Memoriae：一个高效、安全且可靠的大规模去中心化数据存储系统
—— 围绕区块链构建去中心化存储服务

摘要

传统的基于数据中心的集中式存储系统将会面临来自性价比、安全性与可靠性的前所未有巨大挑战。Memoriae 是一个为应对上述挑战并满足未来互联网应用数据存储需求而设计的分散式存储系统。Memoriae构建于区块链之上，组织并管理巨量的分散于世界各地的闲置边缘存储设备，并形成一个分级的低成本、高安全、高可靠的单一存储系统，以服务全球用户。Memoriae的核心构建思路是仅仅将那些最为关键的信息存在区块链中，以充分利用其高安全、高可用、高可靠特性；与此同时，将其他的数据存放在廉价的边缘存储设备中，以避免低性能、低扩展能力的区块链成为系统瓶颈。除此之外，我们提出了一个详尽的技术框架，以降低不可信且不可靠的边缘存储设备引入的安全性与可靠性风险。最后，我们还讨论了一些运营Memoriae的细节问题。

Memory is the treasury and guardian of all things.

——Marcus T. Cicero

1 绪论

在本章之中，我们描述了大规模集中式存储的缺点，讨论了启动Memoriae项目的动机，并阐明了Memoriae的核心思想。

1.1 中心化数据存储的弊端

互联网，尤其是无线互联网的爆炸式发展掀起了一场规模巨大的数据革命。短短十年间，社交网络、人工智能、视觉计算等构建于海量数据的应用如雨后春笋般涌现并快速普及，为人类社会的发展带来了无限可能。然而，随之而来的指数式数据增长也给数据存储——尤其是传统的基于数据中心的集中式数据存储——带来了前所未有的压力。

建设周期冗长、需求变化快速、基建成本高昂、管理投入巨大、设备更替频繁，这一切都将导致集中式数据存储成本的居高不下。一旦集中式的存储设施建成，还需要持续的运营投入以缓解安全风险并保障服务的可用性。

除成本高昂外，数据中心天然的集中性还将不可避免地引入重大安全性与可靠性隐患。受限于用户的技术水平，其上传至上述数据中心的数据大多是未经安全处理的。尽管数据中心的运营企业可能会采取某些技术手段提升保存于其数据中心的数据安全性，如数据加密、数字签名等，但考虑用户对这些技术手段的不可控性与企业运营中的可能遭遇的不可抗力，可以认为数据安全性缺乏基本保障。

此外，尽管数据中心中通常会采用冗余及相关技术以应对频繁发生的软硬件失效，但为存储服务设计与实现有效的容错机制极具挑战。由于其

中心化特点，软硬件失效具有极其显著的相关性 [1]，即大量软硬件失效会在一个较短的时间段内同时发生。出于成本考虑，冗余技术无法高效覆盖这些相关性失效。

1.2 分散式数据存储的契机与挑战

事实上，互联网中存在着大量的廉价存储资源——边缘设备（如，个人计算机）。当边缘设备接入互联网时，设备上大量空闲的存储空间就成为低成本数据存储的最佳候选。

- 无需新建基础设施，自主管理设施设备，设施设备自然更替，所有这些特性都将大幅降低数据存储成本。
- 当采用软件定义的方式管理这些廉价资源时，则能进一步规避建设周期与需求变化的问题。
- 除此之外，基于开放源代码的用户侧安全技术，能够从源头上减小数据泄露风险，而边缘设备的自然分散式特点，则将大幅降低相关性失效发生的概率。

因此，可以预见，整合廉价边缘存储设备以提供互联网级的数据存储服务将是大势所趋。

然而，实现上述目标并非易事。当系统通过互联网管理行星级边缘存储设备并提供全球性通用存储服务时，即使是保障最基本的数据存取操作的可用性也将面临全方位的挑战——这里我们将可用性定义为在预先约定的时间内完成服务目标的概率。

显而易见，可用性不仅取决于安全、可靠等基本存储功能目标的实现，还依赖于存储系统能够达到的性能水准。

就边缘存储设备而言，其拥有者行为的多样性必将导致资源自身的不稳定性，而其薄弱的防护机制则会引入众多的安全隐患。

就网络传输资源而言，网络连接的可靠性，网络延迟的差异性，都可能严重影响存储服务的可用性。

即便是就存储系统中最基本、最关键的寻址功能而言——通过给定键（key，如，文件名）找到对应值（value，如，文件内容），由于缺乏安全可靠的基础设施，保证其正常而稳定的执行也是极其困难的。

1.3 区块链 – –构建分散式数据存储的基石

幸运的是，伴随新兴数字货币涌现而出的区块链技术 [2,3]极有可能成为解决上述问题的关键。区块链通过链式方式组织时序数据，运用密码学特性防止篡改与伪造数据，并采取高冗余避免数据丢失，加之与生俱来的分散式特性，使其能够在低可信条件下提供高安全高可靠服务。加密货币多年的运转更是充分的验证了区块链技术的上述特性。

本文中，我们介绍了Memoriae，一个基于区块链的分散式数据存储系统。Memoriae以区块链为基础，组织并管理广泛存在却备受忽视的廉价边缘存储设备，并为全球用户提供高可用的数据存储服务。

受限于其糟糕的性能与高昂的冗余代价，仅仅使用区块链还无法为成千上万的用户提供聚合吞吐率达每秒数十亿I/O与总容量达到ZettaByte级别的数据存储服务。为了充分利用区块链的安全可靠特性，并避免其成为整个系统的性能与成本瓶颈，在Memoriae 中，只有那些存储系统中最为关键的数据，如，角色（账户）信息、智能合约信息（甚至摘要）等，才会被保存到区块链的主链中，而其它的数据，如数据位置信息、用户数据等，则被存放在性价比更高的边缘存储设备中。与此同时，更加简洁高效的技术将被用于保障存储于边缘存储设备中的数据的安全性与可靠性，从而提

