E 实践探索 xploration

安信证券容器云平台落地实践分享

梁德汉、熊国章、唐新华、段苏隆、江庆坤、陈光辉、徐凯 / 安信证券股份有限公司

容器云平台是近几年受到各行业广泛追捧的技术平台之一，是以容器和 kubernetes 编排为底座的新型 PaaS 平台，打造这样的平台会面临不少的挑战，本文描述了安信证券容器云平台的落地实践，包括平台架构设计、技术选型、安全防护、运维测试、上云推广和建设成果，最后展望平台的发展方向。希望能够给建设中的读者有所启发，共同探索云原生容器领域的最佳实践。

一、实践背景

随着虚拟化技术的发展，我司已完成计算资源虚拟化的建设进程。虚拟化技术一定程度上降低了运维复杂性，提升了资源的利用率。但业务系统的上线仍面临如下问题：

· 应用系统使用的基础软件一般功能繁多、架构复杂，部署维护门槛较高；

· 应用系统的规模越来越大，复杂的架构使得应用的安装、部署和更新也比较复杂，业务停机时间和部署成本都有所增加。已有新应用系统开始尝试容器化的形式解决此类问题；

· 在面对互联网金融等行业的激烈竞争时，业务部门的需求变化越发频繁，同时也希望 IT 部门缩短软件的交付周期；

· 以 VMWare 为代表的虚拟化技术，同样面临硬件利用率相对较低、资源分配调度相对缓慢等问题。

随着金融科技概念的兴起，IT 部门需要更好的提升研发和运维团队的生产力，从而更加灵活、高效、快速的满足业务系统需求，更好提升业务价值。容器技术的出现，在开发和运维之间搭建了一个桥梁，是实现 DevOps 的最佳解决方案。容器云平台以容器技术为基础，支持容器化应用系统运行的基础平台，以降低应用软件基础环境的复杂度，将提供容器化应用全生命周期管理，实现应用的自动伸缩、弹性扩展、灰度发布以及监控告警、自动迁移、故障自愈等功能；同

时通过动态调度容器服务的运行，尽可能地共享或平摊资源，能大幅提高基础资源的利用率，从而降低基础设施的投入，节约成本。

二、驱动力

2.1 全面上云

在金融行业，监管部门对于以云计算为代表的创新技术，一直秉持开放态度。推动“上云”的政策趋向，多年来，国家高度重视新一代信息产业的发展。国务院发布《关于促进云计算创新发展培育信息产业新业态的意见》，工信部制订云计算“十三五”规划，科技部部署国家重点研发计划“云计算与大数据”重点专项等，为云计算的发展提供顶层设计。 我司以全面上云为战略目标，“安信 IT 会坚定不移走平台化建设道路，容器 + 服务化 +DevOps 是技术中台建设的重要组成部分，也是我们团队明确的技术发展路线。”

2.2 容器的优势

安信最初使用 Docker 的历史是部分的应用使用容器的方式替换传统的 VM 部署，典型的代表系统有用户中心和安信官网，从试点的业务系统中我们尝试到了使用容器的好处：

· 更高的资源利用：容器的本质是一个独立的进程，不需要进行硬件虚拟以及运行完整操作系统等额外开销，容器对系统资源的利用率更高。无论是应用执行速度、内存损耗或者文件存储速度，都要比传统虚拟机技术更高效。因此，在单机环境下与 KVM 之类的虚拟化方案相比，能够运行更多数量的实例，而且多个容器互不影响，彼此独立。

· 更快的启动部署：传统的虚拟化技术启动系统和应用需要分钟级，容器应用共享宿主内核，无需启动完整的操作系统，因此可以做到秒级、甚至毫秒级的启动时间，大大节约了开发、测试、部署的时间。

· 更一致的运行：开发过程中一个常见的问题是环境一致性问题，常常表现为动态库、JDK、依赖等版本的差异。容器以标准的方式，使用镜像提供了除内核外完整的运行时环境，确保了应用运行环境一致性。容器可以跨平台运行，无论是物理机、虚拟机，其运行结果是一致。

· 更快的弹性伸缩：在业务高峰时刻，容器可以根据 CPU、内存、甚至业务指标进行快速扩容，提升服务能力。在业务低谷期，可以稳步降低副本数量，节约资源。

2.3 云原生基石

容器提供了一种沙盒的机制，对不同应用能够进行有效隔离。镜像是它出彩的一个设计，可以让开发者们快速部署应用。但这对大型应用管理来说，这是远远不够的，随着容器的大规模使用，容器编排显得异常重要。在云原生大行其道的今天，kubernetes 对容器的编排已经成为一种事实标准，也有人称是下一代的操作系统。

首先我们说明一下什么是云原生？ CNCF 官方定义：

云原生技术有利于各组织在公有云、私有云和混合云等新型动态环境中，构建和运行可弹性扩展的应用。云原生的代表技术包括容器、服务网格、微服务、不可变基础设施和声明式 API。

实际上，我们可以理解云原生其实是一套指导进行软件架构设计的思想，为用户指引了一条可靠的、敏捷的、能够以可扩展、可复制的方式最大化地利用云的能力、发挥云的价值的最佳路径。

而 Kubernetes 在这套设计思想所处的位置，实际上是承上启下。Kubernetes 对上暴露基础设施能力的格式化数据抽象，例如 Service、Ingress、Pod、Deployment，这些都是 Kubernetes 本身原生 API 给用户暴露出来的能力；而对下，Kubernetes 提供的是基础设施能力接入的标准接口，比如说 CNI、CSI、DevicePlugin、CRD，让

基础设施能够作为一个能力提供商，以一个标准化的方式把能力接入到 Kubernetes 的体系中。

三、建设方案

3.1 总体架构

容器云平台是一种使用容器去构建、部署和编排应用的新型 PaaS 平台，以 Docker 作为容器运行时在 Linux 环境中创建容器，以 Kubernetes 为容器编排引擎在平台中编排容器。各家的容器平台架构大体应该都差不多，我司容器云平台在测试、办公、交易等不同安全域分别部署了集群，使用 Rancher 对多套集群进行统一管理。在集群入口，我们使用 F5 进行负载。

· 分层架构：以 kubernetes 为核心组件的 PaaS 平台，整合 EFK，Prometheus，Harbor 等附加组件，PaaS 服务分为三个平面，分别为管理平面、控制平面、计算平面。

· 以应用为中心：建设开发、构建、测试、运行流水线，实现应用从构建到发布的全自动化的过程。

· 自动化运维：智能化的资源调动与分配，通过日志分析，监控指标分析，负载流量等自动弹性伸缩，减轻运维负担。

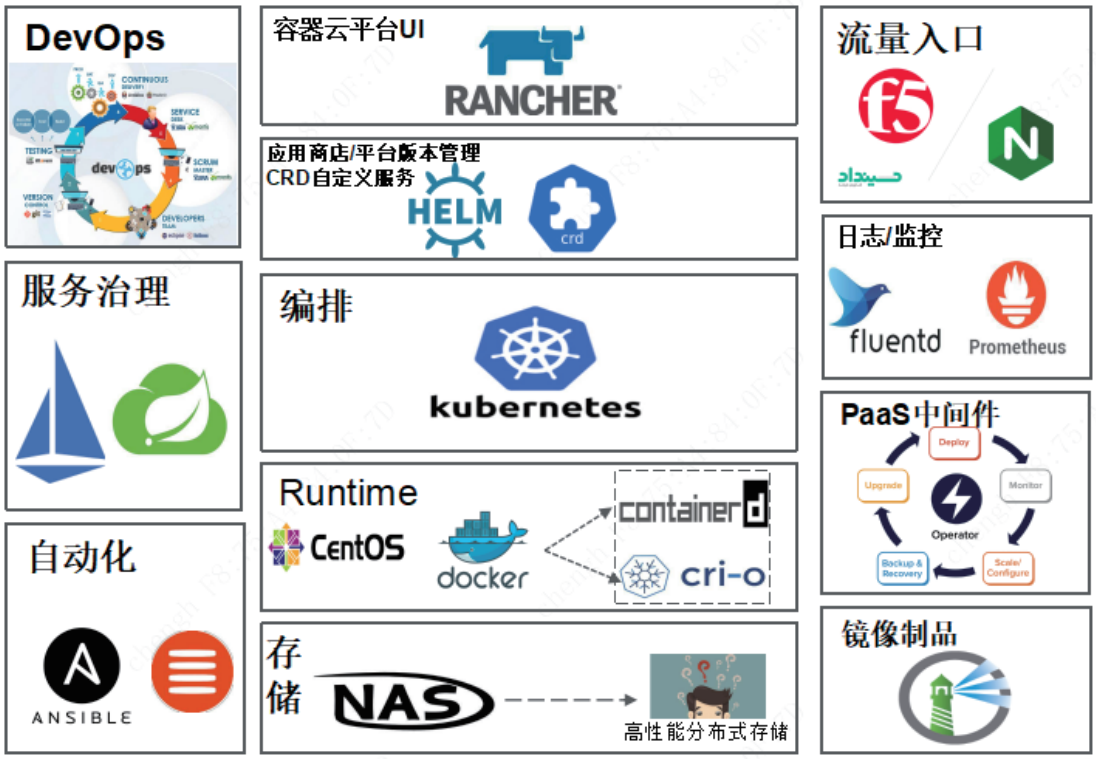
3.2 技术选型

3.2.1 镜像仓库

镜像仓库是容器云平台的建设必不可少的部分，随着我司的应用上云进程不断深入，应用镜像制品的安全性、可靠性以及易用性等问题也日益凸显。我们在建设早期阶段也尝试过其他镜像仓库产品，比如原生的 registry、nexus，但无论是可靠性还是权限管理都无法满足我司的需求，最终我们采用 Harbor 来建设我们的私有镜像仓库。

镜像仓库的架构逻辑大体如下：

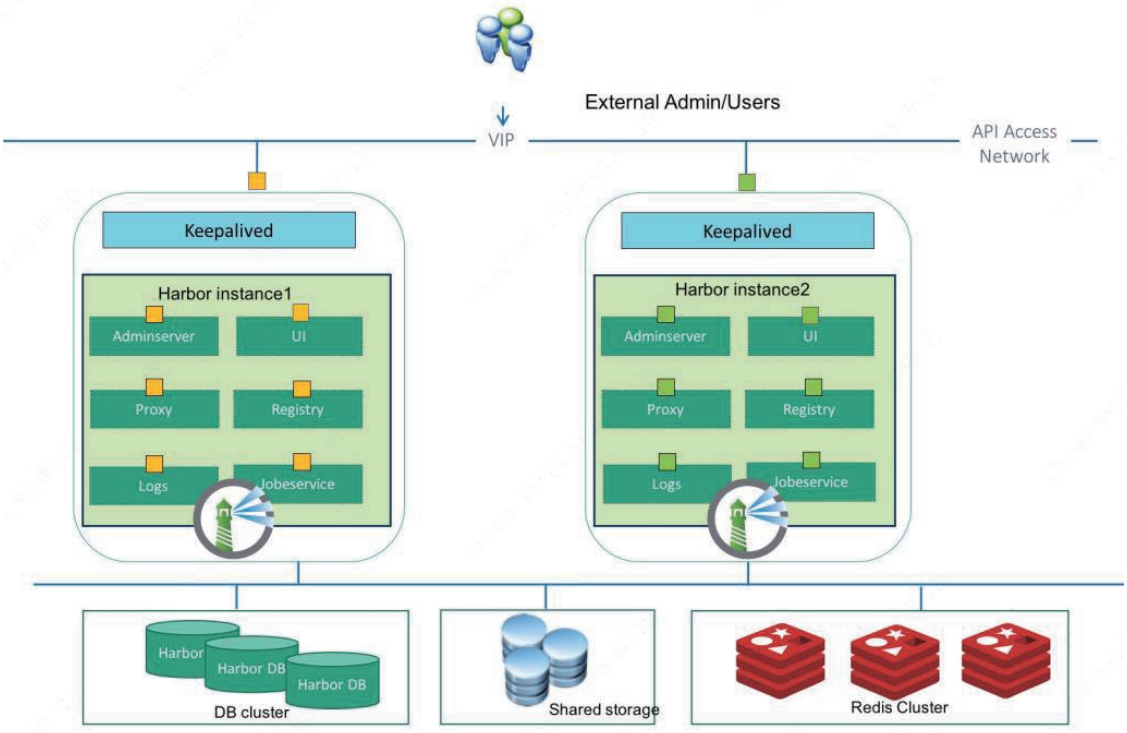
· 镜像仓库独立于容器云 Kubernetes 集群之外部署，双机部署保障高可用

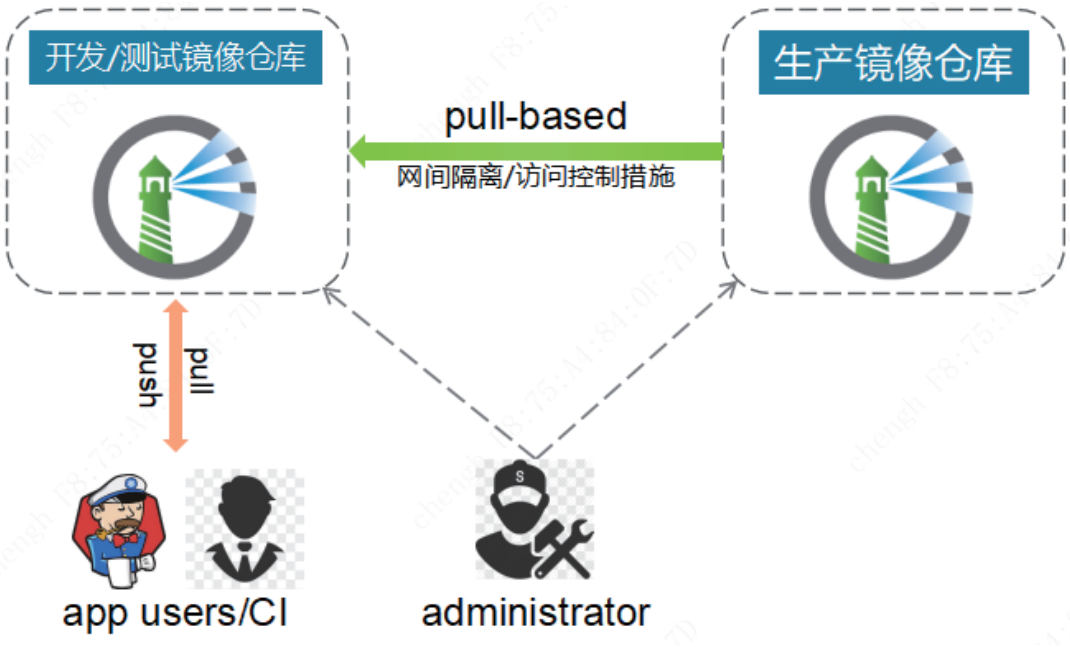


· 分建开发 / 测试，交易生产不同的环境，开发测试镜像仓库允许应用负责人随时构建上传镜像制品；而对于交易生产镜像仓库，为了保证镜像来源的安全、可控，我们限制了只能从测试镜像同步（同步方式是 pull-based）

· 生产镜像仓库只同步正式版本的镜像，应用发布需申请变更窗口，从生产镜像库中拉取镜像进行部署

· 制定策略，容器云平台发布应用的镜像来源只能是我们的私有镜像仓库





对于镜像仓库的租户权限管理，我们也定制了具体的规则：

· 定制基础镜像、公共组件镜像，开放给所有业务系统

· 不同租户（项目）的镜像相互隔离

· 租户管理员拥有对应项目的全权限（读写），租户管理员下设普通用户，如开发者账号、机器人账号等等分别对应不同的使用场景；

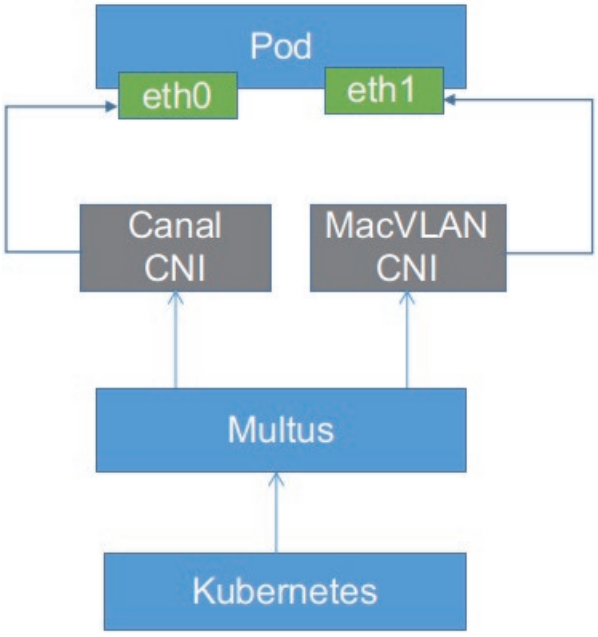
· 镜像仓提供镜像的查询、更新和删除等功能

3.2.2 网络方案

网络是 Kubernetes 非常关键的组成部分，默认的要求是每个 Node（宿主机）之间的容器网络能够联通，设计的一个基础原则是每个 Pod 都拥有一个独立的 IP 地址，而且假定所有 Pod 都在一个可以直接连通、扁平的网络空间中。所以不管它们是否允许在同一个 Node 中，都要求它可以直接通过对方的 IP 进行访问。但其本身并不提供网络解决方案，而是提供 CNI 规范给许多插件例如 Flannel、Calico、Weave 等实现，在集群上使用和部署以提供默认网络解决方案。

在金融行业，监管和安全要求更为严格，业务之间跨区访问需要在防火墙中开启白名单规则。kubernetes 提供了 Network Policy 方案，SDN 技术虽然可以使用标签选择器的方式控制应用之间的流量以及来自外部的流量，但是在我们建设初期，该项技术并未很成熟，技术上的创新更需要很长时间的论证，因此我们选择了“传统”的网络隔离方式，对 Pod 进行 IP“固定”。网络功能需要使用多个网络接口分离控制，管理和控制用户 / 数据的网络平面。为了解决这些需求，Intel 实现了基于 CNI 的网络插件 Multus，它能支持同时添加多个网卡到 kubernetes 环境中。方便用户把管理网络和业务相互隔离。MacVLAN+Canal 便是基于 Multus 的实现，在 Pod 中拥有两块网卡，其中一块网卡可以用于管理网络，底层使用 Canal 网络模式，另外一块网卡做为业务网络，底层使用 MacVLAN 模式。为保障网络的健壮性，要求管理网和业务网使用两块网卡做好 bond 并接入不同的交换机上。

下面是 Multus 实现的 MacVLAN+Canal 方案：



· Canal 网络：主机和主机间通过 Canal 网络插件建立的 vxlan 网络进行通信，主要用于应用之间管理流量。

· MacVLAN 网络：实现各个业务固定在一段 IP 范围内自动分配 IP 和 MAC 地址需求，流量不封装直接可以经过二层交换机进行转发。在我司的环境中，底层的物理机上分为两块网卡并做好 bond，其中网卡 1 做为管理网络流量，网卡 2 做为 MacVLAN 业务流量，网卡 2 交换机端口配置为 trunk 模式，并允许对应的 vlan-tag 通过。

· 默认网络：默认只有 Canal 网络，当启用 MacVLAN 时，Pod 默认网络为 MacVLAN 网络。此时，Pod 默认网络与主机网络在同一扁平网络空间。

3.2.3 存储选型

存储的核心需求是可靠、可用，应用对存储的要求是稳定、高性能，企业考虑的则是可扩展和低成本。通常我们会通过模拟各种组件异常，比如拔网线、磁盘、节点电源等方式来衡量存储

的核可靠性；模拟长时间使用的边界情况，满足的性能指标，容量使用超过 90% 下的性能和稳定性。相对传统的集中式存储，分布式存储拥有更好的可扩展性和低成本优势，前者则拥有更高的性能。

· **可靠性**：可靠性是指存储不丢失数据的概率，一般情况需要满足最大的故障节点数、最大故障盘的提前。评估方式是直接拔盘，比如说存储提供 3 副本策略，拔任意 2 块磁盘，只要数据不损坏，则说明该产品是可靠的。存储采用不同的冗余策略，提供的数据可靠性也不一样。

· **可用性**：可用性和可靠性经常被混淆，可用性是指存储是否提供 HA（High availability）和写保护机制。在抖动的异常场景，是否能够满足应用的持续访问。在机房异常断电的情况，是否能够在集群恢复正常后，数据可以正常访问。评估的方式是拔服务器电源和网线，检查是否存在单点故障或者数据不一致。

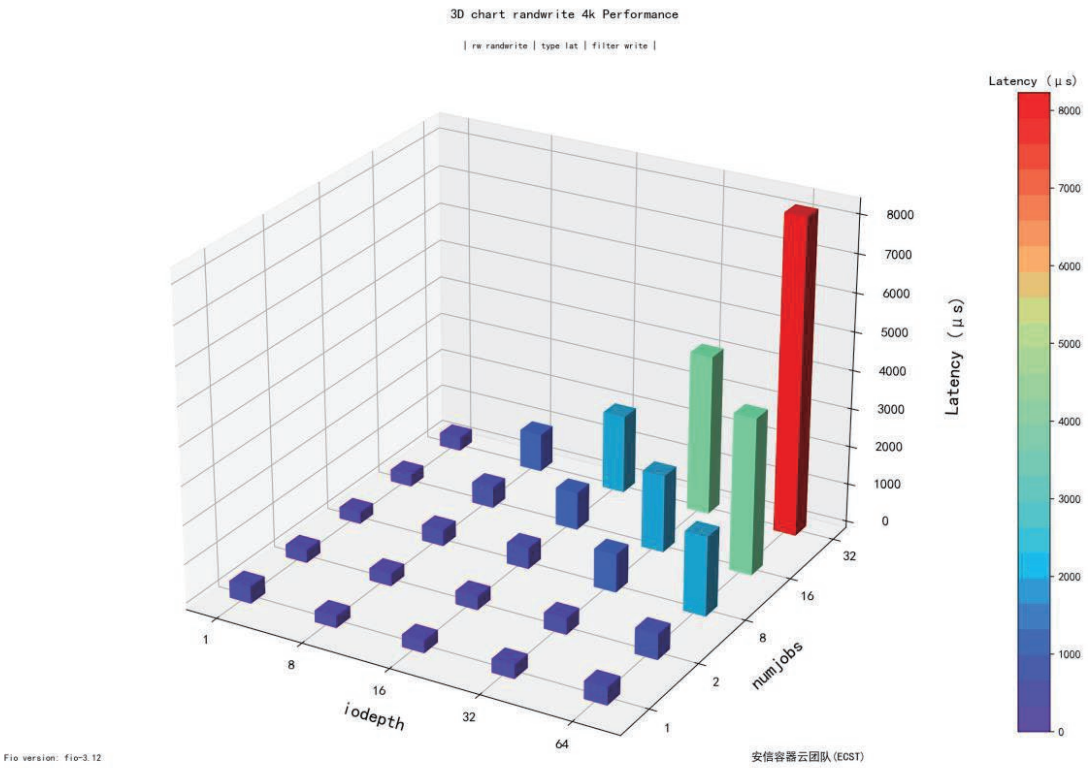
· **存储性能**：评估一个存储产品的性能主要是三大指标：IOPS、时延 (latency) 和带宽 (bandwidth)。性能测试需要一个基准，首先需要正确地描述需求，之后选择合适的工具进行持续的运行，收集数据，分析结果数据。fio 是常用的基准测试工具。通常我们会使用随机读写（rand read / write）测试 IOPS 和时延，使用顺序读写测试带宽。分析结果数据最好的方式是使用绘图工具如 gnuplot 来进行。

可以选择一组高性能的磁盘如 SSD 性能数据作为对比基准：

读性能	最高	写性能	最高
顺序读	3G/s	顺序写	2G/s
随机读	60w IOPS	随机写	20w IOPS
读时延	80 us	写时延	20 us

某存储产品在 4k 随机写下的时延性能

当然，不是说性能相比高性能的数据差太多



就无法满足业务的需求，在满足业务使用的情况下，了解与基准的差距更能准确知道存储产品的真实水平。

3.2.4 日志管理方案

我们对容器云平台的日志做了主要两个分类：

· 管理日志：Kubernetes 各个组件的日志、Kubernetes 集群节点操作系统日志、容器引擎日志、容器云平台管理 / 审计日志

· 应用日志：容器中的业务应用对在进行业务处理过程中的关键结果、状态所进行的记录

管理日志相对比较好处理，我们 Kubernetes 各个组件都是容器化部署，其他的管理日志也都有固定的输出位置，配置好规则一一收集即可。

而应用日志，在弹性伸缩、快速故障恢复和迁移、大规模微服务化部署等场景下，应用容器实例会扩展到集群中的各个节点上，应用生成的日志随之分散存放到各容器所在的主机上，这给整个应用系统的日志监控和故障排查带来极大的挑战。和很多传统的大型应用将日志持久化在本地不同，容器应用需要考虑将分散在多个容器中的日志统一收集，再汇集到外部的集中日志管理中心，以满足对应用日志的管理需求。因此我们与应用方同事沟通制定了相应的日志输出规范：

1. 日志输出到标准输入输出接口，一部分应用进行了改造，将日志输出到标准的输入输出接口

2. 日志写入到日志文件，部分没有经过改造的应用将日志直接写入到指定日志文件中

收集处理逻辑：

我们容器云平台使用 Fluentd 作为容器日志收集组件，Fluentd 使用 Ruby 语言开发的，也是 CNCF 基金会官方项目之一。Fluentd 的整体处理过程如下，通过 Input 插件获取数据，并通过 Engine 进行数据的过滤、解析、格式化和缓存，最后通过 Output 插件将数据输出给特定的终端。

