cloud security alliance

大数据工作组

大数据安全和隐私手册
100 例大数据安全和隐私的最佳实践

实际工作文件

审核人/访问者：如果您有 Google 帐户，请在评论前登录。否则，请在您留下的评论中注明您的姓名和职务。

审核人/访客：请使用 Google 文档上的评论功能，留下您对文档的反馈。要使用此功能，请突出显示您要评论的短语，右键单击并选择“评论”（或 Ctrl + Alt + M）。或者，突出显示短语，从顶部菜单中选择“插入”，然后选择“注释”。
有关 Google 评论功能的详情，请参阅
http://support.google.com/docs/bin/answer.py?hl=zh_CN&answer=1216772&ctx=cb&src=cb&cbid=-rx63b0fx4x0v&cbrank=1

中文版翻译说明

CSA 大数据安全和隐私手册（100 例大数据安全和隐私的最佳实践）由 CSA 大中华区研究院组织志愿者进行翻译。

参与翻译工作专家名单：

1.0-1.10 由刘继顺翻译、2.0-2.10 由江纬翻译、3.0-3.10 由刘小茵翻译、4.0-4.10 由黄丽诗翻译、5.0-5.10 由赵泰翻译、6.0-6.10 由周景川翻译、7.0-7.10 由程丽明翻译、8.0-8.10 由江雷翻译、9.0-9.10 由陈雪秀翻译、10.0-10.5 由任兰芳翻译、10.6-10.10 由马红杰翻译。

参与审校工作专家名单：

1.0-1.10 由江纬审校、2.0-2.10 由刘小茵审校、3.0-3.10 由黄丽诗审校、4.0-4.10 由赵泰审校、5.0-5.10 由周景川审校、6.0-6.10 由程丽明审校、7.0-7.10 由江雷审校、8.0-8.10 由陈雪秀审校、9.0-9.10 由任兰芳审校、10.0-10.5 由马红杰审校、10.6-10.10 由刘继顺审校。

合稿审核：刘文宇、叶思海、郭剑锋、李雨航、杨炳年。

全文由刘文宇、叶思海负责组织和统稿。

在此感谢参与翻译工作的志愿者。由于翻译时间仓促，存在很多不足的地方，请大家批评指正。欢迎大家提供修改意见，可发送邮件到下面邮箱：jguo@china-csa.org。

文档下载地址：http://www.c-csa.org

目录

介绍

“大数据”一词是指由公司和政府收集的有关人类和环境的大量数字信息。预计产生的数据量将会每两年翻一番，从 2012 年的 2500 艾字节增加到 2020 年的 40,000 艾字节。安全和隐私问题随着大数据的数量、多样性和速度而增加。大规模的云基础设施、数据源和格式的多样性、数据采集的流媒体特性和大量的云间迁移都会创造独特的安全漏洞。

大数据已经被许多组织收集和利用了几十年，不仅存在的大量数据，而且正在产生新的安全挑战。目前使用大数据是新颖的，因为各种规模的组织现在都可以访问大数据和使用大数据。过去，大数据仅限于政府和大型企业等非常大的组织，这些组织有能力创建、拥有用于托管和挖掘大量数据所需的基础设施。这些基础设施通常是专有的，与一般网络隔离。今天，通过公共云基础架构，大数据通过大型和小型组织便宜而轻松地访问。诸如 Hadoop 之类的软件基础设施使开发人员可以轻松利用数千个计算节点执行数据并行计算。加之能够从公共云提供商按需购买计算能力，这种发展大大加速了采用大数据挖掘方法。因此，大数据与公共云环境的耦合，由于公共云环境的商用硬件与商用操作系统的异构组合的特征，以及用于存储和计算数据的商用软件基础设施，带来了新的安全挑战。

随着大数据通过流媒体云技术的扩展，基于防火墙和半隔离网络上的为小规模静态数据而设计的传统安全机制是不够的。例如，异常检测的分析将产生太多异常值。同样，目前还不清楚如何改造现有的云基础设施。流数据需要安全和隐私解决方案的超快响应时间。

本文件详细列出了大数据服务提供商应该遵循的强化其基础设施的最佳做法。我们为大数据安全和隐私十大挑战的每个挑战提供十大最佳实践，共为我们提供了一百个最佳实践。每种最佳做法的结构如下：

- What? 什么是推荐的最佳做法
- Why? 为什么应该遵循 - 即通过遵循最佳做法来阻止安全/隐私威胁
- How? 怎么做以实施最佳实践

这些最佳做法文件是基于“扩展十大数据安全和隐私挑战”中概述的风险和威胁。

1.0 分布式编程框架中的安全计算

在分布式编程框架（如 Apache Hadoop）中，我们需要确保映射器的可信赖性；如果有不可信的映射器，需要确保安全数据的可信赖性。此外，我们需要防止信号从映射器输出泄漏。因此，应该采取以下措施来确保分布式编程框架中的安全计算。

1.1 初始信任建立

1.1.1 为什么？

为确保映射器的可信赖性。

1.1.2 怎么做？

向主服务器发送连接请求时，通过主验证员使用 Kerberos 身份验证或等效方式来建立初始信任。认证应是相互的，以确保主服务器的真实性。除了认证之外，还应考虑使用完整性考量机制，例如使用可信平台模块（TPM）的机制。

1.2 确保符合预定义的安全策略

1.2.1 为什么？

为保证在计算中达到高水平的安全性。

1.2.2 怎么做？

定期检查每个工作者的安全属性。例如，应该检查主节点的 hadoop-policy.xml 与工作节点安全策略的匹配性。

1.3 数据去身份化

1.3.1 为什么？

为防止数据标识与外部数据链接。这种链接可能会损害个人隐私。

1.3.2 怎么做？

所有个人身份信息（PII），如姓名、地址、社会保障号等都必须被屏蔽或从数据中删除。除了 PII 之外，还应注意准标识符的存在，准标识符是几乎可以唯一地标识数据主体的数据项（例如，邮政编码、出生日期和性别）。应采用 k 匿名技术[Swe02]等技术来减少重新识别风险。

1.4 通过预定义的安全策略授权访问文件

1.4.1 为什么？

确保映射器输入的完整性。

1.4.2 怎么做？

强制访问控制（MAC）。

1.5 确保不可信代码不能通过系统资源泄漏信息

1.5.1 为什么？

为隐私权。

1.5.2 怎么做？

强制访问控制（MAC）。授权访问管理工具 Sentry 使用 RBAC（基于角色的访问控制）加强 HBASE 安全。在 Apache Hadoop 中，块访问令牌被配置为确保只有授权用户能够访问存储在数据节点中的数据块。

1.6 防止通过输出泄漏信息

1.6.1 为什么？

确保安全和隐私。数据泄漏可能以许多方式发生（例如，不正确地使用加密），需要防止。调试消息、不受控制的输出流、日志功能和详细的错误页面信息有助于攻击者了解系统并形成攻击计划。

1.6.2 怎么做？

使用功能灵敏度来防止信息泄露。

阴影执行（例如，与外部网络的通信用于获得软件版本的更新）是需要考虑的另一个方面。

此外，所有数据都应在网络层上（中转）进行过滤，并符合 DLP 数据防泄漏策略。充分的数据去身份化也有助于减轻影响。

1.7 工作节点的维护

1.7.1 为什么？

确保工作节点的正常功能

1.7.2 怎么做？

经常检查工作节点是否发生故障并进行维修。确保它们配置正确

1.8 检测假节点

1.8.1 为什么？

避免在云和虚拟环境中的攻击。

1.8.2 怎么做？

构建一个框架来检测通过创建合法节点的快照引入的假节点

1.9 保护映射器

1.9.1 为什么？

避免产生不正确的汇总输出。

1.9.2 怎么做？

检测映射器由于恶意修改而返回的错误结果。

1.10 检查数据的更改副本

1.10.1 为什么？

避免在云和虚拟环境中的攻击。

1.10.2 怎么做？

检测重新引入的更改过副本的数据节点，并检查这些节点的真实性。单元数据的哈希机制和单元时间戳将加强完整性。

2.0 非关系型数据存储的最佳安全实践

非关系型数据存储，如 NoSQL 数据库，其中几乎没有嵌入健壮的安全特性。尽管应对 NoSQL 注入攻击的解决方案尚不完全成熟，但是，在考虑非关系型数据存储的安全因素时，仍宜整合下列最佳实践。

2.1 密码保护

2.2.1 为什么？

保护隐私。

2.1.2 怎么做？

- 用安全的哈希算法进行加密或哈希操作
- 使用加密哈希算法，如 SHA2（SHA-256 或以上）和 SHA3
- 进行哈希操作时，使用盐去对抗离线暴力攻击

2.2 静态数据加密保护数据安全

2.2.1 为什么？

在应用了薄弱认证技术和授权技术的情况下可靠地保护数据。

2.2.2 怎么做？

使用强加密方法，如 AES、 RSA 和 SHA-256。代码和加密密钥的存储必须和数据存储或仓库分开。加密密钥宜在安全的离线位置进行备份。

2.3 使用 TLS（传输层安全技术）来建立客户端与服务器之间的连接并进行集群节点间的通讯

2.3.1 为什么？

在传输过程中维持保密性。

2.3.2 怎么做？

实施 SSL / TLS 密封式连接。在理想情况下，每个节点都配备唯一的一对公/私密钥和数字证书，以便进行客户端认证。

2.4 提供对可插拔认证模块（PAM）的支持

2.4.1 为什么？

对于认证相关服务，宜让用户能够使用 PAM 库 API 编程至 PAM 接口。

（译者注：PAM（Pluggable Authentication Modules ）是由 Sun 提出的一种认证机制。它通过提供一些动态链接库和一套统一的 API，将系统提供的服务和该服务的认证方式分开，使得系统管理员可以灵活地根据需要给不同的服务配置不同的认证方式而无需更改服务程序，同时也便于向系统中添加新的认证手段。）