升整个存储系统的性价比。

用户通过基于区块链的智能合约在Memoriae中购买存储服务。交易收益将按比例分配于以下四个方面：

- 租赁边缘存储设备；
- 管理与维护边缘存储设备上存放的数据；
- 基于区块链的事务处理；
- 保障Memoriae相关技术的持续研发。

我们将以可用性与存储用时用量为指标衡量Memoriae的服务质量。具体而言，用户依服务可用性及存储用时用量为基准支付服务费用，服务者（边缘存储设备提供者）依可用性及提供的存储用时用量为基准分配收益。

为了达到上述目标，Memoriae将使用三类共识：

- 关于边缘存储设备上存储的数据的可靠性共识（Proof of Reliability）；
- 关于边缘存储设备上存储的数据的可用性共识（Proof of Availability）；
- 关于边缘存储设备存储数据用时用量的共识（Proof of Space & Time）。

需要强调的是，采用共识用于评估可靠性、可用性与用时用量能够大幅提升系统的可信度，但这些共识会采用链下的方式完成，从而避免为区块链带来支付之外的事务负担。

1.4 术语与内容组织

本文将使用到的术语含义如下：

用户（User）：Memoriae中存储服务的使用者。

存储者（Provider）：Memoriae中边缘存储设备的提供者。

维护者（Keeper）：Memoriae中安全性、可靠性、可用性的维护者，并具备达成共识、维护区块链的能力。

可用性：在预先约定的时间内完成服务目标的概率。在Memoriae中，可用性既是边缘存储设备的关键属性，也是用户所接受的存储服务的关键属性。对于边缘存储设备而言，其可用性使用公式 1描述。对于用户接受的存储服务而言，其可用性使用公式 2描述。

$$\text{边缘存储设备的可用性} = \frac{\text{边缘存储设备及时并成功响应的请求数}}{\text{边缘存储设备服务的总请求数}} \quad (1)$$

$$\text{用户接受的存储服务的可用性} = \frac{\text{用户被及时并成功响应的请求数}}{\text{用户发出的总请求数}} \quad (2)$$

可靠性：在约定服务时间内，存活数据占总数据的比例。在Memoriae中，可靠性用户数据的关键属性，其可使用公式 3描述。

$$\text{用户数据的可靠性} = \frac{\text{存活的用户数据量}}{\text{用户存储的数据总量}} \quad (3)$$

私密性：在约定服务时间内，未泄漏的用户私密数据占用户私密数据的比例。在Memoriae 中，私密性是用户数据的关键属性，其可使用公式

4描述。

$$\text{用户数据的私密性} = \frac{\text{未泄漏的用户私密数据量}}{\text{用户存储的私密数据总量}} \tag{4}$$

完整性：在约定服务时间内，未被篡改的数据占总数据的比例。在Memoriae中，完整性是用户数据的关键属性，其可使用公式 5描述。

$$\text{用户数据的完整性} = \frac{\text{未被篡改的用户数据量}}{\text{用户存储的数据总量}} \tag{5}$$

安全性：包括完整性与私密性两个方面。

时空用量：用户数据占用的边缘存储空间在时间上的累计值。

2 系统总览

本章描述了Memoriae的系统架构及系统中的主要角色。

2.1 Memoriae架构

构建如此规模庞大的分散式存储系统的关键问题在于高效的管理系统角色与关联关系。

系统中存储功能包含的三类主要角色分别为：

- 用户（users）——需要将数据存储到系统中；
- 存储者（providers）——那些保存数据的节点；
- 维护者（keepers）——那些协调用户与存储者的节点。

关联关系包括：

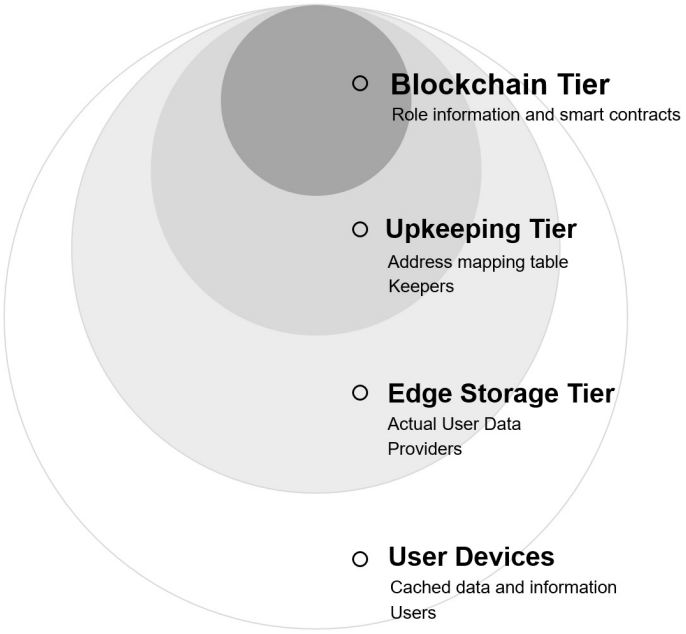

Figure 1: Memoriae整体架构图

- 角色间的关联关系（为支持付费），其也是维持系统持续运转的关键；
- 角色与数据之间的关联关系（为支持数据寻址），其也是实现存储功能的关键。

需要注意的是，角色信息与角色间的关联关系的总量是远小于角色与数据间的关联关系的总量的。

逻辑上，Memoriae系统由区块链、管理设备、边缘存储设备以及用户设备组成。为了最大程度的探索与挖掘区块链的安全性与可靠性，同时保持良好的系统可扩展性与性价比，Memoriae仅使用区块链记录最为关键与稳定的信息，并使用管理设备与边缘存储设备存储其他信息与用户数据。Memoriae的整体架构如图 1 所示。

具体而言，如图 2所示，以下两类信息将被记录于区块链上：

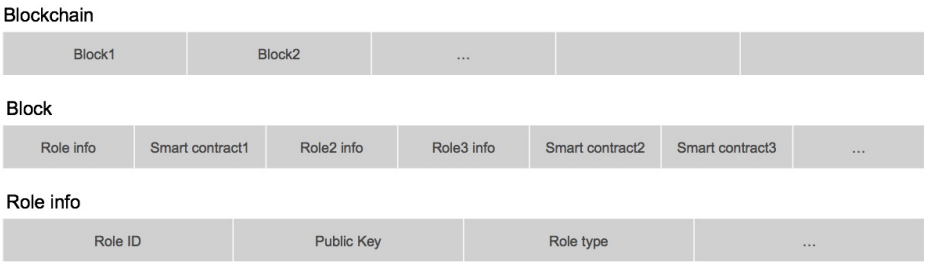

Figure 2: 存储在区块链上的主要信息

- 用户、存储者与维护者的角色信息；
- 角色之间的关联信息，包括用户与维护者间的关联信息（必选），以及用户与存储者间的关联信息（可选）。

如图 3a所示，代表用户与其数据间关系的地址映射信息存储于那些与该用户签署了智能合约的维护者处（维护节点），我们选择将这些地址映射信息保存于keeper之上而非区块链上的关键原因在于：