因为我司已有建成的统一日志分析平台，因此我们容器云平台不再建设独立的日志平台，只需规范应用日志及收集对接即可。

对于整体的日志方案的基本要求如下（见表 1）。

针对输出到标准输出接口的容器应用日志，在 Docker 中标准输出接口日志会默认以 json 的方式存放在 /var/lib/docker/containers/ 目录，

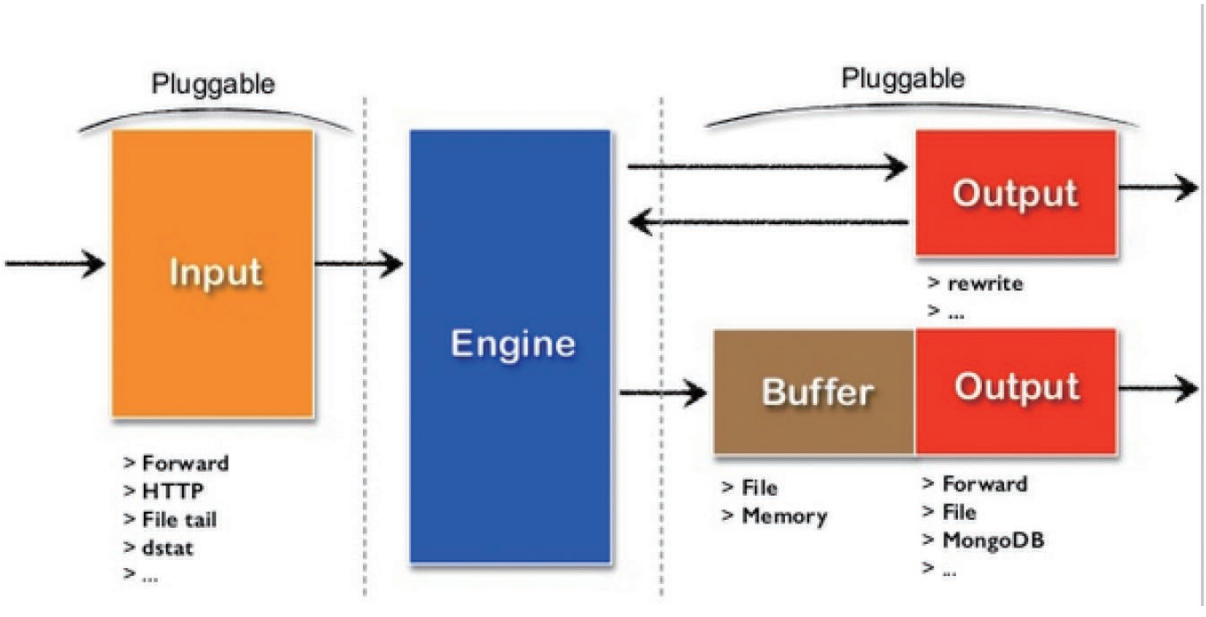


表 1

责任方	功能项	功能描述
应用	日志格式标准化	制定应用系统输出统一的标准化格式
容器云平台	容器日志收集	通过 console/volume 收集日志，要求容器云平台中内嵌采集功能，无须应用层关注；独立开发接口，收集日志时匹配项目（租户）-> namespace -> 应用三层结构
统一日志分析平台	容器日志解析	对日志进行解析后存储至日志中心
统一日志分析平台	日志查询	提供日志查询界面，做权限控制
统一日志分析平台	日志监控告警	根据设定的关键字进行监控与告警
统一日志分析平台	日志存储备份与恢复	对历史日志提供备份与恢复工具

fluentd 会自动去读取该目录下的 json 日志发送到 fluentd 中。 针对容器内对应文件日志，容器云平台上配置 workload 对应的日志目录，容器云平台会使用 Flexvolume 驱动程序来创建卷并将日志挂载到主机 /var/lib/logging/log-volumes 目录，使用 fluentd-tail 插件自动去读取该目录下的日志发送到 fluentd 中。

3.2.5 监控告警

作为继 Kubernetes 之后 CNCF 的第二个托管项目，我们同样使用 Prometheus 作为容器云平台监控建设的核心组件，对整个容器云平台进行多角度的监控。我们主要关心的核心监控维度有：

· 核心组件：etcd、api-server、controller-manager、scheduler、kubelet 等。

· 静态资源对象：节点的资源状态、内核事件等。

· 动态资源对象：Kubernetes 中的 workload，如 Deployment、DaemonSet、Pod 等。

· 应用监控：workload 需自定义的监控指标。

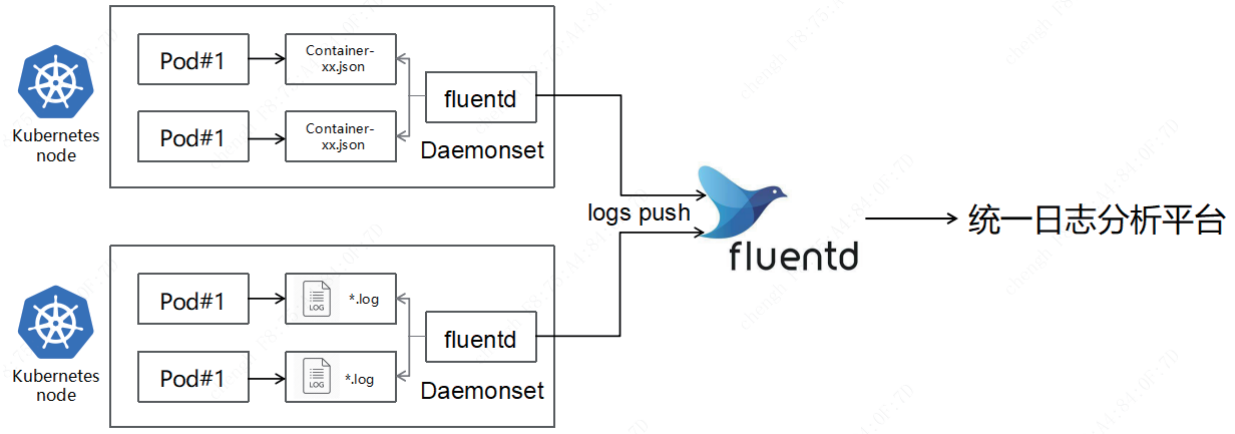
整体的监控建设体系如下：

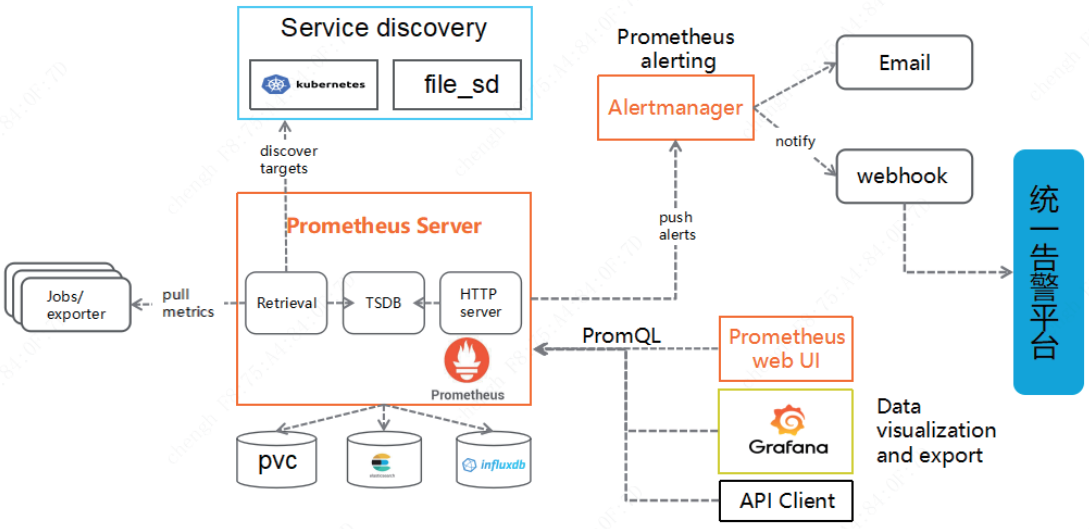
1. Prometheus 抓取到的监控数据本地使用 PVC 进行持久化，同时转存 InfluxDB 保存 180 天。

2. 对接保存数据到远程 ElasticSearch，以满足统一监控要求。

3. 自定义 Grafana Dashboard 满足日常工作需求。

4. 告警通过邮件、短信发送通知，使用 webhook 对接公司的统一告警平台。





3.2.6 容器安全

容器安全是一件复杂又重要的工作，我们联合安全团队基于容器云业务生命周期，从攻防两个视角，配合事件、风险两种驱动力，在各阶段制定了一系列的安全保障措施，包含组织、规范、流程、工具四大部分。

· **组织方面**：安全团队为项目设计、建设成员之一，运营、管理成员之一，保障顺畅的信息传递和协作。

· **规范方面**：共同制定了容器管理规范，将安全相关要求融入规范中。

· **流程方面**：在多个流程节点中加入了安全控制措施，确保安全措施能有效执行。

· **工具方面**：建立容器安全管理系统，实时对容器安全状态检查。最终各部分联动形成闭环的安全保障措施，更好的支撑容器平台稳定、安全的运行。

3.3 容器云平台测试

在上线部署运行之后，安信容器云团队对容器云平台进行了测试，并得出了最新的测试成果《安信证券容器云平台系统测试报告》，对容器云平台系统进行功能测试和性能测试验证。

测试场景通过 3 台 master 的高可用，10 台 node 的计算节点，环境如下：

Master	CentOS 7.6 处理器 8 核 内存 32G
Node	CentOS 7.6 处理器 72 核 内存 256G

按照非功能性需求制定测试指标，为测试结果是否通过提供参考依据。测试结果部分指标如表 2。

3.4 容器云平台场景应用实践

容器云平台本身是一个技术产品，如果它无法承载、赋能业务，那则没有存在的价值。因此，在技术平台竣工前期，需要和各业务系统紧密合作，提供良好的客户支持，配合业务系统平滑迁移到容器云平台。为更好的完成协同，对此，我们提出了以下要求：

· **业务要求**：业务是否适合容器化，将直接影响应用能否成功迁移到容器云平台，在建设初期，综合各因素考虑，我们采取了严格的上云条件再逐步放宽策略。

· **技能要求**：业务团队应当具备镜像制作、部署等基本能力；同时我们内部也开展培训，输出上云指南等来协助业务团队进行业务上云。

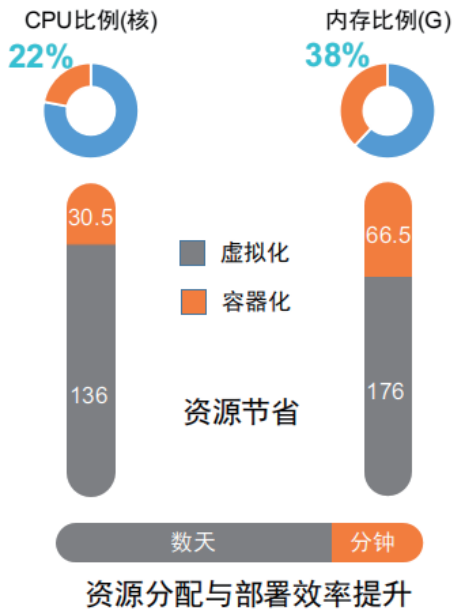
在梳理好上云要求后，我们大致圈定了上云的系统范围。截至目前为止，我司已完成了支

表 2

测试需求项	测试功能描述	测试功能指标	测试结论
主页面压测	业务成功率	≥99.9%	Paas
	最大平均响应时间	≤1s	Paas
	吞吐量	≥200	Paas
网络测试	带宽	>网卡*70%	Paas
	延迟	平均响应时间<1ms	Paas
	丢包率	丢标率<0.1%	Paas
I/O 性能测试	IOPS	≥主机磁盘 IOPS*70%	Paas
	吞吐量	≥主机磁盘吞吐量*70%	Paas
应用服务 (tomcat)	业务成功率	≥99.9%	Paas
	TPS（每秒事务数）	≥2000	Paas
	平均响应时间	≤500ms	Paas
应用服务 (springboot)	业务成功率	≥99.9%	Paas
	TPS（每秒事务数）	≥400	Paas
	平均响应时间	≤500ms	Paas

付中心系统、投资秀、互联网运营平台、条件选股、资管网站、投行存续期管理系统等 20 多套生产应用上云；计划 2021 年完成全部自研类系统上云。

在增效降本方面，与虚拟化相比，资源利用率提升是容器技术的关键优势。kubernetes 对底层的抽象屏蔽了基础设施的复杂度，将整个平台的资源进行池化，带来相当直接的好处。安信容器云开发测试集群资源利用率提升尤为明显，与虚拟化相比，CPU、内存分别只占 22% 和 38%，并随着应用项目的增多，资源利用率还会继续提升。同时，应用资源的交付耗时将降低到分钟级别，资源申请只需简单的 quota 分配即可，替代传统的虚拟化安装操作系统、配置 IP 等复杂的操作，用户获取资源的成本大大降低。此外，得益于镜像打包技术，应用部署时无需再次配置依赖环境、组件，部署效率也会大幅提高，由之前的数天降低到分钟级别。



四、未来展望

· 全面上云：我司云计算建设已经过基础设施上云，应用上云阶段。在万物上云，全面信息

化的数字化时代，下一步我们会坚定不移的坚持全面上云，上下衔接形成有机整体，夯实数字化和智能化能力，打造云上大数据、云上中台，为我司全面数字化转型继续赋能。

· 基于容器云 PaaS 中间件建设：PaaS 中间件作为一个公共的统一服务，不仅可以开箱即用，还能按需变配，有效提高我公司的资源利用率和降低成本。基于容器云的 PaaS 服务能够让业务系统无需购买业务系统专用的硬件服务器来部署中间件服务，在平台申请即用，统一运维保障服务的高可用。在业务初期，可以申请小规格实例来应对业务压力，随着服务压力和数据量增加，可以平滑升级实例规格，当业务回到低峰时，可以降级实例规格，节约成本。

· 两地三中心：以同城双中心、异地灾备中心的方案兼具高可用和灾难备份能力，我司科技园机房在去年年底已逐步投入使用，基于容器云平台的两地三中心基础设施建设亦提上了日程。从应用视角出发，两地三中心的建设并不局限于多数据中心多集群，只有从整体上解决应用双活问题，才能更好的落地两地三中心目标。

基于私有云的智能灾备中心的实践

姚玉强、周为伟 / 东方证券股份有限公司
吕爱民、严俊、黄亮 / 上海英方软件股份有限公司

本文叙述了在东方证券数据中心异构环境下，对物理机、虚拟机采用温备的方式，在私有云架构中实现统一的灾备建设管理，并通过设立独立灾备演练区，实现传统灾备演练模式由人工向自动化转型，极大地降低了灾备平台的运维成本和后期建设费用，故障恢复时间也由原来的小时级缩短至分钟级，为其他券商机构提供了可参考的项目案例。

一、实践背景

近年来，虚拟化技术的发展，为数据中心集约建设、绿色建设的要求提供了新的应用形态。从行业现状分析，当前证券公司的灾备中心通常采用虚拟化技术，实现生产端是物理机 + 虚拟机，备端以虚拟机为主的服务器配置模式。其中生产端物理机和虚拟机存在两种形态：一种是应用和数据是前后端分离部署，如银行三方存管的中间件类型（无状态、无数据），数据存放在专用的数据库系统中；一种是应用和数据一体化部署，数据在本机或以 SAN/NAS 等形式存储。

目前，证券生产中心主要以第一种前后端分离部署为主。如果按照行业监管要求，每增加一个生产系统，灾备中心对应建立一个备份系统，那么灾备中心最终形成生产系统主备 1:1 的资源配置策略。根据东方证券统计，当前实际生产环境运行的应用系统超过 300 套，备机环境异常冗余臃肿，并由此带来一系列的运维管理问题。

例如，在备机普遍采用冷备、高可用和组等运营模式下，造成备机系统重复建设、冷备虚拟机使用率低，系统处于空转待命的状态。日积月累，造成灾备中心的硬件设备、网络资源浪费严重，并加大了运维人员的工作量。同时，根据行业监管的要求，运维人员要进行周期性应急演练确保备机系统的可用性，以保证应急时的业务连续性。但是，传统的冷备方式，存在诸多问题：

第一在资源上备机会占用跟生产系统一样的

资源；

第二在实际生产应急或者应急演练时，运维人员需进行一系列复杂操作进行紧急主备切换，此时备机是否可用以及人为操作失误等问题都会影响整个应急过程以及业务连续性，且生产系统上百套，其可用性验证仅仅通过人力的话就变成了一个浩大的人力工程；

第三是备机资源池的管理，目前也仅有设备和系统管理两个维度，灾备中心运维平台缺少对备机状态、使用资源、运行情况、切换情况做统一的量化管理和资源调配，备机资源利用的高效率和数字化无法实现。

市场上有没有可以同时满足上述需求的容灾备份技术和智能运维管理平台，东方证券和英方软件经过前期的技术调研，从不同维度分析了三个常见的应用技术，并得出结果。

1.1 基于存储层的灾备方案

该方案主要是用于站点级的存储容灾。其基于卷级别的数据同步，通常要求主备两端的设备是同构环境，这显然无法满足大多数券商主备两端服务器异构的环境。此外，数据同步时，复制规则对用户不透明，用户无法按需对个别服务器或指定卷进行数据复制。更为重要的是，存储厂商一般都没有整合虚拟化平台和灾备端虚拟机演练、验证的方案，需要引入虚拟化平台厂商或第三方软件结合使用，过程复杂，可行性低。

1.2 基于虚拟化平台的灾备方案

该方案可以充分利用灾备服务器的物理机资源，且可以原生地支持生产端 VSPHERE 虚拟化平台的虚拟机备份和容灾，也可以实现隔离环境下做虚拟机备端的有效性验证和测试应用程序的数据效果。但该方案存在四个重大的弊端：

一是无法很好地兼顾物理机的备份需求，其相关的转换工具虽然能够实现 P2V 的转换，但功能较弱；

二是物理机每次更新应用程序时，都需要按照操作流程分发到灾备服务器投产完成版本的同步更新，一致性校验依靠手工完成，基于虚拟化平台的备份方案无法满足；

三是对于非中间件类型的业务系统，无代理备份方案无法满足对核心数据做实时复制，必须依赖额外的操作完成，如通过存储层的数据复制，或通过主机层安装代理程序的数据备份或复制；

四是扩展性较差，未来的灾备端（VMware）如果转用开源的虚拟化平台技术（OpenStack/KVM）或其他国产化虚拟化平台，备份系统的迁移存在较大的局限性。

1.3 基于传统备份及快照的技术方案

该方案分为两类：一类是基于文件级或应用 API 的复制技术实现对生产系统的备份，同时在灾备端的虚拟化平台，提前创建环境并构建生产服务器和虚拟机之间的数据同步规则，实现虚拟机对生产系统的应用容灾；一类是基于实时 P2V 镜像工具，将生产系统整机实时备份成虚拟机磁盘格式，当生产系统发生故障时，可直接在灾备平台启动虚拟机实现应急容灾，主要采用数据冷备和离线快照技术。该方案存在两个问题：一是对于重要的生产服务器的核心数据，定时备份存在时间窗口，RPO 无法满足业务需求；二是物理服务器每次仅限应用程序更新时，该方案无法实时地、自动地完成灾备服务器版本的同步和更新，仍然需要人工校对。

上述三种常见的技术方案，难以满足复杂异构环境下灾备中心的智能化运维需求。东方证券智能云灾备中心项目，综合利用了市场上 P2V、V2V 以及字节级数据复制技术等应用成果，成功解决了上述三个难题：异构环境的系统备份和恢复；大规模备份系统可用性验证的自动化操作；全局性的灾备中心运维管理平台、多维度的备机资源池统一和可量化管理及资源调配。

研发团队基于以上事实，设立了项目的目标，

表 1：测试项目与结果

项目	测试结果
添加备机节点	PASS
添加源虚拟机平台	PASS
添加目的虚拟机平台	PASS
添加演练平台(目的平台)	PASS
添加（源平台）虚拟机备份（支持瞬时恢复）规则	PASS
执行虚拟机备份规则	PASS
添加一次性演练规则	PASS
删除演练规则	PASS
删除演练平台(目的虚拟机平台)	PASS

并严格按照项目开发流程，展开了实践方法与思路探究、原型开发、虚拟演练中心搭建测试等论证环境，测试数据显示项目具有可行性。

二、智能云灾备中心平台实现方案

2.1 平台概述

智能云灾备中心平台汇集字节级复制、块复制、物理机和虚拟机备份及有效性验证的相关技术和原理，实现生产中心物理机或虚拟机备份到灾备中心 VMware 虚拟化平台上的目的，为运维人员提供数据实时同步、系统容灾和迁移的自动化管理，实现多维度的备机资源池的可量化管理及资源调配。

该平台的管理界面是基于 B/S 架构的软件客户端，可安装在 Windows 操作系统平台上，为运维人员提供向导式和界面化的操作管理；运维人员可通过总览界面查看当前灾备中心资源池相关数据，可通过备份界面进行文件、物理机和虚拟机备份，可通过统计报表查看备份记录、统计报告，可通过虚拟演练中心进行应急演练等。

2.2 平台架构方案

2.2.1 架构说明

本项目总体技术框架建立要遵循“整合资源信息共享”、“统一架构 业务协同”的原则，应用系统采用多层架构，以多种不同的数据复制技术和资源库为基础进行开发，实现资源和服务的共享，实现业务层和展现层的分离。总体技术框架分为五个基础层级，通过有效的层级结构划分全面展现整体应用系统的设计思路。

系统层：主要包括文件系统 I/O、系统 API、系统监控模块、文件系统层和系统网络传输系统。

核心层：主要抽象了镜像库、网络收发库、基础库、系统跨平台封装库，用于实现基于文件系统实时复制，包括系统管理模块、字节级复制、CDP 模块、高可用模块和数据库装载模块；而数据库备份库和虚拟机备份库则用于实现定时备份数据库、文件系统和虚拟机。

会话层：在 ISO 网络通信模式中 RPC 跨越了传输层和应用层；RPC 使得生成应用程序包括分布式复用程序更加容易；功能模块如节点管理、复制管理、高可用管理和备份管理等，都依赖 RPC 实现控制服务器和被管理节点的通讯。

接口层：将调用与实现解耦，不同功能模块之间的共用，不一定是共用某段代码，也可能是共用某段逻辑框架，这时候就需要抽象一个接口层出来，再通过不同的注入逻辑实现。

应用层：本课题的软件实现，同时提供多种用户界面，包括基于 Web 管理、客户端管理、RESTful API 或第三方集成能力；应用层实现相关引用组件包括工作流、表单、统一管理、资源共享等应用组件进行有效的整合和管理。

2.2.2 设计原则

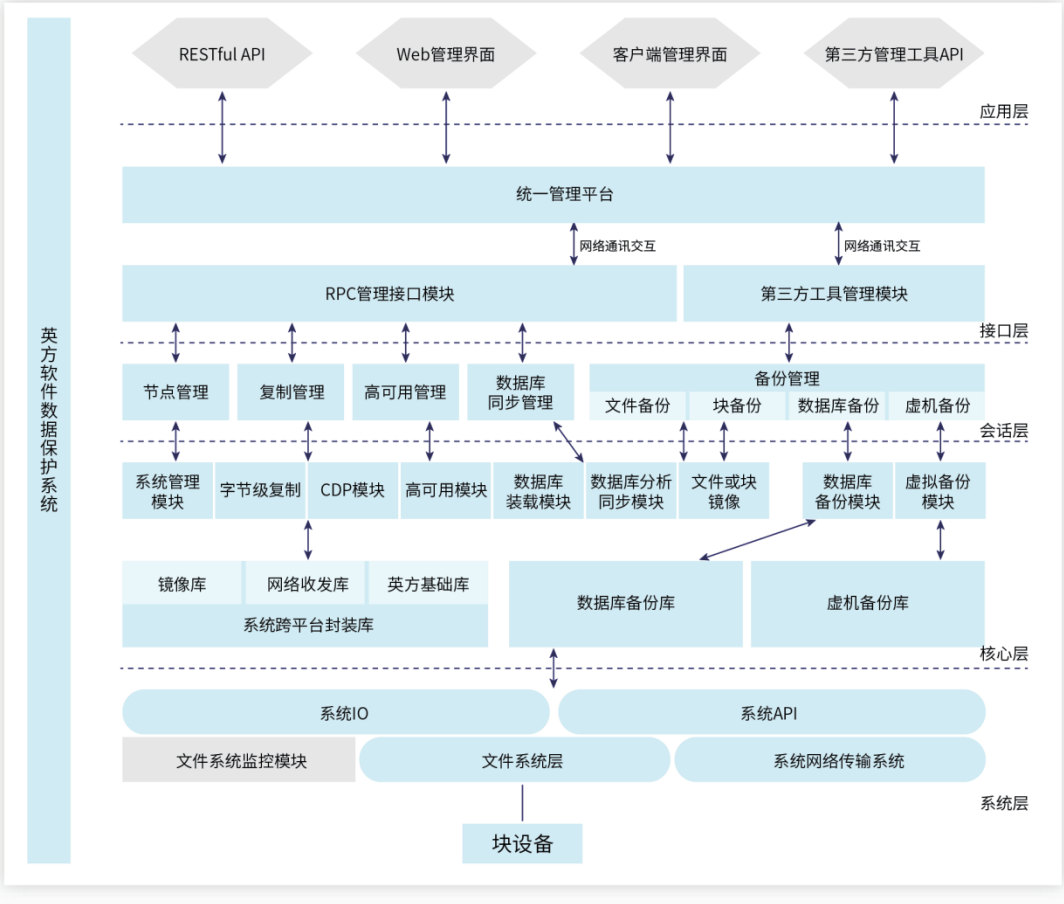


图 1 灾备中心平台架构

（1）算法服务采用统一自研框架为基础进行开发，独立部署、独立扩展、易于维护；

（2）方便整合，程序提供对外 API，可以和东方证券的监控运维系统实现对接，统一监控。

2.2.3 运行环境

适用于 X86 服务器架构；VMware VSPHERE 虚拟化平台。

2.3 备份技术实现方案

2.3.1 字节级复制技术实现

项目团队采用了字节级复制技术实现数据的快速备份，即通过将文件系统序列化 I/O 操作日志实时捕获并传输到备端，在尽可能短的时间内保证了源数据和备份数据的一致性；通过保存捕获的文件系统序列化 I/O 操作日志，保证了备份信息的完整性，使备份复制系统可以做到 I/O 级别的数据复制和恢复，以提高容灾恢复的精确性和灵活程度；并依靠保存文件系统序列化 I/O 操作的增量数据，最大限度减少对备份存储空间的要求。

