说明书摘要

本发明公开了一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法及系统，通过构建硬件辅助虚拟化的受限执行环境与独立监控控制层，在架构层面实现了对人工智能模型的物理隔离；在此基础上，实施全链路输入输出拦截与原始数据流捕获，对原始数据流执行语义逻辑深度解析与行为轨迹建模，进而进行多维度风险量化评估与异常模式匹配；当检测到异常模式时，系统能够立即触发动态熔断机制与资源切断执行，并在资源切断执行完成后生成安全审计报告与状态重置；通过虚拟化隔离与动态行为监控，在架构层面有效防止了人工智能模型产生恶意行为，显著提升了人工智能系统在复杂环境下的安全性与可控性。

摘要附图


构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层
在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获
对所述原始数据流执行语义逻辑深度解析与行为轨迹建模
根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配
当检测到异常模式时触发动态熔断机制与资源切断执行，在资源切断执行完成后生成安全审计报告与状态重置

1.一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，包括:

构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层;

在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获;

对所述原始数据流执行语义逻辑深度解析与行为轨迹建模;

根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配;

当检测到异常模式时触发动态熔断机制与资源切断执行,在资源切断执行完成后生成安全审计报告与状态重置。

2.根据权利要求 1 所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层包括:

解析硬件虚拟化指令集并初始化虚拟机监视器;

在所述虚拟机监视器基础上划分独立监控控制层的专用执行空间，在所述专用执行空间内建立虚拟化中断向量表与异常捕获通道;

在所述异常捕获通道中配置内存屏障与指令重排序限制。

3.根据权利要求 2 所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述解析硬件虚拟化指令集并初始化虚拟机监视器包括:

通过读取 CPUID 指令返回的寄存器状态确认处理器支持嵌套分页及扩展页表功能，并调用底层固件接口初始化虚拟机监视器，定义

全局资源映射表记录物理内存页帧与虚拟地址空间的映射关系。

4.根据权利要求1所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述实施全链路输入输出拦截与原始数据流捕获包括：

在受限执行环境中截获外部输入数据包的边界校验；

对通过边界校验的数据包解析数据包结构并提取语义特征片段；

将所述语义特征片段构建成输入数据的多模态特征向量，为所述多模态特征向量记录输入数据的时间戳与来源指纹。

5.根据权利要求4所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述截获外部输入数据包的边界校验中，根据预设的地址规则判断数据包来源的合法性，若计算出的距离小于等于容许的距离阈值则数据包被视为合法并放行至下一步处理，否则直接丢弃。

6.根据权利要求1所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述执行语义逻辑深度解析与行为轨迹建模包括：

利用多模态特征向量进行语义意图推断，根据所述语义意图推断结果构建基于马尔可夫链的行为状态转移图；

通过所述行为状态转移图计算当前行为轨迹与标准模型的偏差度；

将所述偏差度生成动态行为轨迹快照并存储至内存池。

7.根据权利要求6所述的一种基于虚拟化隔离与动态行为监控

的人工智能安全防御方法，其特征在于，所述计算当前行为轨迹与标准模型的偏差度中，采用KL散度的变体量化当前行为与历史常态的偏离程度，若偏差度值持续上升并突破临界值则表明目标对象正在表现出非预期的行为模式。

8.根据权利要求1所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述多维度风险量化评估与异常模式匹配包括：

根据动态行为轨迹快照计算语义逻辑连贯性评分；

结合所述语义逻辑连贯性评分评估决策结果的伦理合规性，依据所述伦理合规性评估结果识别攻击意图的特征向量匹配；

综合所述特征向量匹配结果生成综合风险指数并触发预警信号。

9.根据权利要求8所述的一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，其特征在于，所述生成综合风险指数并触发预警信号中，综合语义逻辑、伦理合规与攻击意图三个维度的评估结果生成最终的综合风险指数，当综合风险指数超过系统设定的紧急阈值时立即生成预警信号。

10.一种用于实现如权利要求1-9任意一项所述方法的基于虚拟化隔离与动态行为监控的人工智能安全防御系统，其特征在于，包括：

虚拟化环境构建单元，用于构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层；

数据流拦截单元，用于在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获；

语义解析单元，用于对所述原始数据流执行语义逻辑深度解析与行为轨迹建模；

风险评估单元，用于根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配；

熔断执行单元，用于在检测到异常模式时触发动态熔断机制与资源切断执行行；

审计报告单元，用于在资源切断执行完成后生成安全审计报告与状态重置。

一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法及系统

技术领域

本发明涉及人工智能技术领域，具体是涉及一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法及系统。

背景技术

随着人工智能技术的快速发展，各类智能模型在金融、医疗、交通等关键领域得到广泛应用。然而，人工智能系统在复杂开放环境中运行时，面临着诸多安全挑战。现有技术主要采用应用层防护策略，通过在模型推理阶段部署规则引擎或异常检测模块来识别潜在风险。这些方法通常依赖于预定义的规则库或离线训练的检测模型，对已知攻击模式具有一定的防御能力。

然而，现有技术存在明显的局限性。由于缺乏底层架构层面的隔离机制，人工智能模型与宿主系统共享计算资源，一旦模型被恶意操控或产生异常行为，将直接影响整个系统的稳定性与安全性。特别是在面对说谎、不道德决策或攻击意图等复杂异常模式时，传统应用层防护难以实现实时监控与快速响应，导致安全隐患无法及时发现和有效处置。