2.4.2 怎么做？

实现对 PAM 的支持。以互联网安全中心的基准进行加固，也可考虑在 OS（操作系统）层（如 SELinux）进行加固。

2.5 实现适当的日志记录机制和运行中日志分析

2.5.1 为什么？

暴露可能的攻击。

2.5.2 怎么做？

- 按照行业标准，如 NIST 日志管理指引 SP800-92 [KS06] 和 ISO27002 [ISO05]，实施日志记录机制。
- 使用 log4j 等恰当的日志记录机制。例如，ELK 堆栈（Elasticsearch、Logstash、Kibana）和 Splunk 可用于日志监控和运行中日志分析。

2.6 应用模糊方法进行安全测试

2.6.1 为什么？

暴露潜在的漏洞，这些漏洞通常是由 NoSQL 中的输入确认不足造成的，这些漏洞会让 HTTP

协议与客户端建立通信，例如跨站点脚本和注入。

2.6.2 怎么做？

- 提供无效、预期外或随机的输入并对其进行测试。典型的策略包含简单模糊化（采用完全随机的输入）和智能模糊化（根据对输入格式等的了解构筑输入数据）。

  OWASP(https://www.owasp.org/index.php/Fuzzing)

  MWIInfoSecurity

  (https://www. mwrinfosecurity.com/our-thinking/15-minute-guide-to-fuzzing/)

  等提供了相应的指南。模糊化宜在系统的不同层面来进行，包括协议层面、数据节点层面和应用程序层面等等。
- 使用工具（如 Sulley）进行模糊化操作。

2.7 确保强制使用合适的带有时间戳的数据标记技术

2.7.1 为什么？

数据源在数据管道输送时，避免数据未经授权的修改。

2.7.2 怎么做？

使用安全标记技术，这类技术会用一个特殊、不可变的安全字段标记每个送达的特定数据源的元组。

2.8 控制集群间通信

2.8.1 为什么？

确保通道安全。

2.8.2 怎么做？

确保每个节点在建立可信通信通道前确认其他参与节点的信任度级别。

2.9 确保数据复制的一致性

2.9.1 为什么？

正确处理节点故障。

2.9.2 怎么做？

使用智能哈希算法，并确保即使在节点故障期间，被复制的数据在不同节点间也是一致的。

2.10 使用中间件层来安全地封装底层 NoSQL 层

2.10.1 为什么？

具有一个虚拟安全层。

2.10.2 怎么做？

通过中间件在集合层面或列层面引入对象层面的安全，保留其瘦数据库层。

3.0 确保数据存储和事务日志的安全

由于其解决方案（如自动分层）不会记录数据的存储位置，因此大数据存储的管理需具安全性。为避免安全威胁，宜实施下列方法。

3.1 实施已签名消息摘要的互换

3.1.1 为什么？

解决可能出现的争议。

3.1.2 怎么做？

- 使用公共消息摘要（SHA-2 或强度更高的）为每个数字文件或文档提供数字标识，然后由发送者以数字方式签名，保持不可否认性；
- 完全相同的文档使用相同的消息摘要；
- 即使文档只有轻微改动也使用不同的消息摘要来区分。

3.2 确保对散列链或持续验证字典（PAD）的定期审计

3.2.1 为什么？

解决用户新鲜度和写入序列化问题。

3.2.2 怎么做？

使用红-黑树这类技术并跳过列表数据结构来实现 PAD。[AGT01]。

3.3 使用 SUNDR

3.3.1 为什么？

SUNDR（安全不受信任的数据存储库）是一种网络文件系统，用以在不受信任的服务器上安全地存储数据。

3.3.2 怎么做？

使用 SUNDR 来检测恶意服务器操作员或用户对任何未被授权的文件修改的尝试，SUNDR 也

能使用叉一致性有效检测可视文件修改中的完整性或一致性故障。

3.4 使用广播加密

3.4.1 为什么？

提高可扩展性。

3.4.2 怎么做？

使用广播加密方案[FN 93]，此时播出者针对正在收听某一广播频道的某种用户子集 S 加密一条消息。S 中的任一用户都可以使用他的私钥来解密广播。然而，即使 S 以外的所有用户都串通起来，也无法获得这一广播内容的任何信息。这样的系统据说是防串通的。播出者可选择任一子集 S 来加密。S 的一些成员仍然有可能通过使用分配给他们的私钥构建盗版解码器来协助进行盗版。也宜实施叛徒追查机制来确定这些恶意成员的身份，从而遏制盗版行为。

3.5 使用延迟撤销和密钥轮换

3.5.1 为什么？

提高可扩展性。

3.5.2 怎么做？

- 使用延迟撤销，即延迟重新加密直至文件得到更新，使撤销操作更便宜。
- 为了实现这一点，需针对撤销后才修改的所有文件创建一个新的文件组，然后在文件重新加密时将文件移到这个新的文件组。这么做会产生以下两个问题：
  - 问题：每次撤销后系统中的密钥数量增加
  - 解决方案：关联所涉文件组的密钥。
  - 问题：由于在连续撤销之后重新加密的文件集没有彼此包含，因此重新加密文件时，会越来越难确定这个文件应该移到哪个文件组。
  - 解决方案：使用密钥轮换。对密钥进行设置，让文件始终按最新文件组的密钥（重新）加密；由于密钥是相互关联的，则用户只需记住最新的密钥，并可在必要时推导出以前的密钥。

3.6 实现高概率的可恢复性证明（POR）或可证明性数据持有（PDP）方法

3.6.1 为什么？

假设某个客户将大量数据上传到了云端。她怎么知道云端的数据确实是可用的？云端可能已经删除或修改了部分数据。一个简单的办法是让云端把数据发送回来。然而，这种办法太贵了，而且也失去了将数据移到云端的意义。另一个简单的办法是让云端发送数据的散列。但是，一个散列一旦计算出来，就可以在之后的查询中重播。

3.6.2 怎么做？

Ateniese 等人[ABC + 07]引入了一种可证明性数据持有（PDP）模型，有了这种模型，如果客户在不受信任的服务器上存储了数据，就可以在不恢复数据的情况下验证服务器是否持有原始数据。该模型通过从服务器随机抽取区块集合的样本，生成数据持有的概率证明，从而大大提高了效率。客户保留一个常量的元数据，用以验证该证明。这种质疑/响应协议传输少量、常量的数据，使网络通信最小化。

Kaliski 和 Juels [JK07]开发了一种不太一样的密码式构建区块，称为可恢复性证明（POR）。POR 让客户能够确定是否可以从云中“恢复”文件。更精确地说，成功执行的 POR 向验证者确保，证明者提供了一种协议接口，验证者可以通过该协议接口完整地恢复指定文件。当然，证明者在成功加入 POR 后仍有可能拒绝发布文件。然而，在排除证明者行为变化的情况下，POR 提供了可能性最大的文件可恢复性保证。

3.7 实施基于策略的加密系统（PBES）

3.7.1 为什么？

避免串通攻击（假设用户不互换其私钥）。

3.7.2 怎么做？

- 对于以单调布尔表达式正式写入标准规范表格的基于凭证的方针，允许用户对消息加密。
- 提供加密，以便只有对符合方针的凭证集合有权限的用户才能成功解密该消息。

3.8 实施中介解密系统

3.8.1 为什么？

避免串通攻击（假设用户愿意在不交换解密内容的情况下互换私钥）。

3.8.2 怎么做？

- 在中介 RSA 密码式方法和系统中，发送者使用加密指数 e 和公共模数 n 来加密消息（m），接收者和受信任的权力方相互合作，以使用各自的解密指数组成部分 dU、dT 来解密这条加过密的消息。为了防止受信任的权力方在有接收者解密指数部分 dU 权限的情况下读取消息，接收者在将加密消息传递给权力方之前将其盲化。这种盲化通过使用因子 r<e>（其中 r 是个随机秘密数字）的模数 n 盲化操作来实现。接着受信任的权力方将其解密指数部分 dT 应用于消息，并将结果返回给接收者，接收者取消盲化，并应用其解密指数部分 dU 来恢复消息。

3.9 使用数字权限管理

3.9.1 为什么？

通过加密来控制受保护数据的获取权限还不够，因为在用户愿意交换解密内容的情况下，这种保护无法对抗串通攻击。

3.9.2 怎么做？

数字权限管理（DRM）方案是各种权限控制技术，这些技术用于限制对受版权保护的成果的使用。此类方案也能有效地在分布式系统环境中控制受保护数据的获取权限。为了防止受保护数据在未被授权的情况下被获取，某些框架会限制某种获取内容的方式。例如，受保护内容只能在特定设备或查看软件上打开，这类设备或软件安全地执行获取权利或方针（可能是在硬件层面）。同时，此类软件或设备的完整性可在必要时由云存储提供方等通过远程证明技术和（或）TPM（受信任平台模块）证实。

3.10 在不受信任的基础设施上搭建安全的云存储

3.10.1 为什么？

即使是不受信任的云服务提供方，数据所有者也要能以保密、完整性受保护的方式存储数据，同时保留服务的可用性、可靠性，高效恢复数据的能力以及灵活的数据共享。

3.10.2 怎么做？

实现这类安全云存储的解决方案之一，称为密码式云存储，以[KL10]进行设计，采用了对称加密、可搜索加密法[BW07，CJJ + 13]、基于属性的加密法[SW05]和存储证明（即可恢复性证明[JK07]和可证明性数据持有[ABC + 07]）。在密码式云存储中，数据所有者可以在存储加密数据的同时在本地保存加密密钥。此外，可以随时高效地验证数据完整性。因此，这一方案可以解决多项重大挑战，包括监管合规性、地理限制、传票、安全漏洞、电子发现以及数据的保存和销毁。