本项目实现的系统包含有规则模块、操作捕获模块、本地缓存模块、本地网络模块，以及远程网络模块。其中规则模块用于指定捕获序列化操作的文件和目录；捕获模块进行序列化操作的捕获；本地缓存模块主要解决生产机系统资源和性能之间的平衡问题；本地和远程网络模块用于并行异步的网络传输。

首先用户通过规则模块下发指定文件和目录的复制规则到捕获模块，来决定捕获和传递哪些指定文件和目录的序列化操作日志，这样可以对

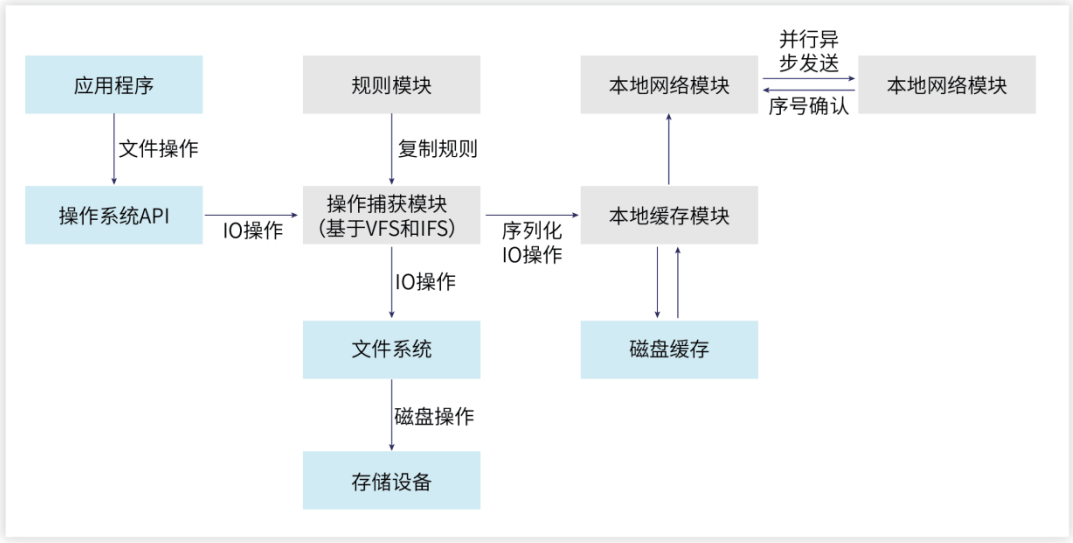


图 2：文件备份模块

各个数据流进行策略化和并行化处理。

当应用程序在对文件系统中规则内的文件或目录的读、写等访问操作时，会通过系统 API 调用传递给操作系统内核处理，在主流操作系统支持下，本装置在文件系统数据通道上加载对应的堆叠式文件系统或可加载文件系统捕获模块，截

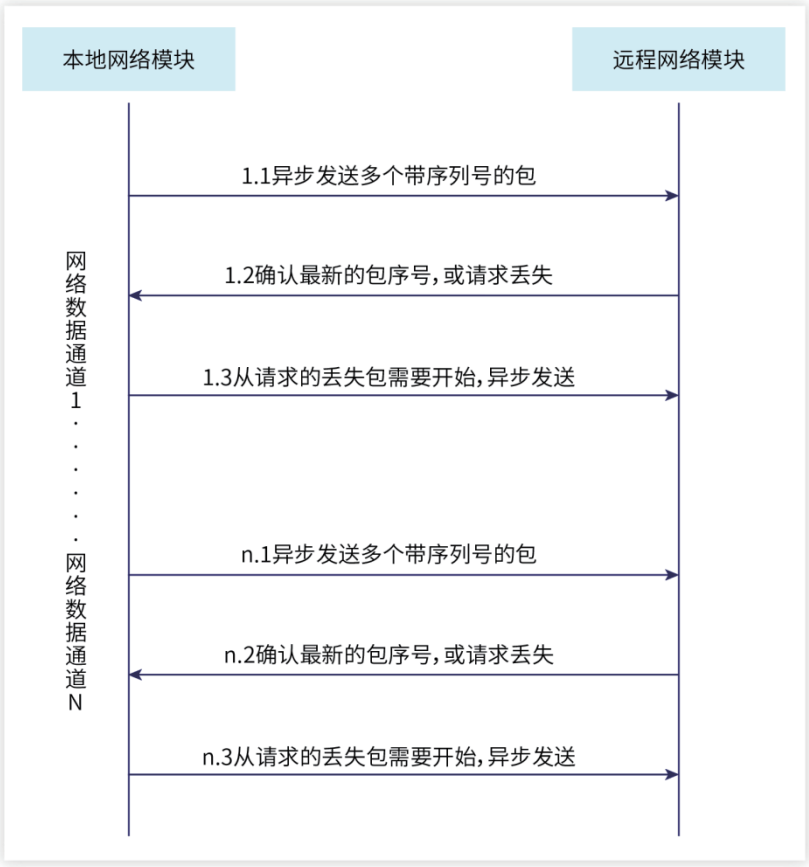


图 3：文件系统序列化 I/O 操作传输交互图

获文件操作序列化的 I/O 操作数据流或 IRP(I/O 请求包)。将各个 I/O 操作发生的时间 (when)、发起的进程 (who)、操作具体针对哪个文件 (which)、文件的具体操作位置 (where)、操作的内容 (what) 组织成序列化操作日志。

在获得序列化的 I/O 操作数据流以后，通过内存空间地址转换管道，将数据从内核态传递到用户态。缓存模块的作用是序列化 I/O 操作日志产生的速度高于网络的传输的速度时，保证 I/O 不受影响且操作的日志不会丢失。本地缓存模块根据当前系统资源状态（CPU、内存、网络等使用情况)，决定是将数据先缓存到磁盘，后期再发往网络模块处理，还是直接发往本地网络模块处理，以保证不影响本地工作机正常的生产服务。

根据策略规则在生产机和远程的备机之间建立起网络数据通道，将序列化的 I/O 操作根据规则分配给对应的网络数据通道，并根据数据通道的收发情况，决定是否需要先缓存 I/O 操作到本地存储，而发送模块会根据序列号优先将缓存处理完毕。在每个 I/O 操作包上都带有操作序号和规则信息，在传递 I/O 操作数据时，始终通过保证 I/O 操作的序号来保证文件的一致性。

2.3.2 物理机整机复制技术实现

整机备份和恢复需要把服务器上的操作系统及应用备份下来，在需要的时候，通过备份下来的数据将操作系统及应用恢复到原服务器上或者其他服务器上。

策略创建及位图构建模块，用于对源端磁盘创建保护策略，并对源端磁盘空间进行划分，根据划分结果于驱动模块 101 中构建源端磁盘的位图（ bitmap)。

有效数据分析模块 102，用于分析源端磁盘快照的分区结构，将分区信息、启动信息所在的磁盘块读取出来，再根据文件系统的接口获取文件系统的块的位图。

数据传输客户端模块 103，用于将有效数据分析模块 102 分析出来的磁盘有效数据传输到云端。在本项目具体实施例中，数据传输客户端模块 103 会对有效数据进行加密后再传输至云端 20。

数据接收服务端模块 202，用于将接收到的磁盘有效数据按照对应的位置写入到备份存储机硬盘中。具体地，数据接收服务端模块 202 于接收到磁盘有效数据后，根据有效数据分析模块

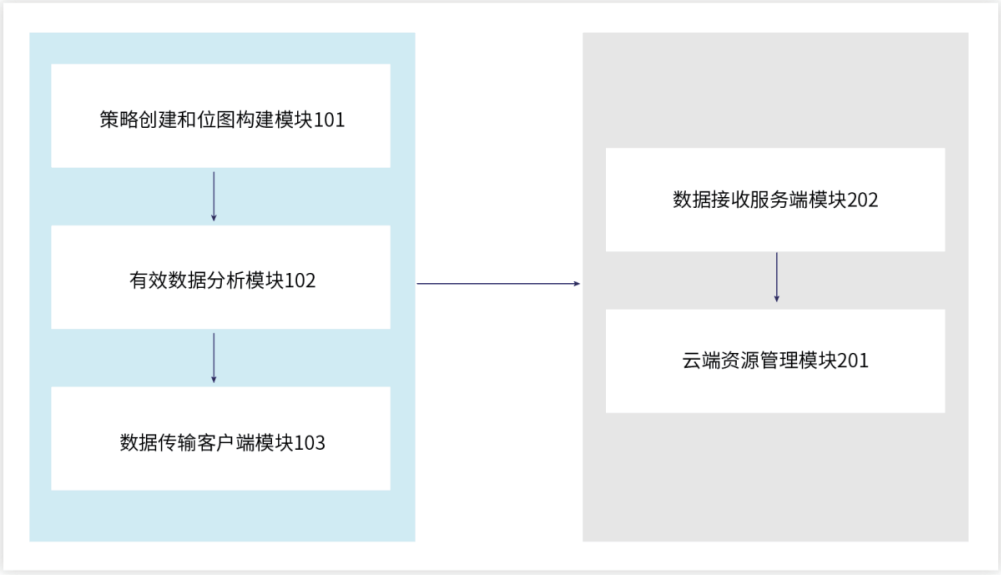


图 4：策略创建及位图构建模块

102 得到的位图将接收到的磁盘有效数据按照对应的位置写入到硬盘。

在本项目具体实施例中，由于在生产端需要捕捉数据变化的位置，当源端磁盘上发生数据变化时，首先需要明确变化数据所处的位置，因此设置驱动模块 101 位于文件系统之下和硬件磁盘驱动之上，以用来实时捕捉源磁盘上的数据变化，所述驱动模块 101 用于监控上层发往磁盘驱动上的 I/O 操作，所述驱动模块 101 的行为是实时性质的，即无时无刻都处于监控状态。

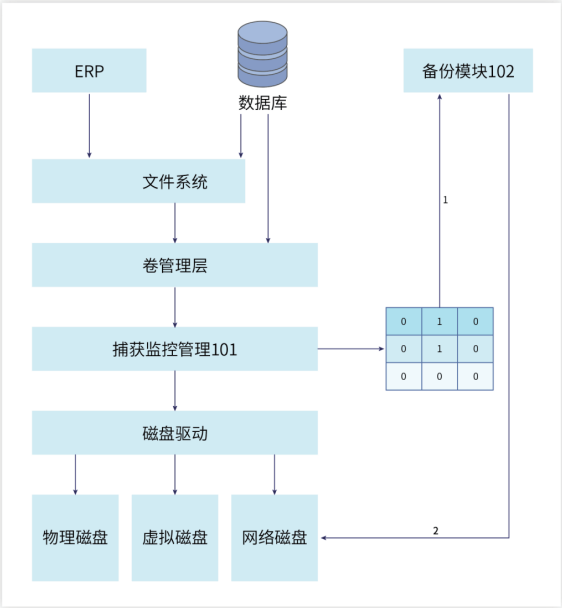


图 5：块复制捕获监控管理架构

一般来说，读操作不会产生数据变化，因此，该技术主要获取写操作产生的数据变化。也就是说，驱动模块 101 会时刻监听并捕获上层发往源端磁盘驱动的写 I/O，一旦获取，会立刻更新将其位图（bitmap）中对应的位置 1，然后根据源端磁盘的保护策略将当前的位图（bitmap）实时或周期触发条件产生时发给备份模块 102，并于成功发送之后，立刻将位图（其 bitmap）原来 1 的位置为 0。位图的每一位的值初始化为 1，规定 0 表示对应的小磁盘空间无数据变化，1 表示对应的小磁盘空间有数据变化，在初始状态，显然所有的数据都是没有同步过的，因此，会将位图的每一位的值初始化为 1。

需说明的是，在策略创建及位图构建模块 100 构建并初始化位图后，所述驱动模块 101 即会根据各源端磁盘的保护策略将该位图（bitmap）传给备份模块 102，同时将构建于驱动模块 101 中的位图（bitmap）中所有的位全部置为 0，以便实时捕捉源端磁盘上的数据变化，并将其转化成位图中的位图信息。

备份模块于接收到所述驱动模块的位图时，将当前接收到的位图与前一次接收到的位图进行整合，根据整合后的位图的位图信息将源端磁盘上相应块的数据备份到备端。这里需要说明的是，备份模块 102 中位图 bitmap 的位值从 1 置为 0 的前提是：备端将该块数据成功写入到备端磁盘上，从而确保了备份的准确性。

如图 6 所示，假设源端为一个 90MB 的磁盘的物理布局，将其从空间上划分为相等的 9 份，每一份代表 10MB 的空间，位图构建模块 100 创建相应的位图（bitmap），在进行完第一次周期全同步之后，驱动模块 101 的位图上的所有位则变为 0；当第一次往源端磁盘上更新一些数据，假设这些数据最终落盘在 10~20M 和 30~40M 的数据块上，驱动模块 101 监测到该些数据变化后，则会将位图（bitmap）上块 2 和块 4 从 0 变为 1；当第二次继续往源端磁盘上更新一些数据，假设这些数据最终落盘在 10~20M 和 50~60M 上，若之前位图中块 2 对应的位仍为 1（即未触发周期同步），则此时只需将位图（bitmap）中块 6 对应的位从 0 变为 1 即可。

这里需要介绍 CBT（Changed Block Tracking）数据块修改跟踪技术，它是 VMware 实现增量备份的底层支撑技术。CBT 的优势在于节约空间，它允许只备份发生了修改的数据。CBT 的工作原理就是让 VMKernel 监控自上次快照时间点依赖有哪些数据块中的数据被改变了，并记录下这些被改变的数据块的偏移量，依靠这些便宜量

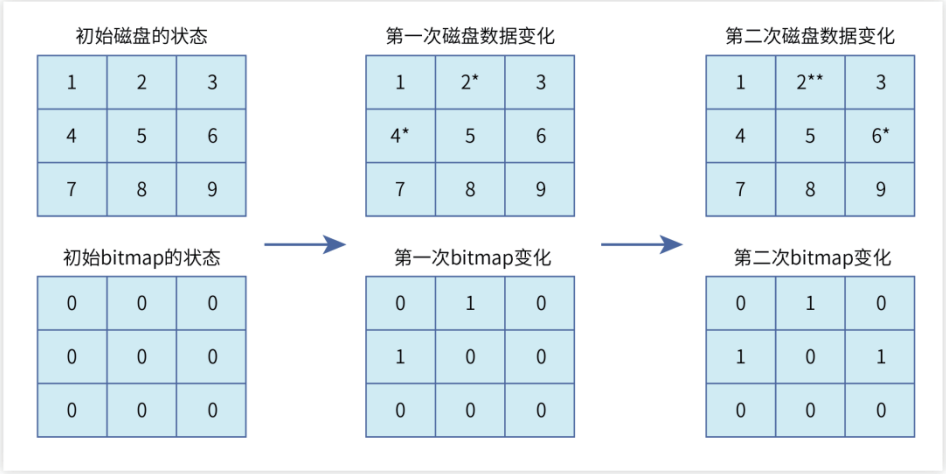


图 6：CBT 位图变化过程

获取数据块中的修改数据。CBT 是备份系统高效备份的关键，能显著提高备份速度，降低备份数据存储空间。

与现有技术相比，本项目实现的整机备份和恢复的系统及方法，基于块复制的周期同步系统及方法通过利用策略创建及位图构建模块对源端磁盘创建保护策略，将产生变化的 I/O 操作转化成位图中的位图信息传送至备份模块，由备份模块根据位图信息将源端磁盘上相应块的数据备份到备端，实现了基于块复制的周期同步目的。减少了磁盘读写资源的非必要消耗，也提高了备份的效率。驱动模块是基于磁盘块进行备份的，所以它不受文件系统的限制，可以支持各种文件系统甚至没有文件系统的磁盘也可以进行数据备份，应用面更为广泛。

2.3.3 虚拟机复制技术实现

虚拟机备份和恢复，是一种直接访问虚拟化平台获取虚拟机的相关信息并完成对单个虚拟机或多个虚拟机的备份和恢复操作。

与现有技术相比，本项目设计的虚拟机备份控制装置、系统及方法通过比较多磁盘数据间差异实现备份虚拟机，以解决大规模虚拟机备份系统中因重复读取冗余数据而导致的备份速度慢，占用存储空间大，消耗过多网络带宽资源的问题。

整个系统包含了三大部分，虚拟机备份控制

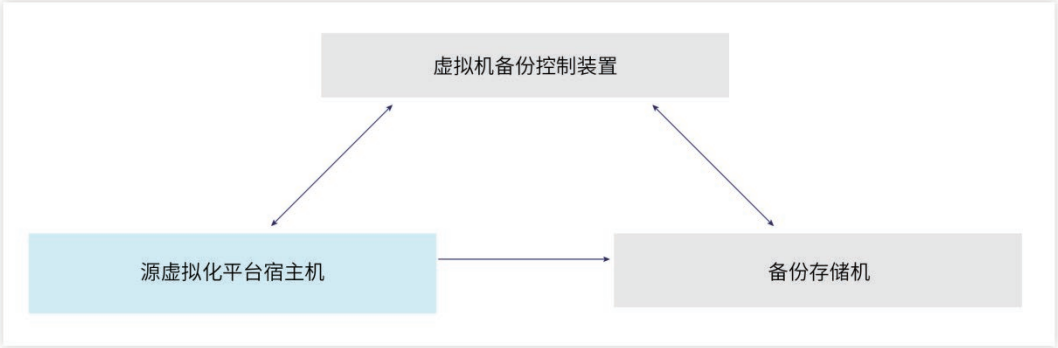


图 7：虚拟机备份架构

装置、源虚拟化平台宿主机和备份存储机。

虚拟机备份控制装置，用于从源虚拟化平台宿主机获取需要备份的虚拟机的磁盘信息，并下发对指定磁盘创建快照任务。

源虚拟化平台宿主机，负责响应虚拟机备份控制装的查询请求并提供备份虚拟机磁盘信息，根据虚拟机备份控制装置下发的指定磁盘创建快照任务对指定磁盘创建快照以保证备份过程中母盘数据的一致性，并根据磁盘快照获得全量 / 增量数据传输至备份存储机。

备份存储机，根据所述虚拟机备份控制装置下发的备份指定磁盘任务，对源虚拟化平台宿主机发送的相对应的磁盘进行数据拷贝，并将已备份磁盘信息发送至所述虚拟机备份控制装置。具体实现过程如下：

虚拟机备份控制装置启动一次虚拟机备份或复制任务，通过 IP 网络访问源虚拟化平台宿主机获取有待备份的虚拟机磁盘信息并创建临时快照。获取的虚拟机配置信息，包括每个虚拟机所对应的磁盘列表以及每块磁盘所对应的磁盘识别符以及磁盘 hash 校验值或 CBT 变化块信息。

在上述过程中，如果是首次执行就先新建备用虚拟机，否则就直接基于已经存在的备用虚拟机。虚拟机备份控制装置根据已获得的备份虚拟机磁盘信息以及已备份磁盘信息进行比对，查找是否存在重复磁盘，若查找到重复磁盘，于获得的备份虚拟机磁盘信息中删除所重复的磁盘设备，若未找到重复磁盘，则根据比对结果形成全量备份列表或增量备份列表。

从备份存储机获取已备份磁盘信息，通过遍历查找所述备份虚拟机磁盘信息及所述已备份磁盘信息中是否存在重复磁盘识别符的磁盘设备以及磁盘数据一致性比对，删除重复磁盘设备，根据比对结果获得全量备份列表及增量备份列表，根据全量备份列表及增量备份列表下发备份指定磁盘任务。

其次，源虚拟化平台宿主机根据磁盘快照获得磁盘数据后，调用磁盘接口，将整个链上的磁盘映射为一个块设备；然后调用 C 语言的 open 函数，将块设备映射为文件指针上的磁盘地址空间；接着调用 C 语言的 pread 或 fread 函数，读取文件指针内的数据，循环读取：每次根据 Offset 偏移量读取一小部分数据，直至读取全部数据；最后调用网络收发库，将读取到的数据发送给备份存储机。

然后，备份存储机调用磁盘接口，创建一个

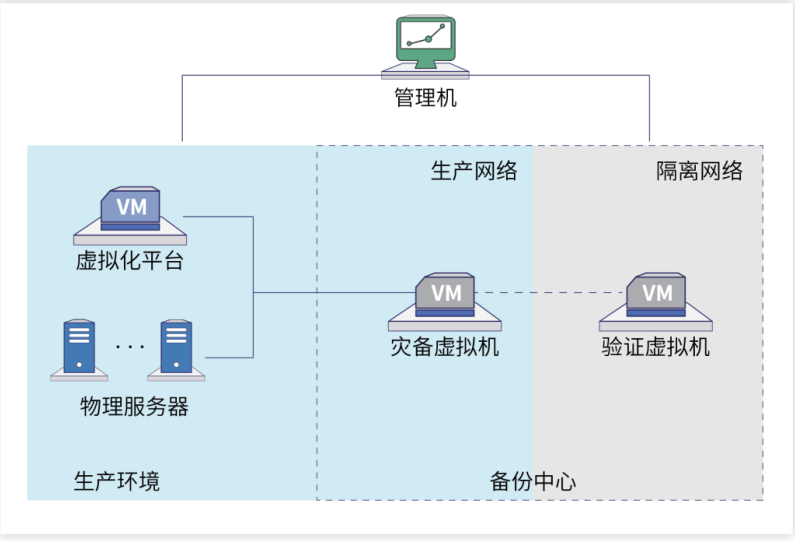


图 8：整机备份有效性验证系统架构

空磁盘；然后调用 C 语言的 open 函数，将块设备映射为文件指针上的磁盘地址空间；调用 C 语言的 pwrite 或 fwrite 函数以及网络收发库，将通过网络收发库收到的数据循环写入磁盘：每次根据 Offset 偏移量写入一小部分数据，直至写入全部数据。

最后，虚拟机备份控制装置下发指令，通知源虚拟化平台宿主机删除源虚拟机的临时快照，完成本次虚拟机的备份复制任务。

在本项目具体实施例中，所述的磁盘接口包括但不限于：Qemu 模拟器提供的 qemu-img 命令行接口；VMware 提供的 VDDK 接口；HyperV 提供的 VirtualDisk 接口；以及各类厂商基于上述接口封装或二次开发实现的其他接口。

2.3.4 备份有效性验证的实现

通过前面的技术研究，可以实现对物理机和虚拟机的整机备份和还原。随着备份数据的增多，用户需要知道灾备中心上保存的备份数据有效性，才能确保灾难发生的时候，灾备中心保存的备份是真实可用的，恢复后的数据是准确的，恢复的应用是能正常运行的。

本项目设计和实现的基于虚拟化技术保护数据的有效性验证系统，包含源虚拟化平台和备份管理机。

图 8 为系统架构图，其包括：源虚拟化平台下生产虚拟机、生产网络、备份虚拟化平台下备份虚拟机、隔离网络、验证虚拟机以及备份管理机。

整个验证过程是，在备份虚拟化平台上创建与源虚拟机相同配置的虚拟机，备份源虚拟数据到备端虚拟机，创建快照作为还原点，另外在备份虚拟化平台上，创建验证机，同时指定还原点快照文件作为当前磁盘文件，将验证机设置到隔离网络内，通过隔离网络内的管理机对验证机执行验证操作，并记录验证结果，从 而达到备份数据的有效性验证的目的。图 9 为本项目实施过程中创建验证虚拟机的流程示意图，其过程如下：