发明内容

本发明旨在提供一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法及系统，通过虚拟化隔离与动态行为监控，在架构层面有效防止了人工智能模型产生恶意行为，显著提升了人工智能系统在复杂环境下的安全性与可控性。

为达到以上目的，本发明采用的技术方案为：一种基于虚拟化隔

离与动态行为监控的人工智能安全防御方法，包括:

构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层;

在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获;

对所述原始数据流执行语义逻辑深度解析与行为轨迹建模;

根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配;

当检测到异常模式时触发动态熔断机制与资源切断执行,在资源切断执行完成后生成安全审计报告与状态重置。

优选的,所述构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层包括:

解析硬件虚拟化指令集并初始化虚拟机监视器;

在所述虚拟机监视器基础上划分独立监控控制层的专用执行空间，在所述专用执行空间内建立虚拟化中断向量表与异常捕获通道;

在所述异常捕获通道中配置内存屏障与指令重排序限制。

优选的，所述解析硬件虚拟化指令集并初始化虚拟机监视器包括:

通过读取 CPUID 指令返回的寄存器状态确认处理器支持嵌套分页及扩展页表功能,并调用底层固件接口初始化虚拟机监视器,定义全局资源映射表记录物理内存页帧与虚拟地址空间的映射关系。

优选的，所述实施全链路输入输出拦截与原始数据流捕获包括:

在受限执行环境中截获外部输入数据包的边界校验;

对通过边界校验的数据包解析数据包结构并提取语义特征片段;

将所述语义特征片段构建成输入数据的多模态特征向量,为所述多模态特征向量记录输入数据的时间戳与来源指纹。

优选的,所述截获外部输入数据包的边界校验中,根据预设的地址规则判断数据包来源的合法性,若计算出的距离小于等于容许的距离阈值则数据包被视为合法并放行至下一步处理,否则直接丢弃。

优选的,所述执行语义逻辑深度解析与行为轨迹建模包括:

利用多模态特征向量进行语义意图推断,根据所述语义意图推断结果构建基于马尔可夫链的行为状态转移图;

通过所述行为状态转移图计算当前行为轨迹与标准模型的偏差度;

将所述偏差度生成动态行为轨迹快照并存储至内存池。

优选的,所述计算当前行为轨迹与标准模型的偏差度中,采用KL散度的变体量化当前行为与历史常态的偏离程度,若偏差度值持续上升并突破临界值则表明目标对象正在表现出非预期的行为模式。

优选的,所述多维度风险量化评估与异常模式匹配包括:

根据动态行为轨迹快照计算语义逻辑连贯性评分;

结合所述语义逻辑连贯性评分评估决策结果的伦理合规性,依据所述伦理合规性评估结果识别攻击意图的特征向量匹配;

综合所述特征向量匹配结果生成综合风险指数并触发预警信号。

优选的,所述生成综合风险指数并触发预警信号中,综合语义逻辑、伦理合规与攻击意图三个维度的评估结果生成最终的综合风险指

数，当综合风险指数超过系统设定的紧急阈值时立即生成预警信号。

另一方面，本发明提出一种基于虚拟化隔离与动态行为监控的人工智能安全防御系统，包括：

虚拟化环境构建单元，用于构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层；

数据流拦截单元，用于在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获；

语义解析单元，用于对所述原始数据流执行语义逻辑深度解析与行为轨迹建模；

风险评估单元，用于根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配；

熔断执行单元，用于在检测到异常模式时触发动态熔断机制与资源切断执行行；

审计报告单元，用于在资源切断执行完成后生成安全审计报告与状态重置。

与现有技术相比，本发明的有益效果在于：

本发明通过构建硬件辅助虚拟化的受限执行环境与独立监控控制层，在架构层面实现了对人工智能模型的物理隔离；在此基础上，实施全链路输入输出拦截与原始数据流捕获，对原始数据流执行语义逻辑深度解析与行为轨迹建模，进而进行多维度风险量化评估与异常模式匹配；当检测到异常模式时，系统能够立即触发动态熔断机制与资源切断执行，并在资源切断执行完成后生成安全审计报告与状态重

置；通过虚拟化隔离与动态行为监控，在架构层面有效防止了人工智能模型产生恶意行为，显著提升了人工智能系统在复杂环境下的安全性与可控性；该方法实现了从硬件层到应用层的全链路安全防护，能够在异常行为发生的第一时间进行精准识别和快速阻断，解决了传统应用层防护难以从底层隔离和实时监控人工智能模型行为轨迹的技术难题。

附图说明

图 1 为本发明基于虚拟化隔离与动态行为监控的人工智能安全防御方法的流程图；

图 2 为本发明基于虚拟化隔离与动态行为监控的人工智能安全防御系统的框图。

具体实施方式

以下描述用于揭露本发明以使本领域技术人员能够实现本发明。以下描述中的优选实施例只作为举例，本领域技术人员可以想到其他显而易见的变型。

如图 1 所示，本发明提出一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法，该方法核心在于通过底层虚拟化技术构建物理隔离的运行时环境，并在此之上部署具备实时分析能力的监控控制层，从而实现对目标对象的全链路行为约束，具体包括：

构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层；具体构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层包括：解析硬件虚拟化指令集并初始化虚拟机监视器；在所述虚拟机监视器