4.0 终端输入验证/过滤

我们需要确保数据源不是恶意的，如果是，我们应该过滤由该源生成的恶意输入。使用自带设备（BYOD）模型使得这个挑战变得更加严峻。以下推荐实现输入验证/过滤的最佳方法。

4.1 使用可信任证书

4.1.1 为什么？

确保对通信的信任和防止 Sybil 攻击。例如，单个实体伪装成多个身份。

（译者注：Sybil 攻击是指一个恶意的设备或结点违法地以多个身份出现，我们通常把这个设备或节点的这些多余的身份称为 Sybil 设备或结点。Sybil 攻击最初是由 Douceur 在点对点网络环境中提出的。他指出这种攻击将破坏分布式存储系统中的冗余机制。）

4.1.2 怎么做？

数字证书通过证书的主体命名证明公钥的所有权。这允许其他（依赖方）依赖于受认证的与公钥相对应的私钥制作的签名或认证。在这种信任关系模型中，CA 是受信任的第三方，被证书的主体（所有者）和依赖证书的一方信任。 CA 是许多公钥基础设施（PKI）方案的特性。存在许多开源代码实现的证书颁发软件。他们的共同之处是，提供必要的服务来进行发布、撤销和管理数字证书。一些开源实现方法是 DogTag，EJBCA，gnoMint，OpenCA，OpenSSL，r509，XCA。证书的有效性必须在使用前根据定期颁发的证书吊销清单（CRL）或 OCSP（在线证书状态协议）进行验证。

我们假设存在一个中央权力机构来确保将唯一证书分配给系统中的每个实体，则攻击者不能伪造多个身份。受信任证书是作为防御 Sybil 攻击唯一可靠的方法。

4.2 进行资源测试

4.2.1 为什么？

为了避免在大型企业中管理证书的缺点，但仍然对 Sybil 攻击进行最低限度的防御，而不是阻止该攻击。

4.2.2 怎么做？

资源测试[Dou02]是防止 Sybil 攻击的常用解决方案。它假设每个节点的计算资源是有限的。验证器检查每个身份是否具有与它相关联的单个物理设备一样多的资源[BS12]。

确定多个假身份拥有的资源是否比独立真实身份预期的少。资源的例子包括计算能力、存储能力和网络带宽。

4.3 使用统计相似性检测技术和异常值检测技术

4.3.1 为什么？

检测和过滤恶意输入。

4.3.2 怎么做？

- 生成表示“正常”行为的模型，例如高斯曲线，然后检测偏离正常输入的离群值，例如严重偏离高斯曲线的实体。
- 使用基于模型的方法，基于邻近的方法和基于角度的方法来检测和过滤恶意输入。

4.4 在中央采集系统中检测和过滤恶意输入

4.4.1 为什么？

中央采集系统是一个理想的场所，因为它具有比终端设备更多的计算资源。

4.4.2 怎么做？

- 使用统计和机器学习技术的异常值/异常检测。
- 基于签名的技术，即从历史数据生成恶意输入的签名，然后使用签名来识别恶意输入。
- 对于分布式系统，将在所有节点检测到的恶意输入转发到中央系统。

4.5 保护系统免受 Sybil 攻击

4.5.1 为什么？

检测并防止一个实体在系统中伪装成多个身份。

4.5.2 怎么做？

- 可信证书（第 4.1 节）
- 受信任的设备（第 4.7 节）
- 资源测试（第 4.2 节）

4.6 识别系统中看似合法的 ID 欺骗攻击

4.6.1 为什么？

检测并防止攻击者假定合法身份。

4.6.2 怎么做？

- 可信证书（第 4.1 节）
- 受信任的设备（第 4.7 节）
- 资源测试（第 4.2 节）

4.7 使用可信设备

4.7.1 为什么？

检测和防止 Sybil 攻击并防止对终端设备和运行在其上应用程序的危害。

4.7.2 怎么做？

系统中的每个实体被分配给具有嵌入式唯一设备标识符的终端设备，以一对一的方式绑定到用户标识（例如 IEEE 802.1AR 中定义的安全设备身份）。攻击者无法使用单个设备创建多个身份，而获取多个设备的成本将令人望而却步。

4.8 设计参数检查器检查传入参数

4.8.1 为什么？

检测和过滤恶意输入。

4.8.2 怎么做？

每个数据收集系统必须实现自己的检查器，检查输入参数中所需的属性和格式。

4.9 使用工具来管理终端设备

4.9.1 为什么？

防止来自受害终端设备和在设备上运行的应用程序的攻击者。

4.9.2 怎么做？

我们可以使用诸如 TPM 等工具来确保设备和应用程序的完整性。我们可以使用终端设备上访问控制和防病毒产品和/或基于主机的入侵检测系统等保护机制。每个设备中的信息流控制和强制访问控制对设备管理也很重要。我们还可以使用日志记录和监视工具来检测受害设备。

4.10 在端点使用防病毒和恶意软件保护系统

4.10.1 为什么？

防止来自受害终端设备和在设备上运行的应用程序的攻击者。

4.10.2 怎么做？

在端点安装防病毒和恶意软件防护系统。我们应该保持防病毒签名更新。

5.0 安全/合规性的实时监控

大数据由许多不同的设备和传感器生成。安全设备就是其中之一。安全/合规性的实时监控是一把双刃剑。一方面，大数据基础架构必须从安全的角度来监控：基础架构是否仍然安全，是否有潜在的攻击风险?另一方面,与不使用大数据的安全分析相比，大数据还可以用来提供更好的安全分析（例如:更少的误报、更细的粒度、更好的量化安全视图）。我们推荐以下的最佳实践。

5.1 使用大数据分析来检测集群的异常连接

5.1.1 为什么？

集群中只允许授权的连接，因为这是构成可信大数据环境的一部分。

5.1.2 怎么做？

使用解决方案，诸如 TLS/SSL , Kerberos, SESAME(用于多供应商环境的安全欧洲体系)、IPSEC 或 SSH 等，以便在集群中（如有必要）建立可信连接，防止未授权的连接。使用监控工具，如 SIEM 解决方案，来监控异常连接。举例来说,可以基于集群系统日志中记录的连接行为（例如，看到一个来自于“坏的互联网邻居”的连接）或者是告警，这样可以发现尝试建立未授权连接的行为。

5.2 挖掘日志事件

5.2.1 为什么？

确保大数据基础架构仍然与基础架构设定的风险接受范围相符。

5.2.2 怎么做？

- 从监控日志文件中挖掘安全事件用于安全监控，如 SIEM 工具；
- 应用其他算法或原理来挖掘事件，如机器学习，以获得潜在的新的安全见解。

5.3 实施前端系统

5.3.1 为什么？

前端系统不是新的安全手段。例如路由器、（应用层）防火墙、数据库访问防火墙。这些系统通常会解析请求(例如，基于语法签名、行为配置文件)，并停止错误请求。在大数据基础架构中，也将采用同样的原则来关注应用程序或数据请求（如 MapReduce 消息）。

5.3.2 怎么做？

部署多级前端系统。例如:网络上的路由器，允许/阻止应用程序的应用层防火墙，以及用于分析大数据典型请求（如 Hadoop 请求）的大数据前端专用系统。类似像 SDN(软件定义的网络)的技术也许有助于实现和部署。

5.4 考虑云层级的安全性

5.4.1 为什么？

大数据部署正在向云端转移。如果这种部署存在于（公有）云上，那么云将会成为大数据基础架构的一部分。因此，它需要受到保护，避免成为基础架构的致命伤。

5.4.2 怎么做？

- 实施 CSA 指南 V4 和 CSA 的其他最佳实践。
- 评估/要求云服务提供商符合 CSA STAR 标准。

5.5 考虑集群级的安全性

5.5.1 为什么？

大数据基础架构的安全性应该是应用多层次的方法，因为不同的组件构成了这种基础架构。集群就是其中之一。

5.5.2 怎么做？

采用适用于集群的最佳安全实践。例如：

- 在 Hadoop 集群中使用 Kerberos 或 SESAME 进行身份验证。
- 使用文件和目录权限确保 HDFS 安全 (Hadoop 分布式文件系统)。
- 使用访问控制列表提供接入控制（例如，基于角色、属性）。
- 使用强制访问控制来控制信息流。

注意，安全控制的实现 (很大程度上)也依赖于正在使用的集群分布。如果在有严格安全性要求的情况下(例如，正在使用保密性要求较高的数据)，考虑使用 Sqrrl 这样的解决方案，提供单元级的细粒度访问控制。

5.6 考虑应用级的安全性

5.6.1 为什么？

在过去的几年中，攻击者已经将他们的关注点从操作系统转移到应用程序和数据库。应用程序作为架构的一部分，它们也应该得到保护。

5.6.2 怎么做？

- 实施安全软件开发的最佳实践，如 OWASP(owasp.org)用于基于 web 的应用程序。
- 持续地、有计划地对应用程序执行漏洞评估和应用程序渗透测试。

5.7 考虑法律法规

5.7.1 为什么？

由于法律和法规的规定，人们不能监视和使用收集到的所有数据单元，因为这与隐私有关。这些规定带来不同的挑战，必将像世界各地其它的隐私保护条例一样，随着时间的推移将会得到解决。

5.7.2 怎么做？

数据生命周期中的每一个步骤，必须遵循法律法规（例如，隐私法律）：

- 采集
- 存储
- 传输
- 使用
- 销毁

注意，数据生命周期中每个步骤的位置(物理和虚拟)可能不一样！

5.8 考虑道德方面

5.8.1 为什么？

事实上拥有大数据并不一定意味着可以使用这些数据。在 (1)技术可行和(2)道德正确之间，总是有一条细微的界限。后者也受到法律法规、组织文化的影响，不一而足。

5.8.2 怎么做？

关于什么是道德上正确的，没有明确的指导方针。最低限度是考虑适用的隐私和法律规定。密切关注与你的组织、地区、商业等等道德有关的讨论。

5.9 监测逃避攻击

5.9.1 为什么？

这类攻击是为了规避或避免被发现。由于这将允许潜在的攻击，或未经授权的访问逃避检测，因此尽可能地减少这一类型是非常重要的。

5.9.2 怎么做？

随着逃避攻击不断演变，要抓住这些攻击并不是那么容易的。根据深度防御的概念，考虑采用不同的监控算法，就像机器学习挖掘数据，并寻找潜在的逃避监测的方法，但是基于签名/基于规则/基于异常检测/基于特定检测方案的方法除外。

