DataVault: 基于可控计算的隐私保护解决方案

版本：1.0

熠智科技

https://yeez.tech

2023 年 7 月

目录

1 需求分析

1.1 可控计算

大数据时代背景下，数字经济正处于蓬勃发展的节点。数据作为数字经济最为核心的生产要素，在社会生产、生活的巨大价值已经不言而喻。数据要素价值的充分发挥在于其有效流通共享，亦已成为人们的共识性认识[1]。

根据数据流通的形式，有多种划分方式：例如根据是否经过计算处理可以分为原始数据和计算结果；是否经过加密分为明文数据和密文数据；传输方式的不同分为离线文件和 API 接口。传统的“复制式”数据流通方式会让企业、个人隐私信息等产生泄露，并且难以防范数据二次贩卖等问题，如何保护数据隐私和商业价值是实现数据要素流通的基础。2022 年，国务院发布《要素市场化配置综合改革试点总体方案》明确指出：“在保护个人隐私和确保数据安全的前提下，有序推动部分领域数据流通应用。探索建立数据用途和用量控制制度，实现**数据使用‘可控可计量’**”。

目前关于数据使用的“可控可计量”尚未有明确的定义，一些研究认为数据使用的“可控可计量”包括了对于数据用途的限制，以及用量的控制，进一步地，实现对于数据使用过程的监督和管理[2]。

具体地，我们以两个实际场景为例：

场景 1: 某企业 A 核心盈利产品为某垂直行业数据集，其销售模式是以数据库的形式部署至客户侧供客户使用，为帮助客户更便利地使用该数据，企业 A 也提供了对应的低代码数据处理平台，数据的查询与分析均可以可视化的方式呈现。对于企业 A 而言，其核心诉求是防止数据在客户侧被二次贩卖或向第三方提供数据服务。

场景 2: 某企业 B 在向政府客户销售数据中台产品时，客户允许企业 B 在系统建设或运营过程中接触数据，但要求该企业保证相关人员无法外泄数据。尽管企业 B 通过建立安全制度可以一定程度上约束这类行为，但在技术层面仍然缺少完整的解决方案。

可以看出，在上述场景中，在数据流通环节中数据供需双方均有限制数据使用途径以及数据流通范围的明确需求，从而防范数据隐私泄露以及约定用途之外使用造成

[1] 《数据要素安全流通白皮书（2022 年）》，华东江苏大数据交易中心 国家工业信息安全发展研究中心等

[2] 《中国隐私计算行业研究报告》，艾瑞咨询

的价值流失。

尽管数据供需双方在基于遵守法律规定和积极履行合同约定的前提下，可以一定程度上减少上述风险，但从进一步降低成本和防患于未然的角度，通过技术手段保证数据使用的“可控可计量”也愈显重要。

因此，本文引入了一种新的隐私计算范式：**可控计算**，即**在数据分析计算过程中，对于数据的流转、分析、处理等实现可度量、可监管的隐私计算技术**。

1.2 可控计算：隐私计算新范式

隐私计算是“隐私保护计算”（Privacy-Preserving Computation）的简称，有时也被称为“隐私增强计算”（Privacy-Enhancing Computation）。目前国内外对于“隐私计算”尚未有统一明确的定义，隐私计算联盟发布报告中定义隐私计算为**在保证数据提供方不泄露原始数据的前提下，对数据进行分析计算，有效提取数据要素价值的一类信息技术**[3]。

在一些学者看来，隐私计算和隐私保护计算的定义存在区别，隐私计算是指**以在保护数据隐私的同时实现计算任务为目的所使用一系列广泛的技术的统称**，而隐私保护计算则包括了更为广泛的概念：**研究计算前对数据的安全获取和管理、计算过程中对数据隐私的保护和度量，以及计算完成后对生产数据隐私的保护以及相关权属与收益的分配的一系列技术**[4]。

隐私计算涵盖了密码学、统计学、计算机系统结构和人工智能等多个学科的技术，根据实现思路主要包括：

- **以密码学为核心的技术实现** 包括多方安全计算（MPC）、同态加密（HE）、零知识证明（ZKP）等密码学技术；
- **基于可信硬件的技术实现** 以基于硬件的信任根，对计算环境进行隔离和度量，主要技术为可信执行环境；
- **融合隐私保护技术的联合建模** 主要面向联合建模场景，是将联邦学习和各类隐私保护技术融合的技术实现，又称可信联邦学习（TFL）。

本文不过多探讨隐私计算（或隐私保护计算）具体定义和涉及范围，但隐私计算不再局限于数据计算过程中的隐私保护已经成为行业共识。目前一个较为通俗的解释认为隐私计算实现了数据的**“可用不可见”**，即在对数据的协作计算过程中，原始数据对除数据所有方以外的其他参与方不可见。