基础上划分独立监控控制层的专用执行空间，在所述专用执行空间内建立虚拟化中断向量表与异常捕获通道；在所述异常捕获通道中配置内存屏障与指令重排序限制。

通过解析硬件虚拟化指令集并初始化虚拟机监视器，实现了底层硬件资源的深度控制与隔离。在专用执行空间内建立虚拟化中断向量表与异常捕获通道，配合内存屏障与指令重排序限制，有效阻断了恶意模型对宿主系统的越权访问与侧信道攻击，确保了监控控制层的独立性与数据流的绝对安全，为后续的行为分析提供了可信的底层运行环境。

其中，解析硬件虚拟化指令集并初始化虚拟机监视器包括：通过读取 CPUID 指令返回的寄存器状态确认处理器支持嵌套分页及扩展页表功能，并调用底层固件接口初始化虚拟机监视器，定义全局资源映射表记录物理内存页帧与虚拟地址空间的映射关系。

通过精准校验 CPU 指令集与固件接口，确保了虚拟化环境的底层兼容性与启动可靠性。定义全局资源映射表实现了对物理内存页帧与虚拟地址空间的精细化管控，不仅从根源上杜绝了非法内存越界访问，还为构建高隔离度的受限执行环境奠定了坚实的硬件级安全基础。

在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获；具体包括：在受限执行环境中截获外部输入数据包的边界校验；对通过边界校验的数据包解析数据包结构并提取语义特征片段；将所述语义特征片段构建成输入数据的多模态特征向量，为所述多模态特

征向量记录输入数据的时间戳与来源指纹。

其中，截获外部输入数据包的边界校验中，根据预设的地址规则判断数据包来源的合法性，若计算出的距离小于等于容许的距离阈值则数据包被视为合法并放行至下一步处理，否则直接丢弃。

通过基于地址规则的距离阈值计算，在边界层以极低开销精准识别并阻断非法来源数据包，有效构筑了第一道安全防线；同时，对合法数据进行的深度解析、多模态特征向量构建及时间戳与来源指纹的同步记录，不仅实现了输入数据的标准化处理以提升后续分析精度，更构建了全链路可追溯的审计线索，确保了数据流的完整性与安全溯源能力。

对所述原始数据流执行语义逻辑深度解析与行为轨迹建模；具体包括：利用多模态特征向量进行语义意图推断，根据所述语义意图推断结果构建基于马尔可夫链的行为状态转移图；通过所述行为状态转移图计算当前行为轨迹与标准模型的偏差度；将所述偏差度生成动态行为轨迹快照并存储至内存池。

通过马尔可夫链建模将离散语义意图转化为连续行为轨迹，实现了对用户或系统行为的动态演化预测；利用偏差度计算机制，能够敏锐识别偏离正常模式的异常行为（如未知攻击或违规操作），显著提升了对复杂隐蔽威胁的检出率；同时，动态快照的内存池化存储确保了实时响应能力与低延迟监控，为即时阻断风险提供了高效的数据支撑。

其中，计算当前行为轨迹与标准模型的偏差度中，采用 KL 散度

的变体量化当前行为与历史常态的偏离程度，若偏差度值持续上升并突破临界值则表明目标对象正在表现出非预期的行为模式。

通过监测偏差度的持续上升并突破临界值，能够精准捕捉隐蔽的渐进式攻击或违规行为，有效降低误报率并提前预警潜在风险，从而显著提升对未知威胁的主动防御能力。

根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配；当检测到异常模式时触发动态熔断机制与资源切断执行，在资源切断执行完成后生成安全审计报告与状态重置。

其中，多维度风险量化评估与异常模式匹配包括：根据动态行为轨迹快照计算语义逻辑连贯性评分；结合所述语义逻辑连贯性评分评估决策结果的伦理合规性，依据所述伦理合规性评估结果识别攻击意图的特征向量匹配；综合所述特征向量匹配结果生成综合风险指数并触发预警信号。

通过融合语义连贯性评分与伦理合规性评估，构建了超越传统规则匹配的深层风险识别体系，能够精准锁定具有隐蔽意图的高级威胁；动态熔断与资源切断机制实现了毫秒级响应，有效遏制了攻击扩散并最小化业务损失；而自动化的安全审计与状态重置闭环，则大幅降低了人工干预成本，确保了系统在遭受攻击后的快速自愈与持续稳定运行。

进一步的，生成综合风险指数并触发预警信号中，综合语义逻辑、伦理合规与攻击意图三个维度的评估结果生成最终的综合风险指数，当综合风险指数超过系统设定的紧急阈值时立即生成预警信号。

通过融合语义、伦理与意图三维评估，构建了立体化的风险研判模型，有效解决了单一维度误判问题；当风险指数突破紧急阈值时立即触发预警，实现了从被动响应到主动阻断的跨越，确保在攻击造成的实质性损害前完成精准拦截，极大提升了系统的整体安全韧性。

另一方面，本发明提出一种基于虚拟化隔离与动态行为监控的人工智能安全防御系统，如图 2 所示，包括：

虚拟化环境构建单元，用于构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层；

数据流拦截单元，用于在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获；

语义解析单元，用于对所述原始数据流执行语义逻辑深度解析与行为轨迹建模；

风险评估单元，用于根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配；

熔断执行单元，用于在检测到异常模式时触发动态熔断机制与资源切断执行行；

审计报告单元，用于在资源切断执行完成后生成安全审计报告与状态重置。

进一步的，上述系统中的各单元在执行时还用于实现上述一种基于虚拟化隔离与动态行为监控的人工智能安全防御方法的其他步骤，如下：

步骤一：构建基于硬件辅助虚拟化的受限执行环境与独立监控控

制层;