- 这些地址映射信息总量极大，会给区块链矿工带来巨大的存储压力；
- 这些映射信息改动频繁，会造成大量的额外链上事务。

如图 3b所示，实际用户数据存放于存储者处（边缘存储设备上）。如图 3c 所示，用户处（客户端）除了保存一部分暂未写入到Memoriae的缓存数据外，还将保存一些热点信息与数据的只读副本，以提升系统整体性能。

相对于传统的区块链而言，更加简洁与高效的手段将被用来保护维护者、存储者以及用户信息与数据，因而保证了整个存储系统的性价比，这些内容将在第三章之中讨论。

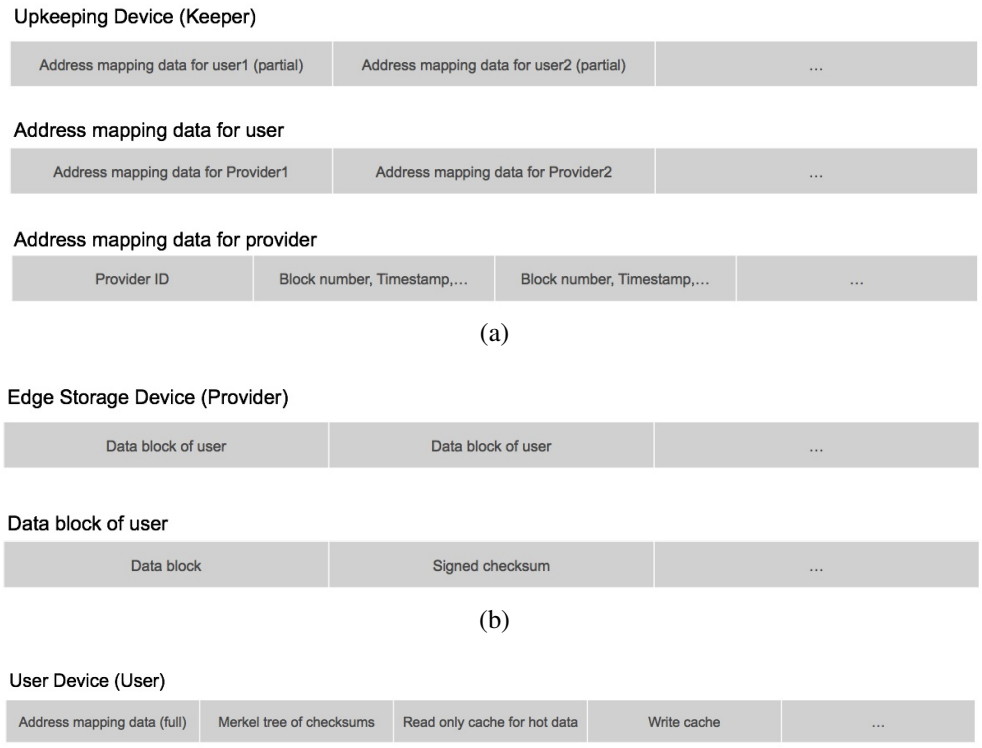

Figure 3: 各类设备上存储的数据与信息

2.2 角色

系统角色的主要功能如下所述：

（1）用户（User）：用户是指Memoriae存储服务的使用者，用户可以通过配置智能合约参数指定需要的服务等级与匹配合适的存储者。用户能够生成自己的智能合约，上传数据到Memoriae，并使用代币支付服务费用。用户也能够下载并管理其存储于Memoriae的数据。

（2）存储者（Provider）：Memoriae中边缘存储设备的提供者。存储者可指明自己的服务能力以匹配合适的用户，并存储这些用户的数据。其还能执行并响应用户的数据访问与管理请求。此外，存储者也能响应来自维护者的挑战并配合维护者修复数据。存储者接受代币作为提供数据存储服务的收入。

（3）维护者（Keeper）：维护者负责保证Memoriae的安全性、可靠性与可用性。维护者除需要保存数据映射信息外，其也使用这些映射信息来挑战存储者并达成管理共识。这些共识包括对存储者可靠性与可用性形成的共识，修复数据时机的共识，以及存储用时用量的共识。维护者同样也接受Memoriae 代币作为其维护服务的收入。

图 4展示了三类角色之间的交互关系。理论上，在系统冷启动期间，对每一个用户而言，其维护者是随机选择的，而存储者则是根据其指明的可用性参数（若考虑存储市场，以及价格参数）以随机的方式匹配得到的。这个随机选择的过程是自动化的。当维护者积累了一定的关于存储者的可用性与可靠性数据之后，则可以建设一个评分系统来匹配用户与存储者。系统将倾向于奖励那些能够遵守合同承诺的存储者，并处罚那些不遵守合同承诺的存储者。对于后续加入的新的存储者而言，其会被分配一个由系统平均服务水平推得的分数作为默认初值。当然，该评分会随着此用户受到的奖励与处罚升高或降低。