[3] 《隐私计算应用研究报告（2022 年）》，隐私计算联盟

[4] 《隐私计算》，陈凯 杨强

首先，在多方协作计算的场景下，很多时候要求原始数据物理意义上的“不可见”并不合理。

例如从特征工程的角度，数据在计算使用之前的清洗、加工和降维等环节，依赖于对原始数据的观察并结合处理者的知识经验，通过反复调试才能达到理想的效果；类似地，在构建视频图像识别模型的监督学习中，对视频、图像样本数据的标注也要求数据可见。

其次，为了保证数据不可见，现有隐私计算解决方案大多会对原有的业务流程产生侵入性，这种侵入性体现在三个方面：

1. 兼容问题：引入隐私计算相关解决方案或产品后，原有的业务系统往往需要调整，即使很多隐私计算产品提供了相应接口，使得用户可以尽可能减少源码层面的修改，但大多数隐私计算系统仍然不能做到应用二进制层面的兼容（即无缝迁移）；

2. 性能下降：数据可用不可见本质上是基于数据加密实现，无论是多方安全计算、还是联邦学习，亦或是可信执行环境，加解密都会造成性能损失（尤其是 MPC 和 FL 解决方案），这在一定程度上影响了应用的可用性，例如实时性要求较高的场景难以使用；

3. 部署成本增加：大多数隐私计算解决方案或产品都需要额外部署隐私计算设备（例如支持 TEE 特性的计算节点、MPC 安全节点或者特定的密码加速硬件）；并且由于底层架构差异（CPU&GPU、x86&ARM），以上定制隐私计算设备难以利用原有业务系统的计算资源，这使得实际部署成本大幅增加。

上述因素使得以数据“可用不可见”为需求驱动的隐私计算解决方案具有一定的局限性，以数据可控为核心需求的数据流通场景是明显有别于传统的隐私计算的，是一种新的隐私计算范式。

1.3 DataVault：基于可控计算的解决方案

本文提出了一种可控计算解决方案：DataVault。本小节将进一步阐明 DataVault 的技术边界和特性，以区别传统的隐私计算解决方案。

可控计算的核心是保证**数据可控**，数据可控是指**数据使用方在数据提供方定义的安全域中对数据进行加工、处理**。如图1所示，物理上，安全域在数据使用方，但是由数据提供方控制。

安全域是一个逻辑上的概念，指由相应密钥和加密算法保护的存储、计算单元。在多数情况下，安全域由数据提供方定义和约束，但相应的存储、计算资源并不由数

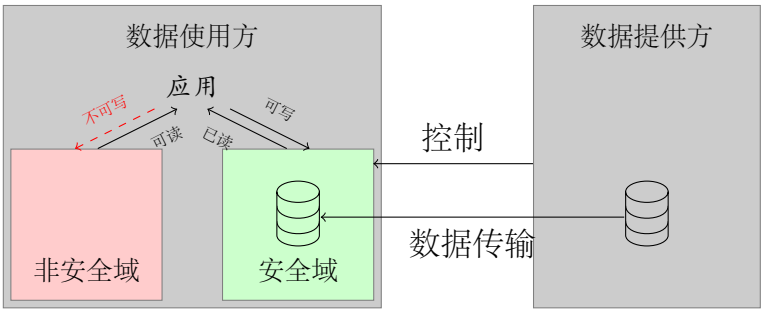

图 1: 数据可控计算示意图

据提供方提供。需要注意的是，加工、处理后的中间数据和结果数据也应在相同的安全域中。

在满足数据可控的前提下，DataVault 还进一步实现了对业务系统的**无侵入性**，这体现在：

- 极佳的通用性：DataVault 提供了二进制兼容，基于 DataVault 的应用无需修改代码，这也包括了现有的主流 AI 模型训练框架;
- 原生硬件适配：DataVault 支持多种 CPU 架构（x86&ARM）以及基于 PCI-e 的计算设备（GPU、FPGA、各种加速卡等）;
- 高性能：远低于其他隐私计算解决方案的性能损失，在大部分应用中，相比原生系统（即不用任何隐私计算技术）整体性能损失不超过 5%。

2 相关技术介绍

由于涉及到的技术较为庞杂，在正式介绍 DataVault 前，本章节先介绍 DataVault 涉及到的技术，并约定相关术语。

注意，本文描述 DataVault 基于 Linux 实现，本章相关技术介绍也以 Linux 为主，但其他操作系统（例如 Windows）也包含类似的模块或功能。因此，理论上 DataVault 可以在其他系统上实现同样的功能。

2.1 TPM

TPM 是可信平台模块（Trusted Platform Module）的缩写，其核心是提供基于硬件的安全相关功能。TPM 芯片是一种安全的加密处理器，旨在执行密码相关操作。该芯片包含多个物理安全机制以使其防篡改，并且恶意软件无法篡改 TPM 的安全功能，使用 TPM 技术的一些优点包括：

- 生成、存储和限制加密密钥的使用;
- 使用 TPM 的唯一密钥将其用于设备身份验证，该密钥已刻录到芯片中;
- 通过获取和存储启动过程的安全度量值来帮助确保平台完整性。