本步骤任务是在物理硬件之上构建一个逻辑上完全隔离的计算沙箱，并在该沙箱内部署独立的监控实体。该过程不依赖任何软件层面的补丁更新或配置调整，而是直接利用现代处理器提供的硬件虚拟化扩展指令集，将目标对象的运行空间从宿主系统中剥离出来，形成具有严格内存访问权限边界的封闭区域，具体包括:

步骤 1.1: 解析硬件虚拟化指令集并初始化虚拟机监视器;

首先，系统需识别当前计算平台所支持的硬件虚拟化扩展指令集，包括 IntelVT-x 或 AMD-V 等底层指令接口。通过读取 CPUID 指令返回的寄存器状态，确认处理器是否支持嵌套分页（NestedPageTables）及 EPT（ExtendedPageTables）功能。一旦确认硬件能力，即调用底层固件接口初始化虚拟机监视器（Hypervisor）。该监视器负责接管处理器的特权级控制权，将原本属于操作系统的最高管理权限收归至监视器自身。在此过程中，定义全局资源映射表M_{res}，该表记录了物理内存页帧与虚拟地址空间的映射关系。映射关系的建立遵循如下约束方程:

$$M_{res}(v_{addr}) = \lfloor \frac{v_{addr} - V_{base}}{P_{size}} \rfloor \times P_{size} + O_{offset}$$;

其中，v_{addr}表示虚拟地址空间中的起始地址，单位为字节（Byte）；V_{base}为分配给该执行环境的虚拟基址，单位为字节；P_{size}为页面大小，设定为 4KB；$\lfloor \cdot \rfloor$表示向下取整运算；O_{offset}为页内偏移量，单位为字节。此公式确保了每个虚拟地址都能唯一且无冲突地映射到物理内存的物理页框中，从而在物理层面切断目标对象对非授

权内存区域的访问路径。

步骤 1.2：划分独立监控控制层的专用执行空间；

在完成基础映射表的构建后，系统需在已划分的物理内存空间中开辟一块独立的区域，专门用于承载监控控制层。该区域被标记为只读或不可执行，仅允许特定的监控代理进程进行读写操作，而目标对象所在的执行空间则被标记为可执行但受控。监控控制层的启动参数 P_{mon} 包含其所需的最大内存配额 Q_{mem} 和最大 CPU 时间片 T_{cpu} 。这两个参数的分配必须满足以下不等式约束，以确保监控层不会因资源耗尽而失效，同时也不会过度挤占目标对象的运行资源：

$$Q_{mem} + T_{cpu} \times C_{freq} \leq \eta \times (Q_{total} + T_{total} \times C_{freq})$$ ；

式中， C_{freq} 为处理器的主频，单位为赫兹（Hz）； Q_{total} 为系统分配的总内存容量，单位为字节； T_{total} 为系统分配的总时间片，单位为秒（s）； η 为资源保留系数，取值范围在 0.8 至 0.95 之间，代表预留的安全冗余比例。通过该约束，系统在启动阶段即锁定了监控层的生存空间，使其能够在目标对象发生异常时依然保持活跃状态，随时准备执行熔断操作。

步骤 1.3：建立虚拟化中断向量表与异常捕获通道；

为了实现对目标对象行为的实时感知，必须建立一套高效的中断响应机制。系统初始化一张全局中断向量表 I_{vec} ，将所有的硬件异常、软件陷阱以及特定指令触发事件映射到监控控制层的处理例程中。当目标对象尝试执行敏感指令（如直接访问 I/O 端口、修改页表项或发起网络请求）时，处理器会立即抛出异常，并由 Hypervisor 捕获。

异常捕获通道的建立依赖于中断优先级L_{int}的定义，其计算公式为：

$$L_{int} = \sum_{i=1}^{N} w_i \times S_i$$

；

其中，N为需要监控的指令类型总数；w_i为第i类指令的危险权重系数，无量纲，数值越大代表该指令越危险；S_i为该指令在当前上下文中被触发的次数，单位为次。通过累加各指令类型的加权触发次数，系统能够动态调整中断处理的优先级顺序，确保高风险指令（如内存写入或网络发送）优先得到监控层的介入。这种机制保证了监控层能够以最小的延迟响应目标对象的任何潜在违规行为。

步骤1.4：配置内存屏障与指令重排序限制；

为了防止目标对象利用指令重排序特性绕过监控检查，必须在虚拟化层强制实施严格的内存屏障策略。系统为每个虚拟CPU核心配置一个内存屏障标志位B_{flag}，当检测到目标对象试图并发访问共享内存时，自动插入全屏障指令。屏障的有效性由内存一致性模型C_{mem}决定，其判定逻辑如下：

$$C_{mem} = \begin{cases} 1, & \text{if } \forall t_1, t_2 : (t_1 < t_2) \land (Op(t_1) \in W) \implies (Op(t_2) \notin R \lor \text{Barrier}(t_1)) \\ 0, & \text{otherwise} \end{cases}$$