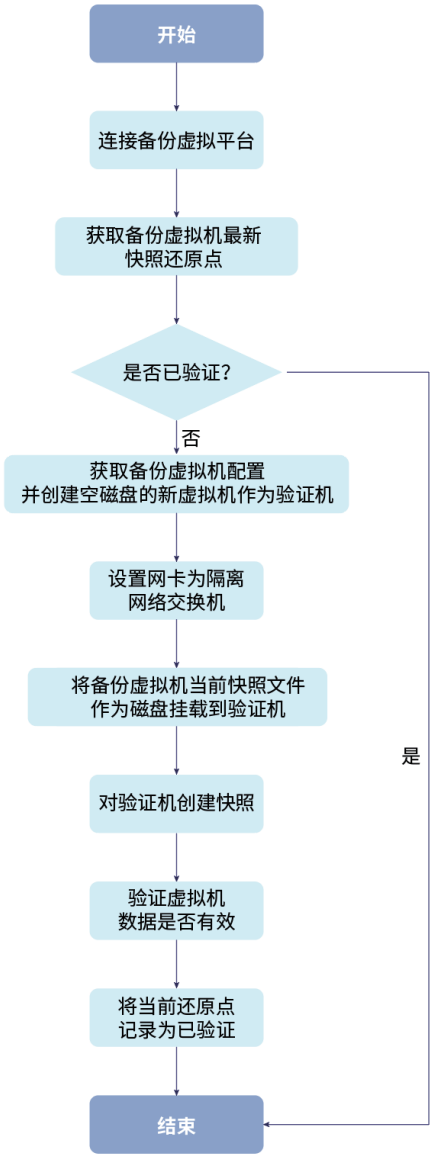


图 9：创建验证虚拟机的流程

首先，连接备份虚拟化平台，获取待验证的虚拟机快照还原点信息。

其次，判断当前快照还原点数据是否已经执行过验证，判断方法为对比管理机内本地记录，每次验证完毕后，保存当前快照点信息到管理机内。如果当前还原点未验证，则针对当前还原点执行验证过程，获取备份虚拟机配置，调用备份虚拟化平台，根据此配置创建空磁盘的新虚拟机，作为验证机。

然后，将验证机网卡设置到隔离网络内交换

机上。接下去，将备份虚拟机的快照还原点对应的磁盘文件，作为新磁盘挂载到验证机上。再者，对验证机创建快照。

最后，开启验证机，执行验证操作，记录当前还原点已经验证。图 10 为本项目实施过程中验证虚拟机执行的流程图，其过程如下：

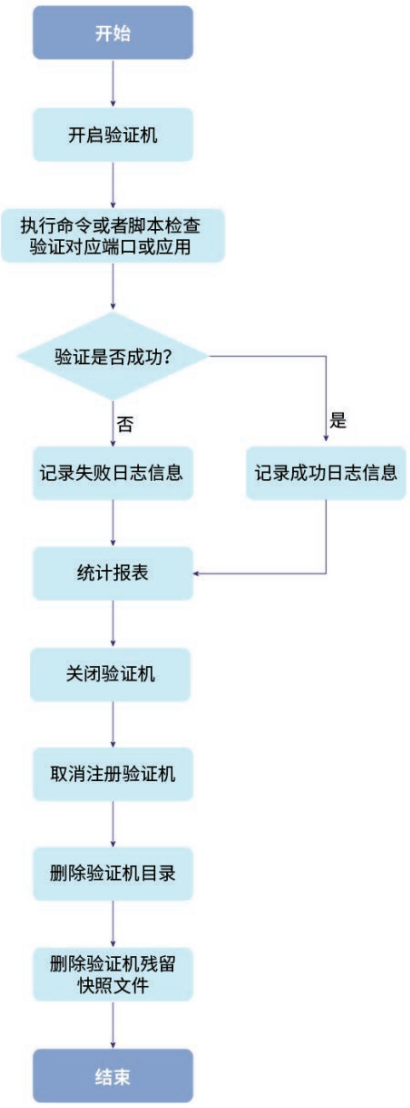


图 10：验证虚拟机执行的流程

首先，将创建的验证机开机。然后，在管理机上通过源生产机器的 IP，访问到隔离网络内的验证机，执行命令或验证脚本，检查验证机上的应用或端口是否正常工作。执行的验证结果，成功或者失败的日志记录到管理机的统计报表中。

其次，关闭验证机，取消备份虚拟平台上注册的验证机，删除存储上验证机对应目录下的虚拟机相关文件。

最后，找到存储中备份虚拟机目录下的验证机残留的快照文件，删除这些残留的快照文件。因为验证机磁盘是关联的备份虚拟机的磁盘，所以新创建的快照文件，默认在备份虚拟机目录下。

综上所述，本系统提供的虚拟机创建和验证方法，可以很好地将验证机设置到隔离网络内，通过隔离网络对验证机执行验证操作。验证机开机不影响生产环境，IP 地址无需配置，保持生产 IP，无需配置网络映射关系，从而实现了全自动化有效数据验证的功能，并且可以设置自动验证脚本及策略、输出演练结果报表供企业参考和评估。验证过程中对源虚拟机备份机制无影响，可以同时执行备份任务。在实际应用环境中是具有较实用的价值。

2.4 资源复用规则引擎实现方案

平台可从业务系统维度出发，对灾备中心的相关数据资源包括镜像资源、VMDK 数据、资源配置、应用环境配置进行分类数据展示。其中可以对虚拟化平台 CPU、内存、存储使用资源、剩余资源和各类业务系统资源覆盖情况进行展示，对集群中每台 Exsi 主机的资源使用情况可以展示。

2.4.1 资源复用规则

为了更好的实现资源复用，本课题设计了资源复用规则引擎，通过该引擎对虚拟化平台 CPU、内存等重要资源进行动态规划，并对资源使用情况和各类应用系统资源覆盖情况进行展示。首先对公司所有业务系统的备份方式、服务时间及服务等级等相关信息进行采集，结合如部分业务系统仅在交易时间段提供服务，部分企业内部系统的系统等级不高等证券行业业务特性进行数据建模。

图 11：资源分配规则引擎设计

按业务系统等级分为 A、B、C 三类，设置业务系统覆盖率，A:a　B:b　C:c，假设可用资源 x，各类业务系统所需资源 A:YA 、B:YB 、C:YC，各类业务系统实际覆盖率计算公式如下：

$$A：\frac{x}{YA} \quad (1)$$

$$B：\frac{(x-YA*a)}{YB} \quad (2)$$

$$C：\frac{(x-YA*a-YB*b)}{YC} \quad (3)$$

服务器恢复的时候需从资源池申请资源进行分配，从而将备份镜像在资源池中拉起恢复，此时资源的分配会根据规则引擎计算的覆盖率来判断资源是否足够，且按比例进行分配。

2.4.2 基于规则分配资源

配置资源，配置备机虚拟资源所需要的 CPU、内存 IP 访问资源，硬盘资源使用（基于现有的 VMDK 文件或者新建硬盘资源），定义配置所属的业务系统，定义资源使用等级并按照时间的维度进行划分。

可以根据配置的资源向灾备中心申请资源，灾备中心根据资源复用规则进行试算，符合使用要求的即完成资源的交付，并自动创建虚拟机。

基于 VMDK 文件创建的虚拟机，可以基于验证结果对硬盘文件的可用性进行校验；可以配置如端口访问等检查规则，并在虚拟机成功激活后按照访问规则进行验证。

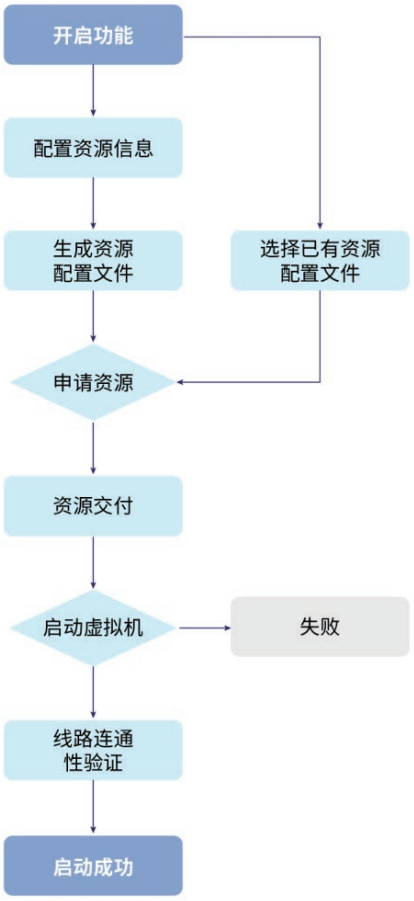


图 12：虚拟机启动及资源配置流程

2.5 平台部署方案

本项目已在我司生产环境部署运行，界面功能清晰，统计数据精准，具体的架构部署与功能实现如下：

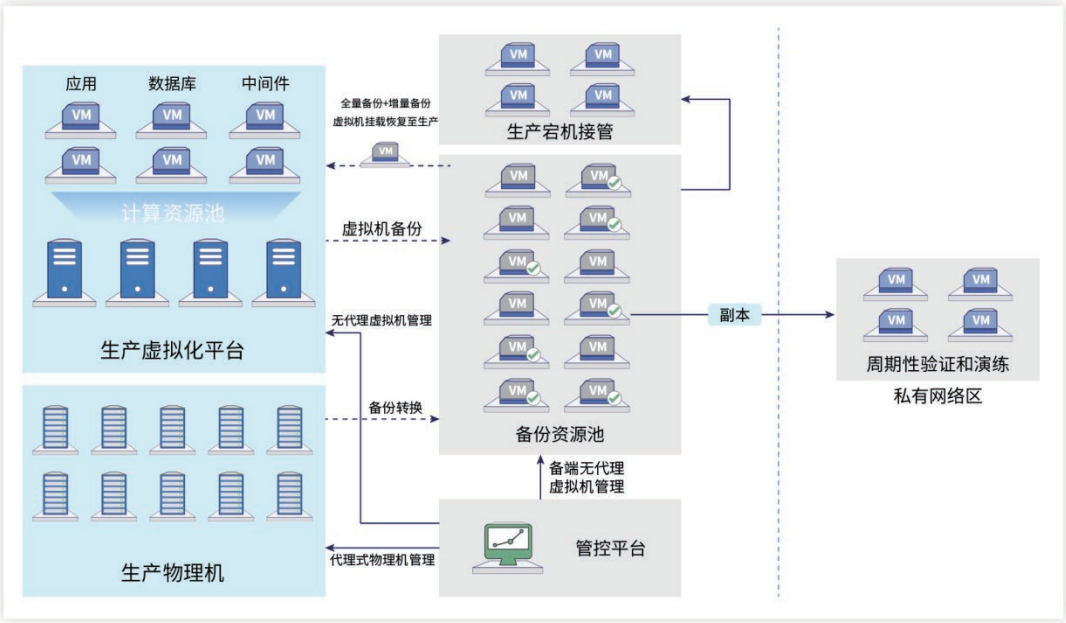


图 13：平台部署架构

在生产区域同时包含 VSPHERE 虚拟化平台和生产物理机，拥有独立的计算资源池和存储资源池，涉及各类应用系统、数据库系统和中间件系统。

备份资源池，即本项目重点建设内容，通过物理服务器和存储部署 VSPHERE 虚拟化平台。在备份资源池提供的计算和存储资源上，通过单独的虚拟机部署智能云灾备中心的“管控平台”。备份资源池提供的存储资源通过 SAN 挂载到生产虚拟化平台。实际过程如下：

针对生产虚拟化平台上运行的虚拟机，管控平台通过无代理的虚拟机备份技术进行备份，采用每周全量备份加每日增量备份的策略保存到 SAN 协议挂载的备份资源池存储资源。当需要进行应急接管时，生产虚拟化平台可以直接加载备份资源池上的虚拟机备份副本。

针对生产物理机，管控平台通过部署代理程序的方式，完成将整机备份到备份资源池。备份过程中不影响生产物理机的业务运行。运维人员可以采用每周全量备份加每日增量备份的策略，保存到 SAN 协议挂载的备份资源池存储资源。当需要进行应急接管时，管控平台通过在备份资源池上提前创建的虚拟机，自动加载生产物理机保存在备份资源池上的整机备份，快速启动并完成业务应急接管。当生产服务恢复上线后，运维人员通过管控平台，可以将正在备份资源池上运行的应急接管服务器中的生产数据，实时反向同步到生产物理机，并在维护时间段完成交割将系统切回到生产物理机。

在日常的运维管理工作，运维人员还可以充分利用备份资源池上创建备份数据的验证。包括生产物理机或虚拟机的备份的有效性，通过自动化的操作方式，在隔离网络内周期性地执行整机备份的有效性和验证。执行时对原有生产虚拟化平台的备份，以及生产物理机的备份完全没有影响。

三、智能云灾备中心场景实践

3.1 系统整机保护

智能云灾备中心打破传统一对一高可用保护方式，综合了整机备份、虚拟机备份和字节级复制等技术，将物理机通过转换成可在虚拟化平台

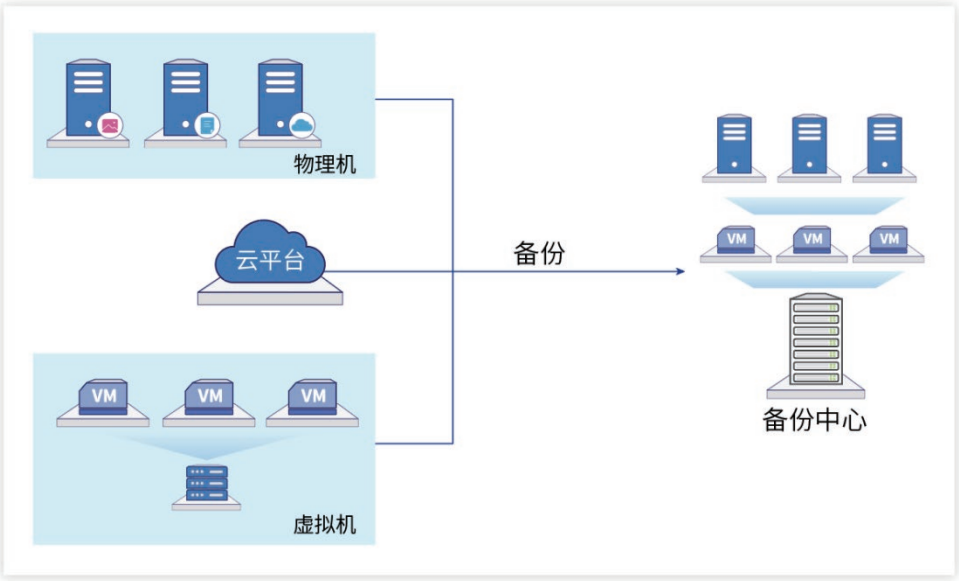


图 14：异构平台系统备份

运行的 VMDK 文件，整机备份到 VMware 虚拟化平台上，虚拟机则以无代理方式备份到 VMware 虚拟化平台上，整个过程以准实时速度进行。在正常情况下，灾备中心的备份系统不启动，不占用计算资源，且管控平台完全基于 B/S 架构，操作简单，全面兼容各种服务器，备份的系统也可以恢复至其他服务器。

该方案具有极大灵活性和创新性，在生产系统发生故障的时候，运维人员点击虚拟化平台上的备份文件即可快速启动系统，降低了 RTO 值。同时由于备份系统属于无开机状态，无需消耗额外的 IP 地址，当系统接管时，IP 地址自动漂移过来。更重要的是，备份文件只占用灾备中心的存储资源，不占用计算资源，减少计算资源使用。

例如 100 台生产系统，在灾备中心只占用 100 台系统的存储资源，当设置允许所有系统同时发生故障的最大概率为 20% 的时候，即可以减少 80% 计算资源的软硬件设备成本，且当系统故障恢复时，备份系统的资源释放后，可以给其他需求复用，做到资源共享，具有非常高的经济性和安全性。

3.2 虚拟演练中心

智能云灾备中心通过设置专门的演练中心，在生产环境网络之外，为了解决备份系统之间 IP 冲突的问题，建立了私有网络作为独立的演练网络。演练准备环节，假设将生产系统 A 作为自动演练的目标系统，在灾备中心克隆一个备份系统 A1，同时通过 A1 在演练环境克隆出备份系统 A2。演练开始后，启动自动演练程序，可通过两种方式进行演练自动化：

一是网络验证，分两种，一种是 Ping 备份系统 A2 本身；一种是通过在演练环境创建代理网关，让备份系统 A2 与代理网关相互 Ping；

二是自定义脚本验证，通过自定义脚本程序验证备份系统 A2 的可用性。

验证演练结束时，平台会自动输出可用性验证报告。如果验证成功，则会输出成功提示，表明系统能开机运行；如果验证失败，则输出失败的原因。演练结束后，备份系统 A2 会自动关机，然后系统自动删除，释放资源。

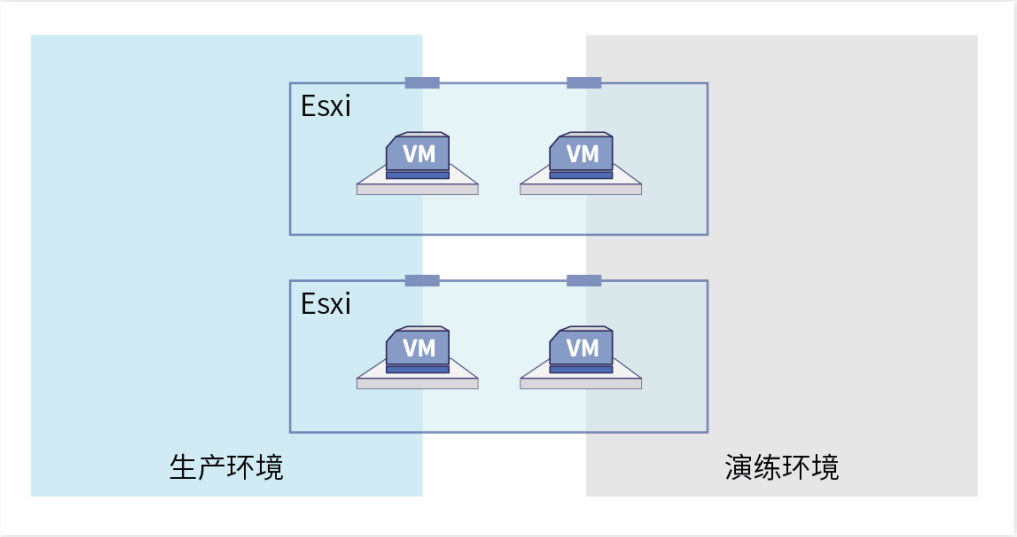


图 15：生产环境与演练环境进行物理隔离

该演练方案具备极大创新性和竞争力，可以在不影响生产系统的情况下，实现平台化的大规模虚备份系统的自动化验证，适合应急演练、可用性验证、测试、复盘等场景。演练平台基于 B/S 架构，简单易用，易维护，可降低运维人员的工作量，达到智能化运维目标。

3.3 快速搭建测试 /UAT 环境

基于智能云灾备中心，针对已有测试环境或 UAT 环境的应用系统，一方面用户如需要在两个环境中部署一样的应用系统，则可通过灾备中心实现应用系统快速迁移，另一方面，如应用系统长期处于停止状态，则可释放所有计算资源，只占用存储资源，当用户需要重新部署时则可通过灾备中心直接将系统拉起。通过这种创新的做法，复用计算资源，减少重复部署应用的劳动量，提

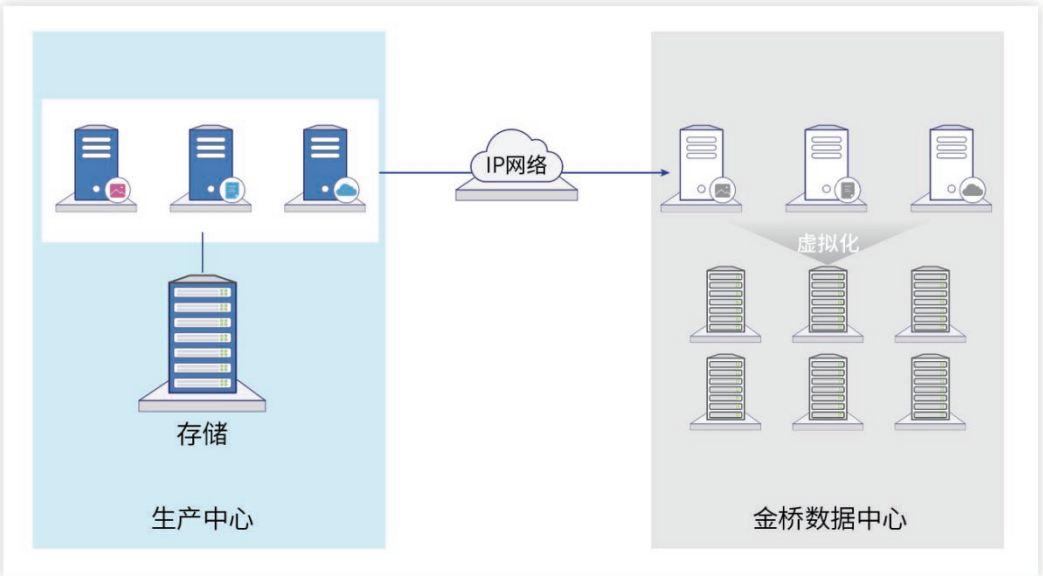


图 16：系统在线热迁移

高工作效率。

3.4 业务系统在线热迁移

智能云灾备中心备份方案结合了全服务器备份、虚拟化备份、块复制和字节级复制等技术，通过 P2V 或 V2V 的方式，在不停机的情况下，一键自动化将系统迁移到金桥数据中心，且迁移过程中 IP 地址和新增数据可实时同步过去。

本方案的系统迁移工具功能强大，自动化程度高，无需人为部署应用系统，两边数据中心无需相同的存储、硬件配置，只需要相同的操作系统，且系统迁移过程对生产系统性能影响小，不需要停机；支持上千台系统及异构硬件的迁移；支持跨云及虚拟化平台迁移；整个迁移过程可加密传输，安全可靠，实现源机与目标机的数据同步，迁移完成后可快速启动系统。

四、总结

本项目从立项到原型开发、测试验证，到方案界面设计和功能实现，到最后落地部署运行，整个过程科学严谨，务实高效。在项目技术和功能实现方面，团队大胆应用新技术与新方案，成功建成了行业首个智能云灾备中心，实现异构环境下的整机和虚拟机备份，实现业务恢复 RTO 降低，以及应急演练和可用性验证演练的自动化，极大地降低了运维人员的工作量，并通过灾备中心基于 B/S 架构的管控平台，对设备和系统数量、备机状态、使用资源、运行情况、切换情况、演练验证等做统一的量化管理和资源调配，实现灾备中心的智能化运维。

实践方案具有创新性、稀缺性、实用性和参考价值，普遍适用于证券机构特别是中大型券商机构灾备中心的建设。

证券公司智能客服云平台探索与实践

肖钢、徐政钧、潘建东、刘逸雄 / 中信建投证券股份有限公司 xuzhengjun@csc.com.cn

在全球数字化经济的大浪潮下，人工智能、区块链、云计算和大数据等技术的发展不断冲击着证券行业的服务模式。证券公司的财富管理业务逐步由线下转移至线上，员工通过 APP、H5 网页和公众号等渠道向客户提供证券经纪、证券投资咨询、融资融券、个股期权以及各类金融产品介绍等服务。因此，搭建客服平台为员工与客户提供咨询通道成为各大券商经纪业务的一项重点工作。

传统客服平台通常使用单体式架构，利用即时通信技术建立员工与客户的沟通渠道。随着技术的发展与架构的演进，传统的客服平台架构已难以满足高可用和稳定性的需求。近年来，微服务架构与云原生技术快速发展，并逐渐成为系统建设的主流技术。微服务架构与云原生技术能够有效提高系统稳定性与可扩展性，为智能客服云平台建设提供了成熟的解决方案。本文阐述客服平台的架构设计与智能客服云平台建设的探索实践，最后针对券商智能客服云平台的建设提出实践建议。

一、概述

近年来，在证券行业利润空间逐渐缩窄、经纪业务佣金收入持续下降的趋势下，众多券商开始进行业务模式转型，逐渐将线下业务转移至线上，以探索新的客户服务方式。在线客服作为一种重要的服务方式，逐渐被各大券商投入实践。在线客服采用即时通信技术建立员工与客户之间的通信网络，通过预定义的报文格式实现信息交互。随着用户量的不断增加，传统的在线客服系统的单体式架构无法满足高可用、稳定性和可扩展性的需求。

另一方面，云计算的不断发展促进了软件开发与部署模式的创新，也使其成为承载各类软

件应用的重要基础设施，是信息化与数字化发展的必然趋势。云计算最初通过虚拟化技术将物理机划分为多个虚拟化资源，分割了计算机内部的实体结构。而随后出现的 OpenStack 等技术实现了对于虚拟机集群的统一管理和分配。近年来，Docker 等容器技术的出现使得虚拟机更加轻量级、细粒度化，具有和物理机几乎持平的运行效率。Kubernetes 等容器编排技术提供完善的集群治理能力。上述云原生技术的不断推陈出新，使得软件设计架构不断优化，从集中化逐渐迈向分布式，以云计算为基础的微服务架构成为目前主流的架构解决方案。

目前，在线客服逐渐走向智能化，以 NLP 技术为核心的智能客服可节省大量人力成本，并且可为客户提供全天候的服务，因而成为各公司客户服务系统建设的重点。如何利用云计算技术与微服务架构，构建高可用、可扩展的智能客服云平台，成为证券公司亟需解决的问题。本文将介绍中信建投证券对于全渠道智能客服云平台的架构探索，并阐述架构探索中的实践与经验。

二、云原生技术介绍

2.1 Docker 容器技术

Docker 是采用 Golang 语言开发并遵从 Apache 2.0 协议开源的一个容器引擎，是目前主流的基于容器技术的轻量级虚拟化解决方案。Docker 以容器作为资源划分与调度的最小基本单位，可用于构建与发布分布式应用。与虚拟机不同，Docker 采用 Linux 内核的 Namespace 实现资源的隔离，采用 LXC 技术实现虚拟化，并通过 Cgroups 实现用户层面的资源管理，可以实现安全的运行时虚拟环境。Docker 实现了软件的快速交付，用户只需将应用程序打包为镜像并上传至仓库，使用者通过下载并运行镜像即可开始使用，省去了复杂的软件环境配置过程。Docker 等容器技术已逐步代替虚拟机成为云计算的主流技术。