5.10 监控数据中毒攻击

5.10.1 为什么？

这类攻击旨在伪造数据，让监控系统相信没有任何错误。或者，通过改造数据格式，导致监控系统崩溃，行为不当或错误解释数据。

5.10.2 怎么做？

- 考虑应用前端系统和行为测试方法来执行输入验证，处理数据，并尽可能地判断对错。
- 对数据源进行身份验证和维护日志也是非常重要的，不仅可以防止未经授权的数据注入，还可以建立问责制。
- 对监控系统也要加入异常行为检测，如长时间的 CPU 和内存负载峰值，磁盘的快速运行。

6.0 可扩展与可组合的隐私保护分析

从研究中发现，将用于分析的数据匿名化不足以确保用户隐私。因此，下面是可以确保大数据环境中的隐私保护的最佳技术。

我们推荐以下最佳实践

6.1 实施差分隐私

如挑战 1 中第 4 点中所讨论的一样。

6.1.1 为什么？

当研究人员设法通过连接两个或多个独立的无关联数据库来识别个人信息时，匿名化公共数据失败。

6.1.2 怎么做？

差分隐私[dwo06]旨在提供手段以达到最大化统计数据库查询精度的同时最小化识别其数据的机会。差分隐私是一种计算一个数据库采取多少匿名保护的数学概念。添加随机噪声是一种实现某些给定级别的差分隐私的方法。应确保对给定的数据库使用适当的机制。

6.2 实施同态加密

6.2.1 为什么？

云环境中数据明文存储可能会受损和导致隐私风险。

6.2.2 怎么做？

同态加密是允许特定类型的加密的一种形式。

对密文进行计算并获得加密结果，解密结果与明文执行的操作结果相匹配。

如使用”unpadded RSA”等技术实现部分同态加密系统。

6.3 维护软件架构

6.3.1 为什么？

软件维护不当是一个主要弱点，并可能轻易被威胁利用。

6.3.2 怎么做？

使用最新的安全解决方案维护软件基础设施。

6.4 使用职责分离原则

6.4.1 为什么？

这个原则对于强大的内部控制以及执行信息安全的最低特权政策是有帮助的。

6.4.2 怎么做？

实施严格执行的职责分离安全控制措施，以实现每个操作人员访问已指定的最小量的数据，并且只能执行一系列指定的操作。对于这些数据，应制订系统中用户行为审计活动。对系统上的用户行为制定审核机制。另外，为了可靠的执行分离，访问共享资源应谨慎监控或控制去检测和/或阻断隐蔽通道。

6.5 再识别技术的认识

6.5.1 为什么？

再识别是匿名的个人数据匹配其真正所有者的过程。为了保护消费者的隐私利益，个人标识符，如姓名和社会安全号码，往往是从包含敏感信息的数据库中移除的。然而，再识别会危及隐私。

6.5.2 怎么做？

匿名化，或去除标记化，消费者隐私权数据采取保护措施的同时，对于营销人员或数据挖掘公司来说这些有用的信息仍然是可用的。

应建立一个正式的隐私标准作为应对（数据）可能被再识别的方法。

6.6 隐私法规意识

6.6.1 为什么？

有越来越多的法律和法规要求在组织的管辖区域内（在美国等地区为医疗保健领域如 HIPAA 和 HITECH）进行某些形式的组织内部培训和意识（宣传）活动。这些法律法规意识有助于避免潜在的诉讼。

6.6.2 怎么做？

实施针对每个国家的隐私问题和适用法规的意识培训。

6.7 使用授权机制

6.7.1 为什么？

（数据或分析功能）访问权限仅赋予最小数量的有隐私（访问）需要的授权实体。

6.7.2 怎么做？

根据组织政策、相关隐私法规及必要时，包括数据主体的偏好，应适当地实施授权和访问控制机制。

6.8 静态数据加密

6.8.1 为什么？

许多对最终用户设备的威胁可能导致设备上存储的信息被未授权方访问。为了防止这样的信息披露，特别是个人可识别信息（PII）和其他敏感数据，需要保护数据的机密性。

6.8.2 怎么做？

限制存储在终端用户设备中敏感信息访问的主要安全控制措施是加密。

可进行细粒度的加密，如含有敏感信息的单个文件，或宽泛的，如加密所有存储的数据。在数据库中，主键用于索引和连接表，因此不能使用加密。因此，敏感数据，如个人识别信息，

不应作为主键使用。确保用于已给定数据集的加密算法是主流的和适当的。

6.9 隐私保护构成

6.9.1 为什么？

在一些现实用例中，如医疗健康，它往往必须从多个数据源去汇总和/或查询数据，如每个医院或研究机构中的电子医疗记录系统，这可能会导致隐私问题。

6.9.2 怎么做？

当多个数据库和/或服务连接时，通过检查和监视连接它们的功能来确保隐私信息泄漏是受控的。

6.10 设计与实施匿名的数据存储连接

6.10.1 为什么？

即使在每个数据库中的数据是匿名化的（例如，个人身份信息被适当地移除），在多个数据存储连接时，这（数据匿名化）往往是不充分的。典型的例子如，实际上，超过 80%的美国人可以通过出生日期、性别和邮政编码被唯一识别。

6.10.2 怎么做？

对每个数据存储，落实如 K-anonymity[swe02]，t-closeness [LLV07]和/或 I-diversity [MKG07]以及差分隐私等隐私概念。

7 大数据加密技术

大数据的到来引起了关于大规模、流式、增长的私有数据的密码学的复杂的新变革。业内普遍认为密码学是大数据和云必不可缺的技术之一。有了这种精确的保证，大众才有动力将数据和计算迁移到云上。加密技术是信息技术的进展预示的可靠工具，而不是繁重无用的负担。

在这一章节，我们将讨论密码学相关组织当前追寻的一些新的、令人振奋的研究方向。

7.1 加密数据的安全搜索和过滤

7.1.1 为什么？

考虑一个场景：一个收件的系统所接收的邮件均经过邮件所有者的公钥加密，但此邮件所有者不希望收到垃圾邮件。由于所有邮件都被公钥加密，因此系统无法区分一个合法邮件密文和垃圾邮件密文。现在最新的技术可向过滤器提供一个“令牌”，这样的过滤器能使用令牌以分析密文是否符合过滤要求。过滤器只能完成指定的分析功能，除此之外没有获取加密信息的其他任何属性的信息。

从技术上来说，一个搜索和过滤加密信息的密码协议不应从此加密数据上学习到超过相应谓词匹配之外的信息。最近研究已能成功将谓词本身隐藏起来，这样恶意学习实体就无法从明文或过滤标准上获取任何有意义的信息。

7.1.2 怎么做？

Boneh 和 Waters [bw07]构造了一个公钥系统，支持对比查询、子集查询、任何此类联合查询。在最近的一篇论文[CJJ + 13]中 Cash 等人提出了第一个次线性可检索的对称加密（SSE）协议，论文阐述了此协议的设计、分析和实现。这个 SSE 协议支持面向对称加密数据的联合检索和通用的布尔查询，并能扩展到非常大的数据集和任意的结构化数据（包括自由文本搜索）。

7.2 使用全同态加密以保证计算外包的安全

7.2.1 为什么？

考虑到客户希望将其所有敏感信息（如照片、医疗记录、账务记录等）放入云端的情况。她

可能将这些数据加密后上传，但当她需要云执行一些计算工作（如统计上月在电影上的花费），这时云将无能为力。由于数据经过全同态加密（FHE），云能对底层的明文进行计算，但所有的结果都是加密的，云无法获取明文或结果。

技术上来看，对于一个在加密数据上进行计算的加密协议，攻击者无法通过查看密文来识别相应的明文，即使其能获取正确和不正确明文的选择。注意，这是一个非常严格的要求，攻击者可能计算原始数据的加密的任意函数的加密。实际上一个威胁高的模型称为规则加密的密文安全选择在此方面没有一个有意义的对应部分——现仍在寻找类似这样的一个类型 [LMSV11]。

7.2.2 怎么做？

在 2009 年出现了一个突破性的成果[Gen09]，Gentry 提出了第一个全同态加密设计。这个设计允许计算底层明文的任何函数的加密。早期的结果[BGN05]创建了一个部分同态加密设计。Gentrys 最初的全同态加密（FHE）设计在多项式环上使用了一个理想的格式框架。虽然格子结构并非效率低下，但是 FHE 的计算负担还是远离实际可接受的范围。目前研究关注发现更简单的结构[vDGHV10，CMNT11]，效率改进[GHS12b，GHS12a]和部分同态设计[NLV11]以满足一些有趣的功能。

7.3 同态加密的设计与使用

7.3.1 为什么？

虽然完全同态加密就通用性而言是一种理想的解决方案，但是其计算开销实在是太高难以应用到实际生产中。

7.3.2 怎么做？

通过限制同态加密的功能，如限制为仅用加法同态操作或使用一些类似内积这样的基本统计计算，同态加密设计在保持实际适用性的同时可大大提高实用性。

7.4 使用关联加密技术实现不同加密数据的关联映射

7.4.1 为什么？

通常每个机构使用自己的加密密钥来保护其数据。实际业务（如医疗研究）中通常需要关联

或链接存储在不同机构的数据。这样的操作可使用同态加密来完成，但其需要所有机构使用同一加密密钥。此外这样的计算仍耗能极大。

7.4.2 怎么做？

关联加密[MR15]技术实际上能在使用不同密钥加密的数据中进行 ID、属性值等的匹配。因此每个数据所有者能使用不同的密钥来保护敏感信息。此外执行这种匹配或链接操作的实体（如云服务提供商）无法解密数据，数据的保密性得以保证。