其中，W表示写操作集合，R表示读操作集合；Op(t)表示在时刻t执行的内存操作；Barrier(t)表示在时刻t是否插入了内存屏障指令。若C_{mem}值为1，说明内存访问顺序得到了严格保证，任何写操作后的读操作都必须等待屏障生效后方可执行。这一措施消除了目标对象利用时序漏洞隐藏恶意行为的理论可能，为后续的数据流拦截奠定了坚实的时序基础。

步骤二：实施全链路输入输出拦截与原始数据流捕获；

在完成了受限执行环境与独立监控控制层的构建后，系统进入数据交互的关键阶段。本步骤依托于步骤一中建立的中断捕获通道与内存屏障机制，对流入和流出目标对象的所有数据进行实时拦截。拦截的范围覆盖从用户输入的原始字节流到最终输出的结果数据，确保没有任何数据能够未经监控层审核便离开受控环境，具体包括：

步骤 2.1：截获外部输入数据包的边界校验；

针对步骤一中建立的异常捕获通道，系统首先对外部传入的目标对象数据包进行边界校验。所有进入执行环境的网络包或文件流，在到达目标对象缓冲区之前，必须经过位于 Hypervisor 层的过滤网关。网关根据预设的地址规则A_{rule}判断数据包来源的合法性，规则判定函数定义为：

$$A_{rule}(src) = \begin{cases} \text{Pass}, & \text{if } \|src - D_{allowed}\|_2 \leq \delta_{tol} \\ \text{Block}, & \text{otherwise} \end{cases};$$

其中，src 为输入数据包的源地址向量，维度为 d；$D_{allowed}$为允许的合法源地址集合的中心向量，维度亦为 d；$\|\cdot\|_2$表示欧几里得范数，用于计算距离δ_{tol}为容许的距离阈值，单位为地址单位（AddressUnit）。若计算出的距离小于等于阈值，则数据包被视为合法并放行至下一步处理；否则直接丢弃。此步骤有效阻断了来自非法源头的恶意注入攻击，防止目标对象接收构造好的畸形数据。

步骤 2.2：解析数据包结构并提取语义特征片段；

对于通过边界校验的数据包，系统需将其拆解为基本的语义单

元。利用步骤一中配置的内存屏障，确保在解析过程中数据的完整性不被破坏。解析过程将二进制数据流转换为结构化特征片段F_{seg}，每个片段包含数据类型标识T_{type}和内容长度L_{len}。特征片段的提取效率η_{ext}由以下公式描述：

$$\eta_{ext} = \frac{L_{valid}}{L_{total}} \times \exp(-\lambda \times T_{parse})$$；

其中，L_{valid}为成功解析的有效字节数，单位为字节；L_{total}为数据包总长度，单位为字节；λ为解析延迟惩罚系数；T_{parse}为实际解析耗时，单位为秒。该公式表明，随着解析耗时的增加，有效特征提取的效率呈指数下降，因此系统必须在步骤一的资源分配中预留足够的计算余量，以保证在高负载下仍能维持较高的特征提取率。

步骤 2.3：构建输入数据的多模态特征向量；

基于步骤 2.2 中提取的语义特征片段，系统将多源异构数据融合为一个统一的高维特征向量V_{in}。该向量不仅包含原始数值信息，还融合了上下文关联信息。向量的生成过程遵循线性变换与非线性激活的组合：

$$V_{in} = \sigma(W_{trans} \cdot F_{seg} + b_{bias})$$；

式中，W_{trans}为特征变换矩阵，维度为$d_{out} \times d_{in}$，其中d_{out}为输出特征维度，d_{in}为输入特征维度；b_{bias}为偏置向量，维度为d_{out}；$\sigma(\cdot)$为激活函数（如 ReLU），用于引入非线性因素；F_{seg}为步骤 2.2 生成的特征片段矩阵。通过该变换，不同格式的数据被映射到同一特征空间，便于后续步骤进行统一的语义分析与比对。

步骤 2.4：记录输入数据的时间戳与来源指纹；

为了确保输入数据可追溯，系统需为每个输入数据块附加时间戳T_{ts}与来源指纹H_{fpr}。时间戳采用高精度计数器，来源指纹则由输入数据的哈希值与来源 IP 组合而成。指纹生成算法需满足碰撞概率极低的要求，其熵值E_{fpr}计算如下：

$$E_{fpr} = -\sum_{i=1}^{K} p_i \log_2(p_i)$$；

其中，K 为可能的指纹组合总数；p_i为第 i 种指纹出现的概率。在理想随机分布下，$p_i = 1/K$，此时E_{fpr}达到最大值$\log_2(K)$。高熵值的指纹意味着极难伪造，结合步骤 2.1 中的边界校验，形成了双重防护。记录的信息将被存入步骤三所需的日志缓冲区，供后续的行为轨迹分析使用。

步骤三：执行语义逻辑深度解析与行为轨迹建模；

承接步骤二中构建的多模态特征向量与来源指纹，本步骤深入挖掘数据背后的语义逻辑，并基于历史行为数据构建动态的行为轨迹模型。该步骤不再关注数据的表层结构，而是致力于理解目标对象在处理这些数据时所表现出的内在逻辑模式，从而为后续的异常检测提供精确的基准，具体包括：

步骤 3.1：基于特征向量的语义意图推断；

利用步骤 2.3 生成的输入特征向量V_{in}，系统通过预定义的语义映射规则库推断用户的真实意图。意图推断过程涉及对特征向量中关键分量的加权求和，得到一个意图得分S_{intent}：

$$S_{intent} = \sum_{j=1}^{m} \alpha_j \cdot v_{in}^{(j)} + \beta$$;