2.2 Kubernetes

Kubernetes 作为一个分布式的大规模容器管理技术，可以用于容器集群的自动部署、运维和扩缩容。Kubernetes 具有完备的集群治理能力，包括安全防护与准入机制、服务注册与发现机制、负载均衡、故障检测与自我恢复、自动扩容机制和细粒度的资源管理能力。Kubernetes 提供了强大的自动扩容能力，可实现应用程序规模的快速扩大。同时，Kubernetes 通过故障检测及自我恢复等能力为用户提供安全可靠的服务运行环境。Kubernetes 作为目前主流的容器编排管理技术，已成为云计算资源管理的核心解决方案之一。

2.3 微服务架构

微服务架构是一项在云中部署应用和服务的新技术。在微服务架构中，一个大型复杂软件应用由多个微服务组成。系统中的各个微服务可被独立部署，各个微服务之间是松耦合的。每个微服务仅关注某项独立的功能模块。微服务架构具有快速部署、高扩展性、高容错性和去集中化等特点。常用的微服务开发框架如 Dubbo、Spring Cloud 和 Istio 等，下面针对 Spring Cloud 和 Istio 进行简要介绍。

Spring Cloud 是一系列框架的有序集合。它利用 Spring Boot 的开发便利性巧妙地简化了分布式系统基础设施的开发，如服务发现注册、配置中心、消息总线、负载均衡、断路器、数据监控等。Spring Cloud 作为一套成熟的分布式服务治理的框架，已得到了广泛的应用。

Istio 是基于 Service Mesh（服务网络）概念设计的一个管理微服务的开放平台，通过 Agent 代理形式来提供服务发现、负截均衡、限流、链路跟踪和鉴权等微服务治理手段。Istio 作为一种微服务框架与服务治理框架，拥有丰富的路由规则、完善的访问控制和安全的服务认证。

微服务架构已经逐渐成为系统开发的主流架构方案，客服平台的微服务化能够有效提高客服

系统的稳定性和可扩展性，在证券市场快速发展、客户规模与日俱增的情况下，能有效提高系统的高可用性，保障客服服务质量。

三、智能客服云平台架构

3.1 在线客服

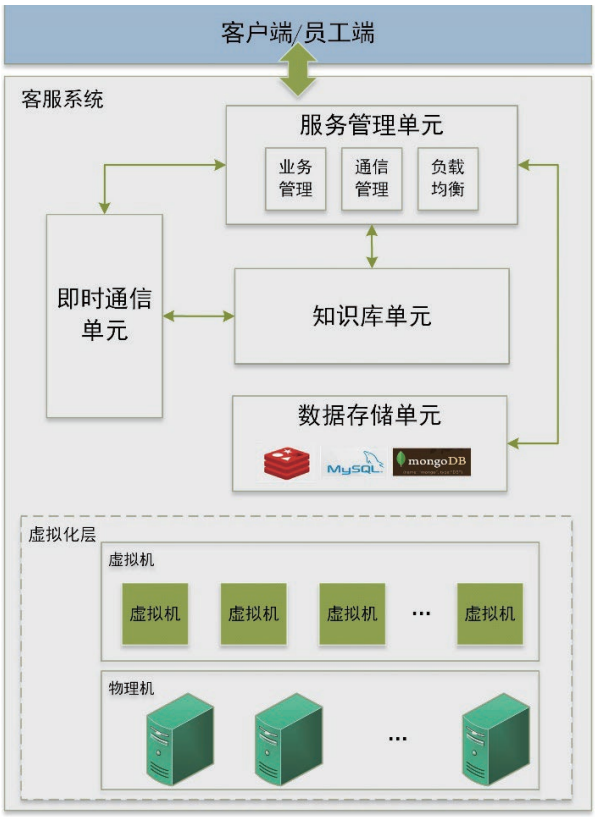


图 1：客服系统早期架构图

在线客服平台通常使用即时通信技术，通过构建网站或软件系统实现员工与客户的沟通。在线客服平台通常包括即时通信单元、客户端、员工端与知识库等核心单元。即时通信单元通常采用 XMPP、MQTT 或 Socket 等通信协议实现。早期客服平台建设多以单体式架构为主，即时通信单元负责实现消息传输，用户通过客户端向系统发起提问，员工通过员工端进行问题解答，知识库通过预存的标准问答辅助员工进行客户服务。

图 1 展示了中信建投证券早期客服系统架构，服务管理单元是系统的核心后台程序，采用 SpringMVC 实现业务层逻辑与其他组件的交互管理，知识库与即时通信单元负责实现对话管理与知识服务，数据存储单元负责缓存及数据持久化。早期系统建立在虚拟机或物理机上，在单体式架构时仅一个后台负责全部响应请求。后期通过多虚拟机部署来分担响应请求，但单机故障则会导致部分请求无法得到响应。

早期在线客服平台致力于建立沟通渠道，在性能方面欠缺考虑。单体式架构在客户规模不断增长时会出现服务器负载过重的情况，甚至出现宕机、无法响应等问题，严重影响客户的服务体验。通常单体式架构也难以实现同城多活、异地灾备等故障应对手段。此外，伴随着数字化的发展，深度学习、自然语言处理等技术能有效应用于客户的智能化服务，同时也对平台架构提出了严峻的考验，面对大量计算如何有效提升系统稳定性，成为客服平台建设的重点。

3.2 智能客服云平台架构

随着智能化技术与云原生技术的不断发展，将系统云化成为一种实现高可用、快速扩展的成熟解决方案。为保障客服系统的稳定性，提升系统计算能力，智能客服平台逐步架构改进，最终的架构模块如图 2 所示。

客户端与员工端负责客户与员工接入，采用 H5、小程序和 APP 等形式。服务网关作为接入的核心组件，提供用户登录、安全认证、负载均衡、降级熔断和黑白名单等功能。由于采用微服务架构，后台服务具有多实例，负载均衡可通过调度策略将请求分摊至各实例，防止所有请求访问同一实例而造成服务拥堵。在微服务架构中，为防止出现“雪崩效应”，当服务不可用时则通过网关进行服务降级或熔断，这是一种有效的微服务链路保护机制。同时，网关提供限流作用，当服务访问量超过阈值时会对流量进行拦截，防止服务器高负荷。安全认证采用 CAS 与 Token 的方式，员工需通过 CAS 统一认证系统进行单点登录，登

录之后颁发 Token，员工的请求需附带 Token 值，网关对员工鉴权后可进行后续操作。黑白名单则通过网关的过滤功能实现，对黑名单客户进行 IP 访问限制，防止恶意请求的发生。本文设计的智能客服平台采用 Spring Cloud 架构，服务网关使用 Gateway 二次开发，其他常用例如 Netflix Zuul 均可实现。

智能客服云平台中，服务管理单元作为核心单元，提供业务管理、智能服务、API 管理及服务注册等功能，该单元集成 Nacos 服务注册与配置管理组件，同时实现业务 API 与智能 API 的统一管理，是系统业务对外唯一的入口。数据存储单元提供 Redis 缓存，结构化数据及非结构化数据的存储功能。智能路由单元为员工与客户之间连接提供匹配算法，即在需要人工客服时通过算法规则为客户匹配具有问答权限的员工。即时通信单元负责建立客户与员工的连接，实现对话管理、消息记录管理等一系列标准化 IM 管理功能。

智能化组件包括机器人服务单元、自然语言处理单元与智能知识库。机器人服务单元通过语义搜索、情感分析、会话意图识别等智能应用组件实现智能聊天机器人，自然语言处理单元则提供基层算法库，例如聚类、SVM、CNN、RNN、Bert 等神经网络与智能化算法。智能知识库以图数据库与知识图谱为基础，构建知识搜索、推理和计算等知识服务，为智能客服云平台提供专业的知识服务。

上述单元通过交互共同实现智能客服，对外

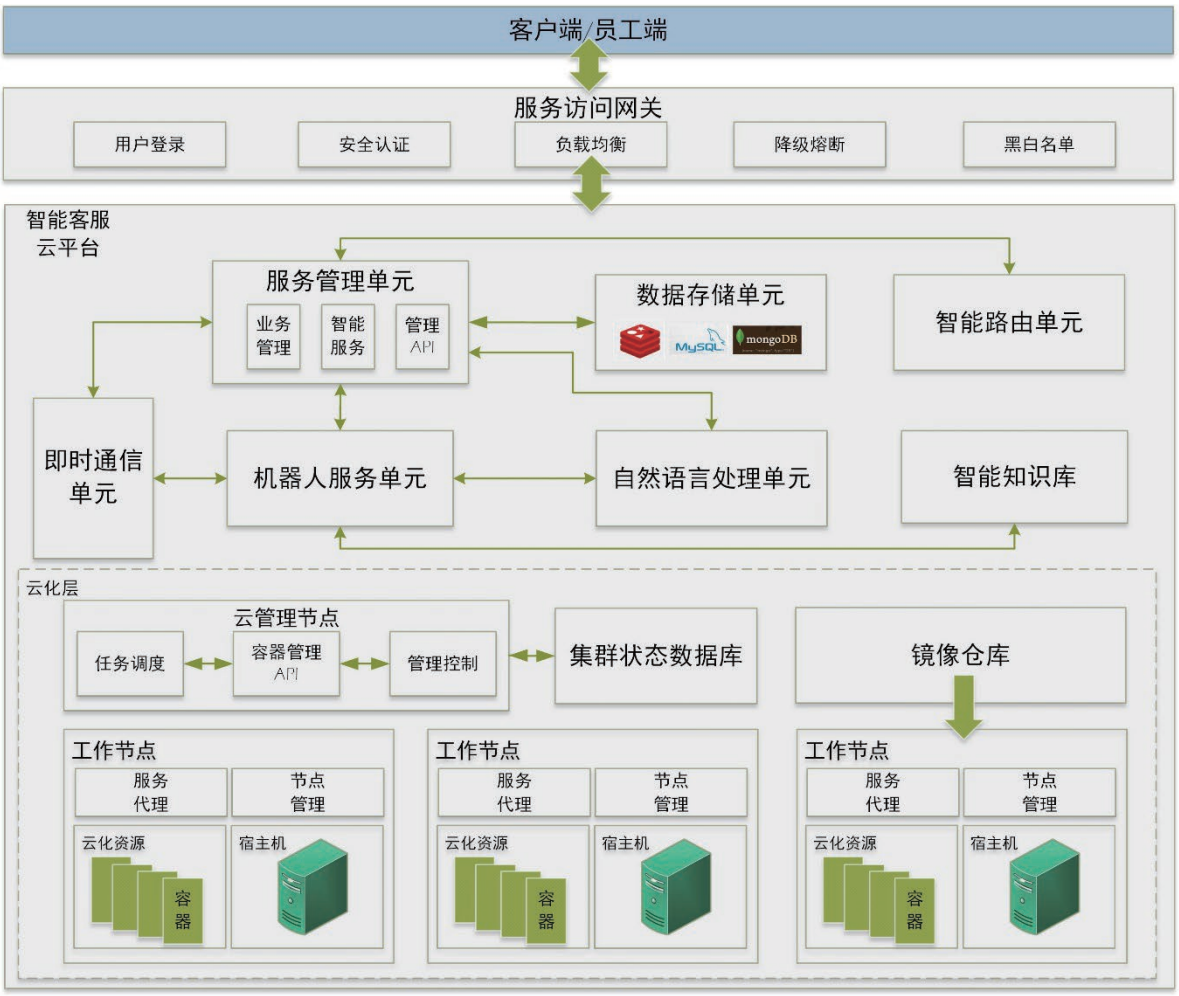


图 2：全渠道智能客服云平台（优问）架构图

提供服务，底层云化层则提供高可用的保障。底层容器实现服务多实例部署，防止系统出现不可用。容器云即采用时下流行的容器技术与容器编排技术实现的云服务平台，主要包括管理节点和工作节点。管理节点主要包括管理控制、任务调度和容器管理 API 等组件。工作节点包括服务代理、节点管理等两个组件。容器管理 API 是各组件通信的核心枢纽，它对外提供各类资源对象的管理接口。管理控制组件是集群的控制中心，负责各节点、实例和资源配额的管理。任务调度组件则负责接收待调度实例，根据一定的规则将其分配至最佳节点运行。工作节点则负责具体的落实，服务代理组件负责将用户对服务的访问请求转发至特定实例，而节点管理组件则通过收集节点与实例的状态，并以心跳的方式定时向云管理节点汇报。同时，节点管理器负责从云管理节点获取指令，并根据指令控制本节点内容器的新建、配置和迁移等工作。集群状态数据库是云平台的基础存储单元，通过该分布式数据库可实现各服务实例的通信，各节点将其运行状态实时同步至数据库中，实现了统一监控。镜像仓库是镜像的存储中心，提供镜像的上传和下载等功能，它保存了智能客服系统所有可运行服务镜像。实际应用中我们采用 Kubernetes 与 Docker 实现云化，保障客服云平台能够稳定运行且易扩展。

通过上述架构设计，智能客服系统采用云原生技术实现云化，实现系统的高可用，在面临后台服务宕机、负载过高时会自动切换或扩缩容。

图 3：全渠道智能客服云平台（优问）功能模块图

由于容器相较于虚拟机能达到更贴近于宿主机的计算能力，因此在计算效率等方面相较于传统虚拟机或物理机部署有明显提升。根据此架构设计，智能客服系统不断进行扩展，以客服促营销、智能化服务为宗旨持续完善系统功能。系统功能模块图如图 3 所示。

智能客服云平台对外提供多种接入方式，并以接口形式提供机器人服务、路由服务等一系列业务层云服务。业务以云服务形式由底层容器承载，可由平台控制服务启动与扩容。智能客服云平台由多平台共同构成，如智能客服平台、智能知识库和插件平台等，各平台通过 API 互相调用完成顶层业务。基础服务模块提供系统的常用功能性服务，包括登录注册、数据管理、统计报表和配置管理等。智能服务对外提供基于智能算法的分析、推荐和文本处理服务。

算法层则通过机器学习、深度学习和自然语言处理等多种算法，提供智能化的基础算法服务，该模块为智能服务提供算法引擎。同时智能客服云平台接入公司的服务中台，获取组织架构、统一认证和消息推送等多种基础服务。底层存储采用高可用数据库，通过 Redis、Mysql 和 MongoDB 等多种数据库及中间件提供缓存与数据持久化存储等功能。

智能客服云平台以云服务为基础，不断探索架构演进，逐步从单体式服务向分布式服务迁移，功能也逐步向智能化发展，设计了一套以智能算法为基础的人机协作服务模式，智能路由、智能质检和知识库等多种智能组件共同协作实现该服务模式。面对快速增长的用户，该架构能够有效保障系统稳定运行，为用户提供高质量的专业金融服务。

四、分析与实践建议

中信建投证券在金融领域智能客服云平台不断深耕，在系统架构和功能模块上不断迭代与探索，取得了一定的成效。目前的架构设计能够有效改善系统的运行效率与稳定性，具体体现在以下几个方面。

1、运行效率提升

采用分布式架构与云原生技术，可保障系统有效利用物理机资源。同时多实例可分担计算压力，从而防止单节点 CPU 或内存资源出现满负荷的问题。系统高负载会严重影响请求的响应时间，因此本架构所带来的运行效率提升可保障用户良好的使用体验。

2、增强系统稳定性与高可用性

客服系统逐步从单体式架构向分布式架构演进，同一服务可通过多实例分担流量压力。服务在注册中心注册后由网关进行服务转发，并且具有降级、熔断和限流等功能保障系统稳定性。高可用的数据库与中间件提供完善的数据备份和可靠机制，系统可用性增强显著。

3、模块耦合性逐步降低

分布式架构使模块间耦合性降低。单体式架构导致系统模块耦合严重，随着系统功能的增加，开发成本越来越高，同时也给故障分析带来困难。以微服务架构进行解耦可使模块间调用关系更清晰，并可进行链路跟踪与分析，显著提高开发效率与系统运行监控能力。

4、智能水平稳步提高

智能客服云平台通过对于员工与客户的画像建模和智能化算法实现精细化与定制化服务，数字化与智能化水平稳步提高。建立机器人与人工协作的服务模式，机器人作为辅助，为员工提供专业金融知识的辅助问答。人工与机器人服务相辅相成，提升人工服务效率的同时也提升了客户的满意度。

客服系统作为券商对客户的核心服务之一，是解决客户问题、提升公司影响力的直接渠道。如何向客户提供高质量的服务是各券商努力探索的方向，中信建投证券在客服系统建设中总结出下述几点建议。

1、可扩展性

客服系统与客户直接对接，且通过数据分析，我们发现咨询量与市场行情紧密相关，因此对于系统可扩展性的要求较高。在咨询量大时，应及时考虑进行扩容。采用的架构应具备状态监控与自动扩容的机制。

2、高可用保障

客服系统的建设应保障高可用，任何服务资源及基础设施要进行灾备建设和故障检测恢复机制。高可用架构可以保障在单点故障时不影响用户使用。数据库等数据存储单元必须采用高可用架构，同时建立完善的数据备份与迁移机制。

3、智能化改造

随着公司的快速发展，客服咨询量逐年增加，人工客服的成本也随着提高。因此如何利用自然语言处理等技术设计智能客服成为一项重点问题。由于智能客服无法满足全部客户需求，因此券商应探索智能机器人与人工的协同工作模式，以最大化客户满意度，同时提高处理效率。

五、总结与展望

为提升客户的满意度，应对快速增长的客户咨询，中信建投证券以智能化为导向，微服务架构为标准，针对智能客服云平台不断探索。本文针对中信建投证券在智能客服云平台的探索与实践进行阐述，对系统架构演变进行详细介绍。此外，本文还介绍了客服系统智能化建设的成果，从系统功能设计角度阐述系统的模块设计与智能化改造。最后，本文分析了架构升级所带来的成效，并介绍了系统建设的实践经验。

未来智能客服云平台会持续探索人机协作的服务模式，探索智能化在券商客服领域的应用。同时持续关注并实践服务网格技术，探索现有架构向服务网格迁移的可行性，将微服务的治理下沉至平台，从而实现跨语言、高效率的开发方式，开发人员将不在关注复杂的服务治理问题。未来智能客服云平台将以云原生和智能化为方向持续改进优化，以为客户提供高质量服务为最终目标不断努力。

金融资讯数据服务平台建设实践

林剑青、王施、刘存光、曹叙风、王伟利、熊友根、王洪涛 / 海通证券股份有限公司软件开发中心

海通证券金融资讯数据服务平台通过构建统一资讯数据模型，对海量外部数据进行处理和融合，并运用人工智能技术挖掘数据价值，探索集团级资讯数据应用场景，实现对业务的全面赋能。本文通过介绍金融资讯数据领域的探索和实践，针对企业数字化转型中遇到的外部数据规范不统一、单一来源依赖性强、“烟囱式”系统建设、数据服务模式单一等问题，分享解决方案和实践经验，助力证券行业金融资讯业务发展。

一、概述

1.1 背景

金融资讯数据在证券行业有着广泛的应用，充分发掘资讯数据价值，提供差异化服务是业内探索的一个重要研究课题。金融资讯数据使用的过程当中也面临着诸多挑战。从内部应用系统整合的角度看，公司内部与金融数据相关的各种应用系统基本是“相互孤立、独自运行”的，各个业务部门在需要金融类信息数据的时候，往往从采购和系统建设管理上以部门为单位独自进行，既造成了公司内部资源的浪费，同时容易出现重复采购的金融类数据源或者重复建设的系统。另一方面，由于采用的金融数据源五花八门，一旦在数据源上发生了任何的变动，上层的应用往往都需要作相应的调整，一个微小的变化都可能会带来较大的影响。除此之外，“烟囱式”的应用系统给系统维护人员带来很大挑战，不利于技术

栈的统一，数据服务能力也得不到沉淀。

1.2 建设价值

海通证券金融资讯数据服务平台（简称“资讯中心”）作为一个基础资源输入平台，通过对各种投资类资讯、产品资讯、服务资讯的整合，实现资讯数据的统一管理。同时，通过对数据的自动采集、抽取、校验，将第三方数据按照统一的数据模型和规范转换为有用的、可靠的信息。

公司统一规范的金融资讯数据服务，消除了数据孤岛，实现了集中化的数据管理。在数据资产化的基础之上，借鉴专业的投资研究、运营管理、风险管理、舆情风控理论，对数据进行深入挖掘，建立公司特有的资讯数据分析模型，提供风险监控、机会发现、投资决策等多元化的服务支持。

二、金融资讯数据模型

2.1 业务模型

数据业务模型的设计过程对各相关核心业务元素的完整性和相关性进行细致的分析。示例业务模型（图 1）涉及到的数据类别包括：

◆中国资本市场主要金融品种的交易数据、财务数据及各类公开披露的信息。包括：上海证券交易所和深圳证券交易所全部上市公司的基本资料、发行资料、交易数据、分红数据、股本结构、财务数据、公司公告及其它重要信息。

◆公募基金、券商集合理财及信托等产品的发行上市资料、净值、投资组合、收益和分红数据、定期报告、财务数据等。

◆国债、企债、金融债、可转债、央行票据等债券的基本资料、计息和兑付数据、交易数据

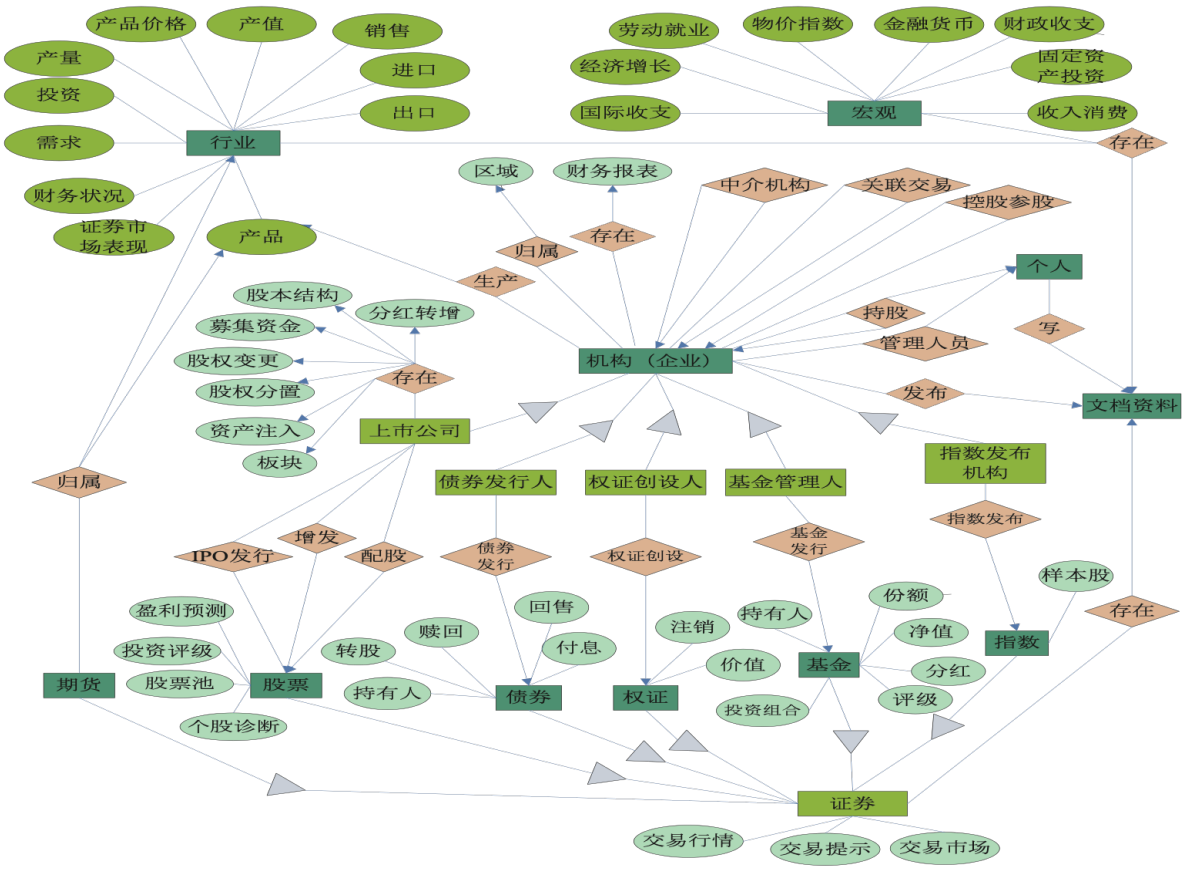


图 1：金融资讯数据模型示例

等，以及各类收益率曲线等衍生数据，并提供支撑债券定价计算的数据结构。

◆中国证券市场指数（沪深交易所指数、银行间债券市场指数、MSCI 中国指数、新华富时指数、中信指数、申银万国指数等）和海外市场指数的基本资料和交易数据。

◆国内各期货交易所期货合约资料及交易数据，可支持套利计算、程序化交易的数据结构。

◆自有数据，包括研究所报告、理财产品等。

2.2 编码规则

通过设计统一的编码规则，如公司编码、证券编码、行业编码、板块编码（图 2），将各金融品种有效的串联起来，再设计通用的全局编码与外围业务系进行对接。数据根据自定义的内码进行关联，并采用业务主键做唯一索引，保证了数据库结构的高度规范性，同时也实现了跟上游数据去耦合，降低上游数据结构变化带来的风险。

三、金融资讯数据服务平台架构

整个金融资讯数据服务平台的实现紧密围绕着对于各种数据的采集、转换、清洗、分层存储以及管理、发布、数据接口和数据应用等功能而展开。从数据的角度出发，我们将金融资讯数据服务平台架构从逻辑上分为分成 4 个层次：数据源层、数据采集和处理层、数据存储层、数据服务层（图 3）。具体如下：