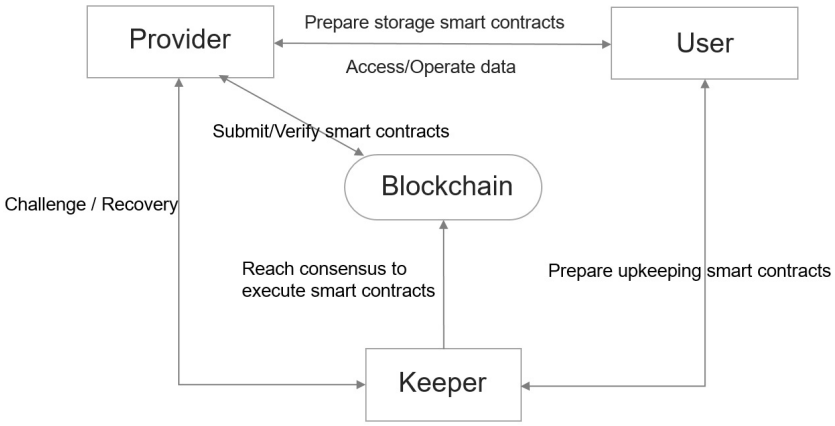

Figure 4: 三类角色之间的交互关系

除区块链之外，Memoriae的主要功能被实现于的三个软件模块。这三个模块分别为运行于用户处的客户端模块，运行于维护者处的维护模块，以及运行于存储者处的存储模块。

3 关键技术

数据存储功能是Memoriae的基石，而数据安全性与可靠性则是数据存储功能的基石。在本章中，我们描述了Memoriae中保障数据隐私性、完整性与可靠性的关键技术。各类主流的密码学技术，如对称式加解密 [4]、防碰撞哈希函数 [5] 与数字签名技术 [6]，被应用以满足严格的数据隐私性与完整性需求。上述密码学技术与数据冗余及修复技术将为系统提供足够的弹性以容忍各类可能发生的软件与硬件故障以及小规模的恶意攻击，如试图入侵维护、存储与用户设备，从而篡改、销毁或公开数据与信息。同时，我们还会采用久经考验的POW技术 [7]与更加新颖的POS技术 [8, 9]，并辅之以第三方安全工具，以进一步提升系统安全性。

3.1 数据安全性

在本节中，我们不但描述了将会采用的数据安全策略，还将讨论访问控制技术。

3.1.1 数据隐私

对称式数字加密是一种保障数据隐私性的有效方法。在Memoriae中，我们的数据隐私策略要求数据在其源头——客户端——即被使用用户密钥加密，之后才会写入到边缘存储设备中。因此，无论是在网络上传输的数据，还是在边缘存储设备上存储的数据都是密文形态。

```
def MR_FWrite(DataBuffer, ...):
    ...
    EncryptDataBuffer=Encrypt(DataBuffer, ...);
    MR_PutData(EncryptDataBuffer, iProvider, ...);
    ...
```

只有那些被读取到客户端的密文数据，才能使用用户密钥自动解密。

```
def MR_FRead(DataBuffer, ...):
    ...
    MR_GetData(EncryptDataBuffer, iProvider, ...);
    DecryptDataBuffer=Decrypt(EncryptDataBuffer, ...);
    ...
```

3.1.2 数据完整性

无论是以明文还是密文形式存储或传输的数据，其完整性都必须得到保障。防碰撞哈希与数字签名技术会被用作识别损坏的数据。冗余数据则将被用作修复损坏的数据。

客户端软件将使用用户私钥签名由数据生成的哈希值。签名后的哈希值将与数据一起被传输与存储。图 5 展示了存储隐私数据的主要过程。

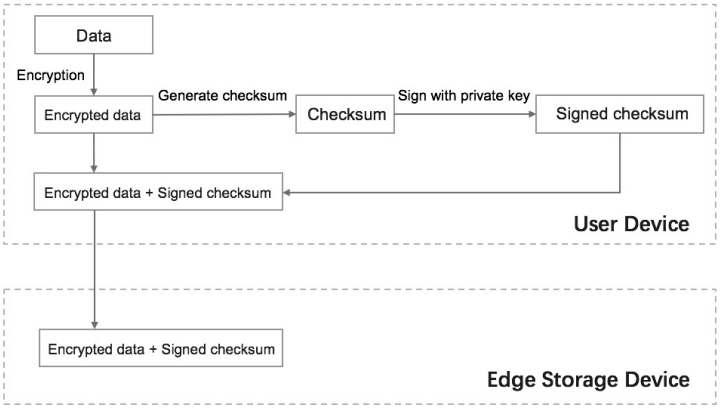

Figure 5: 存储私密数据的主要步骤

```
#attach a signed checksum to private data
def MR_Put_Checksum(EncryptDataBuffer, ...):
    ...
    checksum=MR_Make_Checksum(EncryptDataBuffer, ...);
    EncryptChecksum=MR_Sign(checksum, ...);
    Connect(EncryptDataBuffer, EncryptChecksum, ...);
    ...
```

为了验证数据的完整性，任何系统角色（包括用户、维护者或存储者）必须使用数据拥有者的公钥来验证哈希值的真实性。而哈希值则能够被用于验证数据的完整性。图 6与图 7分别展示了读取与挑战私有数据的主要过程。值得注意的是，当选取的防碰撞哈希函数支持同态运算时，对于多个数据块的挑战实际上是可能合并的。即，在挑战时，可以一次传输多个数据块的加和，而非这些数据块本身，这样能够进一步减少通信开销 [10, 11]。

```
#verifying the integrity of data
def MR_FRead(EncryptDataBuffer, EncryptChecksum, ...):
    ...
    checksum1=MR_Make_Checksum(EncryptDataBuffer, ...);
    checksum2=MR_VerifySign(EncryptChecksum, ...);
```

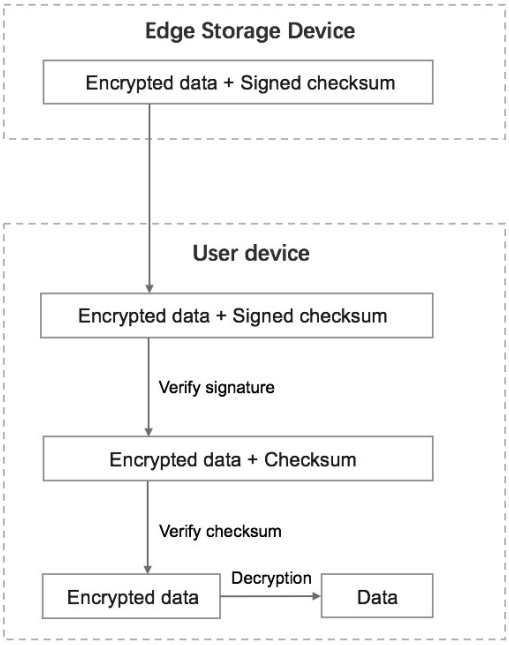

Figure 6: 读取私密数据的主要步骤

```
if(checksum1 != checksum2) :
    #the data are tampered and they should be recovered
    ...
else:
    #the data are integral
    ...
...
```