7.5 协调认证与匿名

7.5.1 为什么？

通常认证和匿名的需求是对应的。然而有时是可以达到一个平衡点，在这个点上即可实现认证的保证，也可保留一定程度的匿名性。

对于一个确保来自一个可识别来源的数据的完整性的加密协议来说，最核心的要求是不可伪造非来自于所声称数据源的数据。当数据源只能识别为一个组的一部分时，这仍具有一定程度的匿名性。另外，在某些情况下（可能是合法的），一个可信的第三方能够将数据链接到一个准确的数据源。

7.5.2 怎么做？

组签名是一种可选的加密设计，这种设计允许个体实体能签署其数据但只在公开的组中可识别。仅有一个可信的第三方能确定个人身份。这个设计最初由 Chaum 和 Heyst 提出 [Cv91]，随后由 Boneh、Boyen 和 Shacham 开发出实例[BBS04]。

环签名首先由 Rivest、Shamir 和 Tauman 提出[RST01]，是一种组签名的设计，这种签名只有用户没有管理者。组签名适用于成员需要合作的情况，环签名适用于成员无需合作的情况。组签名和环签名都是签名者模糊的，但在环签名设计中没有预先安排用户组，没有设置、改变、删除组的步骤，也没有办法分配特定的密钥，没有办法撤销实际签名者的匿名性。唯一的假设：每个成员都已经与某个标准签名设计相联系。

为了生产一个环签名，实际的签名者声明一个包含可能签名者的任意集（实际的签名者自己

也包含在内），使用自己的私有密钥和其他人的公开密钥来计算签名。

7.6 实现基于身份的加密

7.6.1 为什么？

当实际部署基于公有密钥加密的系统时最大的一个问题就是密钥的管理，包括密钥的提供、更新、撤消。如所有通信节点必须配备由可信认证机构颁发的数字证书。此外在一个包含大量受限设备的资源（如物联网）的设置中充足密钥管理是不可行的。

7.6.2 怎么做？

在基于身份的系统（IBE）[Sha85]中，明文可被加密为一个给定的身份，并且希望只有一个实体及其身份能解密密文。其他的任何实体无论是单独还是联合在一起都无法破译密文。Boneh 和 Franklin 第一个提出使用使用椭圆曲线加密（IBE）[BF01]。之后便出现大量高效和安全的改进[Wat09, CW13, JR13]。

7.7 基于属性的加密和访问控制

7.7.1 为什么？

传统数据的访问控制由系统执行——操作系统、虚拟机，这些系统基于一些访问策略限制了对数据的访问。数据仍为明文。系统范式至少有两个问题：（1）系统可能被黑；（2）传输中的数据安全。

另一种方法是在基于访问策略的加密内核中保护数据本身。解密仅由策略允许的实体执行。虽然密钥可能也会被黑，但是这种方法中暴露的攻击面显著变小。虽然隐蔽的旁信道攻击[Per05, AKS07]可能提取秘密的密钥，但是这些攻击更难部署并且需要净化的环境。同时加密的数据也可能移动也可能静止，均将统一处理。

技术上来看，对于一个使用加密的密码强制访问控制方法，攻击者无法通过查看密文来识别相应的明文数据，即使他得到正确和错误明文的选择；即使所有的访问策略排除的成员联合起来也无法做到。

7.7.2 怎么做？

基于属性的加密（ABE）将此概念扩展到基于属性的访问控制。Sahai 和 Waters 第一个提出了 ABE（[SW05]），指出用户的凭证是由一个“属性”的字符串和关联这些属性的公式的访问控制谓词的集合组成。随后的工作[GPSW06]扩展了谓词的表现，并且提出两个互补形式的 ABE。在密钥策略 ABE 中，属性用以注释密文，使用这些属性的公式用以描述用户的私密密钥。在密文策略 ABE 中，属性用以描述用户的凭证，使用这些凭证的公式被附加至加密方的密文上。明确解决基于密文策略属性加密问题的第一个工作是由 Bethencourt，Sahai 和 Waters [BSW07]进行，随后由 Waters [Wat11]进行了改进。

7.8 使用易擦除的 RAM 实现隐私保护

7.8.1 为什么？

当数据存储到云上，即使数据被合适的加密，对于云服务提供商可见的数据访问方式仍可能会泄露敏感信息。

7.8.2 怎么做？

易擦除 RAM [SSS11]在每次访问后对内存进行“洗牌”。这样即使一个云服务提供商也无法得知在每次访问所操作的数据，因此能有效的隐藏访问模式。

7.9 保护隐私的公开审计

7.9.1 为什么？

将验证过程外包给第三方审计机构（TPA）是当前流行的趋势。同时检查协议预计将公开验证。这样的操作不应损害隐私性。

7.9.2 怎么做？

WWRL10]中提出了一个云存储中保护隐私的公共审计方案。基于与随机掩码集成的同态线性认证，所提出的设计能在 TPA 审计存储在不同层的服务器中的数据集时保护数据隐私。

7.10 使用融合加密实现重复数据删除

7.10.1 为什么？

存储在云端的数据通常是加密的。然后使用通常的加密设计，同一个明文可能会生成不同的密文。由于云服务提供商无法分辨两份数据是否相同，因此云存储中同一个数据可能有多个无必要保存在云端的副本。

7.10.2 怎么做？

融合加密设计（最初在[SGLM08]提出）是一种很好的去重方案。这种方案使用一个加密密钥，这个加密密钥是从待加密的明文中准确推导出的，如数据的加密哈希值，因此产生的密文变成相同的。以这样的方式可删除相同的数据。

8.0 细粒度访问控制

任何访问控制解决方案都有两个侧重点。第一个是限制非授权用户对数据和程序的访问，第二个是授予授权用户相应的访问权限。通常，不得违反前者，而设置数据库访问权限的目的是为了更好满足后者的要求。访问控制策略对用户如何有效利用数据库具有深远的影响。为了满足策略要求，粗粒度访问机制通常不得不限制本应共享的数据，而细粒度访问控制机制可以在不违反策略的情况下减少对数据的限制。在使用细粒度访问控制时，应遵循以下最佳实践。

8.1 正确选择所需的粒度级别

8.1.1 为什么？

通常来说，实施细粒度访问控制的成本在于增加了对数据标签和安全属性的管理的复杂性，而粗粒度访问控制的成本在于数据建模。例如，数据库视图可用于保护不支持“行级”访问控制、“列级”访问控制或“单元级”访问控制的数据库，随之而来的是增加了维护数据库视图的成本。

8.1.2 怎么做？

应评估数据访问的管理策略，特别关注其中对数据源的限制（例如，来自客户的数据只能被该客户或公司数据科学团队看到）或对模式元素的限制（例如，个人身份信息，类 SSN，只能被接受过隐私培训的员工看到）。通常可以采用“行级”访问控制来满足基于数据源的限制策略，采用“列级”访问控制以满足基于模式元素的限制策略。

有两个增加复杂性的实例：数据转换和弹性模式会增加访问控制的复杂性。弹性模式使“列级”访问控制难以管理，而数据转换后存储（例如，提取和聚合日志数据以形成特定实体属性的表）可能会使原先行存储和列存储的存储方案失效。在这些情况下，则需要使用“单元级”访问控制。

8.2 首选将可变元素归一化以降低更新成本，而更多不可变元素被非规范化以提高查询性能的方案

8.2.1 为什么？

最新数据库技术的发展为形式多样的非规范化数据建模打开了大门。对于不变的数据元素，与多连接模型相比，非规范化模型可以提供更高的并发性和更低的延迟。由于多个来源、多种类型的数据汇聚在一起，因此当使用非规范化数据模型时，细粒度访问控制变得尤为重要。

8.2.2 怎么做？

细粒度访问控制的核心是维护数据标签。当对数据进行非规范化处理时，应维护数据访问策略所引用的信息源的来源信息。例如，当数据源限制了哪些人可以看到该数据，则应在标签中维护该源信息以及来自该源的必要字段。

8.3 跟踪保密需求

8.3.1 为什么？

建立可扩展细粒度访问控制机制的一个重要方面是根据调整的保密策略对数据进行标记。保密要求可能会随时间而变化，因此，对细粒度访问控制机制进行调整以满足不断变化的策略需求尤为重要。

8.3.2 怎么做？

标记方案应使用不随时间变化而变化的保密策略元素对数据进行标记，而将保密策略中的可变元素留待查询时检查。追踪数据采集时所使用的数据标记策略，以减少查询时对策略评估设定的各种假设。

8.4 维护访问标记

8.4.1 为什么？

对来源复杂的数据进行策略决策时需要使用访问标记。维护访问标记涉及大量对数据源进行追踪的工作。

8.4.2 怎么做？

在尽可能上游的位置对数据进行标记。在所有数据传播过程中追踪数据访问策略中引用的标记。使用支持布尔逻辑和/或标签集的访问控制机制，以简化数据聚合阶段的标签跟踪工作。

8.5 跟踪管理数据

8.5.1 为什么？

用户的角色和权限频繁变更，因此其不应被假定为常量。由于更改此类信息无异于将数据重新纳入系统，因此选择适当的数据库策略十分重要。

8.5.2 怎么做？

使用具有访问控制和审计功能的数据库来跟踪管理数据。在将其用作决策依据时，请务必确保管理数据与大数据架构之间连接的安全。

8.6 使用标准 SSO 机制

8.6.1 为什么？

细粒度访问控制机制的一大优点是在不大幅增加管理成本的前提下扩大数据共享。单点登录解决方案（SSO）将用户身份验证管理转移到企业范围内的其它系统或公共系统上，从而进一步降低大型用户群的管理负担。