DataVault 基于 TPM 2.0 标准（2014 年发布）实现。相比于 1.0 版本，TPM 2.0 支持自定义的密码算法，包括国密标准算法。目前，TPM 2.0 已经被广泛应用于 PC 和服务器中，Windows 10 和 Windows 11 将 TPM 2.0 列为必须要求；英特尔平台与 AMD 平台对 TPM 2.0 功能的描述各有不同，Intel 的 CPU 中内置了 PTT，AMD 的 CPU 中内置了 fTPM，都支持相应的 TPM 2.0 标准。此外，也可以通过在主板上安装额外的独立芯片获得相应的 TPM 2.0 功能。

出于对信息安全的考虑，我国在关键领域通常会采用国产设备或标准，可信平台模块也不例外。

1999 年我国出台的《商用密码管理条例》明确规定：国家任何商用密码的研制、生产和销售都必须接受专控管理。任何国外生产的商用密码产品未经许可都不允许中国市场上销售。因此，为了满足自身的安全要求，我国提出了自己的可信密码模块(Trusted Cryptography Module,TCM)，TCM 参考了 TPM 1.2 等国际规范，与 TPM 1.2 有很多的相同点，是替换了其核心算法后的产品，同时，TCM 也按照我国相关证书、密码等政策提供了符合我国管理政策的接口。

在可信计算中，TPM 提供了信任根，包括密钥、证书和可信度量。Linux 系统和 grub（>=2.06 版本）都使用到了 TPM 的度量功能[5]。

2.2 IMA

IMA 全称为完整性度量框架（Integrity Measurement Architecture），是 Linux 内核中的一个子系统，能够基于自定义策略对文件进行度量，度量结果可被用于本地或远程证明，或者和已有的参考值比较以控制对文件的访问。2004 年，IBM 发表的论文中首次提出 IMA 架构[6]。

可控计算的重要环节就是对系统平台进行完整性度量，简单来说，IMA 保证了当系统及应用程序的二进制文件放生变化时，度量值也会发生变化。

IMA 的度量机制依靠 TPM 的 PCR（Platform Configure Register）的重置与扩展操作完成。在安装了 TPM 的系统上，IMA 使用 PCR 对度量结果进行记录。由于 TPM 的 PCR 寄存器只支持重置与扩展，因此恶意代码无法进行“任意”的篡改。

[5] https://uapi-group.org/specifications/specs/linux_tpm_pcr_registry/

[6] “Design and Implementation of a TCG-based Integrity Measurement Architecture”, 13th Usenix Security Symposium

IMA 每次度量结果的存储均依靠扩展运算

$$Digest\ of\ (PCR\ old\ value||data\ to\ extend)$$

实现，同时将度量的文件名称、路径以及度量结果存入度量日志/列表。

注：IMA 使用 PCR 10 存放系统的度量值，其中也包含了 PCR 0～PCR 7 的值。

2.3 LUKS

LUKS （Linux Unified Key Setup）是 Linux 系统下常用的磁盘加密技术之一。LUKS 具有以下特点：

- 支持多密码对同一个设备的访问；
- 加密秘钥不依赖密码，因此可以改变密码而无需重新加密数据；
- 基于数据分割技术来保存加密秘钥，保证秘钥的安全性。

LUKS 需要区分密码和加密密钥两个概念。LUKS 卷包括了一个存储加密密钥，即 LUKS **主密钥 (**Primary Key**)** 的头部（Header）和存储加密数据的数据部。写入数据部的数据使用 LUKS 主密钥加密，主密钥则使用用户提供的密钥，即密码（Keyphrase）加密，加密后的 LUKS 主密钥则写入头部。

LUKS 主密钥在解密时会在内存中存储，不会存储在任何持久化存储设备上，以保证 LUKS 主密钥的安全。用户可以同时使用多种方式加密 LUKS 主密钥，包括使用密码、密钥文件等，也可以使用多种加密算法。因此，LUKS 会在头部同时存储多个加密后的主密钥，这些加密后的主密钥被存放在头部的槽（slot）中。LUKS 同时支持 8 个槽。为了区别这些概念，本文称这些用于加密 LUKS 主密钥的密钥为 LUKS **槽密钥**，LUKS 槽密钥可以是密码、普通的密钥文件、甚至额外的硬件设备（如 UKey）。

LUKS 在应用读写时，数据的解密、加密操作，对上层应用是完全透明的。

由于加解密操作的存在，LUKS 在一定程度上会对系统性能产生影响。但现代 CPU 通常有密码操作加速特性，因此，上述性能损失并不大。根据一些公开的数据 [7]，以块大小为 64K 为例，在使用机械硬盘的情况下，读性能损失为-0.06%，写性能损失为-1.32%; 在使用 NVME 的情况下，读带宽损失为 71.76%，写带宽损失为 70.36%。考虑到 IO 操作在整体应用中的比例，基本可以认为使用 LUKS 对系统整体的性能影响在 5% 以内。