损坏的数据将被丢弃，我们将采用3.2节中描述的方法恢复这些被丢弃的数据。

3.1.3 访问控制

访问控制机制将被用作补充前面提到的方法，以进一步提升数据安全性。

理论上，存放在边缘存储设备的数据只能被其拥有者（用户）、维护者

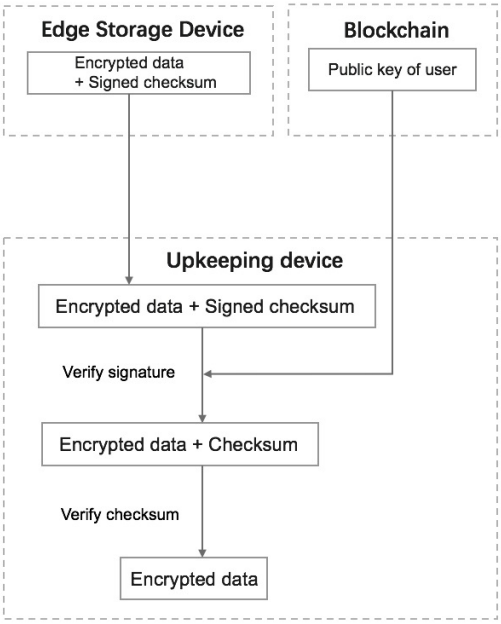

Figure 7: 检验私密数据完整性的主要步骤

或存储者访问。其中，用户对其拥有的数据有着完整的访问权限。维护者对其维护的数据具有读取的权限，且仅对已经被判别为损坏的数据具有修复及写入权限。存储者只能够在得到数据拥有者或数据维护者授权的情况下才能读取或写入数据。而其他任何人都无权访问数据。且在任何情况下，任何人都无法访问明文形态的隐私数据。

访问控制机制保证了数据能够被其拥有者或其维护者与存储者协作访问，同时不被其他用户、或无关的维护者与存储者访问。值得注意的是，访问控制机制对于存放数据的存储者而言是没有意义的，因该存储者实质上拥有对自身设备的所有访问权限。

智能合约的存储有两种选择：一是如同现有的绝大多数解决方案一样存放在区块链上；二是如未来将会流行的解决方案一样，采用签约各方链下保存而链上仅存放指纹的形式。这个版本的白皮书中，仅围绕第一种方

式展开讨论。

当智能合约被存储于区块链上，可被任何人访问。因此，对任何存储者而言，数据的拥有者及维护者的身份实际上是可查的。请求发送者（数据的拥有者或维护者）的ID 会采用其私钥签名，并包含在请求之中。服务者只需使用请求者的公钥就能验证请求者的身份，并依据区块链上的智能合约判断是否应执行请求。此外，请求中将会植入时间戳，以防止重放攻击。

访问控制策略并不禁止存储者访问他们保存的数据。但这个对于数据安全性的影响极其有限。由于存放在边缘存储设备的隐私数据数据是加密存放的，因此数据隐私得以保障。根据现有的密码学理论，在没有获取私钥的情况下，依靠算力从密文暴力破解出明文的概率低到可以忽略不计。对于那些没有加密就存放在Memoriae中的数据，其实质上等同于已经放弃了隐私性要求。同时，无论是私有数据还是公开数据，其完整性均由前面描述的防碰撞哈希与数字签名方法保障。以下给出了提交一个请求与验证其合法性的步骤。

```
#User or Keeper
def MR_MakeRequest(RequestType, RoleID, SequenceNO, PrivateKey, ...) :
    ...
    request.type=RequestType;
    request.id=RoleID;
    request.content=RequestContent_Encrypt(BlockID, RoleID, SequenceNO, PrivateKey, ...);
    ...

#provider
def MR_HandleRequest(request, ...) :
    ...
    if(READ_REQUEST == request.type) :
        PublicKey=MR_GET_PublicKey(request.id, …);
        RequestContent=RequestContent_Decrypt(request.content, PublicKey);
        BlockOwnerID=MR_GET_BlockOwner(RequestContent.BlockID);
```

```
        BlockKeeperIDs= MR_GET_BlockKeepers(RequestContent.BlockID);

        if((BlockOwnerID != RequestContent.id) &&
            RequestContent.id is not in BlockKeeperIDs) :
            #this request is illegal
            ...

        If(RequestContent.SequenceNO is not unique) :
            #this request has been handled
            ...

        ...
    else if(WRITE_REQUEST == request.type) :
        PublicKey=MR_GET_PublicKey(request.id, …);
        RequestContent=RequestContent_Decrypt(request.content, PublicKey);
        BlockOwnerID=MR_GET_BlockOwner(RequestContent.BlockID);

        if(BlockOwnerID != RequestContent.id) :
            #request is illegal
            ...

        if(RequestContent.SequenceNO is not unique) :
            #this request has been handled
            ...

        ...

    ...
```

3.2 可靠性

本节描述了Memoriae保障可靠性的方法。我们的目标是防止数据在用户设备、维护设备或/与边缘存储设备故障时丢失。本节不会讨论网络故障相关问题，原因在于：1.网络故障可以视作设备故障处理；2.当设备上的数据可靠性得到保障时，网络传输过程中丢失的数据可以通过重传处理。

3.2.1 维护设备上的容错

单个或少量维护设备上损坏的与丢失的地址映射信息几乎没有影响，原因在于：当智能合约执行期间，每个用户的地址映射信息将会在其所有的维护者间完全复制。

当一个维护设备上的地址映射信息完全损坏或丢失时，维护者必须查询区块链上的事务记录，以找到那些与其服务对象相同的维护者，随后，即可从那些维护者处同步完整的用户地址映射信息，并重新生成自己的地址映射信息。