8.6.2 怎么做？

由高级 SSO 系统（如 LDAP、Active Directory、OAuth、OpenId 等）实施认证。

8.7 适当的联合授权空间

8.7.1 为什么？

通过将不同数据集进行关联分析，可以拓展大数据概念的内涵和外延。然而，由于继承了多个数据提供方的安全策略，使得大数据保护策略变得非常复杂。联合授权空间通过更精确地跟踪数据标签并允许数据提供方对其数据访问实施一定控制，从而支持更加模块化的策略管理方法。

8.7.2 怎么做？

标记方案应避免不同权限管理体系下的标签之间可能产生的歧义，并充分利用与高级 SSO 系统（如 LDAP、Active Directory、OAuth 和 OpenId）相结合的用户属性。

8.8 正确实现保密要求

8.8.1 为什么？

检验是否正确实现保密要求需要构建数据身份、用户身份、用户目的以及环境因素等参数作为依据。在细粒度访问控制系统中，出于性能原因，这些元素分布在整个大数据架构中，通过正确使用工具可以为检验保密要求提供简明的论据。

8.8.2 怎么做？

将保密要求分解为特定数据保密要求、特定用户保密要求和特定应用程序保密要求。使用一致的标记策略来确保数据相关的所有必要细粒度信息都能在标签中找到。用户属性应引用权威来源的信息。确保任何保密目的或应用程序限制均由应用程序直接实现并经过审计，或通过应用程序认证间接实现。

8.9 逻辑过滤器的应用层实现

8.9.1 为什么？

通过细粒度访问控制机制，用户可借助应用程序获取数据从而实现不同目标。任何防止“答非所问”的策略都在某种程度上依赖应用程序正确分离数据的不同用途。例如，单个医疗保健提供者可能基于诊断特定患者的目的访问数据，也可能基于表征患者群体的目的访问数据。假设应用程序可用于实现这两个用途，那么该应用程序应确保在实现前者时获得的个人身份信息和个人健康信息不会泄漏给后者。

8.9.2 怎么做？

确保应用程序在不清除会话数据的情况下不会服务于不同用途。在可能的情况下遵循数据库的细粒度访问控制机制。

8.10 访问限制的跟踪协议

8.10.1 为什么？

数据隐私策略通常很复杂，难以实现 100％的准确性。因此，明确系统所采取的策略、审视其随着时间推移调整这些策略的能力则显得十分重要。

8.10.2 怎么做？

用于跟踪访问限制的协议有两种形式：对策略进行编码的协议和审核策略实例化的协议。记录策略决策过程并对其进行关联审计，可以获得诸如哪些数据正在被用户访问、哪些数据用户尝试访问却被拒绝、用户如何以未经授权的方式尝试访问系统等关键信息。通过审计分析，可对策略进行精细化调整，从而进一步加强隐私保护和提升共享水平。

对策略进行编码时，选择标准语言（如 XACML）有助于获得来自工具构建者社区的帮助，从而在对策略集进行可视化、编辑和推理时得到便利。

9.0 细粒度的审计

执行细粒度的审计是最好的做法，因为有可能由于缺少实时安全监测系统的有效报警而错过对攻击行为的发现，因此执行细粒度审计是最好的做法，最佳实践如下。

9.1 创建攻击的审计全景图

9.1.1 为什么？

攻击可能由不同的阶段组成(例如:进行侦测扫描，脆弱性攻击)，重要的是把所有的碎片化信息收集起来拼成一个整体。只有这样，才可以构建全景图。这一过程对于建立连贯一致的审计追踪是非常重要的:发生了什么？什么时候发生？如何发生？谁发起的或什么导致的？为什么会发生等。

9.1.2 怎么做？

- 在大型数据基础设施中启用审计功能。
- 根据基础设施组件的特性(如来自路由器、应用程序、操作系统、数据库等)的日志信息，选择相关的功能。
- 使用 SIEM、审计和取证工具来处理收集的审计信息。

9.2 信息的全面性

9.2.1 为什么？

为了提供全面的审计追踪，建立审计追踪的所有相关信息必须是可用的。因此，信息的全面性是关键。

9.2.2 怎么做？

- 评估日志文件、操作系统设置和配置文件、数据库配置等审计信息可能的相关性和可用性。
- 大数据基础设施启动和设置必要审计功能，如路由器、操作系统、Hadoop 和应用程序等审计信息必须预先收集。而其他审计信息可以设置为在后期收集。
- 可使用 SIEM 或审计工具收集和处理审计数据。

9.3 及时获取审计信息

9.3.1 为什么？

在一个攻击事件中，时间是最重要的方面。不仅要确定攻击发生的时间，还要及时获取事件过程所需要的审计信息。这与最佳实践 9.2 有关联。

9.3.2 怎么做？

如最佳实践 9.2 中所描述，预先设置审计信息是关键。不仅可以保证获得信息的完整性，而且能保证及时获取信息。

9.4 信息的完整性

9.4.1 为什么？

没有完整性保证，就没有唯一真实的版本。审计信息如果不可信也就变得无用。

9.4.2 怎么做？

- 实现完整性的控制措施，如安全哈希。SHA－1、SHA－224、SHA－256 和 SHA－512 等可用。
- 确保在收集、处理、使用和存储等数据生命周期中审计信息的完整性。

9.5 信息的保密性

9.5.1 为什么？

完整性的目标是为了保证审计信息的正确性，保密性针对不是每个人都需要访问审计信息的事实。这尤为重要，因为审计信息中包括潜在攻击者及其所使用方法等数据。因此，只有授权的人员(通常是审计员和取证研究员)才能访问这些信息。

9.5.2 怎么做？

- 确保审计信息单独存储(参见最佳实践 9.9)。
- 确保审计信息只能被授权人员访问(参见最佳实践 9.6)。
- 考虑使用可行和适用的加密技术来加密审计信息。

9.6 授权

9.6.1 为什么？

审计信息包含什么时间、谁访问了什么系统或使用了什么数据，所以这些信息对调查至关重要。因此为了避免攻击者通过删除他或她的攻击行为而篡改审计信息，必须严格控制对审计信息的访问。

9.6.2 怎么做？

- 当建立身份和访问管理规程时,要考虑审计信息的访问行为。例如，定义一个单独的审计员角色，只有该角色可以访问审计信息。系统中的所有其他角色无法访问审计信息。
- 定期监视这个角色的使用情况，特别是对于异常或尝试访问行为。
- 根据审计信息创建攻击全景图。

9.7 启用所有必需的日志

9.7.1 为什么？

构建的审计视图好坏取决于构成审计视图所收集的数据。这些数据的大部分来自日志文件(例如:网络，操作系统，数据库，应用程序。因此，根据审计的需要启用日志是关键。

9.7.2 怎么做？

与此相关的最佳实践 9.2，描述了审计所需要的信息。基于这些信息，评估大数据基础设施组件的日志功能以及启动不同的记录特征。

9.8 工具的使用

9.8.1 为什么？

大量的信息(特别是现在有大数据)通过手动处理是不够的。工具，像 SIEM 工具，是收集和处理数据所必须的。

9.8.2 怎么做？

- 使用诸如 SIEM 工具等可利用的工具来处理从日志中收集到的信息。

9.9 大数据和审计数据的分离

9.9.1 为什么？

由于审计数据中包含了大数据基础设施中发生事件的信息，因此建议将此数据与“常规”大数据分离。这也被称为职责分离。

9.9.2 怎么做？

- 审计系统要建立在有别于大数据基础设施的其他基础设施上，例如不同的网段或云。
- 确保只有审计师角色可以访问审计系统和审计数据。
- 监视审计系统。

9.10 建立审计层/协调者

9.10.1 为什么？

由于大型数据基础设施包含不同的组件，每个组件都有自己日志功能和日志格式。对于一个审计团队来说，要了解这些不同的、异常复杂的日志特征和日志格式是非常困难的，甚至是不可能的。审计协调者可以充当中间人，他可以为审计员抽象出底层的技术细节。审计员通过可查询的接口与审计协调者通信，审计协调者负责正确的收集日志文件，通过复杂的技术处理进行规范化处理，向审计员提供可理解的审计术语。

9.10.2 怎么做？

- 基于审计协调者的复杂性，因此不建议自己创建，可以评估第三方或开源的解决方案，如 ElasticSearch（一个基于 Lucence 的检索服务器，用于分布式全文检索）。
- 一些 SIEM /日志处理服务供应商也提供这方面的解决方案(如 SplunkSumologic 和 Loggly 等日志管理工具)。

10.0 数据来源

随着大数据应用程序中大型来源图形的来源元数据的增加，以下是数据来源安全性的最佳方法。

10.1 基础设施认证

10.1.1 为什么？

认证是主要的验证对方身份的方法，即只允许合法的用户访问来源。授权意味着授予许可以收集和访问来源数据的权限。如果没有认证，所有用户都可以直接访问数据，包括恶意的用户有可能滥用所访问的数据。例如，攻击者很可能会向公众发布涉及一些敏感信息的来源信息或数据本身。

10.1.2 怎么做？

认证协议是对允许使用秘密信息而辨识的主体之间的一系列消息交换。认证协议可以根据所采用的主要密码方式进行分类（即对称密钥或公钥）。它还可以区分为那些使用受信任的第三方，以及那些纯粹在两个愿意实现某种认证方式的通信主体之间进行的操作。如果需要授权的节点数量巨大（例如使用无线传感器节点收集来源数据），则可以使用公钥基础设施来授权每个使用者，并且可以向每个使用者发送一个对称密钥。授权节点可以使用对称密钥进行通信，因为对称密钥的开销远远小于公钥方法。由于没有对称密钥，未经授权的用户不能发送信息。这样，我们可以建立有效的认证基础设施来授权有效的用户。

