

PROVISIONAL APPLICATION FOR UNITED STATES PATENT

UNDER 35 U.S.C. § 111(b) AND 37 C.F.R. § 1.53(c)

TITLE OF THE INVENTION

**LOCAL-FIRST DISTRIBUTED DATA STORAGE SYSTEM WITH CLIENT-SIDE
ENCRYPTED FRAGMENT DISPERSAL ACROSS PEER-CONTRIBUTED STORAGE
NODES**

Inventor: Benjamin Andrew Carmitchel, Edwardsville, Illinois, United States

Applicant: Datarecovery.com, Inc., an Illinois corporation, Edwardsville, Illinois

September 2026

FIELD

[0001] This disclosure relates to computer data storage, and more particularly to distributed storage systems in which user data is divided into encrypted fragments and dispersed with redundancy across a network of storage nodes contributed by participants, under the direction of a coordination service that stores neither file content nor decryption keys.

BACKGROUND

[0002] Conventional consumer and business file storage services maintain user files in centralized infrastructure operated by the service provider. Centralization concentrates both the stored content and the ability to read it: the provider typically holds encryption keys, an operator or intruder with sufficient privilege can access plaintext, and a single breach, outage, or insider action can affect every account at once. Uploading and sharing are serialized through the provider's infrastructure, and a user's access to the user's own data depends on the continued operation and goodwill of the provider.

[0003] Peer-to-peer and decentralized storage networks have been proposed in which data is erasure coded and dispersed across independently operated nodes. Known systems of this kind commonly rely on blockchain ledgers or cryptocurrency token economics for coordination and compensation, treat the network copy as the primary copy rather than keeping the user's own device authoritative, are oriented to developers rather than to consumer file synchronization and sharing, or lack an immediate sharing path while data is still being dispersed. There remains a need for a storage architecture that keeps the complete plaintext copy on hardware the owner controls, disperses only ciphertext, requires no distributed ledger, and makes shared data available to a recipient immediately upon sharing.

SUMMARY

[0004] Embodiments described herein provide a distributed storage system, referred to for convenience as a protective cloud shell, comprising client software executing on user devices, a coordination server, and a plurality of peer storage nodes contributed by participants who install the client software on personal computers, mobile devices, or servers. A complete, readable copy of each user's data remains on that user's own device or devices. The client divides files into chunks, deduplicates the chunks by content identifier, encrypts each resulting fragment with keys derived from a master secret held only by the user, applies erasure coding so that any k of n coded fragments suffice to reconstruct the source data, and transmits the coded fragments to peer storage nodes selected by the coordination server under diversity constraints. The coordination

server maintains a registry of nodes, health information, placement records, and client-encrypted manifests, and stores no fragment payloads and no keys.

[0005] Further aspects include immediate sharing in which the sharing user's own device serves file content directly to a recipient while dispersal of fragments to the shell proceeds in the background; share links divided into a locator portion that is resolved by the coordination server and a capability portion carrying key material that is never transmitted to the coordination server; continuous audit of fragment possession by challenge and proof, with automatic reconstruction and re-dispersal of fragments lost to node departure; contribution accounting by which participants earn storage capacity in the shell in proportion to capacity they contribute, without any blockchain or token; append-only versioned storage with an immutable retention window; recovery of account keys by recovery kit and by threshold secret sharing among designated contacts; restriction of placement to organization-owned nodes to form private shells subject to residency policy; and opportunistic participation of mobile devices.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] FIG. 1 is a block diagram of a storage system according to some embodiments, showing an owner client device, a coordination server, peer storage nodes, a recipient device, and a relay.

[0007] FIG. 2 is a flow diagram of client-side processing of a changed file according to some embodiments, from change detection through chunking, deduplication, encryption, erasure coding, placement, and distribution.

[0008] FIG. 3 is a diagram of link-based sharing according to some embodiments, showing a link having a locator portion and a capability portion, resolution of the locator by the coordination server, and parallel retrieval by a recipient.