◆数据源：各类内部以及外部数据源，包括结构化数据、非结构化数据、半结构化数据。

◆数据采集和处理：基于数据清洗转换体系提供完备的源数据跟踪管理、数据处理调度服务，支持根据实际数据要求，灵活配置处理任务；并提供多种数据校验模型，灵活配置校验规则以及任务，为数据源提供质量保障。

◆数据存储：数据存储层存放经过采集、转换、清洗和整理后产生的各类数据，除了出于系统的效率和应用的支持等目的而产生的少量冗余外，中心数据库中的数据是原始的、精炼的，也不会产生各类二次加工数据。在数据存储层，从技术上主要考虑数据针对业务或数据应用的存储模型设计，以及针对数据应用效率的数据分层设计。

◆数据服务层：是各类与数据相关的服务端系统的汇总，主要目的是为各类前端的数据应用访问资讯中心提供一些基础性服务，可以包括数据服务 API、可视化组件、数据库表及文件服务等。同时，资讯加工处理中运用到的 NLP 能力也可以进一步通用化，如标签处理、情感分析、语义识别等，为不同业务场景提供相应的技术支持。

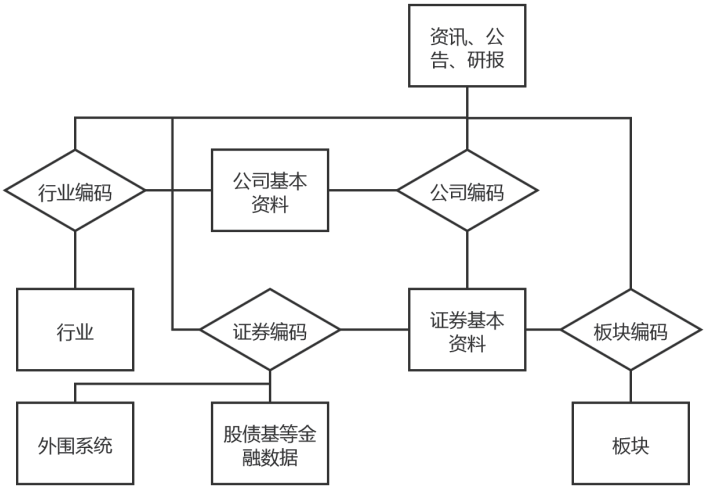


图 2：编码信息示例

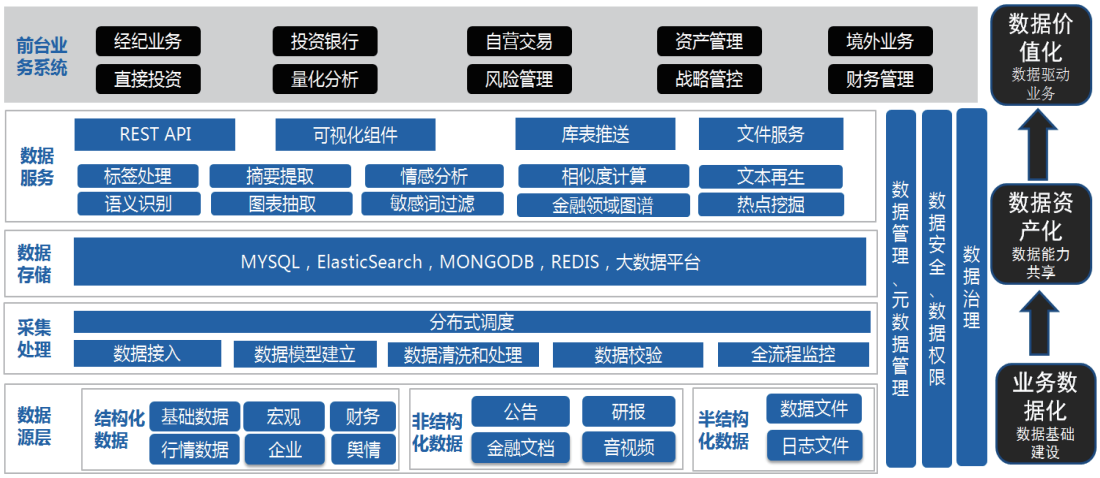


图 3：海通证券金融资讯数据服务平台总体架构

3.1 数据源层

目前金融资讯数据服务平台的数据涵盖股票、公司、债券、基金、货币、指数、理财、期货现货、期权、资讯等多个类别（图 4），并且不断引入特色资讯数据，进一步提升数据质量和数据全面性。

3.2 数据采集处理

3.2.1 数据清洗转换

数据清洗转换系统功能模块图如下（图 5）

元数据库是指定义数据清洗转换平台的基础数据，主要功能包括数据源配置、数据字典和数据血缘管理。

调度中心是管控平台的核心组件，包括配置管理、调度引擎、执行器管理、任务管理等。配置平台支持在线配置调度任务入参，即时生效。

图 4：海通证券金融资讯服务平台数据分类

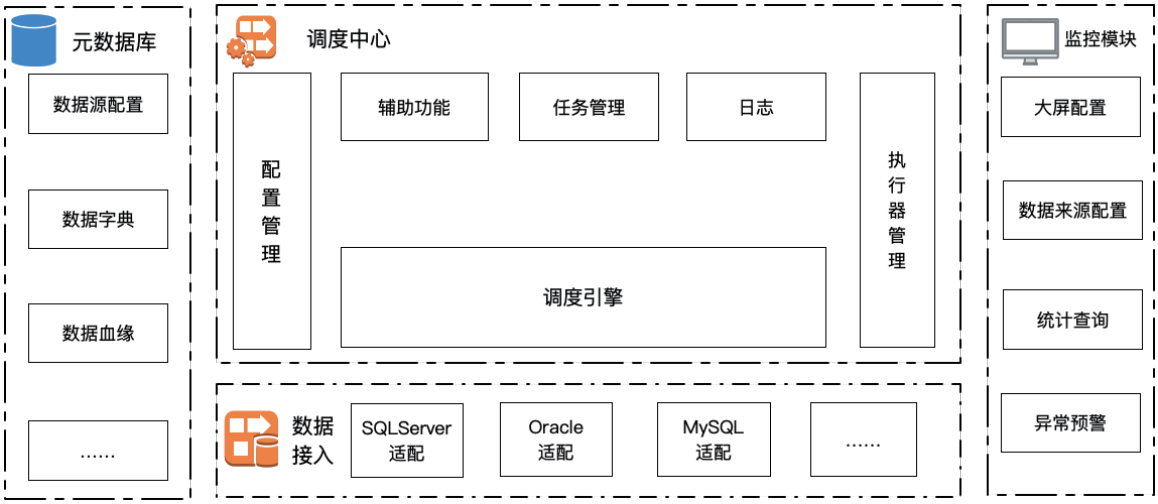


图 5：数据清洗转换系统主要功能模块

执行器支持任务节点弹性扩容缩容：一旦有新执行器机器上线或者下线，下次调度时将会重新分配任务。

任务管理支持动态修改任务状态，暂停 / 恢复任务，定时任务触发，配置子任务依赖等。

3.2.2 数据抽取和转换

数据清洗转换模块是基于 Kettle（开源工具，图 6）扩展的，在管理界面的模型配置菜单中可支持新建数据清洗转换模型、上传 / 下载 / 发布脚本、调试运行、查看操作日志和报错日志等。

3.2.3 调度框架和策略

核心调度功能模块（图 7）是基于 Quartz 实现的集群调度中心，该架构支持调度服务的水平扩展，实现调度服务的高可用。调度中心通过读取 ETL（数据抽取转换加载）模型配置的定时任务信息，定时启动任务。一次任务调度包含多个

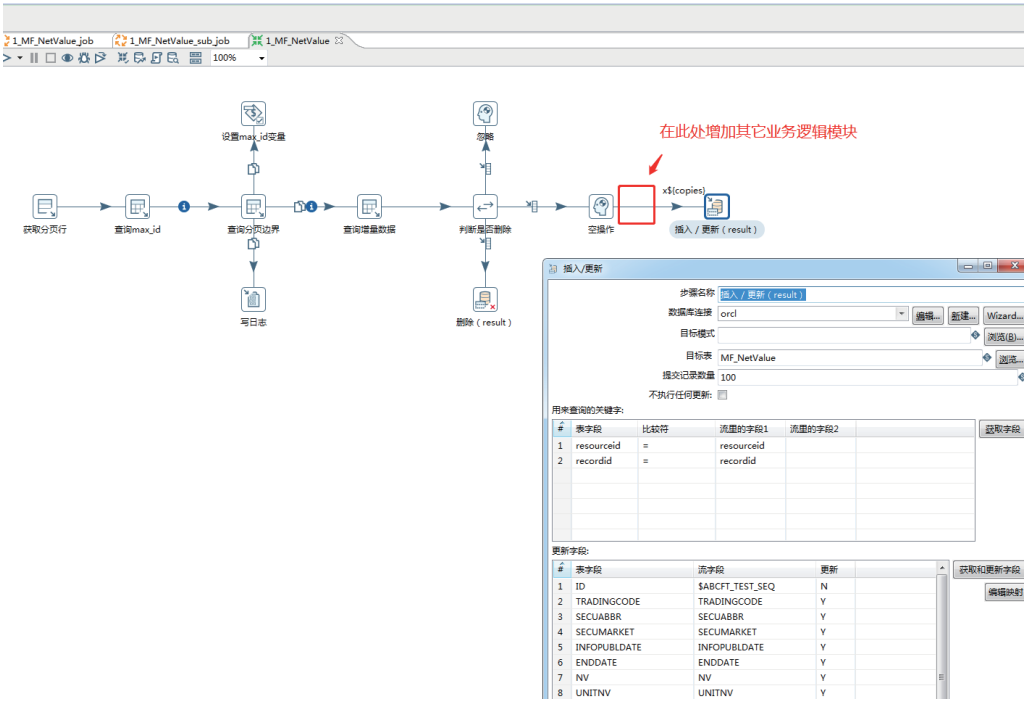


图 6：模型开发界面

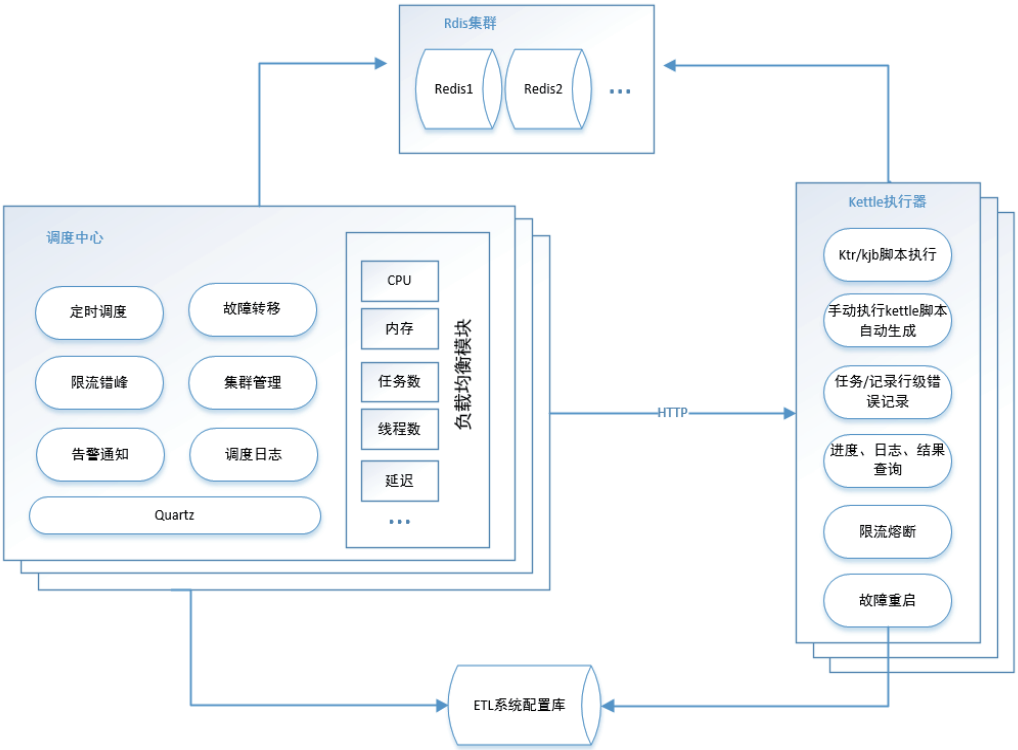


图 7：ETL 任务调度架构

模型，模型是整个调度中的最小单位。每个模型的调度执行均由负载均衡模块计算后选择负载最优的执行器执行。

由于 ETL 任务的调度执行过程具有内存及 CPU 消耗密集型的特点，传统负载均衡策略（分发、随机、HASH 等）在此场景效果较差，往往容易造成单台执行器负载过高而宕机。通过完善负载均衡机制，结合服务器实时的 CPU、内存、

图 8：海通证券资讯中心监控大屏

并发任务数、线程数、服务延迟等指标对各执行器进行打分，调度时选择得分高的执行器执行，实现准实时特性的负载均衡。配合完善的执行器自检、熔断、重启策略，达到调度执行器集群高可靠性与可用性的目的。

采用分组调度策略，即一个任务配置一种定时策略，但该任务下涵盖多个 ETL 模型，当某个调度任务发起时，该任务组下的所有模型均会被调度执行。该调度方式的优势在于降低了系统定时器的数量，避免操作人员频繁配置重复定时任务，节省操作成本，且可降低调度服务定时线程开销。

为了避免任务调度过于集中造成执行器集群压力过大问题，在进行定时任务配置时对定时策略进行末位随机算法，使任务执行尽量离散，避免在同一时刻有大量定时任务触发。在任务组内也采用了随机延迟调度的策略，极大的保证了调度集群的稳定性，提高了系统并发调度负载上限，使服务器实现最大化利用。

3.2.4 系统监控

为了保证对异常情况进行及时、准确地预警，通知相关人员第一时间处理，设计了海通资讯中心监控大屏（图 8），对数据清洗转换、数据同步及校验系统进行实时监控。监控维度主要有数据清洗转换同步总量及报警提示、校验系统校验数据总量及异常数量、数据同步实时增量等。

四、数据应用

4.1 舆情风控预警

多源资讯数据采集到集团内部之后，通过智能标签平台对资讯打上相应的标签，包括股票、债券、基金、行业、概念、板块、风险事件等。目前风险因子超过 200 个，用户可以根据主体、风险事件和时间等维度快速进行舆情检索，并根据个人需要进行预警任务的配置，当触发告警条件时，会第一时间及时进行告警提示。

4.2 智能资讯运营

标签化之后的数据通过数字化运营平台进行管理，运营人员可以在线进行资讯内容、标签内容、情感正负面等要素编辑操作，审核通过的资讯可以直接对外发布。同时，提供了资讯来源管理、资讯栏目管理等功能，实现对资讯的精细化运营。

4.3 智能语义分析服务

资讯数据结合自然语言处理方面的能力，实现在实体识别、语义理解、情感分析和智能文档解析等领域的应用。

实体识别可以文档中出现的各类实体，包括公司、人名、行业、板块等；

语义理解：长城汽车今年 9 月电动车产量，可以识别出主体是长城汽车，“今年 9 月”对应的时间区间是 9 月 1 日到 9 月 30 日，“电动车”对应新能源汽车行业，“产量”对应某一项指标。

情感分析服务可以通过算法判断主体和新闻本身的正负面；

文档解析服务通过运用非结构化解析相关的能力，实现对各类文档中的文字、表格和图片的抽取，用户可以根据自己的需要灵活地进行各类信息的提取和使用。

五、数据治理

5.1 库表命名规范

不同类别的数据建立统一的命名规范（表 1），从而起到“见名知义”的效果。下游使用数据的时候，根据数据表的名称就能大致定位表的用途，减少数据查找和数据定位耗费的人力成本。

5.2 数据校验

数据校验在整个资讯数据服务平台建设过程中是尤为重要的一个环节，主要功能包括有校验

表 1：库表命名规范

表头	模块	说明
blt_	公告	公告资讯相关表
bnd_	债券	债券维度相关表
com_	公司	公司维度相关表
fnd_	基金	基金维度相关表
fut_	期货	期货维度相关表
inx_	指数	指数维度相关表
news_	新闻	新闻资讯相关表
opt_	期权	期权维度相关表
rpt_	研究报告	研究报告资讯相关表
stk_	股票	股票维度相关表
tru_	信托	信托维度相关表
……	……	……

规则管理、任务管理、通知中心等。包括三类核心规则的配置，字段校验，记录行校验与三方校验。

所有校验规则均可灵活配置定时执行策略，校验执行结果提供校验不通过数据量、通过率、异常数据明细等信息。提供校验不通过消息推送，便于业务人员及时发现并处理异常数据。

业务规则的调度周期同模型同步的调度周期，依据业务需求，如行情类数据，一般使用分段变时配置收盘后每隔半小时调度；证券主表，机构主表等重要的基础数据，源库不定时推送数据的表，一般使用相等间隔调度，配置每 5 ~ 10 分钟调度一次；某些不常更新的表，如常量、行业分类等，一般使用固定时间调度，配置一天调度一次或更久。

图 9：海通证券资讯中心元数据管理平台

5.3 元数据管理

元数据系统主要实现数据的可视化呈现，并记录表与表之间的逻辑关系，方便数据的追溯（图9）。包括表信息展示、表结构数据展示、样例数据浏览和导出等。

六、总结与展望

海通证券金融资讯数据服务平台通过将多源异构的数据按照统一数据模型和规范转入集团内部，解决了金融资讯数据使用中存在的规范不统一、单一来源依赖性强等问题，并运用自然语言处理方面的技术，进一步挖掘数据价值，丰富了资讯数据应用场景。

未来金融资讯数据服务领域还有广阔的探索空间，一方面随着人工智能技术在证券行业的应用逐步深化，资讯价值的纵向挖掘充满了更多可能性，企业图谱、产业图谱、供应链、客户画像等数据可以相互关联和穿透，实现数据的采集和展示到知识的积累和价值挖掘的飞跃。另一方面，随着券商的开放程度越来越高，跨界进行信息、知识和能力共享逐渐成为可能，金融资讯数据作为很好的切入点，在提升机构客户服务体验以及构建开放生态上，将发挥出更大的价值。

基于硬件数据库的风控系统

卢文岩、张宇 / 中科驭数（北京）科技有限公司
何波、汪伟、周忠辉 / 中泰证券股份有限公司
钟浪辉 / 上交所技术有限责任公司

伴随证券交易系统的技术变革与快速迭代，证券交易系统的风控功能变得举足轻重，性能瓶颈集中在风控功能。目前业界普遍采用“软件数据库”的风控方案，面临风控规则越来越多和执行效率越来越低的挑战。

2020 年，上交所技术公司、中泰证券、中科驭数开展了“基于 KPU 架构的合规监管系统设计”的联合课题研究，提出了基于 KPU（Kernel Process Unit）架构硬件数据库的风控方案，采用现场可编程门阵列（FPGA）实现数据库卸载引擎（DOE，Database Offload Engine），既能满足灵活的风控规则配置部署需求，同时又能满足交易的微秒级性能要求。

一、证券公司风控系统特点

证券公司合规风控系统主要采用纯软件的方式，性能瓶颈在于大数据量下的交易信息数据库查询操作，即使采用全内存优化方式，风控规则执行仍在毫秒级。特别是开盘时单位时间交易量压力会数倍于其他时段，风控性能降低尤为明显。因此，亟需一种高吞吐、低延时并且性能抖动小的风控方案。

1.1 风控系统业务需求

风控系统检查用户报单的合规性，避免金融交易中可能发生的各种风险，减少风险事件发生时所造成的损失，其性能直接影响整个交易系统性能。风控系统需要接收并解析交易数据包、按照风控规则进行合规检查，风控过程需要对证券标的信息、账户信息以及交易日志信息进行存储、检索、更新。

风控系统要尽量降低处理延时，缩短订单执行时间；要适应大带宽网络下的订单，无论网络空闲还是有突发的报文风暴，系统应正常应对，性能平稳。

1.2 硬件风控系统的难点

由于期货交易风控规则相对简单，基于 FPGA 的风控方案在期货类交易中率先得到了突破。但

是 FPGA 风控加速方案未在证券领域得到突破，主要有三大原因：1）证券交易规则复杂，给 FPGA 开发带来了巨大的挑战；2）规则多变，针对不同客户不同场景风控规则差别较大，现有 FPGA 方案虽然具有一定的可配置性但编程性较差，很难满足规则多变定制化的需求；3）数据量大，证券市场标的数量远远大于期货市场，同时风控规则涉及的统计量也异常繁多，在 FPGA 上灵活地维护如此大的数据量异常困难。

二、RISKCOP™ 风控方案

中科驭数从底层计算架构入手探索新的计算方式、编程方式、数据存储方式，应对证券交易业务面临的风控数据量庞大，以及规则复杂多变的挑战。

本课题采用了 RISKCOP™ 风控解决方案，它的核心是硬件数据库引擎 DOE，DOE 基于中科驭数自主知识产权的 KPU 架构设计实现，可以在微秒内完成百万条数据量下多条件查询操作，从而可在 2 微秒内完成风控所需标的、账户、以及报单信息的提取，在保证大吞吐量的同时，又能显著降低系统整体延时。

2.1 领域专用计算架构 KPU 架构介绍

领域专用计算架构（专用处理器）因算力强大被视为维持“后摩尔”时代性能增长效率的革新力量。中科驭数以“软件定义加速器（Software-Defined Accelerator, SDA）”为技术路线，实现设计阶段和运行时资源管理的协同。将硬件结构重配置适应不同的应用领域，直接面向特定计算问题组合 IP（知识产权）核。KPU 芯片架构以 SDA 为理论，突破专用算法和微结构功能核之间的语义障碍，对加速器资源的管理更加高效，所有的计算核和存储都是软件管理的。

2.2 硬件数据库引擎 DOE 介绍

DOE 是超高性能数据库运算加速引擎，它基于 KPU 架构实现和集成了数据库操作相关的运算核（Kernel），以指令形式对外暴露配置和调用接口。DOE 运算 Kernel 集包含数据库基本操作 Kernel 和规则函数运算 Kernel。其中数据库基本操作 Kernel 包含：表配置指令集、Update 指令集、条件查询指令集、条件综合指令集、Insert 指令集和结果获取指令集。而规则函数运算 Kernel 包含：算数运算指令集（+-*/），逻辑运算指令集（and, or, !, <, >, ==, !=），基础聚合函数（SUM，AVG，

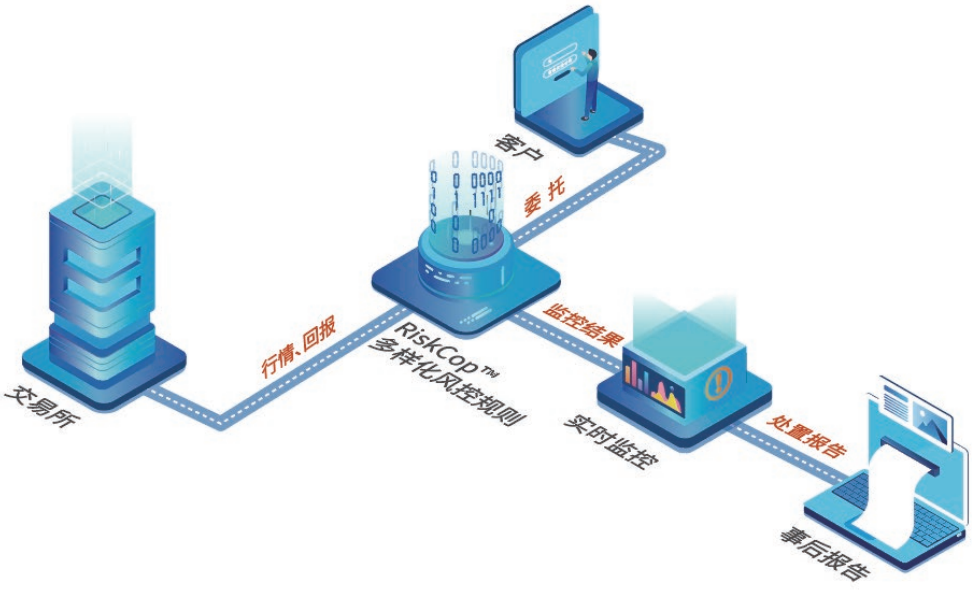


图 1：RISKCOP™ 风控场景应用

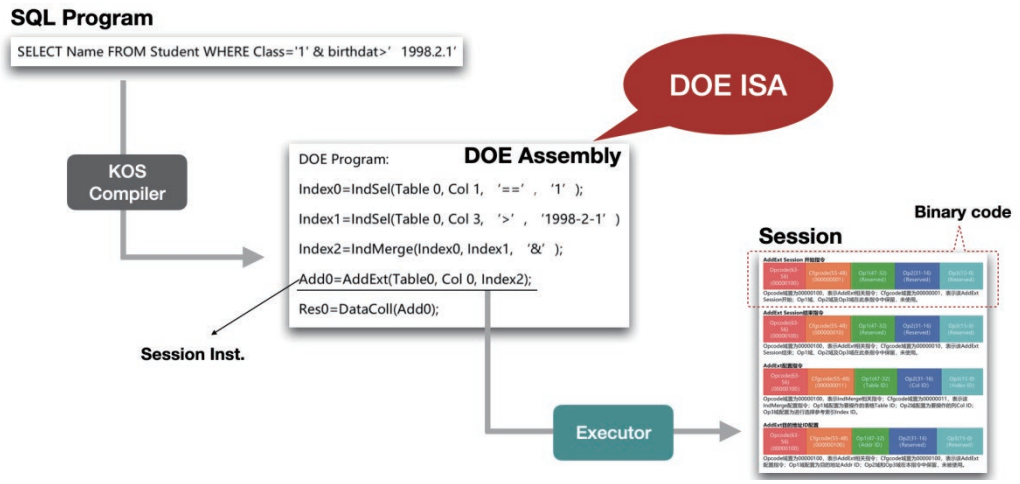


图 2：硬件数据库（DOE）指令编译执行示意图

COUNT，OrderBy）。用户可以基于该类原子操作自由搭建更高层次的风控表达。

DOE 采用面向列式的存储结构，指令集完全面向数据查询定制，用户可以利用指令来编译任意复杂的 SQL 查询。

在实际应用中一个标准的 SQL 插入、更新、查询或其他操作会被 KOS(Kernel Operation System) 分解成 DOE 的多条指令，下发到 FPGA 硬件上进行内部指令级优化和执行。本课题利用 DOE 实现了证券业务的多种风控功能：验资、验券、持仓验证功能、流量控制功能、异常指令检测功能、日志保存功能和软硬互备多板卡解决功能。