[7] https://scs.community/2023/02/24/impact-of-disk-encryption/

2.4 LSM

LSM，即 Linux 安全模块（The Linux Security Module），是 Linux 内核中用于支持各种计算机安全模型的框架，其与任何单独的安全实现无关。这个框架使用 GNU 通用公共许可授权，并且从 Linux2.6 开始成为 Linux 内核的一部分。

LSM 的核心是 Hook（钩子）技术，LSM 提供了诸如文件打开、关闭等几十种操作的钩子函数。LSM 开发者通过这些钩子函数检查用户操作是否满足自己定义的安全策略。

3 系统原理

在本章，我们将描述 DataVault 系统的原理。我们不会在一开始讨论系统的初始化问题，例如密钥如何生成等。本章先关注单机环境下的数据可控方案，对于分布式应用将在7.3节讨论。

首先给出安全域的正式定义。在本文，安全域是指**使用了同一组密钥加密的一个或多个磁盘分区**，用二元组可以表示为：

$$D = \{K, \Delta\}$$

其中 K 是称为安全域的密钥，Δ 称为安全域的范围，包括了多个磁盘分区 δ_i, $\delta_i \in \Delta$。安全域的范围可以动态增加，即将更多的磁盘分区使用 K 加密。系统中的分区不一定都放在安全域中（不一定都加密），没有放在安全域中的分区称为**非安全域**。

一个进程从执行开始到执行结束，会读取、生成一系列数据（文件）。从内核的角度，我们将文件的操作定义为一个三元组 α，

$$\alpha = \{p, t, o\}$$

其中，p 是进程标识，t 是文件所在磁盘（文件系统）分区的标识，o 是操作，只能是read或write操作。对于一个应用程序，可能包含多个文件操作，例如

$$\begin{aligned} &\{ \\ &\quad \alpha_0 = \{p_0, \texttt{/dev/sda}, \text{write}\} \\ &\quad \alpha_1 = \{p_0, \texttt{/dev/sda}, \text{read}\} \\ &\quad \alpha_2 = \{p_0, \texttt{/dev/sdb}, \text{write}\} \\ &\} \end{aligned} \tag{1}$$

注意一个父进程可能会包含多个子进程（`fork` 子进程），对于子进程的操作，同样使用父进程的进程标识。下文描述的进程也统一指父进程，而不是子进程。

DataVault 不会改变应用操作，仅通过 LSM 中定义的钩子对相应的操作进行判断。当通过检查后，相应的系统调用方能正常进行；反之，相应的系统调用则会失败。相应地，如果进程妥善的处理了对应错误（例如，打开文件失败），则进程能够正常执行；反之，进程可能会意外退出。

在 DataVault 中，**一个读取了安全域中的文件的进程，仅能在安全域中进行写文件操作**。值得注意地，对于某进程 p_i，所读取的其安全域 Δ_i 外的分区，均称为 p_i 安全域外 $\overline{\Delta_i}$（简称为域外），这既包括其他安全域的分区（例如 $\delta_k \notin \Delta_i$），也包括系统中的非安全域。

当进程试图在安全域外中写数据（以写权限打开文件）时，相应的操作则会失败。DataVault 并不试图判断写的数据是否和从安全域中读取的数据有关联，这是因为这种关联的判定是十分困难的，需要对数据流进行分析。因此，即使写的数据实质上与安全域中的数据无关（或不泄露隐私），DataVault **也不允许进程在其安全域外进行写操作**。

尽管上述约定是直观的，但在实际情况中还有读写顺序问题：在安全域外先进行写操作，文件关闭后，读取安全域中的数据，然后再在安全域中进行写操作。这种情况下，在安全域外进行的写操作不涉及任何安全域内的敏感数据，因此不会造成安全域中数据的泄漏，因此，这种情况是允许的。但这种情况也是实现相关的：即使没有实现这种细粒度的判定，也不会降低安全性，只是会在一定程度上产生误判，导致本可以执行的进程不能正常执行。

在 DataVault 中，进程可以同时读取多个不同的安全域中的数据，这种情况仅发生在该进程不写任何数据的情况下。例如一个进程读取了两个安全域中的数据，进行分析处理后，仅在终端上输出了结果，这种情况是允许的。如果一个进程需要写数据，则该进程只能使用一个安全域，且读写均发生在该安全域内。

举例来说，进程 A 读取了安全域中的数据后，`fork` 了子进程 B，由于 Linux 的 fork 机制，子进程 B 有父进程 A 在内存中的数据，因此子进程 B 可以在安全域外写该数据。这种情况也视为数据的泄漏，因此也是需要避免的。以前述的情况（式1）为例，假设 `/dev/sda` 为数据提供方定义的安全域，则上述应用在试图进行 α_2 操作时失败。