每个用户的所有维护者必须周期性的对地址映射信息做共识，以修复那些可能未能的同步的信息。对于那些数次未能参与共识的维护者，其将丧失维护资格，并受到惩罚。

如前文所描述，在整个系统中，元数据量与数据量相比微不足道，且维护者存储的元数据的复制因子（副本份数）将远低于区块链上存储数据的复制因子，因此，即使是采用副本技术对元数据进行容错，也不会造成存储用量的大幅上升。而考虑到元数据的访问频率将远远高于数据本身，因此，采用副本方式容错能够大幅提升系统的访问处理能力。

3.2.2 边缘存储设备上的容错

副本、纠删码 [12]与其它数据冗余技术能够被用来保障边缘存储设备上存放的数据的可靠性。无论是副本还是校验都是由那些已签名的原始数据生成。签名的数据与冗余数据共同构成冗余组。冗余组内的块被分发到不同的边缘存储设备上。当某些边缘设备故障导致数据/冗余丢失时，其它存活的设备上同组数据/冗余将被取出，并用作修复丢失的数据。同时，当设备暂时离线时，冗余还能够提升可用性。

尽管系统同时支持副本与纠删码两种容错技术，但绝大多数的用户数

据会采用纠删码而非副本方式处理，其原因在于，相较于副本技术，纠删码能够大幅降低冗余带来的空间成本，进而降低用户购买存储服务的价格。只有那些会被频繁访问数据才会使用副本技术处理，以提升访问性能。

除数据冗余机制之外，数据修复方法也是决定数据可靠性的重要因素。Memoriae独创的数据恢复方法RAFI [13] 被用作进一步提升存储于边缘存储设备的数据可靠性。RAFI是一类适用于所有分布式存储系统的数据修复方法，不管目标系统是数据中心中部署的存储集群，还是Memoriae这类的分散式存储系统。

RAFI工作的原理在于，通过快速发现那些具有较高丢失风险的数据，有效缩短数据修复总时间从而提升数据可靠性。

RAFI技术对可靠性的提升幅度随容错度（如，副本数量）的增加而增加。例如，在（6，3）纠删码存储集群中，RAFI能够将数据可靠性提升一个数量级左右，而在容错度更高的分散式存储中，RAFI有能力进一步提升数据可靠性。关于RAFI技术的细节，可参考对应论文 [13]。

4 存储生态系统

服务费用由存储累计用时用量、访问系统的费用与存储服务质量决定，而存储服务质量则以可用性衡量。绝大多数的服务费用会按比例分配给存储者、维护者以及区块链矿工，只留一小部分收入用于Memoriae日常管理与开发。此外，区块链矿工还能通过POW挖矿获得代币。

4.1 服务收费

传统存储系统的服务目标涉及一系列指标，包括成本、容量、可扩展性、接口、可管理性、性能、安全性以及可靠性等等。

我们将会使用区块链管理分散在世界各地的低成本边缘存储设备，以

提供通用存储服务。此设计基本涵盖了成本（单位成本）、容量（总容量）与可扩展性，并包含了对象访问接口支持。今后的版本中还会添加文件与块级访问。引入维护者是为了外包管理，以便自动化处理海量管理任务，并自动维护边缘存储设备。

Memoriae系统架构能够大幅降低数据泄露风险，此外，数据篡改与数据丢失将按照同样的方式处理。因此，用户的可调节服务目标主要有两方面：可靠性与性能，这两者均由可用性指标决定。

总的来说，如果不考虑系统设计因素，数据副本越多，性能越有保障。在一定的技术条件下，挑战越频繁、数据冗余越多，可靠性就越高。但显然，更高的冗余度（副本也能被视为一类冗余数据）与更频繁的挑战，消耗的系统资源也会越多，这反过来又会增加服务成本。

存储累计用时用量是服务成本的另一个决定因素。固定大小的存储空间在给定的时间间隔内，只能存储相同大小或者更少的数据（排除压缩技术），这是由于数据存储的独占特性所致。单个用户的存储累计用时用量是评估用户Memoriae 存储资源消耗的一个最直观指标。因此，决定服务成本的指标除了可用性之外，还包括存储累计用时用量。

当维护者对某存储者在一定时间段内（周期性，例如每天或每周）的归一化IO总量、可用性与空间使用量达成共识时，存储者所服务的用户需按约定支付相应费用。

$$userfee = MRCalFee(\#IO * Availability, Spacetime) \tag{6}$$

4.2 收入分配

Memoriae的存储部分设有三个角色，加上区块链矿工，则共有四个角色。用户为使用的服务付费，存储者、维护者与区块链矿工就所提供的服

务收费。此外，Memoriae服务赚取的收入的绝大部分分配给存储者，剩余收入用作向维护者支付管理费用、向区块链矿工支付交易费用、以及开发与维护系统。角色之间的收入分配如下所述。

（1）不同角色间的收入分配：除一小部分收入将用于开发与维护外，大部分收入分配给存储者、维护者与区块链矿工。假定系统规模和提供服务与系统管理所需的工作呈线性增长，则每个角色的收入占总收入的比例是固定。例如，总收入中的5%用于系统管理和开发以及支付交易费用，85%分配给存储者，5%分配给维护者，剩余5%则作为交易费分配给矿工。

$$Puserfee = Preserved + Pproviderincome + Pkeeperincome + Pminerincome \tag{7}$$

（2）相同角色间的收入分配：同一角色间基于各自可量化的贡献按比例进行收入分配。存储者核心功能是提供数据存储空间并响应用户请求。存储者的贡献可通过使用的存储用时用量、归一化IO数量以及可用性进行量化。维护者的主要功能发起挑战，报告挑战结果，修复还原数据并确保在维护者间达成共识。维护者的贡献可通过有效的挑战数、修复与/或还原的数据量以及达成共识的数量进行量化。

$$providerincome = MRCalPvrPay(\#IO, Availability, Spacetime, ...) \tag{8}$$

$$keeperincome = MrCalKprPay(\#Challenges, RecoveredData, \#Consensus, ...) \tag{9}$$

$$minerincome = MrCalKprPay(\sum Puserfee, ...) \tag{10}$$

5 链下共识

Memoriae 在实际操作中应用三种共识。需要再次强调，由于区块链的低可扩展性，这些共识并非由区块链矿工于链上达成，而是由维护者于链下达成，这样可以在保证可信度的同时，有效提升系统处理能力。