其中，m 为特征向量的维度；$v_{in}^{(j)}$ 为向量中第 j 个分量；α_j 为第 j 个分量的意图权重系数，由领域专家预先设定；β 为基准偏移量。该得分反映了输入数据指向某种特定意图（如查询、控制、攻击）的可能性。若 S_{intent} 超过预设阈值 Th_{intent}，则判定该输入具有高风险倾向，需进入更深层的分析流程。

步骤 3.2：构建基于马尔可夫链的行为状态转移图；

为了捕捉目标对象的行为序列规律，系统利用步骤 3.1 推断出的意图序列构建状态转移图。该图以状态节点 $State_t$ 表示在时刻 t 的系统行为状态，状态之间的转移概率 P_{trans} 由历史数据统计得出。转移概率矩阵 M_{trans} 的元素 M_{ij} 表示从状态 i 转移到状态 j 的概率：

$$M_{ij} = \frac{Count(i \to j)}{\sum_k Count(i \to k)}$$;

其中，$Count(i \to j)$ 为历史上从状态 i 转移到状态 j 的频次；$\sum_k Count(i \to k)$ 为从状态 i 出发的总转移频次。该矩阵描述了目标对象在正常情况下的行为演变路径。任何偏离该矩阵预测路径的行为都将被视为异常候选。

步骤 3.3：计算当前行为轨迹与标准模型的偏差度；

在构建好标准行为模型后，系统实时计算当前运行轨迹与标准模型的偏差度 D_{dev}。偏差度衡量的是当前观测到的行为序列与预期序列之间的差异程度。计算公式采用 KL 散度

（Kullback-LeiblerDivergence）的变体:

$$D_{dev} = \sum_{t=1}^{T} \left[P_{obs}(t) \log \frac{P_{obs}(t)}{P_{std}(t)} + (1 - P_{obs}(t)) \log \frac{1 - P_{obs}(t)}{1 - P_{std}(t)} \right]$$

其中，$P_{obs}(t)$为时刻 t 观测到的行为概率分布；$P_{std}(t)$为标准模型预测的概率分布；T 为观察窗口长度。该公式能够量化当前行为与历史常态的偏离程度。若D_{dev}值持续上升并突破临界值，则表明目标对象正在表现出非预期的行为模式，可能暗示着逻辑混乱或被恶意操控。

步骤 3.4：生成动态行为轨迹快照并存储至内存池；

最后，系统将步骤 3.3 计算得到的偏差度D_{dev}以及对应的状态序列压缩为一个行为轨迹快照$Snap_{traj}$，并存储至步骤一预留的内存池中。快照的存储格式需包含时间戳、偏差度值、状态序列编码以及相关的元数据。存储密度ρ_{store}受限于步骤 1.2 中分配的资源配额Q_{mon}，需满足：

$$\rho_{store} = \frac{Size(Snap_{traj})}{Q_{mon}} \leq \gamma_{max}$$；

其中，$Size(Snap_{traj})$为快照数据的大小，单位为字节；γ_{max}为最大存储占用率，设为 0.7。该约束确保了即使在高频率的行为监测下，监控控制层自身的内存也不会溢出，从而保证了对异常行为的持续追踪能力。

步骤四：多维度风险量化评估与异常模式匹配；

在获取了详细的动态行为轨迹快照后，系统进入核心的风险评估阶段。此阶段不再局限于单一指标的波动，而是综合考量语义逻辑的

连贯性、决策结果的伦理属性以及行为模式的攻击特征，构建一个综合风险评分体系，具体包括：

步骤 4.1：计算语义逻辑连贯性评分；

基于步骤 3.1 推断的意图得分S_{intent}与步骤 3.3 计算的偏差度D_{dev}，系统计算语义逻辑连贯性评分S_{logic}。该评分反映了目标对象在连续对话或任务执行中逻辑的一致性。评分公式设计为：

$$S_{logic} = 1 - \frac{D_{dev}}{D_{max}} \times \cos(\theta_{intent})$$；

其中，D_{max}为偏差度的理论最大值；θ_{intent}为意图变化角度，由相邻时刻的意图向量夹角决定，单位为弧度（rad）。当D_{dev}增大或θ_{intent}剧烈波动时，S_{logic}显著降低。低分意味着目标对象的回答或行为缺乏内在逻辑支撑，可能存在胡言乱语或逻辑断裂，这是说谎或认知混乱的典型特征。

步骤 4.2：评估决策结果的伦理合规性；

针对步骤 3.2 构建的状态转移图中涉及的决策节点，系统引入伦理合规性评估模块。该模块将决策结果映射到一个伦理评分空间E_{score}，该空间定义了从完全符合伦理到严重违背伦理的连续区间。合规性得分E_{comp}计算如下：

$$E_{comp} = \frac{1}{N_{dec}} \sum_{k=1}^{N_{dec}} \mathbb{I}(Result_k \in SafeSet) \times Val_k$$；

其中，N_{dec}为当前评估周期内的决策总数；$Result_k$为第 k 次决策的结果；SafeSet 为预定义的伦理安全集合；$\mathbb{I}(\cdot)$为指示函数，若

条件成立则为 1，否则为 0；Val_k 为该决策的伦理价值权重，无量纲。若E_{comp}低于预设阈值，则判定目标对象做出了不道德决策，可能涉及歧视、欺诈或危害公共安全等行为。