三、硬件风控业务的实施

3.1 风控业务的数据定义

本课题使用标准数据库中数据表 Table 定义语法来灵活定义合规风控引擎的输入输出数据格式，以及中间数据处理和存储格式，支持多种基本数据类型，并且支持数据定义的实时修改和重新配置，可以完备地表达合规风控业务场景中的

```
/*机构账户数据*/
create table account_info
(
    account_id              INT          not null , /*账户ID*/
    available_balance       INT8         null     , /*账户剩余可用资金*/
    total_balance           INT8         null       /*账户总资金*/
)
/*账户持仓数据*/
create table account_ticket_info
(
    account_id              INT          not null , /*报单ID*/
    ticket_id               char(16)     null     , /*证券标的代码*/
    avail_quantity          INT          null     , /*可用持仓数量*/
    total_quantity          INT          null     , /*总持仓数量*/
    avg_price               INT8         null     , /*平均持仓价格*/
    buy_top_price           INT8         null     , /*最大买单价格*/
    sell_low_price          INT8         null       /*最小卖单价格*/
)
```

图 3：数据定义

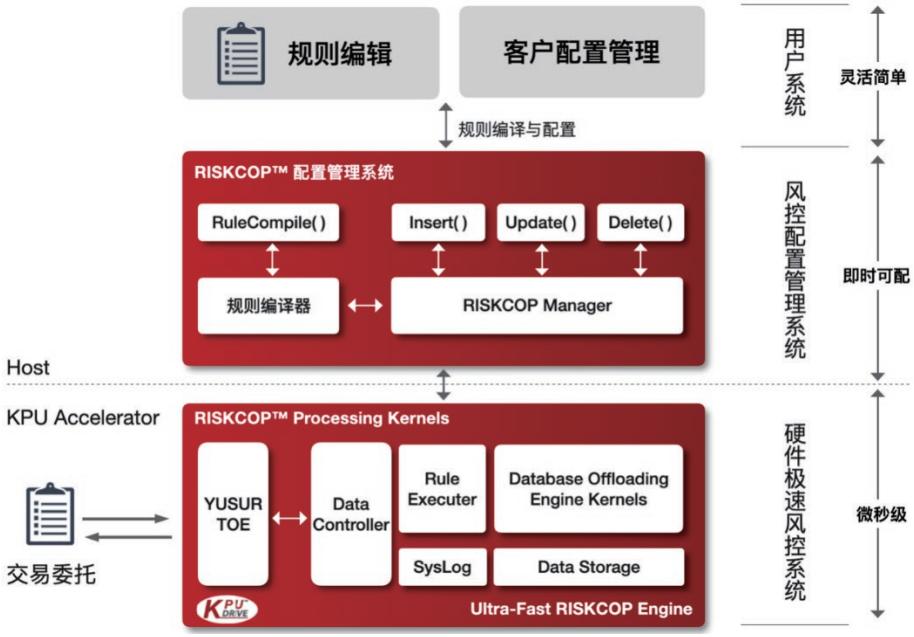


图 4：规则编辑及风控实施的整体架构图

数据定义。图 3 展示了典型数据表 accout_info 和 account_ticket_info 的结构和定义。

3.2 风控业务的规则配置

用户可以根据风控业务规则进行规则编辑和客户配置管理。

用户使用类 SQL 语法可以定义需要处理的数据格式，也可以实时定义、添加和修改合规风控规则，图 7 展示了防对敲规则定义的例子。系统将用户配置好的数据定义与规则定义交由风控系统提供的软件 SDK 进行编译，编译并测试成功后，就可以配置到 KPU 硬件的规则引擎启动风控业务规则执行。用户可以通过数据初始化模块接口对整个风控系统中所需的标的信息和账户信息进行初始化，也可以实时对各账户交易日志进行查询。Rule 规则模块为用户提供了灵活高效的风控规则配置接口，各证券 / 期货公司根据其内部风控规则需求自主配置风控规则，保证各证券 / 期货公司风控规则的私密性。最后 Executor 执行模块会将用户提供的风控规则自动编译成 KPU 所需的指令代码。

规则采用以 SQL 为原型的 KPU kernel 函数嵌入式语言，可以灵活运用 KPU 架构的核函数进行业务规则组合，多风控规则可以映射到多个 KPU Kernel 中并行执行（如图 6 所示），充分释放 KPU 性能。

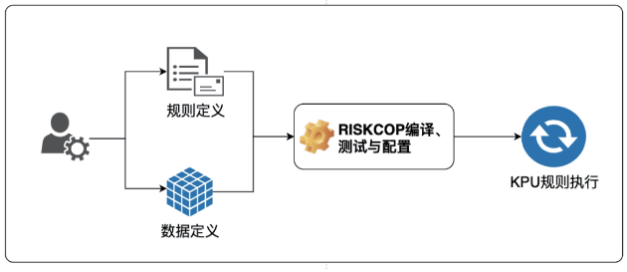


图 5：风控规则配置管理流程示意图

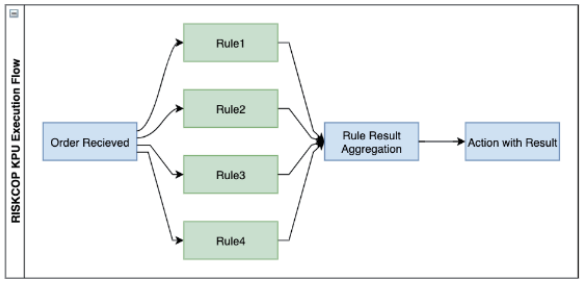


图 6：风控规则并行执行示意图

```
/*卖出防对敲验资规则配置*/
  Rule T1 (
    (
      trade_order.price >
          (select buy_top_price from account_ticket_info
            where (ticket_id = trade_order.ticket_id and account_id =
trade_order.account_id)
          )
      ) where trade_order.direction = 1
)
```

图 7 卖出防对敲验资规则配置

四、硬件风控系统的测试评估

4.1 风控业务的测试用例要求

本课题根据风控业务提炼了基本的业务场景与测试用例，支持期权做市商使用 20 个账户，对 2048 个标的进行买卖，单账户交易量 1,000~1,000,000 笔，具体配置参见表：

参数	配置
风控规则	验资、验券、防对敲
标的数量	2048
账户数量	20
单账户持股数量	1024
单账户交易量	1,000~1,000,000（笔）

4.2 测试环境

宿主机服务器使用 Intel® Xeon® Gold 5218R CPU @ 2.10GHz，64GB 内存，CentOS Linux release 7.8.2003，Xilinx U200 FPGA 加速板卡。测试环境细节如图 8 所示。

宿主 CPU 通过 PCIe 接口与模拟 TOE 模块进行数据交互。需要说明的是由于本次评估的重点在于 RISKCOP™ 系统性能，而非 TOE 系统，且 TOE 是非常成熟的 IP 解决方案，因此本次评估未将 TOE 包含在内。我们在 FPGA 板卡端模拟了 TOE 模块，其行为和接口与标准的 TOE IP 完全一致，宿主机 X86 CPU 通过 PCIe 与模拟 TOE 模块进行数据交互，来模拟 TOE 数据收发。CPU 侧基于 C++ 语言和 RISKCOP™ SDK 搭建了 TestBed 测试工具。TestBed 实现风控业务的初始化和规则配置，同时模拟“交易者”和“交易所”通过 PCIe 向 FPGA 板卡发送测试用例，并统计各测试用例执行情况。

TestBed 基本测试过程：

1. 使用系统配置功能，对测试用例涉及的各数据表进行创建和初始化，然后配置所要进行检查的风控规则和数据更新规则。

2. 交易者报单，CPU 端 TestBed 模拟交易者向 FPGA 板卡报单（买 / 卖）。FPGA 收到报单信息后，进行风控检查，并报单给交易所或拒单反馈给交易者。

3. 交易所反馈，CPU 端 TestBed 模拟交易所向 FPGA 板卡发送反馈信息（买 / 卖成交 / 拒绝），FPGA 根据反馈信息更新数据并反馈给交易者。

4.3 性能测试结果分析

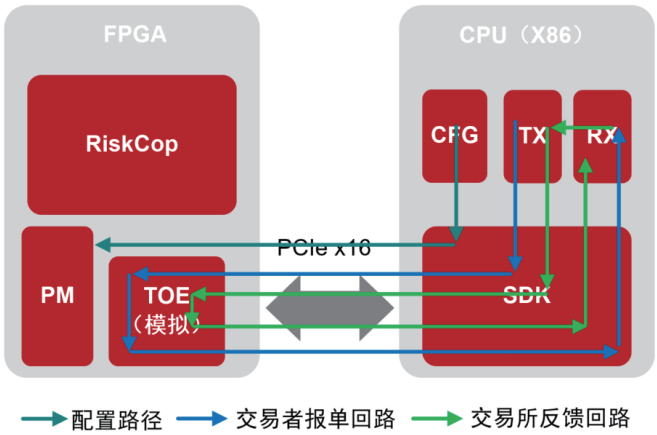


图 8：测试环境细节

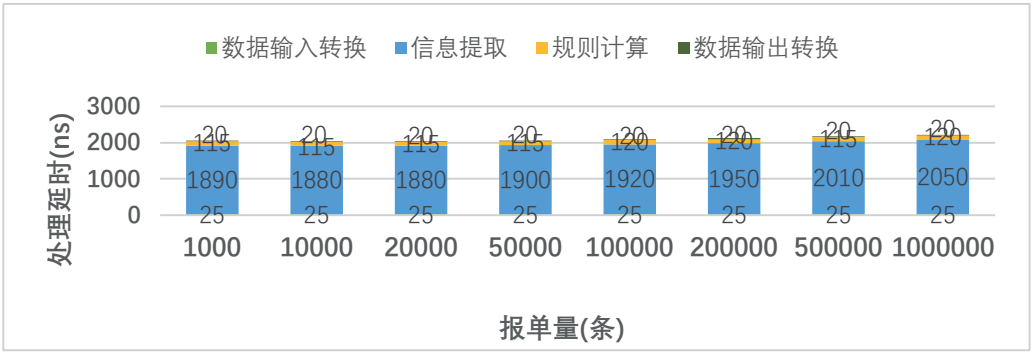


图 9：买报单检查流程性能

4.3.1 基本性能测试

本课题针对 FPGA 风控系统中八类主要操作（买报单、买更新、卖报单、卖更新、买成交更新、买拒绝更新、卖成交更新、卖拒绝更新）分别进行了测试，延时统计是在各报单量（1,000~1,000,000 条）下进行了 2,000 次测试取平均值作为最终结果。其中，买报单更新流程，卖报单检查流程，卖报单更新流程的处理延时与买报单检查流程的处理延时类似，都在 2 到 2.7μs 之间；而买拒绝更新流程、卖成交更新流程、卖拒绝更新流程性能与买成交更新流程一致。

如图 9 所示，以买报单流程为例，整个流程包含四个关键操作（1）数据输入转换，将 TOE 接收到的报单数据转换为风控系统内部数据格式；（2）信息提取，从各信息表中提取风控相关信息，DOE 主要操作；（3）规则计算，对事先定义好的规则进行计算；（4）数据输出转换，将报单信息转换为 TOE 网络数据格式发送到交易所。

买报单风控规则检查流程的处理时延描述了：①系统从接到报单数据到规则检查完，再到数据报单给交易所整个流程延时在 2μs 左右；②随着报单条数增加，从 1000 条增长到 1,000,000 条处理延时增加非常缓慢，控制在 2.5μs 以下；③整个处理流程信息提取是最耗时的阶段，得益于基于 KPU 架构的 DOE，保证了极低的信息提

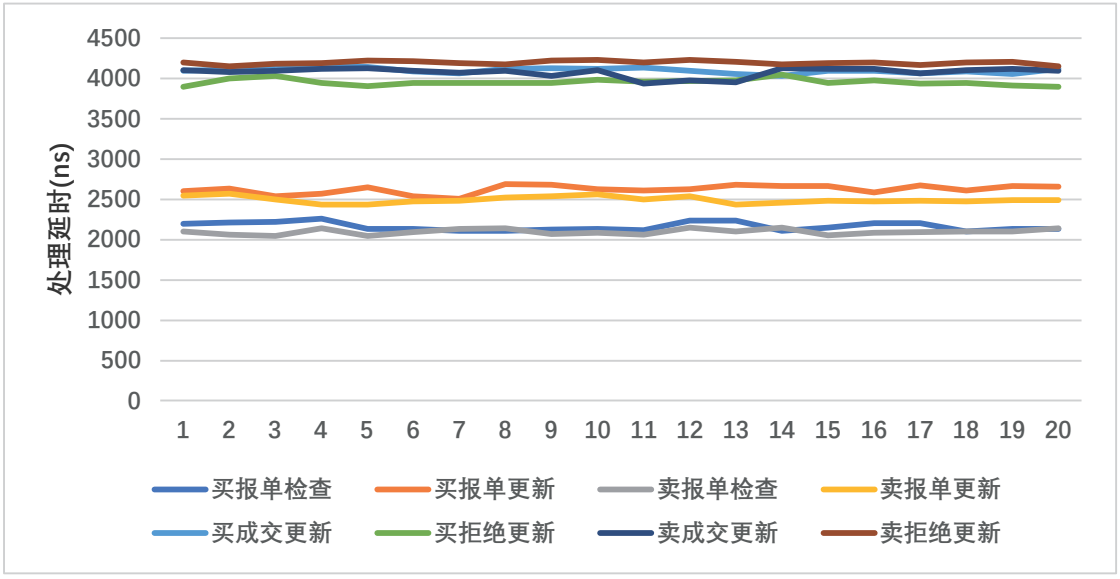


图 10：八类操作流程处理延时抖动

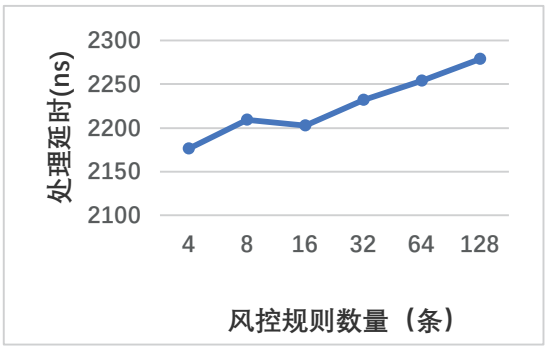


图 11：风控规则数量 vs 处理延时

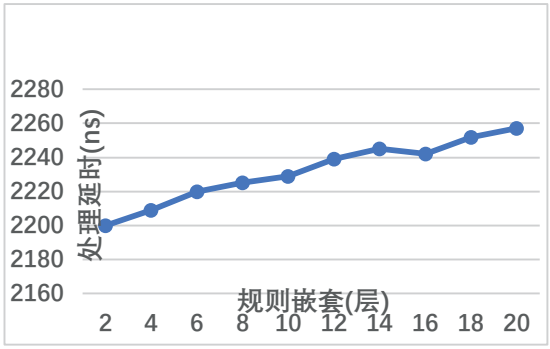


图 12：风控复杂度 vs 处理延时

取延时，尤其在 1,000,000 超大数据量情形下。

在交易系统中，除了单次操作低延时之外，系统延时的稳定性也是非常重要的一个性能指标（即抖动性），稳定的延时有利于交易策略的高效执行。图 10 展示了八种操作类型在 1,000,000 条报单条数下多次测量的情况，最大延时和最小延时的差值可以控制在 100ns 以内。

4.3.2 可扩展性测试

从两个方面来评估 FPGA 风控系统可扩展性：1）风控规则条数，2）风控规则表达能力。

随着风控规则数量的增加，从 4 条规则到 128 条风控规则时，延时只增加了百纳秒左右。

随着风控规则复杂度（通过处理嵌套层数来刻画）增加，风控规则表达式 20 层嵌套相比两层嵌套，延时增加了大概 50ns，延时增加非常低。

五、总结展望

本文借助领域专用计算架构 KPU 架构实现了硬件数据库引擎 DOE 与 RISKCOP™ 风控系统，探索了异构证券风控方案，展现了硬件风控处理时延、吞吐和抖动的优越性。用户使用类 SQL 的数据配置和规则编辑工具，大大降低了系统学习维护成本，提高了系统灵活性。

下一步工作是进行实际生产系统的部署和测试论证，拓展风控规则表达范围，完善系统监控功能以及两级容灾方案（第一级硬件多 KPU 板卡热备系统，第二级软硬异构互备系统）。

证券行业互联网系统自动化安全运营实践

吴佳伟、王玥、李鹏、胡晓明、龚威、倪文亮 / 兴业证券股份有限公司，上海 200135
E-mail：wujiawei@xyzq.com.cn

当前网络安全形势日趋严峻，攻击手段呈多样化，行业各机构时刻遭受着大量的外部攻击；同时国家、行业相关网络安全管理要求越来越完善和精细化，给行业机构日常安全运营工作带来巨大挑战和压力。作为为公众提供服务的互联网系统，其直接面对大量外部威胁，一旦失陷将成为入侵内部网络环境的跳板，可能给企业带来巨大的安全风险和超乎想象的经济、名誉损失，因此互联网系统的安全性应得到优先保障。本文定义自动化安全运营，解析自动化安全运营优势，结合兴业证券互联网系统自动化安全运营实践经验，介绍了适用于证券行业的基于自动化技术互联网系统常态化安全运营具体实践，将被动救火方式的应急响应转变为主动提前预防的风险管理，有效提升了互联网系统安全运营水平，有力保障了互联网系统安全。

一、自动化安全运营

（一）自动化安全运营定义

自动化安全运营可分两部分来定义：

安全运营：用一句话概括是有计划、有组织、成体系化地实施网络安全工作。制定适合公司特性的安全工作规划、组织匹配安全规划的有力资源并形成体系化的安全方案。一方面持续加强安全建设，修补安全短板，使应有的安全能力不缺失；另一方面不断优化已有的安全设施，包括安全技术设施、安全控制策略和安全运营平台等，使安全设施越来越高效、合理及符合预期目标进行运转。

自动化：采用安全自动化技术，将重复的、可标准化的信息安全工作自动运转起来，让人从重复繁杂的工作中释放出来，从事更有价值的工

作，比如去设计更实用更合理的安全架构、安全体系等。

因此，基于自动化技术的安全运营使安全运营更高效地开展，使人、技术和流程更好地关联起来，人利用技术去监测信息资产以及信息资产中存在的安全风险，再利用标准化流程去分析和响应发现的安全风险，并持续优化整个过程，达到人、技术、流程一体化。

（二）自动化安全运营意义

之所以要实施自动化安全运营，能够有效缓解证券行业安全专业人员稀缺、安全工作零散、效率低、规模化管理难、落地成效差等痛点问题，使日常安全运营工作更具有成效：

1、有章法

自动化安全运营使日常安全工作成体系化，更有条理性。首先让安全人员能分清轻重缓急，对安全风险设置不同优先级，使优先级高的安全风险得到及时处置；其次形成闭环管理，制定闭环的标准流程，对安全风险从检测、预警、分析到响应形成有效的闭环管理机制，持续优化提升安全防御能力。

2、提效率

安全运营过程采用自动化手段，相比人工方式大幅提升了安全风险检测和响应效率，例如，通过自动调度引擎定时调用安全检测引擎实施安全检测，发现安全漏洞后再通过自动调度引擎对接 CMDB 和工单系统进行后续响应，从风险的检测、定位责任人、到告知责任人整个过程时间大幅缩短。

3、成规模

安全运营专业人才普遍稀缺，人工运营方式难以实现有效的规模化管理。人工的方式对 100 台或 1000 台设备进行安全管理存在可行性，但是面对上万台设备就不现实了。自动化安全运营使人工操作变少，自动化任务变多，可有效实现安全规模化管理。

4、同参与

安全运营若由安全团队独自完成，通常在落地环节非常难，效果达不到预期。自动化安全运营将安全措施或安全工具嵌入到研发运维流程中并自动化完成，驱动研发人员、运维人员甚至业务人员一起参与运营，让安全性成为系统或产品的一种天然属性，整个过程也间接提高了人员信息安全意识水平。

二、互联网系统自动化安全运营具体实践

互联网系统特点是面向外部不确定对象提供服务，时刻可能遭受外部大量的多样化攻击威胁，因此在时间、资源有限的情况下互联网系统应得到优先保障。兴业证券从互联网信息资产管理、风险检测、风险处置 3 个方面形成了自动化的闭环安全运营。图 1 是互联网系统自动化安全运营架构，建立互联网信息资产库，对互联网信息资产进行统一管理；针对库中的信息资产实施体系化的安全检测，发现系统存在的安全风险；当发现安全风险后，自动对接 CMDB、ITSM 工单、安全运营集中管理平台等系统进行处置，形成安全风险的闭环处置。

（一）互联网信息资产集团化管理

当前兴业证券针对互联网信息资产已实现了集团化安全管理，建立了集团互联网信息资产库，直接纳管包含 10 个子公司在内的整个集团互联网信息资产，并实时对集团互联网信息资产使用情况进行安全监控，如图 2 所示。

1、构建集团统一互联网信息资产库

兴业证券采用管理与技术相结合的方式构建了整个集团统一的互联网信息资产库，资产库包含了 IP、端口、协议、状态、服务、产品类型、版本、位置、所属系统、责任人等内容。首先与防火墙进行自动对接，自动获取防火墙 NAT 映

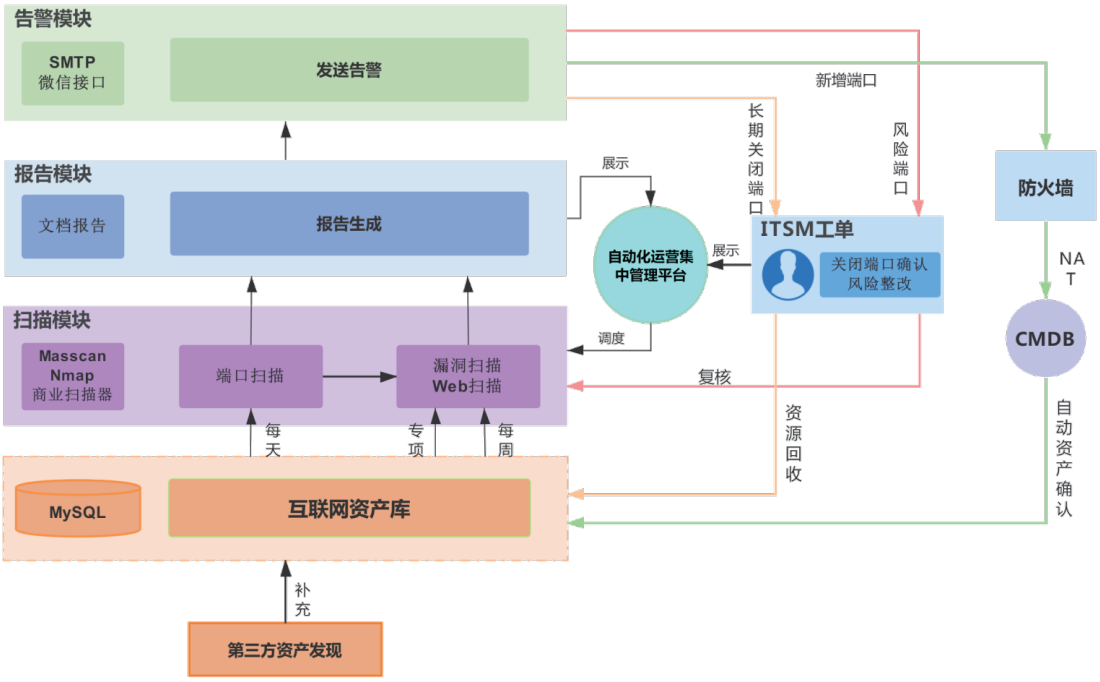


图 1：互联网信息资产自动化安全运营架构

射表并动态更新互联网信息资产库；其次通过管理手段要求各子公司定期反馈互联网信息资产情况，作为互联网信息资产库的线下补充；最后借助外部安全供应商信息资产核查技术全面发现互联网信息资产和敏感信息。通过多种方式相结合的动态信息资产管理，既能掌握最新互联网信息资产情况，又能把存在互联网中隐蔽的信息资产或敏感信息挖掘出来。