在实际的系统中，还需要考虑其他诸多因素，例如利用 `ptrace` 查看进程中内存的内容; 使用软链接、硬链接访问文件等。对于通过进程间数据共享技术（共享内存、管道、网络）传输数据，DataVault 同样会跟踪这些系统调用。这里有两种实现方式：一种是进一步跟踪发生数据共享的对手进程（peer process），要求对手进程不能在安全域外进行写操作；另一种是简单的禁止进程使用进程间数据共享相关的系统调用，

这种实现更为简单粗暴，但依然有足够的安全性。

DataVault 不修改应用，作为代价，DataVault 也不知道应用对于数据的使用情况。即使写到安全域外的数据和敏感数据无关，DataVault 也认为会造成潜在的数据泄露。这在某些情况下可能会过于严格，例如，某些应用需要生成日志文件或自动备份文件等，对此，可以引入白名单机制，允许特定的应用写入特定的文件，以保证应用的正常使用。

写入安全域的数据可能是中间数据，也可能是结果数据。通常，总是存在某些结果数据需要被移出安全域的情况。对此，DataVault 通过计算文件 hash 识别需要被移出安全域的文件。如果一个文件需要被移出安全域，DataVault 就不再认为该文件属于相应的安全域，读取该文件后的任意写操作都不再受到影响。

需要注意，**安全域是一次性的**，也就是说，当一个敏感数据被使用完后，用户应该将结果数据移出，并将安全域销毁，不能使用该安全域处理另外的敏感数据。销毁安全域时，需要将相应的分区从安全域中移除，并重新格式化。

4 系统架构

本章节会从系统组成、运行流程以及安全防范机制等方面介绍 DataVault 的整体架构。

4.1 系统组件

如图2所示，在数据使用方的系统环境中，DataVault 主要包括三个组件：DVUtil，DVAgent，以及 FTrammel。

- DVUtil 是供用户进行操作、配置的工具，其提供了诸如初始化安全域、扩展安全域、删除安全域、从安全域中导出数据等操作;
- DVAgent 是运行在系统中的守护进程，功能包括初始化系统，为 DVUtil 处理相关的操作，为内核提供关于安全域的配置，以及与数据提供方通信;
- FTrammel 是运行在内核中 LSM 模块，FTrammel 按照 DVAgent 给出的配置检查内核中文件系统的读写操作。由于 LSM 的要求，FTrammel 需要在 Linux 内核中编译，为区分，我们将包含了 FTrammel 的 Linux 内核版本称为 YeeZLinux。

在数据提供方，DataVault 包括两个组件：DVConfig 以及 DVServer。

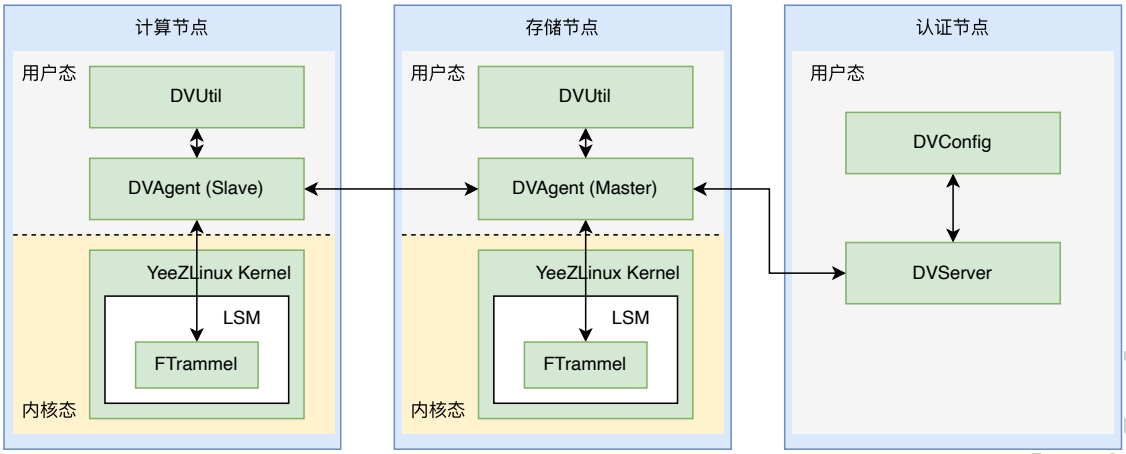

图 2: DataVault 软件架构

- DVConfig 是供数据提供方进行操作、配置的工具，其包括了密钥管理、数据管理、安全域管理、白名单管理等功能;
- DVServer 是运行在系统中的守护进程，负责与 DVAgent 通信，完成相关配置的下发及状态收集功能。

系统初次安装后，需要使用 DVUtil 配置、初始化相应的安全域。安全域配置完成后，数据提供方将数据传输到相应的安全域中，这里有三种数据迁入方式，见第6.1节。

4.2 系统流程

DataVault 系统依赖于 TPM 的度量启动、引导装载程序（Boot Loader）的度量启动、以及操作系统完整性模块。系统完成初始化后（包括系统重启），DataVault 会通过 TPM 计算整个系统的度量值，并存储在 PCR 的 13 寄存器，我们将 DataVault 对系统的度量值称为 DV **度量**（DataVault Measurement，DVM），度量启动详见第5章。