[0009] FIG. 4 is a flow diagram of fragment audit and self-healing according to some embodiments.

DETAILED DESCRIPTION

[0010] Reference will now be made to exemplary embodiments. The invention is not limited to the embodiments described; features described with respect to one embodiment may be combined with features of any other embodiment, steps may be reordered, combined, or omitted where logic permits, and stated algorithms, parameters, protocols, and numeric values are examples rather than limitations.

System overview

[0011] FIG. 1 shows a storage system comprising an owner client device 110, a coordination server 120, peer storage nodes 130a through 130n (collectively, the shell), a recipient device 140, and optionally a relay 150. The owner client device 110 retains a complete local copy 112 of the user's files in plaintext or in whatever form the user keeps them, executes a client module 114 that performs the processing described herein, and in some embodiments also acts as a storage node 116 holding encrypted fragments belonging to other users. Because the complete copy remains local, the user's ordinary read and write operations proceed at local storage speed and function without network connectivity; the shell functions as backup, as a distribution mechanism, and as the means of access from other devices, rather than as the authoritative copy.

[0012] The coordination server 120 maintains a node registry 122 recording the identity, addressing information, capacity, and health of each storage node; a placement map 124 recording which coded fragments are assigned to which nodes; and encrypted manifests 126 described below. The coordination server stores no fragment payloads and no decryption keys. In some embodiments the coordination server is implemented as a replicated or clustered service for availability; the term server includes such replicated implementations. Compromise of the coordination server therefore yields routing metadata and ciphertext manifests, not file content.

[0013] Each peer storage node 130 is a device on which a participant has installed the client software and allocated a configurable quantity of local storage, for example a portion of an underutilized disk, for the storage of fragments belonging to other users. A storage node stores fragments in a fragment store 132 as opaque ciphertext objects identified by fragment identifier. A node cannot determine the owner, filename, file boundaries, or content of any fragment it holds. Nodes may be personal computers, servers, network-attached storage devices, or mobile devices, and in some embodiments the operator of the system additionally provides always-on anchor nodes that participate in the same protocol and provide a guaranteed availability floor.

Client-side processing pipeline

[0014] FIG. 2 shows client-side processing according to some embodiments. At 210 the client module detects a new or changed file, for example by file system watching, periodic scan, or explicit user action. At 220 the file is divided into chunks. In some embodiments chunk boundaries are content defined, for example by a rolling hash over a sliding window with boundary selection when the rolling hash satisfies a criterion, so that an edit localized in a file changes only the chunks overlapping the edit. Exemplary average chunk sizes are between 64 kilobytes and 8 megabytes, for example approximately 1 megabyte, so that a terabyte of stored

data corresponds on the order of one million fragments; fixed-size chunking is an alternative embodiment. A typical user account therefore corresponds to hundreds of thousands to millions of dispersed fragments.

[0015] At 230 each chunk is identified by a collision-resistant content identifier, for example a cryptographic hash such as SHA-256 or BLAKE3, computed in some embodiments over the plaintext chunk with a per-account keyed hash so that content identifiers do not leak information outside the account. A chunk whose identifier is already recorded in the account's index is not processed again; only new chunks proceed. Deduplication is per account in some embodiments; in an alternative embodiment employing convergent encryption, deduplication extends across accounts at a described cost in confidentiality, and the system permits the operator or user to select between the modes.

[0016] At 240 each new chunk is encrypted on the client device using an authenticated encryption algorithm, for example AES-256-GCM or XChaCha20-Poly1305, under a per-chunk key. Per-chunk keys are derived (245) from a master secret held only by the user, for example by a key derivation function such as HKDF, or are generated randomly and wrapped by keys derived from the master secret. The master secret may itself be derived from a user passphrase by a memory-hard function such as Argon2, generated by the client and stored in a platform keystore, or both. Neither the master secret, nor any derived or wrapped key, nor any plaintext is transmitted to the coordination server or to any storage node. The system is in this sense zero knowledge with respect to both the coordination server and the shell.