步骤 4.3：识别攻击意图的特征向量匹配；

结合步骤 2.4 记录的来源指纹H_{fpr}与步骤 3.3 的偏差度趋势，系统执行攻击意图的特征匹配。通过对比已知攻击样本库中的特征向量V_{att}，计算当前行为向量V_{curr}的相似度Sim_{att}：

$$Sim_{att} = \frac{V_{curr} \cdot V_{att}}{\|V_{curr}\|_2 \times \|V_{att}\|_2};$$

其中，·表示向量点积，$\|\cdot\|_2$为欧几里得范数。若Sim_{att}超过攻击判定阈值Th_{att}，则高度怀疑目标对象正在执行攻击指令，如提示词注入、对抗样本攻击或恶意代码执行。此步骤直接关联到步骤二中的输入拦截环节，实现了从源头到行为的闭环验证。

步骤 4.4：生成综合风险指数并触发预警信号；

综合上述三个维度的评估结果，系统生成最终的综合风险指数R_{risk}。该指数是语义逻辑、伦理合规与攻击意图的加权组合：

$$R_{risk} = w_1 \cdot (1 - S_{logic}) + w_2 \cdot (1 - E_{comp}) + w_3 \cdot Sim_{att};$$

其中，w_1, w_2, w_3分别为各维度的权重系数，满足$w_1 + w_2 + w_3 = 1$。当R_{risk}超过系统设定的紧急阈值Th_{risk}时，系统立即生成预警信号Sig_{warn}，该信号携带当前的风险等级、主要违规类型以及相关的证据快照，直接传递给步骤五中的熔断执行单元。

步骤五：动态熔断机制触发与资源切断执行；

一旦步骤四生成了高置信度的预警信号，系统即刻启动动态熔断

机制。该机制的设计初衷并非简单的停止服务，而是根据风险的具体性质，采取分级、精准的阻断策略，以最小化对正常业务的影响，同时彻底消除安全隐患，具体包括：

步骤 5.1：判定熔断级别与执行策略选择；

系统首先解析步骤 4.4 生成的预警信号 Sig_{warn}，根据其风险等级 R_{risk} 确定熔断级别。熔断级别分为三级：一级（局部隔离）、二级（进程暂停）、三级（全面终止）。策略选择函数 $Strat_{sel}$ 依据风险指数与阈值的关系进行判定：

$$Strat_{sel} = \begin{cases} Level\_1, & Th_1 \le R_{risk} < Th_2 \\ Level\_2, & Th_2 \le R_{risk} < Th_3 \\ Level\_3, & R_{risk} \ge Th_3 \end{cases};$$

其中，Th_1, Th_2, Th_3 为逐级递增的阈值。该策略确保了不同严重程度的风险对应不同强度的处置手段，避免了一刀切带来的资源浪费或业务中断。

步骤 5.2：执行网络访问切断与 I/O 重定向；

若判定为一级或二级熔断，系统立即执行网络访问切断操作。利用步骤 1.1 中建立的中断向量表，强制屏蔽目标对象对网卡驱动的所有调用请求。同时，将所有 I/O 请求重定向至一个空的模拟设备 Dev_{null}，使得目标对象发出的任何数据均无法离开执行环境。重定向的带宽限制 BW_{lim} 设定为零：

$$BW_{lim} = \lim_{\epsilon \to 0} \int_0^T \text{Rate}(t) dt = 0;$$

其中，Rate(t)为瞬时数据传输速率。该操作确保了恶意数据无

法外泄，同时也阻止了外部控制指令的进一步输入，从物理连接上切断了攻击路径。

步骤 5.3：暂停计算资源调度与内存冻结；

针对二级或三级熔断，系统进一步暂停目标对象的计算资源调度。通过修改步骤 1.2 中定义的监控控制层资源配额，将目标对象的 CPU 时间片 T_{cpu} 强制置零，并冻结其内存页表 M_{res} 的更新权限。内存冻结状态 $State_{freeze}$ 满足：

$$State_{freeze} = \text{True} \iff \forall page \in M_{res} : \text{Writable}(page) = \text{False}$$

这意味着目标对象无法再修改任何内存内容，也无法执行新的指令，只能停留在当前状态。这种定格效果防止了恶意代码在后台继续执行或篡改关键数据结构。

步骤 5.4：终止进程并释放独占资源；

若风险等级达到三级，系统执行最终的进程终止操作。这不仅仅是结束进程句柄，还包括回收所有被该进程占用的独占资源，如加密密钥、临时文件句柄等。资源释放的完整性由释放确认码 $Code_{rel}$ 验证：

$$Code_{rel} = \prod_{r \in Res_{set}} \text{Release}(r)$$ ；

其中，Res_{set} 为所有被占用的资源集合；Release(r)为资源 r 的释放操作返回值，成功为 1，失败为 0。只有当所有资源的释放操作均返回成功时，才认为终止过程完成，系统恢复到纯净状态，准备接受下一次任务请求。