5.1 可靠性证明（Proof of Reliability）

数据修复能力与数据冗余一样重要。数据修复通常包含错误识别与数据恢复两个阶段。必须先正确找出失效数据，然后才能对数据进行恢复。边缘存储设备在可靠性方面可谓声名狼藉。引入维护者就是为了更好的驱动数据修复进程。遗憾的是，维护者自身也不可靠。如果失效数据仅依靠一位维护者识别，维护设备故障或者恶意维护者可能导致无法及时识别出失效数据或者假数据失效。未能及时识别的数据失效将增加数据丢失风险，而假数据失效则会增加维护成本。

因此，Memoriae要求在恢复数据之前，多名维护者需就数据/设备故障达成共识。

5.2 可用性证明(Proof of Availability)

可用性是衡量服务质量的核心指标，也是分配收入时考虑的重要因素之一。

由于用户也不是完全可信的，因此不能仅由用户报告边缘存储设备的可用性。否则，用户报告的可用性很可能会低于实际水平，从而影响存储者的收入。

另一方面，维护者可以在存储者在线时对其进行质询。维护者发起的质询实质上等效于数据访问，因此维护者也能参与判定存储者的可用性。

Memoriae要求维护者验证用户报告的可用性与其独立判定的可用性是否相符。维护者必须就用户费用与收入分配两方面达成共识。

5.3 用时用量证明（Proof of Space & Time Utilization）

存储累计用时用量是衡量用户所消耗资源的一个直接指标。

每个维护者维护多个用户完整的地址映射数据，包括这些用户存储的数据块的创建时间戳。持有地址映射数据的维护者必须形成共识才能给出权威性的用时用量数据。

6 智能合约

智能合约用于维持维护者和存储者的身份和服务信息，确定维护者和存储者提供的服务是否适当，也规定了如何向用户收费。Memoriae 智能合约是基于用户输入、维护者和提供者在区块链中记录的信息为用户自动生成的。

6.1 角色智能合约

在任意一个维持者或存储者加入系统之前，都需要声明自己的角色（维持者和存储者）和相应的服务级别。这一声明存储在角色智能合约中，任何一个参与者都可以查询。

合约所需主要数据如下：

- 声明的可用性
- 声明的可靠性
- 声明的最大可用存储容量
- 声明的可用时间长度和频率

6.2 维护智能合约

6.2.1 合约所需数据

所需存储者服务级别：

- 可用性
- 可靠性
- 最大可用存储容量
- 边缘存储设备可用的时间长度和频率

所需维护者服务级别：

- 可用性
- 可靠性
- 服务能力

所需维护者服务：

- 基于存储者的可用性证明和可靠性证明选择存储者
- 结合存储数据量和存储者可用容量确定所需的存储者数

6.2.2 工作流程

在Memoriae 中存储任何数据之前，用户必须选择理想的存储服务级别。存储服务级别将决定所需的最少维护者和存储者。所需的维护者数是由用户基于其处理地址映射信息（该指标基于维护者处理的地址映射信息条目进行计算）的能力和可以提供的服务水平选择的。

在选择了维护者和存储者后，用户创建并签署维护智能合约，约定维护者如何向存储在存储者边缘存储设备上的数据提供维护服务以及用户收费标准，以及用户和存储者之间的存储服务以及付费条款。

用户在签署维护智能合约之后，可以请求在P2P 存储网络上存储数据。从合约中的存储者中挑选合适的存储者进行存储。用户数据的所有维护者必须就存储者的可用性证明、可靠性证明和用时用量证明达成共识后才能出示结果。然后，维护者将触发维护智能合约向维护者和对应的存储者进行付费。

6.2.3 存储者

存储者数目可能会由以下两个原因发生变化：一是随存储数据量增加而增加；二是存储者由于故障导致可用存储者数目不足，需要添加新的存储者。

在验证存储者符合理想服务级别后，有两种方式添加存储者：一是用户向维护合约中添加存储者；二是由维护者对此存储者达成一致，然后使用环签名技术来添加存储者。

7 Memoriae 区块链

区块链是Memoriae 的一个重要组成部分。它用于促成用户、存储者和维护者之间的交易，以确保存储者和维护者能基于服务按比例获得报酬，得到激励并保持积极性。Memoriae 的终极目标是组织大量的边缘设备为庞大的用户群体提供存储服务。区块链必须能扩展到每秒处理上万起（如果做不到几十万的话）交易才能满足存储服务的需求。

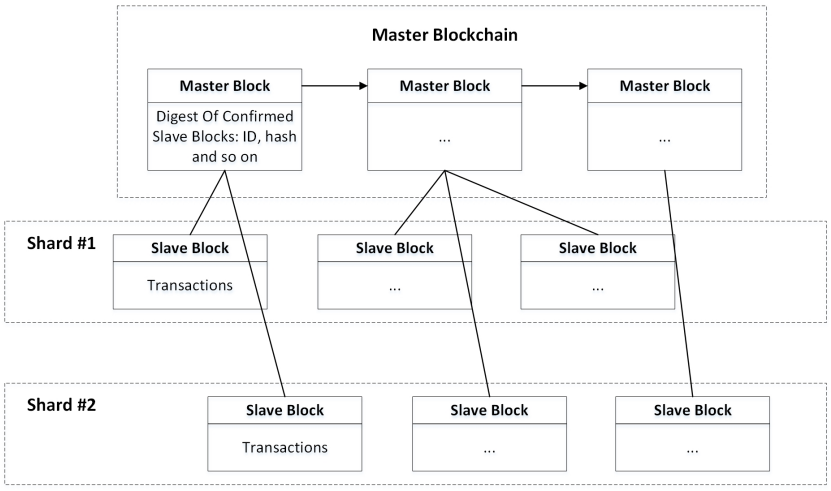

Figure 8: 使用分片技术后的Memoriae区块链

7.1 架构

Memoriae 区块链采用标准技术来改善性能和吞吐量，如图 8所示。我们的区块链由一个主根区块链和多个从属区块链（分片）组成。每个分片都有一个锚定到主根区块链的从属区块链。主区块链上的区块仅记录分片ID、从属区块链区块的区块地址和哈希值。主区块链不处理任何实际交易，它只确认或者拒绝从属区块链建议的交易。交易先由从属区块链确认，预先确认过后再提交到主区块链进行最终确认。