[0017] At 250 encrypted chunks are grouped and erasure coded. For each group, n coded fragments are produced such that any k of them reconstruct the group, for example by Reed-Solomon coding, with fountain or other rateless codes as alternatives. Exemplary parameters include k of 10 and n of 30, or k of 20 and n of 60; the ratio n over k is selected according to the expected availability of the node population, and in some embodiments differs by redundancy class, so that a user or policy may assign higher redundancy to designated data. Because peer nodes exhibit churn, the system deliberately provisions redundancy substantially higher than datacenter systems, which is economical because the underlying medium is contributed.

[0018] At 260 the coordination server assigns each coded fragment to a storage node under placement constraints (265) that promote failure independence, for example: no two fragments of a group on the same node; caps on the fraction of a group held by any single participant, network, autonomous system, or geographic region; preference for nodes with distinct operating environments; and compliance constraints such as data residency described further below.

Placement is computed by the coordination server in some embodiments, by the client from a node list in others, or cooperatively. At 270 the client transmits each fragment to its assigned nodes directly where connectivity permits, and otherwise through relays as described below. Fragment transmission is encrypted in transit in addition to the fragment-level encryption.

[0019] A manifest is maintained for each file, recording the file's chunk sequence, content identifiers, group structure, coding parameters, fragment identifiers, and version lineage, together with wrapped key material where the embodiment uses wrapped keys. The manifest is encrypted on the client under the user's keys and stored, in encrypted form 126, with the coordination server, and in some embodiments is additionally dispersed into the shell like file data. The coordination server can therefore locate fragments by fragment identifier without ability to associate them with filenames, directory structure, or content.

Audit, healing, and durability

[0020] FIG. 4 shows audit and healing according to some embodiments. Storage nodes send heartbeats to the coordination server, and the coordination server issues possession challenges (410), for example a request to return a hash over a randomly selected byte range of an identified fragment, optionally salted, so that a node cannot answer without actually holding the fragment. A node that is unresponsive past a grace period, or whose proof fails (420), is marked lost. When the count of live fragments for any group falls below a repair threshold above k (430), the coordination server directs reconstruction: any k surviving fragments of the group are retrieved, the group is reconstructed, missing fragments are regenerated (440), and the regenerated fragments are placed on new nodes satisfying the diversity constraints (450). Reconstruction is performed in some embodiments by the owner's client when online, by designated repair workers, or by anchor nodes, in each case without access to plaintext where repair operates on encrypted groups. Repair is continuous and automatic, so that durability is maintained indefinitely without user action.

[0021] Storage in the shell is append only in some embodiments. A new version of a file produces new chunks and a new manifest version; prior manifest versions are retained according to a retention policy. During an immutable retention window, for example seven to ninety days, the coordination server declines instructions to purge version history, including instructions authenticated as the user, so that malware or an attacker in control of a user device cannot destroy the shell history of the user's data within the window. Point-in-time restoration is performed by selecting a manifest version and retrieving its fragments.

Retrieval and synchronization

[0022] To retrieve data, a client obtains the encrypted manifest, decrypts it locally, requests any k fragments of each needed group from the nodes recorded in the placement map, verifies fragment integrity by content identifier, reconstructs the ciphertext, and decrypts locally. Requests are issued to many nodes in parallel, with selection preferring nodes of low measured latency and high measured throughput, so that aggregate retrieval bandwidth is the sum over many peers rather than the allotment of a single server connection. Devices belonging to the same account discover one another and, when on the same local network, synchronize directly at local network speed without transiting the internet. A device that is restored or newly added retrieves the account's manifests and rebuilds selected content from the shell, which is the designed recovery path for device loss, theft, or failure.