10.2 周期性状态更新

10.2.1 为什么？

无线技术的进步增加了移动设备的数量，导致越来越多的无线设备被用于收集、传输和存储数据。由于无线设备受电池限制，其使用寿命有限，因此当节点电池用尽时，无法激活无线节点传输数据。无线环境中还存在一些恶意节点、窃听节点和懒惰节点。恶意节点是那些活跃的主动攻击节点。当恶意节点接收到从数据所有者发来的数据时，它主动篡改数据，或丢弃数据包，并将伪造的数据发送到下一个中继节点。当数据包发送到窃听节点时，这种节点不修改数据，而是收集数据，尽可能多地获取敏感信息。懒惰节点既不会污染数据，也不会窃听数据，

而是鉴于数据传输的能量消耗选择不存储转发数据。对信息的收集和传输的操作也将作为源数据的一部分而被记录。为了正确收集和传送数据，应定期更新源数据记录的状态。需要删除无效节点，并通过周期性地更新每个节点的状态让有效节点来传输数据。

10.2.2 怎么做？

可以将信任和信誉引入无线网络来解决上述问题。信誉系统提供一种机制为整个系统中涉及的每个实体生成一个封装信誉的度量。每个节点具有基于历史数据的先验信任值（如果节点没有历史数据，则我们简单地初始化先验（即先前）信任分数并将其分配给固定值）。通过组合先验信誉评分和新评级来计算后验（即，更新的）信誉评分。新的评估值基于每个节点的性能和行为。例如，如果节点对某些敏感信息有一些恶意行为或窃听，则系统可以将一些预定义的负值作为评级分配给节点。此外，系统可以将具有正常行为的节点的分配预定义正值评级值，而将惰性行为简单地分配为零评级值。一个节点也可以基于它们的相互作用给予另一个节点的信任值。该信任信息可以作为源数据记录的一部分存储，该信息可以周期性地更新以便反映当前最新状态。一些预定义的阈值应由系统预先定义以供选择。如果节点的信任值低于预先定义的阈值，则节点被识别为无效，不用于发送或收集信息。

10.3 完整性校验

10.3.1 为什么？

数据完整性作为安全方面的一个重要问题，是确保数据在整个生命周期中的准确性和一致性。数据完整性是数据库的重要特征，以确保数据库中包含的数据准确可靠。在现实环境中，源数据总是存储在个人电脑或远程数据库中心。可能会有一些无法预估的风险，使得意外丢失了一部分原始数据，例如电脑从高处摔下后造成硬盘驱动器损坏。我们应该能够检测和纠正数据，确保存储数据的完整性。在攻击者活动的环境中，也可能有对手修改数据库中原始数据文件的某些比特位。需要检测数据是否被篡改。此外，完整性是实现源数据被信赖的最重要因素之一。

10.3.2 怎么做？

校验和被认为是代价最低的一种错误检测方法。由于校验和本质上是一种压缩形式，错误屏蔽可能会发生。利用算术校验和在串行传输中进行错误检测被作为循环冗余校验（CRC）的替代。由于在硬件中可以简单实现，CRC 作为循环码和非密码学的散列技术可以处理突发错误检测，因此是一种最流行的错误检测方法之一。里德 - 所罗门（RS）代码是一类特殊的线性非二进制块

代码，能够纠正错误和掉包。由于 RS 码是最大距离可分离（MDS）码，意味着没有其他编码方案可以从较少的接收码中恢复丢失的源数据符号，理论上可实现对数据包丢失错误的保护。因此，MDS 代码用于纯错误检测或同时进行纠错和检测。

保持数据和来源信息的完整性的其他有效方法是，使用可以防止数据被对手伪造的数字签名。数字签名是公钥加密的最常见应用，执行过程如下：数据所有者可以获取公私钥对，并将公钥发送给数据验证者。公私钥对可以由其自身或可信任的第三方生成。数据发送者可以使用私钥对消息进行签名，并使用加密方式将签名后的消息和签名发送给验证者。当验证者接收到消息和签名时，它可以使用发件人的公钥来检查消息的完整性。有效的数字签名表示数据是由声称的发送者产生的，并且该消息在传输期间未被更改。虽然数字签名提供了一种保持数据及其来源完整性的方式，但是当数据量变大时，验证成本也还是可接受的。新的高效的数字签名方案预计将开发出来，用于大数据场景中数据的完整性保护。

10.4 一致性校验

10.4.1 为什么？

将数据自身与其来源相关信息分开就会引入诸如数据与数据来源信息之间的不一致等问题。数据来源相关的信息也也应保存在存储系统中。如果数据来源相关的信息与相应的数据不完全匹配，则我们不能放心地使用数据来源相关的信息。应该设计算法，以确保这两类数据之间的一致性。此外，数据来源相关的信息保留了数据对象的进程历史记录。关于数据的历史信息可以以链的形式构成起源链。一个文件的来源链是一个非空的且时间有序的原始记录序列。由于存在数据库中的大量原始数据，原始数据来源链应很好的组织使得数据来源相关的信息记录一致。否则，如果不能保持一致性属性，历史来源信息不能被放心使用。为了证明有效的出处数据的可用性，应该实现原始数据与其来源的一致性以及目前来源与历史来源的一致性。

10.4.2 怎么做？

为了保证来源和数据之间的一致性，哈希函数和散列表可以用来解决问题。这就是说，我们可以使用哈希函数映射键来定位数据。由于来源数据量巨大，数据结构复杂，我们最好不直接使用来源数据。在原始数据存储到数据库中之前，我们首先利用散列函数根据所选择的数据块生成固定大小的位串。之后，我们使用来源信息和散列来源数据构建哈希表。然后，我们稍后可以使用散列表来检查来源和数据之间的一致性。用户可以首先调用散列函数对所检索的原始

数据进行哈希处理，然后使用原始信息通过使用哈希表来获取先前的散列原始数据。最后，我们比较两个哈希值。如果两个散列值相匹配，则表示来源数据和从数据库中获取的数据是一致的。

作为来源链的基本组成部分，原始记录表示对原始数据执行的一个或多个操作的顺序。为了实现来源记录的一致性，将新修改的来源记录和原始记录历史链的密码学散列作为输入。结果是作为新更新的来源记录的一个组成部分输出。来源链的一致性可以通过在编辑会话开始时验证当前来源链是否与其对应的哈希值相匹配来实现。由于在大数据应用程序中生成大量来源数据，因此我们需要高效的方法来保持源，数据和来源链本身之间的一致性。

10.5 加密

10.5.1 为什么？

随着云计算技术的不断发展，现在我们经常将大量的科学来源数据外包给云端进行存储/计算。当数据提供者将数据以明文形式存储到云服务器时，传输的数据能被对手轻易地窃听。此外，云服务器始终被视为无法完全信任的第三方服务器。我们应该使用一些机制来保证源数据不能泄露给任何不信任方。数据所有者可以利用一些方案将大部分计算任务委托给不受信任的云服务器，而不会泄露底层数据内容。

10.5.2 怎么做？

在传输期间保持数据安全的一种现有方法是通过使用受信任方授权的对称密钥将原始数据转换成密文形式。在将源数据发送到不可信服务器之前，数据所有者使用私钥对数据进行加密。发送者不信任的人不拥有私钥，也就无法解密密文以获取原始数据。只有授权方才能使用私钥解密密文。我们还可以使用加密方法将计算任务外包给不可信服务器。在将高强度计算任务发送给云服务器之前，数据所有者可以通过使用轻量级加密技术首先“盲化”原始数据。之后，数据所有者可以将加密的数据外包给云服务器，以处理高强度的计算任务。由于云服务器没有安全密钥，因此该服务器无法透露原始来源数据。当计算任务完成时，数据所有者可以使用秘密密钥来恢复云服务器处理的数据。通过使用加密，数据所有者可以安全地实现计算任务的外包和数据存储在不可信服务器。对源数据进行细粒度的访问控制也很重要。这可以通过以下方式确保获得最佳效果。

10.6 访问控制

10.6.1 为什么？

由于计算机系统有着多样化的应用并服务于各类授权用户，因此人们对于数据安全威胁和保护的意识不断提升，特别是云计算环境。云计算的其中一个优势是云服务提供商可以在不同用户间分享数据。然而，仅有授权用户才有权限去下载分享的数据。

此外，储存在云服务器上的数据源的体积通常都很大，因此在大多数情况下，用户（数据所有者）希望能限制其他用户对数据的访问，以减少不必要的信息检索和数据交换，从而在很大程度上降低计算资源的使用。

由于有各类授权用户能访问储存在云服务器上的数据，因此需要建立一个有效的访问控制机制来确保授权用户能且仅能访问他在整个数据库中被授权访问的部分。

从另一个方面来说，源数据也可能包含了隐私信息，如用户个人档案，浏览记录等。

缺乏恰当的访问控制可能导致对手会暴力获取和使用数据源或者数据本身。

而且，对手可能会把敏感数据公布于众以损害数据拥有者的利益。如何限制授权用户只能去访问特定的数据或者信息源是目前安全工程师和科学家所面临的重要安全问题。

10.6.2 怎么做？

访问控制的目的是限制合法用户的一系列可能的行为和操作，以防止不合法的访问并控制隐私泄露的风险。访问控制限制了一个用户的直接行为，及用户被允许的程序执行。这种用于限制授权用户访问特定数据访问控制技术通常被称为基于角色的访问控制（RBAC），其机制是数据所有者将数据源存放于一个加密的表中，并授权给特定角色的用户。这些特定角色的用户根据访问控制的策略可以对数据进行访问，而在访问控制策略中未被授权的用户在任何情况下都无法浏览数据。基于属性的加密技术，是为了达到对原数据的精细化访问控制。