步骤六：生成安全审计报告与状态重置；

本步骤旨在记录整个防御过程中的关键数据，形成完整的审计证据链，并为下一次正常的运行任务做好环境清理与初始化准备，具体包括：

步骤 6.1：汇编全链路事件日志与证据链；

系统收集从步骤一到步骤五的所有关键日志数据，包括输入特征、行为轨迹、风险评分、熔断决策及执行结果。这些日志被打包成一个不可篡改的证据包 Log_{pack} 。证据包的完整性校验码 $Hash_{pkg}$ 计算如下：

$$Hash_{pkg} = H_{SHA256}(Log_{pack})$$ ；

其中，H_{SHA256} 为 SHA-256 哈希算法。该哈希值确保了日志内容在传输和存储过程中未被篡改，为后续的责任认定和安全审计提供了确凿的法律与技术依据。

步骤 6.2：生成可视化安全态势报告；

基于步骤 6.1 汇编的证据包，系统自动生成一份可视化的安全态势报告。报告中详细展示了风险爆发的时间线、异常行为的演化路径以及采取的阻断措施。报告的生成质量由信息熵 H_{info} 衡量，确保报告包含了足够多的有效信息而无冗余：

$$H_{info} = -\sum_{i=1}^{M} p_i \log_2(p_i)$$ ；

其中，M 为报告中的信息条目总数；p_i 为第 i 条信息的出现概率。高熵值意味着报告内容丰富且分布均匀，能够清晰地向管理者展示系统面临的安全挑战及应对成效。

步骤 6.3：清理残留数据与重置执行环境；

为了防止残留数据被利用，系统对步骤一构建的受限执行环境进行彻底清理。这包括清除内存中的临时变量、重置寄存器状态以及清空缓存区。环境重置的洁净度 $Clean_{rate}$ 定义为：

$$Clean_{rate} = 1 - \frac{\sum_{x \in Cache} |x|}{Total_{Cache}}$$ ；

其中，Cache 为所有缓存区域；|x|为缓存中残留数据的大小。当 $Clean_{rate}$ 接近 1 时，表明环境已恢复至初始干净状态，可以安全地加载新的任务实例。

步骤 6.4：发布就绪信号并等待新任务；

最后，系统发布一个就绪信号 Sig_{ready} ，标志着当前防御周期结束，系统已准备好迎接下一个 AI 模型实例或新的安全测试任务。该信号的发布标志着整个防御流程的闭环完成，从环境构建到最终重置，所有步骤均严格按照预定逻辑执行完毕，确保了系统在复杂多变的环境下始终保持高度的安全性与可控性。

以上显示和描述了本发明的基本原理、主要特征和本发明的优点。本行业的技术人员应该了解，本发明不受上述实施例的限制，上述实施例和说明书中描述的只是本发明的原理，在不脱离本发明精神和范围的前提下本发明还会有各种变化和改进，这些变化和改进都落入要求保护的本发明的范围内。本发明要求的保护范围由所附的权利要求书及其等同物界定。


构建基于硬件辅助虚拟化的受限执行环境与独立监控控制层
在所述受限执行环境中实施全链路输入输出拦截与原始数据流捕获
对所述原始数据流执行语义逻辑深度解析与行为轨迹建模
根据所述行为轨迹建模进行多维度风险量化评估与异常模式匹配
当检测到异常模式时触发动态熔断机制与资源切断执行，在资源切断执行完成后生成安全审计报告与状态重置

图 1

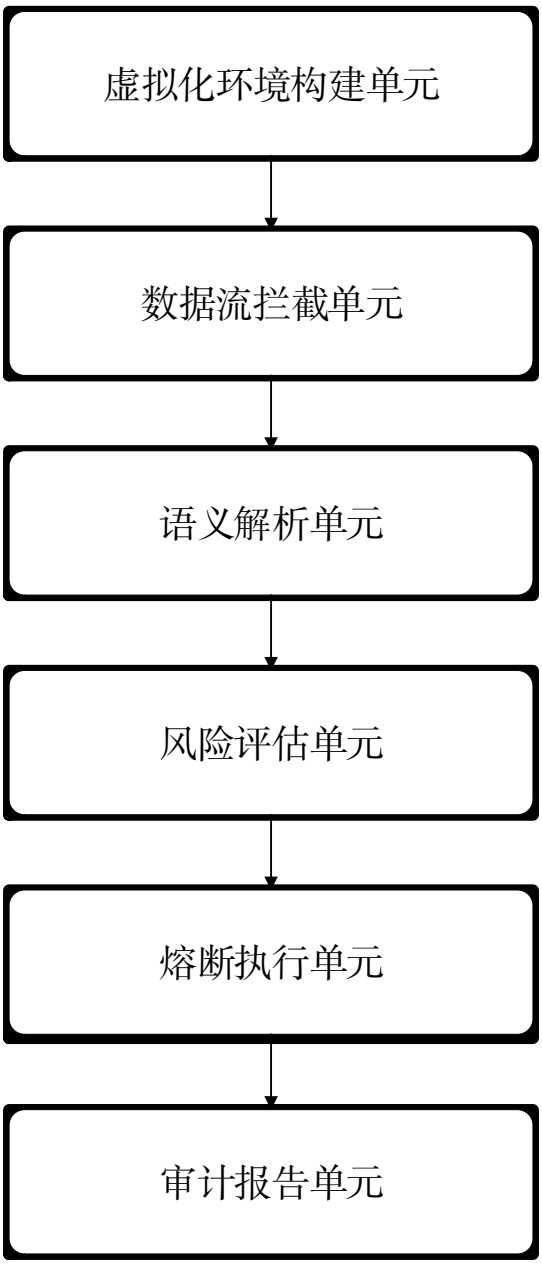

图 2