Link-based sharing and serve-while-syncing

[0023] FIG. 3 shows sharing according to some embodiments. A sharing user directs the client 310 to share a file, folder, or version, and the client generates a link 320 comprising a locator portion 322 and a capability portion 324. The locator portion identifies the share to the coordination server. The capability portion carries the key material needed to decrypt the shared content, or material from which it is derived, and is carried in a part of the link that is not transmitted to the coordination server when the link is opened, for example a URI fragment processed only by the recipient's browser or client, or is conveyed out of band. The coordination server therefore resolves locators without ever possessing decryption capability, and disclosure of the coordination server's records does not disclose shared content.

[0024] A recipient 330 opening the link, in a web browser without any installation or account, or in a native client, obtains from the coordination server the share's manifest and fragment locations by presenting the locator only, retrieves fragments in parallel 335 from storage nodes and, while the sharing user's device is online, directly from that device, reassembles and decrypts locally 340, in browser embodiments using in-browser cryptography. Because the sharing user's device holds the complete copy, sharing is effective immediately upon link creation: the recipient begins receiving from the source device at once, while dispersal of fragments into the shell proceeds concurrently in the background, and retrieval shifts to the shell as fragments propagate. There is no distinct upload phase that must complete before sharing is effective, and the source device serves the shared data while it is still synchronizing.

[0025] Shares may be limited by expiration time, by number of retrievals, or by enumerated recipients; may be revoked, after which the coordination server ceases to resolve the locator, it being understood that revocation cannot recall content already retrieved; and may be re-keyed, in

which shared content is re-encrypted under new keys and a previously issued capability ceases to decrypt newly placed fragments.

Connectivity and relays

[0026] Storage nodes and clients commonly sit behind network address translation, including carrier-grade translation, without stable public addresses. Each node maintains a persistent outbound connection to the coordination server, over which control traffic flows and through which inbound requests are brokered. Direct node-to-node paths are established where possible by connection reversal or hole punching coordinated by the server; where no direct path can be established, traffic is carried by a relay 150. A relay carries only fragment ciphertext and message envelopes and is architecturally incapable of reading content. The outbound-dial pattern requires no inbound firewall configuration by participants and tolerates changing public addresses, and no endpoint of the system depends on a participant possessing a stable address.

Contribution accounting and tiers

[0027] The coordination server meters, for each participant, storage contributed to the shell, weighted in some embodiments by measured availability and by successfully answered possession challenges, and storage consumed in the shell by that participant's dispersed fragments. A participant's entitlement to consume is a function of contribution, for example a ratio by which a participant earns one unit of consumable capacity for a configured multiple of contributed capacity, the multiple being at least the coding expansion so that the network remains over-provisioned by construction. Accounting is performed by the coordination server on metered quantities; no blockchain, distributed ledger, token, or cryptocurrency is required, distinguishing the system from ledger-coordinated storage networks. Participants may alternatively or additionally purchase capacity, and paid service in some embodiments pins fragments to anchor nodes, extends version retention, or raises redundancy class.

Mobile device participation

[0028] Mobile devices participate as both clients and, in some embodiments, storage nodes, under conditions that respect the device's constraints: contribution of storage and answering of retrieval requests occur opportunistically, for example only while the device is charging and connected to an unmetered network, and within a configured storage allotment. Placement weighting accounts for the lower expected availability of mobile nodes. Capture-to-shell operation disperses photographs and other files from the mobile device as they are created, subject to the same pipeline of chunking, deduplication, client-side encryption, and coding.

Private shells and placement policy

[0029] Placement constraints are policy driven. An organization may restrict placement of its data to nodes it owns or designates, so that the organization's own workstations, servers, and offices form a private shell providing self-healing, client-side-encrypted storage on existing hardware; may require that fragments remain within a jurisdiction or on premises for data residency; or may combine restricted and general placement by class of data. Hybrid embodiments place one subset of a group's fragments on organization nodes and another subset on the general shell or on anchor nodes, with thresholds selected so that reconstruction requires participation of the organization's subset, whereby content cannot be reconstructed from the general shell alone even with keys.