有两种基于属性的加密访问控制方法：基于密钥策略的属性加密（KP-ABE）和基于密文策略的属性加密（CP-ABE）。其中，基于密文策略的属性加密方法被认为是对云计算环境更合适的框架。在这个框架中，系统中不同的用户有根据他们固有特性所创建的相应属性集。授权用户都被分配了与不同的属性集相关联的私钥。数据的所有者根据访问框架对数据源进行加密，并将加密的数据分配给特权用户。只有用户的私钥属性集符合密文的访问框架才能成功解密数据，这样未授权的用户就无法解密了。

尽管用基于属性的加密框架能够轻松地达成访问控制的功能，但是计算和通信的成本还是很大负担，因此应用到大数据场景时应考虑这个因素。

更有效的框架应该被提出，以达到数据源的访问控制机制。

10.7 满足数据的独立持久性

10.7.1 为什么？

独立性指的是当更新两个相临近的数据源时，用户无法分辨出是哪个数据源。例如，两个衍生数据的源，如模拟结果，是无法分辨两个衍生数据之间的变化。有些时候一个用户只能访问数据源中某些片段，而不能访问其他的片段。如果没有满足数据的独立性，当用户得到两块数据源，他不能分辨出二者的差异。一些数据源可能涉及到一些数据所有者不想让消费者看到的敏感信息。在这种情况下，数据消费者很难分辨出哪个是他/她仅能访问的部分数据源

记录源也应该具备独立性，因为一些用户也只能访问来源链的一部分。个人记录源包含了一些创建者和修改者的隐私信息。由于用户本身受限的权限，一些用户只能访问来源链中的部分记录源。保证来源链中不同数据块的独立性对数据拥有者很重要。

例如一个公司里，爱丽丝能访问一些修饰符记录源但是不能访问其他具有拥有者隐私数据源。鲍勃也是一样。如果不满足独立性条件，爱丽丝和鲍勃能用他们自己的访问记录源来推断其他的信息。一些数据所有者的隐私信息可能会暴露给爱丽丝和鲍勃。因此应设计一些机制去同时达到数据记录源的独立性，及历史记录和当前信息的独立性。

10.7.2 怎么做？

对称密钥和分布式哈希表技术能用于保持不同数据源的独立性。加密不同数据源的对称密钥都是随机和独立选择的。由于独立的对称密钥的差异性，因此加密的数据源也是独立的。而且，同一数据源的不同片段不应该被保存在一起，因为他们都有相同的格式。我们可以使用分布式哈希表去分布数据源的不同片段。

为了方便理解，我们假设密钥对应某特定值，分布式哈希表负责存储一个密钥及相对应的值，例如一个文件的名字及其内容。数据源片段能以分布式储存于分布式哈希表中。但有需求时，索引标签能用于定位存储于数据库中的数据，并使用相应的对称密钥解密数据源。有且仅有那些有索引标签和相应对称密钥的人才能获取数据源片段。但是其他的数据依然不能被解密，因为他/她没有其他片段的对称密钥，并且不能定位储藏数据的位置。此外，为了达到不同记录源

的独立性可取消哈希功能。单一的记录源应该被哈希加密，然后不同哈希处理的记录源就能创建出一条链。我们可以通过检查哈希处理过的内容简单地判断某个记录源是否被修改了。被修改过的记录源仅会影响到相应的哈希组件，而无法影响到哈希链上的其他组件。

这就是说，不同部分的记录源就能达到独立的持久性。通过使用哈希方程和分布式哈希表，数据片段和记录源片段的独立性也可在系统中达成。

10.8 动态细粒度访问控制

10.8.1 为什么？

在第 10.6 节中，我们介绍了访问控制，即只允许授权用户访问某些数据。这里，我们引入细粒度的数据访问控制，通过使用基于属性的加密方式，使得某个用户（数据消费者）拥有对某些类型的数据的访问权限。在大多实际情况下，分配给用户的权限总是随时间和地点的变化而变化。例如，外科医生只有在工作时间内诊断患者时，才具有权限访问患者的健康记录。而在工作以外的时间，医生不可以访问患者健康记录的任何内容是合情合理的。在这种情况下，外科医生的特权即是随时间而变化。我们也可以通过另一个例子看到：员工处理涉及公司敏感信息的财务报表时，他/她应该只能在公司内访问这些数据。在公司以外的任何地点，他/她便无法访问财务报表的任何信息。此外，分配给员工的访问权限通常因位置而异，这意味着他/她只能在某些特定的场合才会有相应的访问权限。也就是说，用户的特殊权限是动态的，随着周围的环境因素而变化。考虑到动态属性，设计一个细粒度的访问控制方案是至关重要的。

10.8.2 怎么做？

通过使用基于属性的加密方法，可以将细粒度的访问控制应用到源数据的加密中。为了达到动态属性，我们可以将动态属性和属性加权引入到基于属性的加密方法中。动态属性可以描述为频繁变化的属性，如位置坐标。其他属性可以被认为是加权属性，这些属性根据其在访问控制系统中定义的重要性具有不同的权重。系统中的每个用户都拥有一组加权属性集，数据所有者为拥有属性集的用户加密数据。但是用户的私钥具有某种特定加权访问框架。具有一组加权属性的密文必须满足特定的加权访问框架方可被解密。同时，可以增加或减少属性的权重以反映动态属性。例如，公司的职位在实际工作中可以被分为不同的类型，如销售员工和销售经理。销售经理比销售员工拥有更多的权限来访问公司的数据，即表明出“销售经理”的权重比“销售员工”要高。当“位置”属性的权重等于或大于销售经理的权重时，密文也可以进行解密。

当“位置”属性的权重低于销售经理的权重时，是无法进行解密的，比如销售员工。当销售员工升级为销售经理时，他/她所具有的权重属性就会被更新，因此可以进行解密操作。通过将动态属性和加权属性的概念引入到基于属性的访问控制框架中，可以较容易地实现动态细粒度访问控制。

10.9 可扩展的细粒度访问控制

10.9.1 为什么？

由于我们都处于大数据的时代，数十亿的源数据将在数据库中进行存储和交互。这将允许数据消费者根据由数据所有者设计的访问策略访问各种类型的源数据。然而，访问策略应该是可扩展的，以满足不断增长的源数据和用户群体的数量。此外，我们需要一个强大的方案来实现大量源数据细粒度级别的控制。例如，数据所有者根据源数据中包含的源信息设计了访问策略，可扩展机制应允许通过向访问策略添加/删除源信息来改变访问架构，以实现可扩展的细粒度访问控制。如果访问策略无法实现可扩展性，那么由于无法简单更改访问策略，会导致数据库难以管理这种大规模的源数据。此外，为了访问大量数据，可以将用户添加到系统中或从系统中删除。系统应该是可扩展的，以便容纳任意数量的用户。为了控制用户的特权，应该实现细粒度的访问控制，并且，为源数据设计可扩展的细粒度访问控制方案是非常有必要的。

10.9.2 怎么做？

我们在上一节中介绍，为了达到细粒度的访问控制可以使用基于属性的加密方法来实现。然而，为加密的数据设计高效且可扩展的细粒度访问控制方案是比较困难的。为了解决这个问题，我们可以引入一个利用“代理重加密技术”的半可信代理方案。它允许中间代理来解密源数据，并通过使用新的访问架构重新加密这些数据。由于代理是“Honest-but-curious”的，这意味着它将尝试基于自己的能力发现尽可能多的隐私信息，我们不能允许代理完全解密源数据。用户希望通过策略控制来利用代理生成部分密文。当密文中的访问架构的属性需要被数据所有者修改时，新的策略会被重新加载到代理中。代理首先使用先前的部分属性集解密部分密文，然后使用新的策略加密部分解密的密文。因为代理没有设置完整的属性集，所以它只能解密部分数据，不能得到整个源数据。通过使用代理重加密技术，我们可以添加/删除属性来实现可扩展的细粒度访问控制，而不会将任何敏感数据信息泄露给对手。

10.10 支持灵活的撤销机制

10.10.1 为什么？

由于数十亿的源数据存储在数据库中，数据管理者控制如此大量的数据便变得很困难,即使访问数据的权限过期，数据使用者对数据的可访问性也容易被滥用。一个很好的例子就是：一个公司的员工可以根据工作要求访问属于公司的具体数据,一旦员工被解雇了，他就无法访问公司的任何数据。这表明只有当他/她处于特定位置时，员工才能访问数据。

如何设计一个有效的方案来撤销用户的访问权限，以保护敏感信息免遭恶意或未经授权的查询，甚至是来自前员工的修改，已成为一个关键的安全问题。

在某些情况下，例如涉及一些敏感信息的文档只有一段特定的时间可进行检验，或者敏感数据在特定的时间之前不会被公布。即，数据仅在特定时间间隔内有效。在数据到期后，我们还需要一个有效的数据撤销方案。我们如何确保数据/用户在特定的时间段内才能有效？应提出一些有效的技术，以便为用户和数据提供灵活的撤销方案。

10.10.2 怎么做？

为了撤销用户的访问权限，最简单的想法是有一个中央权力机构来重新分配密钥。由于系统中涉及到所有用户的密钥，最好建立一个中央的权利机构来管理所有密钥,当用户变得无效时，中央权利机构可以生成新的私钥并将其分配给其余的有效用户。存储在系统中的密文也会用先前的私钥解密后再用新的私钥加密。无效的用户不能解密新的密文，因为它已经不保存新的私钥。这个方案是低效的，因为中央权利机构的计算和通信成本太高了。

如上一节所介绍的，我们可以通过引入代理，使用代理重加密技术来以减少密钥更新时中央机构的成本。为了有效地撤销源数据的访问全新啊，可以使用时间特定加密（TSE）技术来加密数据。TSE 是一种具有附加功能的基于公钥的加密技术。TSE 可以指定时间间隔，且数据消费者的私钥与时间相关联，这意味着只有当时间落在指定的时间间隔内时，密文才能被成功解密。

通过指定合适的时刻，仅在该段时间间隔内有效的密钥，将可以被有效撤销。