Memoriae分片大小被Memoriae 网络内的维护者和提供者数以及总交易数控制。分片的大小和数目不断发展变化以平衡网络。总之，维护者和存储者将始终在所有分片间随机均衡分布。

7.2 共识算法

Memoriae 的主区块链采用通用的工作量证明(PoW) 共识算法，其工作原理与大家熟知的其它PoW 区块链非常相似。主区块链的矿工必须

解决复杂加密问题，才能在主区块链上创建新的区块，从而“挖”到新的Memoriae 代币。

另一方面，Memoriae从属区块链（分片）采用服务证明(PoSs) 共识算法。存储提供者在从属区块链中充当铸造者。他们不“挖“新的Memoriae代币，而仅在所提交的区块在主区块链上得到确认后挣取交易费。

每一轮都会在每个分片内随机选取维护者来推举铸造下一区块的存储者。存储者的提名基于其服务证明(PoSs)，而PoSs 则由存储者在最近N 个区块中的可用性证明和用时用量证明决定。提名必须获得三分之二维护者集体签名才有效。此时被提名的存储者才有资格为分片创建下一区块。分片内至少51% 的存储者必须对被提名的存储者建议的区块达成共识，而分片内不低于51% 的存储者集体对新区块签名则表示已达成共识。该区块由存储者签名后，被提名的存储者必须提交分片ID、区块地址和哈希值以及集体签名，它才能包含在主区块链区块中。

采用两种不同的共识算法导致很难在Memoriae 区块链上进行分叉攻击。攻击者必须获得51% 以上的哈希算力才可以对主区块链进行分叉。这已被证明非常困难。如果要建议新区块，攻击者还必须获得分片内所有存储者至少51% 的哈希算力。但是，建议的任何不属于主区块链最长链的分片区块将被自动拒绝。

7.3 主链矿工

Memoriae 的主区块链矿工数量M（每次竞选时确定，不超过1024）个，节点需要通过竞选才能成为矿工参与主链共识。竞选活动周期性举行，每N个出块周期举行一次，N 为可调参数。竞选条件为提供的存储空间大小（可在链上查询PoST证明）和抵押代币数量，经过计算综合得分最靠前

的M个节点成为矿工，而恶意参与竞选（提供存储空间和抵押代币数量显著小于最低竞选要求）的节点将会burn掉部分代币作为惩罚。

7.4 外部公链接入

Memoriae 支持外部公链接入。外部公链用户可以直接使用Memoriae的存储服务而不需了解Memoriae的存在，甚至可以使用外部公链的代币进行支付。

支持外部公链接入的逻辑如下：

- Memoriae运行跨链服务节点MCS（Memoriae Cross Server），它们负责交易的跨链同步。这些MCS 同时运行在Memoriae 链和外部链上。
- 在外部链上部署跨链智能合约MCSC（Memoriae Cross Smart Contract），负责外部链上处理用户的交易。
- 用户给MCSC提交存储交易、支付代币、并向MCS发送需要存储的数据。
- MCS收到数据后，通过MCSC确认用户交易信息，如果交易合格则重新在Memoriae发起存储交易，并把数据发送给存储者，完成交易更新MCSC状态；如果交易不合格，则拒绝交易并更新MCSC状态。

8 结论

本文介绍了Memoriae 这一新颖的去中心化分布式存储系统，它提供成本低、安全可靠的大规模存储服务。基本设计理念是利用区块链来组织并管理分算在互联网的大量边缘存储设备，为世界各地的用户提供数据存储服务。Memoriae 包括如下功能：

- 使用经验证的区块链技术确保关键存储系统信息安全可靠并可用。
- 发掘大量存在但基本被忽视的低成本边缘存储设备的潜能，用以改善存储性能并降低存储成本。
- 使用加密算法、访问控制、数据冗余、数据恢复以及其它技术确保存储在边缘存储设备上的数据的安全性、可靠性和可用性。
- 使用简单可测量的标准对存储服务级别进行量化分类并按比例分配存储服务收入。

除了存储系统中都存在的用户和存储者角色，Memoriae还引入了另一角色——维护者。维护者接管了用户的大部分数据管理任务。这不仅极大改善了去中心化而且扩展了Memoriae范围。与其它依赖用户管理自己数据的存储项目不同，Memoriae将用户从繁重的复杂数据管理任务中释放出来。

Memoriae将继续致力于系统模块的创新和集成，增强其可用性以提供大规模的公共存储基础结构。

References

[1] D. Ford, F. Labelle, F. I. Popovici, M. Stokely, V. Truong, L. Barroso, C. Grimes, and S. Quinlan. Availability in globally distributed storage systems. In *USENIX OSDI 2010*, page 61 – 74, 2010.

[2] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. http://www.bitcoin.org/bitcoin.pdf, 2009.

[3] Ethereum. https://github.com/ethereum/, 2016.

[4] Advanced encryption standard. https://en.wikipedia.org/wiki/Advanced_Encryption_Standard.

[5] Secure hash algorithms. https://en.wikipedia.org/wiki/Secure_Hash_Algorithms.

[6] Digital signature. https://en.wikipedia.org/wiki/Digital_signature.

[7] Proof of work. https://en.wikipedia.org/wiki/Proof-of-work_system.

[8] Proof of stake. https://en.wikipedia.org/wiki/Proof-of-stake.

[9] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *ACM SOSP 2017*, 2017.

[10] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. *J. Cryptol*, 17(4):297 – 319, 2004.

[11] H. Shacham and B. Waters. Compact proofs of retrievabilty. In *Proceedings of ASIACRYPT' 08*, page 90 – 107, Berlin, Heidelberg, 2008.

[12] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, page 300 – 304, 1960.

[13] J. Fang, S. Wan, and X. He. RAFI: Risk-aware failure identification to improve the ras in erasure-coded data centers. In *USENIX ATC 2018*, Boston, MA, USA, 2018.