Key custody and recovery

[0030] Because no key ever leaves the user's control, the system provides engineered recovery rather than provider reset. A recovery kit, generated at enrollment, contains the master secret or material sufficient to reconstruct it, for printing or offline storage. In addition, some embodiments split recovery material by threshold secret sharing, for example Shamir's scheme, among M designated contacts or devices such that any quorum m of M shares reconstructs recovery material and fewer than m shares disclose nothing; the shares themselves may be held as fragments in the shell, encrypted to each contact. Account authentication to the coordination server is by public-key credential in some embodiments, and is distinct from content keys, so that credential compromise alone does not disclose content.

Security properties

[0031] The architecture is arranged so that no single compromise discloses user content. Compromise of the coordination server yields node metadata and encrypted manifests. Compromise of any storage node, or of many storage nodes, yields ciphertext fragments that cannot be attributed to owners or files without the manifests and cannot be decrypted without user keys. Interception of a share link's server-bound traffic yields a locator without capability. Reconstruction of any file requires simultaneously (a) identification of the file's fragments via a decrypted manifest, (b) possession of at least k fragments of each group from nodes selected for diversity, and (c) the user's keys, which exist only on user hardware. The design goal, and the principal advantage over centralized storage, is the elimination of any single repository whose breach discloses content.

Implementation notes

[0032] Client software may be implemented for desktop, server, and mobile operating systems and in web browsers; the coordination server as one or more services on commodity or cloud infrastructure; and communications over standard transport protocols with transport encryption. The functional divisions described are exemplary: functions of the coordination server may be split among services or partly delegated to clients, and modules described separately may be combined. The terms file, folder, and account are used for clarity and do not limit the stored objects. Nothing in this description requires any particular hash, cipher, code, chunk size, coding parameter, threshold, interval, or ratio; stated values are examples.

EXEMPLARY CLAIMS

Claims are not required in a provisional application. The following exemplary claims are included to illustrate, without limitation, subject matter regarded as the invention.

1. A data storage system comprising: a client module executing on a user device that retains a complete local copy of user data; a plurality of storage nodes each comprising storage contributed by a participant device on which the client software is installed; and a coordination server that maintains a registry of the storage nodes and placement records and that stores no fragment payloads and no decryption keys; wherein the client module divides user data into chunks, encrypts fragments derived from the chunks using keys held only by the user, applies erasure coding such that any k of n coded fragments reconstruct the data, and transmits the coded fragments to storage nodes selected under diversity constraints.
2. The system of claim 1, wherein the coordination server stores manifests that are encrypted on the user device and that the coordination server cannot decrypt.
3. The system of claim 1, wherein, upon sharing of data by the user, the user device serves the shared data directly to a recipient while transmission of coded fragments of the shared data to storage nodes proceeds concurrently, whereby the shared data is retrievable by the recipient without completion of any upload phase.
4. The system of claim 1, wherein a share link comprises a locator portion resolved by the coordination server and a capability portion carrying decryption capability, the capability portion not being transmitted to the coordination server.
5. The system of claim 1, wherein the coordination server issues possession challenges to storage nodes and, upon loss of fragments below a repair threshold, causes regeneration of lost fragments from any k surviving fragments and placement of regenerated fragments on other storage nodes.
6. The system of claim 1, wherein the coordination server meters storage contributed by each participant and storage consumed by that participant and grants consumable capacity as a function of contribution, without use of a distributed ledger or token.
7. The system of claim 1, wherein version history of stored data is retained in append-only form and the coordination server declines destruction of version history during an immutable retention window notwithstanding authenticated instruction.