如前所述，安全域的一个重要属性是安全域的密钥，安全域由数据提供方（DVServer）控制。因此，数据提供方首先通过 DVServer 生成或导入**昌钥**，昌钥是一对非对称密钥（昌公钥、昌私钥），做为 LUKS 槽密钥使用，用于加密 LUKS 中的主密钥，表示安全域的控制权。昌私钥需要加密、传输，因此不能使用智能卡、UKey 等不能导出私钥的设备生成昌钥。

在 DVAgent 中，TPM 根据 DVM 的值生成**文钥**，文钥是一对非对称密钥，文私钥被封存在 TPM 中（由封存在 TPM 中的随机数和 DV 度量生成），文公钥则可以公开。相应的，TPM 会为文钥生成证明 (Quote)，即**文钥证明**，文钥证明不但包括了对于公钥的签名，还包括了 PCR 的值的签名。文钥是平台相关的，当 DVM 的值发生变化时，文钥也会发生变化，如果使用文钥加密了一个数据，那么文钥发生变化后，

该数据不能解开。

DVAgent 将文公钥及文钥证明发送给 DVServer，DVServer 对其进行验证，以证明相应的文私钥是不可获取的。验证通过后，DVServer 将 DVConfig 生成的**昌私钥**使用文公钥加密后，返回给 DVAgent，DVAgent 使用文私钥解密（通过 TPM）、得到相应的昌私钥，并写到一个内存文件中，称为**昌私钥文件**。DVUtil 可以通过 DVAgent 将昌私钥文件初始化为安全域（LUKS）的槽密钥，从而完成指定安全域的初始化。昌钥是会话相关的，也就是说，一个数据提供方可能有多个数据文件，这些文件需要交付给不同的数据使用方的多个安全域，对每个安全域应使用不同的昌钥。

安全域完成初始化后，数据提供方将数据传输到相应的安全域中，关于数据的迁入内容详见第6.1章。

DVAgent 会将系统中的安全域信息、白名单信息配置到 FTrammel。在应用发出文件打开操作时，LSM 的钩子会调用 FTrammel，由 FTrammel 按照前述的规则对打开文件操作进行判定，如果符合安全规则，则运行文件打开操作，否则，文件打开失败。同样的，FTrammel 也会判断进程对共享内存、网络传输的调用是否符合安全规则。

对于结果数据（或需要移出安全域的数据），数据使用方需要得到对应文件的 hash，并将文件 hash 出示给数据提供方[8]，数据提供方（在 DVServer 中）使用昌私钥对文件 hash 签名后，发送给 DVAgent，DVAgent 验证后，配置到 FTrammel，使 FTrammel 不再跟踪该文件。关于数据的迁出内容详见第6.2章。

4.3 系统安全

DataVault 的系统安全策略考虑到多个层面，能够应对不同级别的作恶行为。下面讨论一些典型的作恶行为，以及 DataVault 是如何防范的。

1. 篡改操作系统：DataVault 要求使用 YeeZLinux Kernel，如果节点没有部署安装相应的内核，或者篡改了内核或配置，相应的 DVM 的值也会发生变化。DVServer 仅对白名单中的 PCR 的值通过验证，因此一旦 DVM 的值发生变化，DVServer 就不再返回加密后的昌私钥。
2. 篡改硬件、固件、引导装载程序：等同于前述的篡改操作系统。
3. 在运行时更新硬件：指运行时对设备（例如网卡）的热拔插，例如增加一个带有嗅探功能的网卡[9]，对此，可以使用 TPM 对设备进行认证，仅允许通过认证的硬件执行。

[8]数据提供方可能需要文件的原文以验证文件 hash，该步骤不是必须的

[9]例如`https://github.com/enjoy-digital/pcie_analyzer`

4. 私自使用昌私钥文件：昌私钥文件的路径是公开的，因此系统中的其他进程可能会使用该文件（读取该文件的内容）。昌私钥的文件读写权限仅属于 DVAgent，不能被其他进程访问。昌私钥文件作为槽私钥，需要确认 LUKS 使用槽密钥时，是哪个进程在使用，对此，可以在 FTrammel 中加以限制。

5. 私自移除安全域：安全域的增加、移除操作仅由 DVAgent 进行，FTrammel 中对于系统调用的主体有限制。

6. 伪造移出文件的信息：安全域中的白名单相关操作仅由 DVAgent 进行，DVAgent 在增加白名单时，需要检查相应的签名。

7. 伪造 DVAgent：用户修改 DVAgent 的行为，泄漏昌私钥。DVAgent 是 Linux 系统（IMA）的度量对象，并且开机启动，因此不会被修改。

5 度量启动

度量启动保证了系统是完整的、未经篡改的。包括

1. 硬件未经篡改;
2. 固件（firmware）未经篡改;
3. 启动引导程序 (bootloader) 未经篡改;
4. 操作系统内核、内核配置、启动参数未经篡改;
5. DVAgent 未经篡改。

