individual, and that is why this architecture divides the inside of it. Different spheres have no path by which to query each other, and neither do two spheres belonging to the same person. What one service has been permitted is only the defined schema inside its own sphere, so the surface of exposure does not grow as services multiply. The node itself runs in a confidential execution environment where neither the cloud operator nor Newnal can see plaintext — and above all, one node is one person, which is different economics from one server that yields tens of millions.

Q18. How would we know if Newnal quietly changed the node's code?

That is what remote attestation is for. A confidential execution environment produces evidence, confirmable from outside, that the code running inside it right now is exactly the code that was published. The party doing the verifying does not have to be Newnal. This is why the claims in this document are objects of audit rather than objects of belief.

> This architecture does not rest on “trust Newnal.” The records of ownership, permission and access can be verified by anyone; the information needed for abuse does not exist in the system; and Newnal, too, cannot make use of a fan's data without that fan's permission.

B Technical Summary

The components that appear in the diagrams in the body are collected here in a single table. It is a summary for those conducting technical review, and is not required in order to follow the claims made in the body.

Component	What it does	What it does not do
Personal Data Node	The primary location of personal data. It belongs to the user's DID, and creating or reading a record requires an access grant carrying the owner's signature. It is divided into a public repository and sphere repositories, and inside it data is encrypted and stored redundantly. The cloud provider is selectable and the node can be moved at any time.	Does not hold personally identifying information. Does not put another person's data in the same table.
Confidential VM Container	An execution environment whose memory is encrypted and sealed at the hardware level. Remote attestation lets an outside party verify that the published code is the code running.	Shows plaintext to neither the cloud provider nor Newnal's own operators.
Personal AIOS Device	Holds the private repository that never leaves the device, and the root secret key carrying final ownership of the DID. Final authority over personal data originates here.	Never exports the root secret key that carries final ownership of the DID.
Sphere Owner Node	Defines the boundary and the participation eligibility of a service domain's data space — the sphere — and issues access credentials.	Holds no authority to read participants' data.
Service Agent Node	Where a tenant's data service is implemented. Instead of pulling originals into its own database, it calls through the published API contract.	Cannot reach spheres it has not been permitted. Does not receive customer identity or original data.
InfraBlockchain public ledger	The DID ledger (signing keys, recovery keys, node addresses), the URAuth ledger (domain – owner DID – data schema and API contract), and the token contract (the DID-based settlement book).	Does not hold original data. Is not Newnal's internal database, and no one can alter it at will.

One backend, many services

Services of quite different natures operate on the same personal-data backend without ever touching each other's data.

- **YALI** — an artist AI device service built on fans' personal data. A fan's stories, feelings and purchase records stay in the YALI sphere inside that fan's node. The agency searches the permitted scope and generates personalized artist content, without taking originals to its own servers.
- **ILLI** — a content-sharing and appless AI agent device service for seniors and their families: sharing family media, recommending products and booking taxis without going through apps, and so on. Several family members' DIDs share a single sphere while each keeps a separate repository.

What the two services share is the same personal data node, the same DID, the same ledger. What they do not share is data.

From the tenant's side — a platform where you build no backend

Register the data schema and API contract on the URAuth ledger, open your sphere, and tenancy is complete. You build no member database, manage no identities and store no personal information, so you are not a subject of breach liability either. Settlement is executed automatically by a DID-based ledger.

From the user's side there is no app, no account and no login — the device and the DID take their place. The relationship is inverted: it is the service that logs in to the user's node and sphere (Reverse Login).