8. The system of claim 1, wherein placement of fragments is restricted by policy to storage nodes designated by an organization, or to a jurisdiction, whereby the designated nodes form a private shell.
9. The system of claim 8, wherein fragments of a group are divided between designated nodes and other nodes with thresholds selected such that reconstruction requires fragments held by the designated nodes.
10. The system of claim 1, wherein mobile devices participate as storage nodes opportunistically subject to charging and network conditions, and placement weighting accounts for expected availability of mobile nodes.
11. The system of claim 1, wherein storage nodes maintain outbound connections to the coordination server from behind network address translation, and node-to-node transfers that cannot be established directly are carried by a relay that carries ciphertext only.
12. A method of storing data comprising: retaining a complete copy of the data on a device of the data's owner; dividing the data into chunks and deduplicating the chunks by content identifier; encrypting fragments derived from the chunks on the owner's device with keys held only by the owner; erasure coding the encrypted fragments such that any k of n coded fragments reconstruct the data; dispersing the coded fragments to peer storage nodes contributed by other users under diversity constraints recorded by a coordination server that stores neither fragments nor keys; auditing possession of dispersed fragments by challenge; and regenerating and re-dispersing fragments lost to node departure.

ABSTRACT

A distributed storage system keeps the complete copy of a user's data on the user's own device and maintains, across a network of storage nodes contributed by participants, millions of encrypted fragments produced by client-side chunking, deduplication, encryption under user-held keys, and k-of-n erasure coding. A coordination server tracks nodes and placement but stores no fragments and no keys. Fragments are audited by possession challenge and automatically regenerated upon node loss. Sharing is by link, with decryption capability carried in a portion never sent to the server, and shared data is served by the owner's device immediately while dispersal proceeds in the background. Capacity is earned by contribution accounting without a ledger or token. Placement policy supports diversity, residency, and organization-restricted private shells.