为了保证系统的完整性，DVAgent 会在启动后，将系统的度量（PCR8、PCR9、PCR10）扩展（extend）到 DVM（PCR 的 13 寄存器）中，并进一步将 DVAgent 的二进制文件（包括链接库）的 hash 扩展到 DVM 中，并将 PCR8、PCR9、PCR10,PCR13 的 Quote 发送到 DVServer。DVServer 对 TPM 的 Quote 进行验证，并通过已知的操作系统内核、启动命令参数、DVAgent 的二进制文件 hash 验证相应的 PCR 值是否合法。如果非法，则返回错误; 如果合法，则继续。

DVAgent 在收到系统度量合法通知后，则根据 PCR13 中的值，在 TPM 中生成相应的文钥，并对文钥进行签名，生成文钥证明，相应的公钥和文钥证明发送给 DVServer，用于加密昌私钥。DVAgent 收到加密的昌私钥后，直接存储到内存文件中。

DVAgent 收到加密的昌私钥后，也可以将密文存储在文件系统中。但是，系统中的其他进程可能会调用 TPM 中的文私钥解密相应的密文，从而造成昌私钥的泄漏。

一种可能的做法是在 FTrammel 中将该文件的打开权限限制为 DVAgent，从而避免其他进程访问该文件。这种做法的优点是如果不是第一次启动，DVAgent 则从 DVM 恢复相应的文私钥, 并试图解密存储在文件系统中加密的昌私钥。如果解密成功，则说明 DVM 的值较上次启动未发生变化，即系统是完整的; 否则，说明系统在某个层面被篡改，从而导致安全域不能被加载。

昌私钥解密成功后，DVAgent 会将其明文写到临时的内存文件（昌私钥文件）中，并以该文件作为 LUSK 槽密钥加载安全域中定义的磁盘分区，从而完成系统的启动。

6 数据的迁入和迁出

6.1 数据迁入

数据的迁入是指数据提供方如何将数据迁入到数据使用方划定的安全域中。根据数据传输方式的区别，有三种数据迁入方式：分区挂载、数据拷贝以及网络传输。

6.1.1 分区挂载

分区挂载是指数据提供方直接提供一个物理硬盘（通过邮寄或其他方式），然后数据使用方直接将硬盘加载到系统中。这种方式适合交付数据大的场景，例如待交付数据达到几十、上百 TB。

数据提供方使用 cryptsetup 等工具直接将物理硬盘初始化为 LUKS 分区，并将昌私钥作为 LUKS 槽密钥。数据提供方直接将明文拷贝到该物理硬盘中，LUKS 会自动将数据加密。数据使用方收到该物理硬盘后，可以直接以热插入的方式挂载硬盘，硬盘挂载后，使用 DVUtil 将其添加到安全域中。

6.1.2 数据拷贝

数据拷贝是指数据提供方对数据加密后，将加密后的数据以物理介质（U 盘、光盘等）交付给数据使用方，数据使用方使用 DVUtil 将加密数据解密后导入到安全域中。

数据提供方加密数据时，使用昌公钥。DVUtil 导入数据时，实际的解密操作由 DVAgent 执行，DVAgent 使用接收到的昌私钥解密，并将解密后的数据写入到相应的安全域中。该步骤的关键是 DVAgent 不能泄漏数据，考虑到系统的完整性保护，这是容易的。

6.1.3 网络传输

网络传输方式是指数据提供方对数据加密后，将加密后的数据以网络传输方式交付给数据使用方，数据使用方将加密数据解密后导入安全域中。

类似于数据拷贝方式，数据提供方使用昌公钥加密数据，DVAgent 使用昌私钥解密数据。不同的是，数据提供方需要使用 DVConfig 工具，通过 DVServer 直接和 DVAgent 通信，数据使用方不需要通过 DVUtil 执行任何操作。

网络传输方式更为灵活，适合大多数情况，但需要 DVAgent 和 DVServer 之间的连接和数据传输。

6.2 数据迁出

当原始数据完成处理后，需要将结果数据迁出安全域。由于结果数据依然可能包含潜在的隐私，违反数据提供方的安全策略，因此，结果数据的迁出需要数据提供方的授权。

数据使用方通过工具得到对应文件的 hash，并通过 DVUtil 调用 DVAgent 将文件 hash 发送给数据提供方，数据提供方根据自己的策略对文件 hash 进行校验。这种策略可能是基于文件原文，也有可能基于付费，还有可能没有任何限制。数据提供方（在 DVServer 中自动进行）使用昌私钥对文件 hash 签名后，发送给 DVAgent，DVAgent 验证后，调用 FTrammel 相关的系统调用，添加到 FTrammel 的白名单，使 FTrammel 不再跟踪该文件。DVAgent 完成相关操作后，DVUtil 返回相应的操作结果等信息。最后，数据使用方可以使用 `cp` 命令将数据移出。