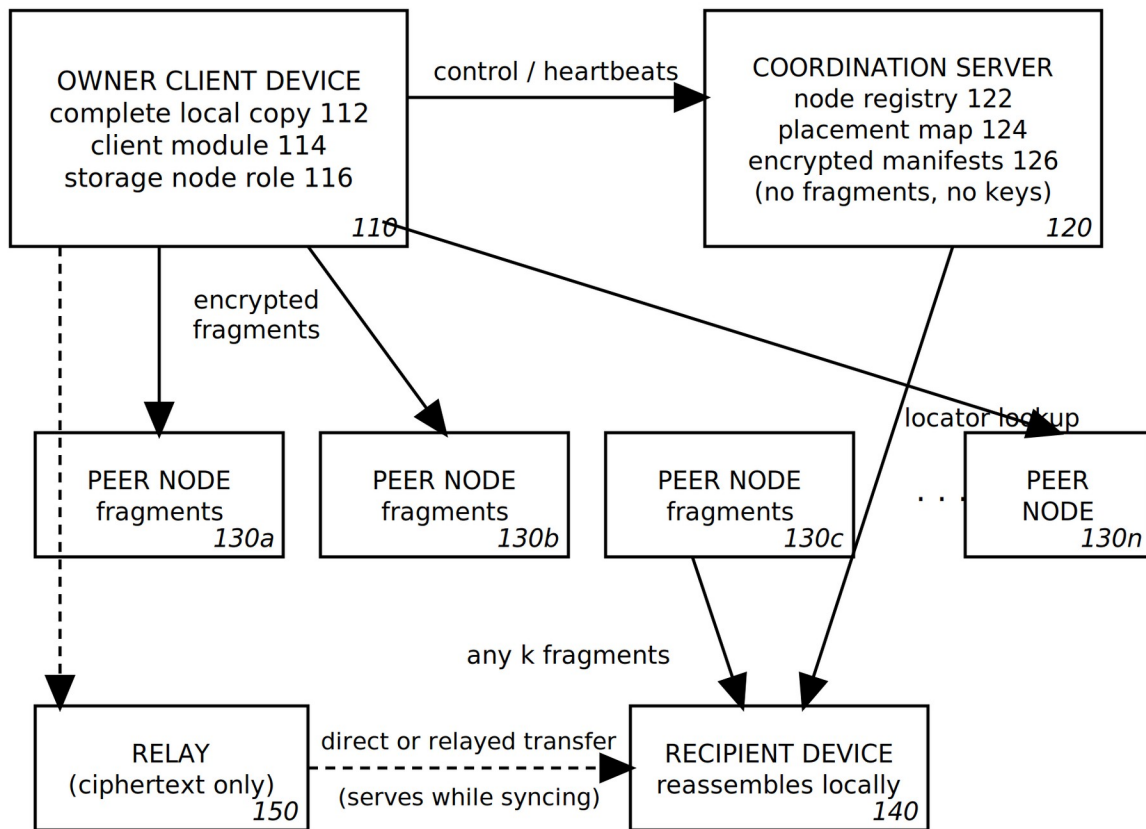


FIG. 1

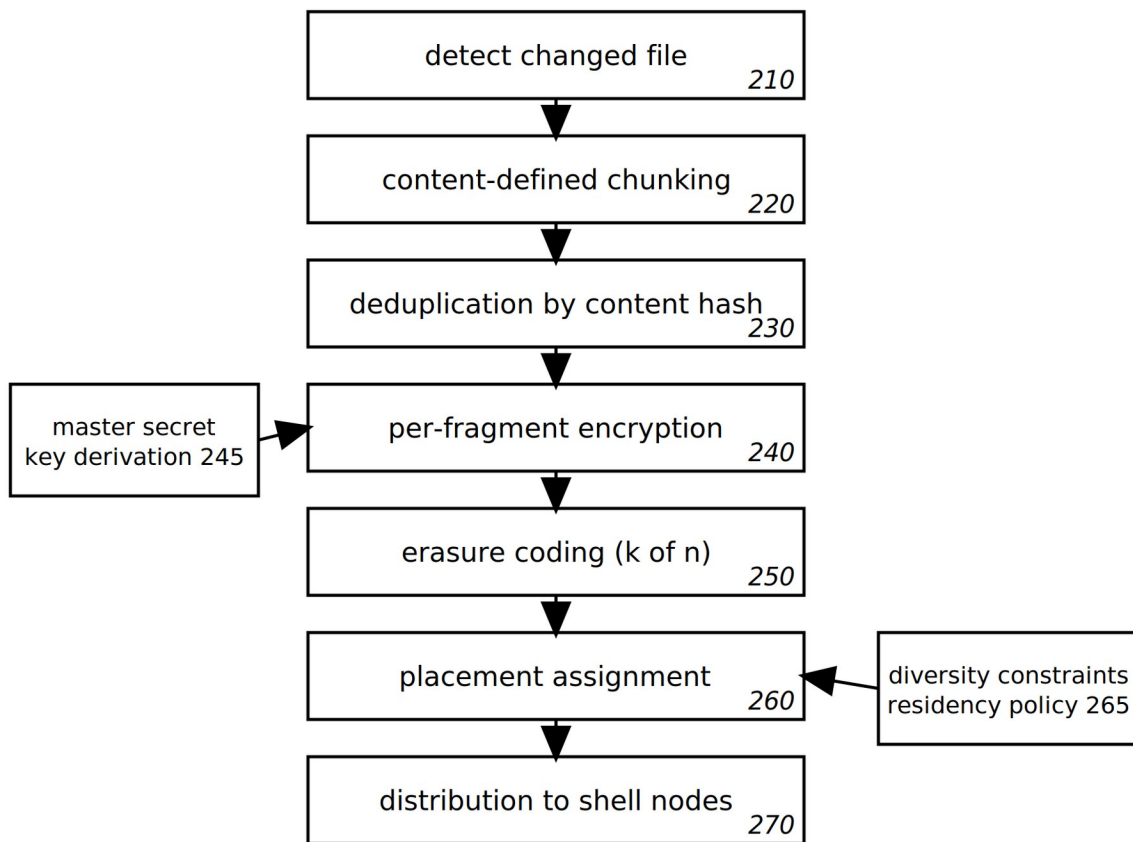


FIG. 2

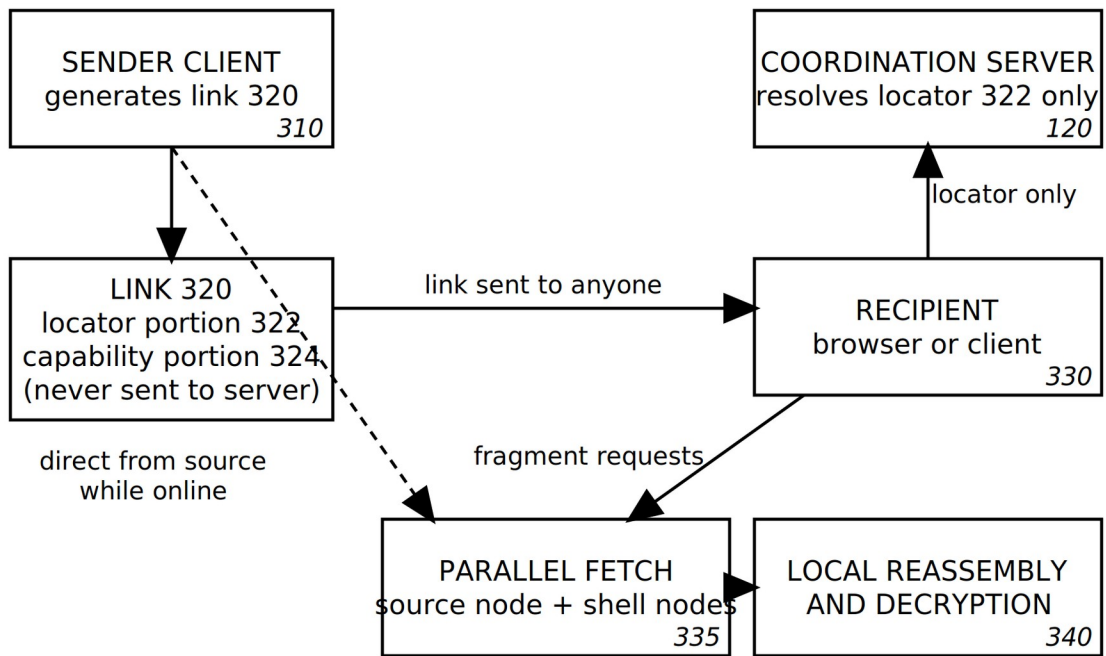


FIG. 3

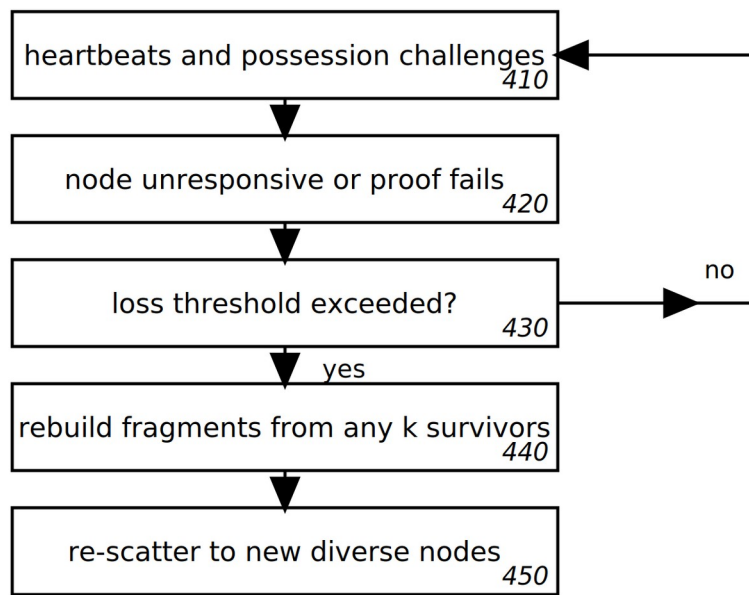


FIG. 4