7 FTrammel

如前所述，FTrammel 通过 LSM 中定义的钩子对相应的操作进行判断：一个读取了安全域中的文件的进程，仅能在安全域中进行写文件操作。

FTrammel 会在 LSM 中对打开文件的操作 (`hook_file_open`) 进行判断，对打开文件的进程（如果是子进程，则记录父进程）记录打开的文件，以及打开文件的权限。出于性能考虑，FTrammel 并不跟踪每次读写，而且限制只有在安全域中才能以写权限打开文件。

7.1 FTrammel 的安全问题

FTrammel 中包含了安全域的新增、删除操作，同时也包括了对于迁出数据的白名单的修改，因此不能被其他进程任意访问，只能被 DVAgent 访问。这可以在 FTrammel 的系统调用中，对调用进程的文件 hash 进行判定，只允许给定 hash 的进程进行系统调用。该 hash 可以在 Linux 的启动命令行参数中给定，当该参数不给定时，FTrammel 不限制调用进程，但此时系统是不安全的。

7.2 使用 FTrammel 完成度量

FTrammel 还应该完成对 DVAgent 的度量，并扩展到 TPM 中，生成相应的 DVM 的值。在 DVAgent 进行度量是不安全的。

7.3 在 FTrammel 中支持分布式应用

对于分布式应用，需在在多个计算机节点上共享、传输数据，各个计算机节点上接收的数据也包含了数据隐私，FTrammel 也支持分布式应用的可控计算。完整的介绍 FTrammel 是如何支持分布式应用的，超出本文的范围，此处仅对其原理进行简单的介绍。

我们称数据存储的节点为 DVAgent-Master，简称为 Master，通过网络接收数据的节点为 DVAgent-Slave，简称为 Slave。在系统完成初始化后，Master 节点和 Slave 节点都需要部署 DVAgent 和 FTrammel，并且两个节点的 FTrammel 建立连接，并完成验证（attestation）。当 Master 和 Slave 建立连接（TCP）或发送数据（UDP）时，FTrammel 首先会判定目标地址是否已经通过 FTrammel 和自身建立了连接，如果没有建立连接，则组织 Master 和 Slave 之间的数据传输；如果建立了连接，则告知 Slave 节点，要求接收数据的进程只能在安全域内写数据，不能在安全域外写数据。从而控制敏感数据的传播范围。

Master 和 Slave 节点需要使用同样的昌私钥建立安全域，这意味着 Slave 节点需要从 Master 节点获取昌私钥。

Master 节点和 Slave 节点的数据通信为明文，不存在额外的加解密过程，因此不会增加分布式应用中的通信开销，也不会带来性能损失。

8 总结

本文提出了一种新的隐私计算范式:可控计算,并提出了一套解决方案,DataVault。DataVault 将 TPM 做为信任根，保护了系统的完整性；使用 LSM 技术使得安全域内的数据只在可控范围内使用。在此基础上，DataVault 使用 Linux 提供的全盘加密技术将数据置于安全域内，并设计了完整的密钥分发以及签名授权等密码协议，进一步保证了数据的可控。

最后，DataVault 的实现保证了其无侵入性的特点，包括

- 性能高：相比于无数据隐私安全防护的系统，性能损失不超过 5%；
- 通用性高：支持多种数据处理框架、模型训练框架，且二进制兼容；
- 兼容性好：支持多种专用加速卡，包括多种 CPU、GPU、FPGA 等硬件。

附录

A 可信计算

虽然本文将数据可控描述为一种新的隐私计算模式，但还是有必要区分几个概念隐私计算、机密计算、可信计算。隐私计算（Privacy Preserving Computing）在于保护隐私，但是隐私的定义是模糊的，因此更常见与市场营销中; 机密计算（Confidential Computing）则严格定义了数据的机密性要求; 而可信计算则不同，不同国际组织对可信（Trusted）做了不同的定义。

- 可信计算组织（TCG）的定义：一个实体是可信的，它的行为总是以预期的方式达到预期的目标。
- 国际标准化组织与国际电子技术委员会定义（1999）：

  参与计算的组件、操作或过程在任意的条件下是可预测的，并能够抵御病毒和一定程度的物理干扰。
- IEEE Computer Society Technical Committee on Dependable Computing 定义：

  所谓可信，是指计算机系统所提供的服务是可被论证其是可信赖的，可信赖主要是指系统的可靠性和可用性。

简而言之，可信就是系统按照预定的设计和策略运行，不做其他事情。一个可信计算系统由信任根、可信硬件平台、可信操作系统和可信应用组成，它的基本思想是首先创建一个安全信任根（TCB），然后建立从硬件平台、操作系统到应用的信任链，在这条信任链上从根开始，前一级认证后一级，实现信任的逐级扩展，从而实现一个安全可信的计算环境。可信计算采用的是白名单机制，即只允许经过认证的内核、内核模块、应用程序等在系统上运行，如果发现程序已发生更改（或本来就是一个未知的程序），就拒绝其执行。