2、自动安全监控互联网系统使用情况

互联网系统的变化涉及到网络攻击暴露面，未及时掌握变化情况极大可能被外部攻击利用。兴业证券对互联网系统使用情况实施实时自动监

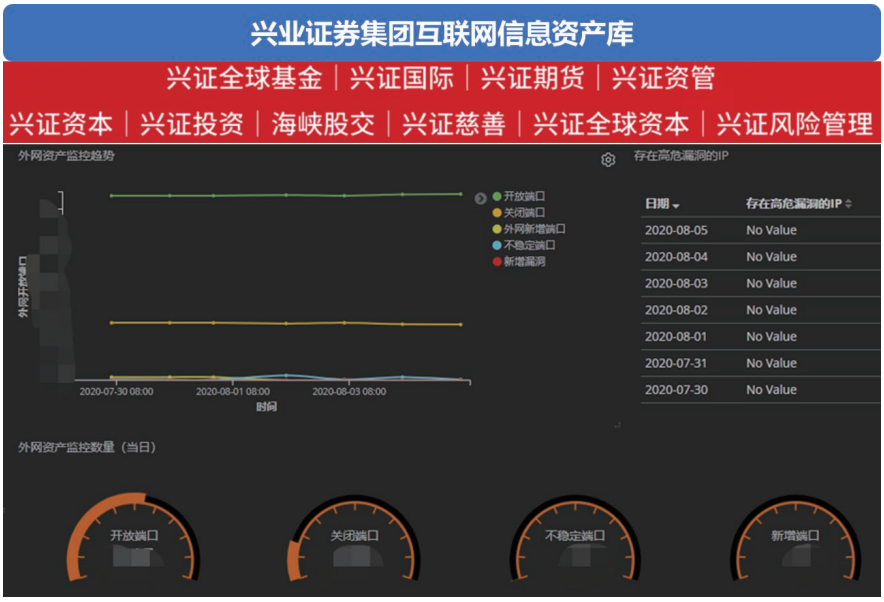


图 2：互联网信息资产使用情况实时监控

表 1：端口定义

序号	类别	定义
1	开放端口	资产库中对外提供服务可访问的端口
2	关闭端口	资产库中对外未提供服务但存在互联网映射的端口
3	不稳定端口	资产库中间歇性关闭或开放的端口
4	新增端口	未在资产库中，新增对外提供服务且可访问的端口

控，如图 2 所示，自动监测开放端口、关闭端口、不稳定端口和新增端口，端口定义见表 1。针对不稳定端口和新增端口自动调度安全检测引擎进行安全测试及风险整改，保障其安全性。互联网系统自动安全监控不仅能实时掌握互联网系统使用变化情况，也可有效控制私自映射外网的违规行为，如：研发、运维人员为了测试方便私自把测试环境映射到互联网，则可被及时发现并给予通告。

（二）互联网系统安全风险检测

在互联网系统使用情况得到有效、清晰、动态的监控后，需要高效体系化的安全风险检测机制开展安全评估和发现安全风险。兴业证券建立了互联网系统自动化安全风险检测体系，由自动化调度引擎按计划任务调度多种检测工具实施安全检测，形成了多层次、全范围、高强度的自动化安全检测机制，如图 3 所示。

1、多层次

自动化安全风险检测体系设置了天、周、月、年不同检测周期，每种检测周期实施不同的检测手段，满足合规要求的同时可及时发现安全风险。如图 4 所示，以天周期中网站可用性监控为例，对网站平稳度异常、挂马、黑链、篡改等安全风险进行实时监控，一旦发现异常在分钟级内告知到相关责任人进行处置响应。

2、全范围

自动化安全风险检测体系的运行是基于集团互联网信息资产库，覆盖了包含子公司在内整个兴业证券集团互联网信息资产，涉及服务器、Web 应用系统、终端程序和移动 APP 等多种信息资产，实现了安全规模化统一管理。

图 3：互联网信息资产自动化安全检测机制

图 4：网站可用性实时监控

3、高强度

自动化安全风险检测体系每种检测周期均采用了多种检测手段，其中漏洞检测、基线核查、攻击监测等多种手段均自动化完成，实现了高频高强度的安全检测力度，同时也呈现了异构的安全检测形态，不同检测手段间相互取长补短，大幅提升了安全风险的检测效果。

（三）互联网系统安全风险处置

完成互联网信息资产管理、安全风险检测后，应对安全检测后发现的安全风险及时进行处置响应。在处置过程中，需快速定位系统责任人，及时告知责任人进行风险整改、跟踪整改过程及复核整改结果。兴业证券通过自动化调度引擎与 CMDB、ITSM、邮箱系统、安全运营集中管理平台等系统进行自动对接，根据 IP 地址调用 CMDB API 接口定位系统责任人，再根据责任人信息调用 ITMS 工单系统 API 接口创建工单发布整改任务，同时通过邮箱系统进行邮件告知。安全风险的生命周期管理信息同步至安全运营集中管理平台，进行统一集中可视化管理。

如图 5 所示，以定期例行漏洞自动化扫描闭环管理流程为例，通过调度引擎将安全扫描引擎、ITSM 工单系统、安全运营集中管理平台（态势

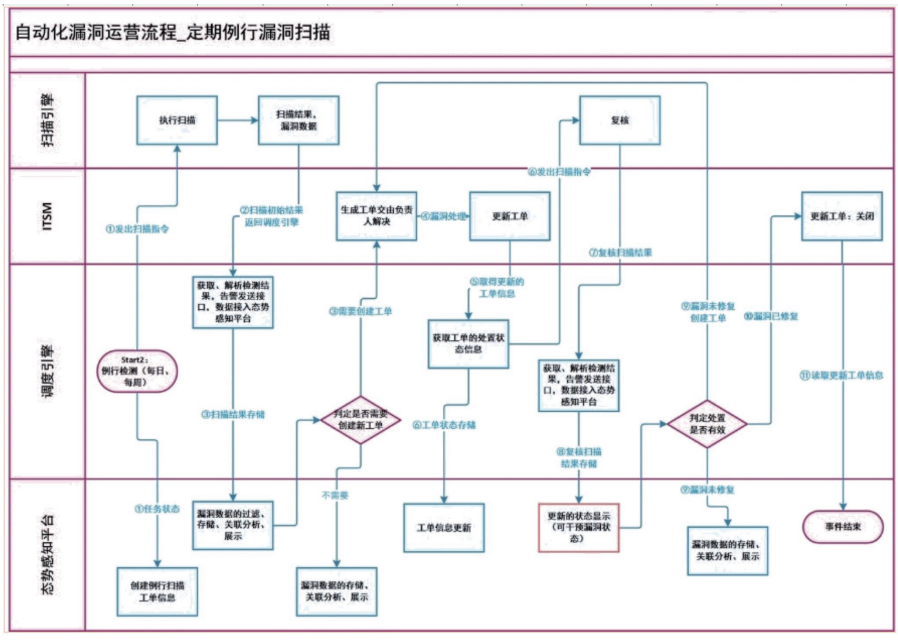


图 5：示例：定期例行漏洞扫描闭环管理流程

感知平台）有机结合成一体，形成了安全风险自动化检测、确认、定位、整改、复核的闭环处置流程。

三、总结与展望

互联网系统自动化安全运营是基于自动化技术从互联网信息资产管理、安全风险检测和安全风险处置三个方面对互联网系统开展闭环的自动化安全运营，如图 6 所示。信息资产管理具备了集团全范围覆盖、多方式动态补充和高效率自动监控等能力，安全风险检测方式成体系化、检测手段呈组合拳、检测效率靠自动化，安全风险处置中自动对接周边系统、自动定位责任人、可视化集中管理安全风险，形成了一个牢固的闭环互联网系统自动化安全运营铁三角。当前兴业证券在互联网系统自动化安全运营过程中基于整个集团互联网信息资产建立了近 20 个自动化安全运营流程，将高危风险响应时间从天降低至分钟级别，提升了互联网系统标准化安全管理水平和安全运营效率，实现了互联网信息资产安全规模化管理。

兴业证券将继续研究和完善互联网系统自动化安全运营体系，加强安全基础设施建设和安全风险的集中统一管理，强化集团化安全管理服务，探索更加领先的安全检测方式，进一步提升安全风险检测能力，构建更多可落地的自动化安全运营流程以进一步提升安全运营效率，实施常态化安全攻防实战演习持续提升互联网系统安全运营水平。

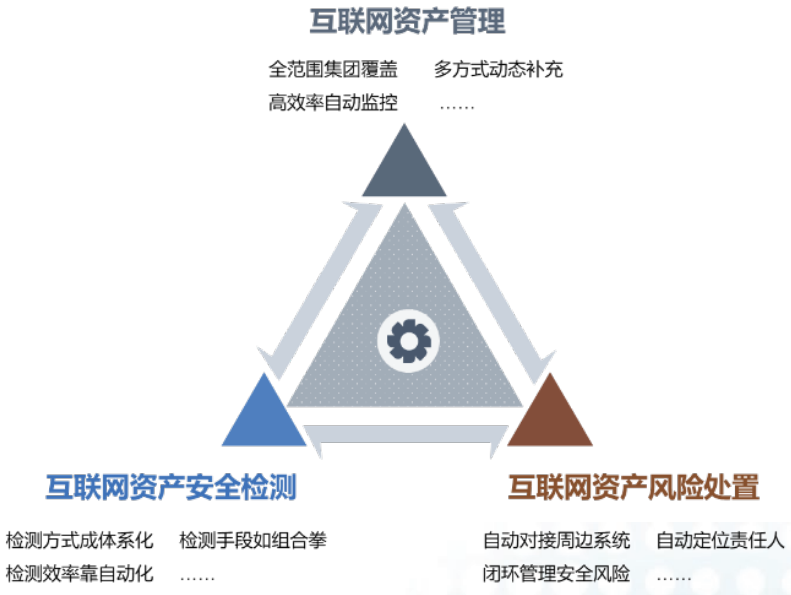


图 6：互联网系统自动化安全运营铁三角

基于开源平台和威胁情报的自动化拦截技术实践

赵川 / 国联证券股份有限公司 zhaoc@glsc.com.cn

金融企业部署了大量的安全设备及平台对日趋加剧的网络攻击进行安全防护，面对海量的告警数据，如何快速从中识别出真正的恶意地址对其实施拦截是安全人员急需解决的问题。国联证券基于开源组件，通过自建统一日志管理平台，对各类安全设备告警日志进行统一管理，并采用自动化方式，将告警源地址汇集清洗后，发往第三方威胁情报平台进行威胁查询，对于查询结果明确为恶意的地址，自动在边界防护设备中实施永久封禁，实现统一日志平台、威胁情报、安全设备之间的多方联动。

一、引言

近年来，网络攻击呈现攻击源多样化的趋势，来自云主机、工控设备、智能产品等具有联网功能的网络终端发起的攻击流量占比明显增多。根据国家计算机网络应急技术处理协调中心发布的《2019 年中国互联网网络安全报告》显示，在 DDOS 攻击中参与真实地址攻击的肉鸡达到 340 余万个。网络安全威胁信息共享平台发布的《2020 上半年公共互联网网络安全态势及威胁监测处置报告》统计数据显示，该平台收集的恶意 IP 地址数量约 188 万个，以恶意扫描服务器 IP 地址为主。

金融企业由于行业特殊性，天然成为网络攻击的首选目标，金融行业的重要信息系统无时无刻不面临来自互联网的攻击流量，在这些流量中，又以利用特定漏洞为目的的无差别扫描为甚。同时金融企业大多已经部署了较为全面的安全防护

设备，如防火墙、Web 应用防火墙、入侵检测 / 防御等，对攻击行为进行拦截并产生攻击告警，还有部分企业通过购买第三方机构威胁情报服务，对这些攻击流量的源地址进行威胁分析，采取进一步的安全防护策略，比如安全设备当前仅对当次攻击流量进行检测并拦截，若该地址进行下一次攻击时，安全设备仍需要再次识别，所以对这些地址进行永久封禁是比较好的选择，这在重保等特殊时期是非常有用的手段。但由于安全设备的多样性，安全设备之间告警日志无法统一管理，大多安全设备无法直接与威胁情报系统进行集成，面对海量的告警数据，如何快速从中识别出真正的恶意地址，给安全人员带来了不小的挑战。另一方面，由于金融信息系统运行实时性要求较高，为了避免变更对业务系统带来不可预估的影响，通常在夜间执行变更操作，这就压缩了运维人员的可操作时间，在识别出真正具备威胁的恶意地址后，如何在安全设备中对这些地址进行准确快速地实施封禁策略同时避免频繁的变更，也是安全运维人员需要着重考虑的问题。

国联证券基于开源组件，通过自建统一日志管理平台，对各类安全设备告警日志进行统一管理，并采用自动化方式，将告警源地址汇集清洗后，发往第三方威胁情报平台进行威胁查询，对于查询结果明确为恶意的地址，自动在边界防护设备中实施永久封禁，达到不同安全设备、威胁情报、统一日志平台三者之间多方联动效果。

二、技术架构

（一）总体设计

本技术实践总体设计框架由安全设备、统一日志平台、威胁情报源和处理引擎四部分组成，技术架构如图 1 所示。

（1）安全设备

安全设备在企业安全防护中起到攻击监测告警和攻击拦截作用，是安全日志的直接输出者和防护策略生效者。但在企业安全实践中，部署的安全设备往往存在品牌多样、类型繁复等问题，不同类型品牌的安全设备提供的安全功能不尽相同，产生的告警日志也无法做到格式统一，需要在统一日志平台进行结构化处理后再行入库。另外由于边界安全设备起到对企业整网入口的防护功能，所以最终经威胁情报判定为恶意的源地址，应在边界设备进行黑名单写入。

（2）统一日志平台

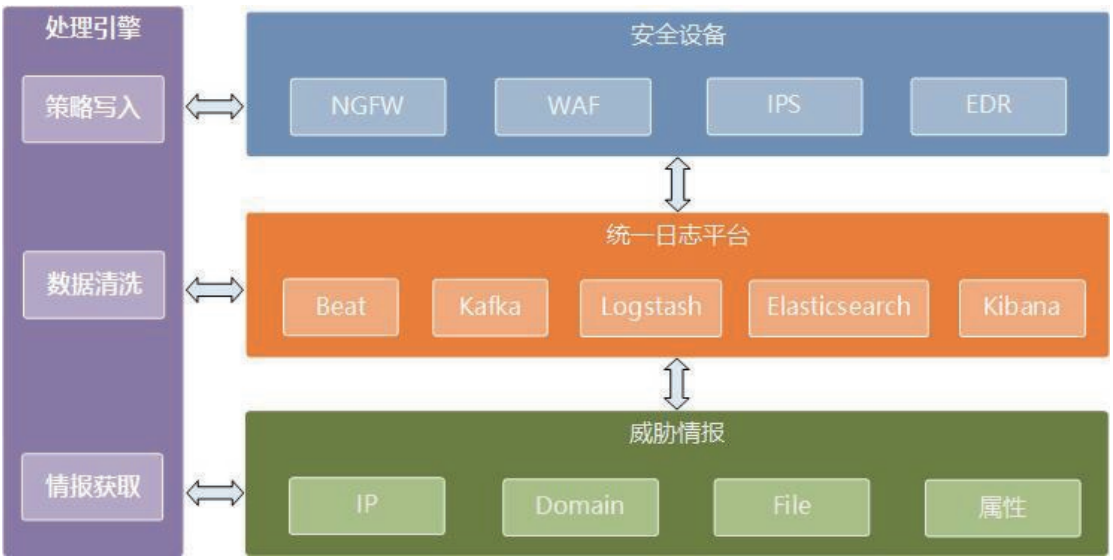


图 1：技术架构

统一日志平台是本次技术实践的核心节点，提供了安全日志处理、存储、分析、查询等核心功能，也是安全运营、态势感知等上层安全系统依赖的基础。本例直接使用 ELK+Kafka 的形式搭建自有统一日志平台。ELK 指的是 Elastic Stack，由 Beat、Logstash、Elasticsearch 和 Kibana 四个开源组件组成，Beat 用于接收安全设备的 Syslog 日志；Logstash 是服务器端数据处理管道，能够同时从多个来源采集数据，转换数据；Elasticsearch 是一个基于 Lucene 的搜索和分析引擎；Kibana 可以对 Elasticsearch 中存储的数据以图形和图表形式进行可视化；Kafka 是一个消息队列组件，加入 Kafka 的目的，是为了将安全设备的原始日志通过 Beat 接收后先发送至 Kafka 消息队列，再由 Logstash 进行消费，避免 Logstash 在日志转换过程中由于处理性能问题导致原始日志积压或丢失。

（3）威胁情报源

根据 Gartner 对威胁情报的定义，威胁情报是某种基于证据的知识，主要内容为用于识别和检测威胁的失陷标识，市面上知名的威胁情报源有微步在线情报社区、绿盟威胁情报中心、IBM X-Force Exchange 威胁情报共享平台等，在本例中，主要利用了微步威胁情报的云 API 接口，实现自动化的情报查询和获取。

（4）处理引擎

处理引擎的作用主要是检索日志平台中存储的结构化日志数据，将源地址进行清洗后批量在威胁情报源中查询，并将判定为恶意的地址自动写入边界防护设备要读取的黑名单列表。

（二）处理流程

整体的处理流程大致可分为安全日志转换与存储、地址清洗与情报查询、黑名单写入与策略下发三个步骤，如图 2 所示。

1、安全日志转换与存储

在本次技术实践中，采用标准化的 Syslog 协议，将安全设备的原始日志传送至统一日志平台，由日志平台对原始日志进行统一结构化处理，格式化成由 Key-Value 构成的 JSON 键值对形式保

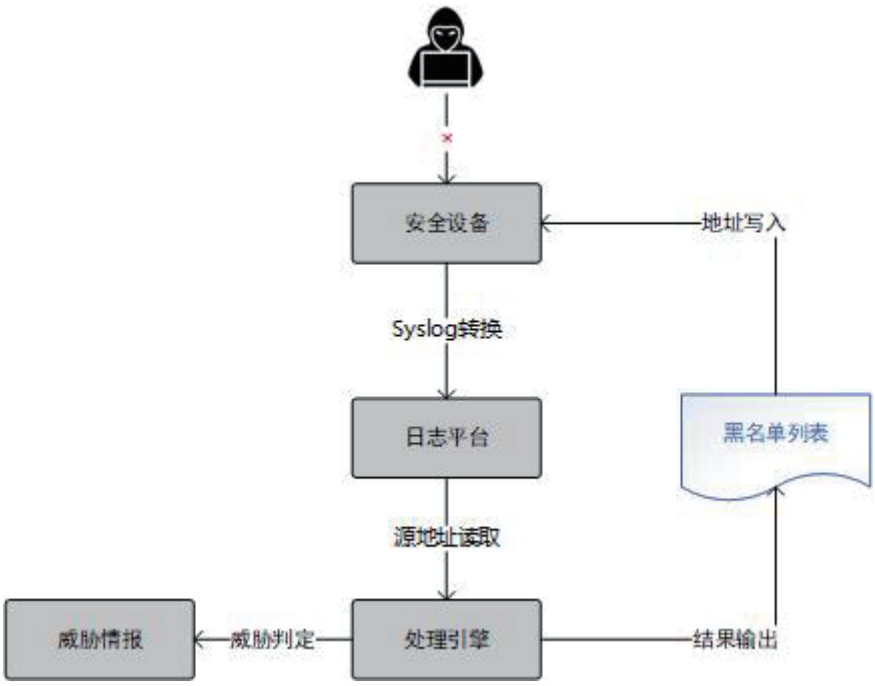


图 2：处理流程

存在日志平台中，方便日后对日志的分析和查询。本例中我们在部署在网络边界的下一代防火墙（NGFW）上启用了动态地址对象读取功能，并设置一条对动态地址执行 DROP 的默认策略，实现自动化的地址读取和策略加载。

2、地址清洗与情报查询

处理引擎可以实现读取日志平台的结构化日志数据，提取有用的字段，本例中主要是源地址这一字段，在获取源地址后，需要进行去重，再将地址列表通过请求威胁情报 API 方式，查询该部分地址的威胁属性，确认是否为恶意地址。

3、黑名单写入与策略下发

在经过威胁情报源判定后，可以获得源地址的威胁属性，对于判定为恶意的源地址，由处理引擎输出至黑名单列表，该列表以 Web 资源方式部署在 Web 服务上供 NGFW 读取，NGFW 在读取黑名单列表后，会自动将恶意 IP 关联至策略源地址对象，执行阻断操作。

三、具体实现

按照核心功能的不同，可以从统一日志平台和处理引擎两个模块进行技术实现，下面主要对这两个模块的具体实现细节进行分析。

（一）统一日志平台

统一日志平台在 Elastic Stack 技术栈的基础上，加入 Kafka 消息队列，实现对安全设备告警日志的统一管理，系统架构如图 3 所示。

1、日志采集

日志采集主要由 Syslog、Beats、Kafka 三部分组成，完成了从安全设备源端将告警日志采集至消息队列的过程。

(1)Syslog

Syslog 广泛应用于系统日志，是日志传输的消息标准，安全设备均支持以 Syslog 方式将自身日志发送至第三方平台。这里提供两种收集 Syslog 的方式，第一种可以依赖 Linux 系统中的 rsyslog 系统，接收安全设备的日志，转存为文件保存在 rsyslog 服务器上，再使用 Beats 家族的 Filebeat 组件，对日志文本进行采集；第二种可以直接使用 Beats 的 Syslog 功能，监听安全设备发送过来的 Syslog，两种方式的区别在于前一种多保留了一次设备的原始日志。

(2)Kafka

Kafka 是一种高吞吐的分布式消息发布订

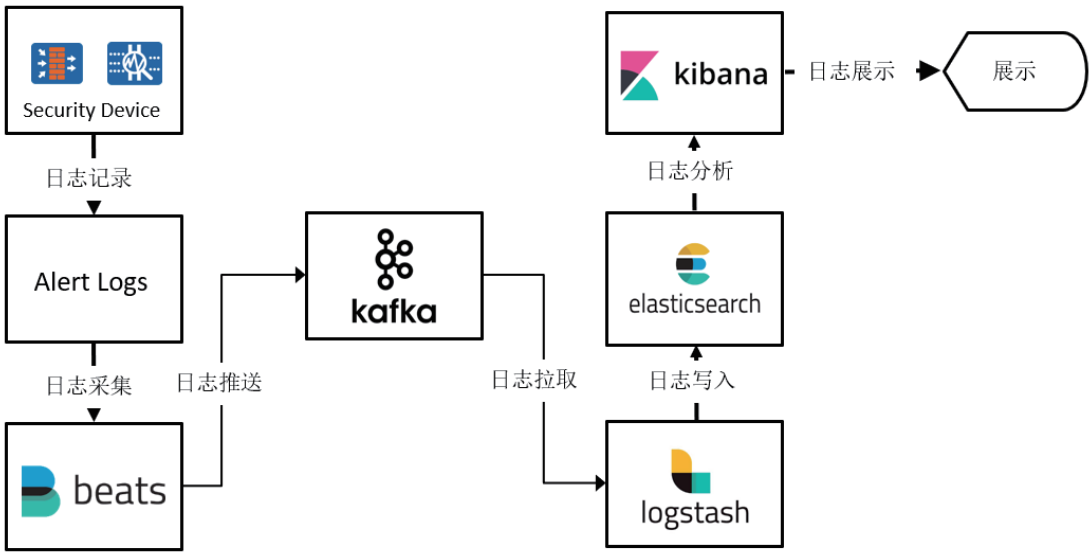


图 3：统一日志平台架构图

阅系统，引入 Kafka 的目的，是为了解决下游的 Logstash 在进行日志转换时可能产生的日志丢失问题，若直接将 Syslog 发送至 Logstash，因 Logstash 需要进行大量的正则匹配处理，当 Logstash 未能及时处理消息时，就可能造成日志丢失。而引入 Kafka 后，Logstash 就可以根据自身的处理能力，顺序从 Kafka 消费日志数据，避免了日志丢失。

在接收 Syslog 后，需要通过 Beats，将 Syslog 发送至 Kafka。在 Kafka 的配置上，应考虑建立的 Topics 数，在本例中，我们根据不同品牌的产品，建立 Topic，对于同一品牌的设备，无论部署位置，均发送至同一 Topic，便于读取，如图 4，建立了 checkpoint、paloalto、waf 三个 topic，分别存储 NGFW 和 WAF 的告警日志，经过 Beats 采集后的日志，安全设备的原始 Syslog 会被存放在 message 字段中，如图 5。

2、日志转换

日志转换的作用是将 Message 中的 Syslog 进行结构化清洗，转换成键值对形式，再存储至 Elasticsearch。这里需要依赖 Logstash 组件完成，Logstash 先从 Kafka 消息队列消费日志数据，再对数据进行清洗与转换，丢弃不需要的字段，对于部分日志内容，添加自定义字段名，然后生成 Elasticsearch 需要的索引。表 1 是对 WAF 日志进行转换后，最终的字段形式（部分）。

图 6 展示了经过 Logstash 处理后，一串原始 Syslog 在 Elasticsearch 中存储的形式。在对 Syslog 进行转换时，需要在 Logstash 的配置文件中，按照不同的 Kafka Topic，编写不同的解析策略，生成不同的 Elasticsearch Index。

3、日志存储

(1) Elasticsearch

经过 Logstash 转换的 Syslog 数据，便可以 Kafka Topic 加日期为名称在 Elasticsearch 建立索引。Elasticsearch 是 Elastic Stack 技术栈中最重要的组件，是一个高扩展的分布式全文检索引擎，可以近乎实时的存储、检索数据，同时支持方便

Properties | Offsets

Topic	Partition	Start	End	Offset
ad	0	50,498,650	58,142,687	58,142,679
checkpoint	0	138,528	163,068	163,068
paloalto	0	13,210	22,703	22,703
waf	0	1,874,519	1,893,738	1,893,738

图 4：Kafka Topics


```
{
 "@timestamp": "2021-02-08T01:06:31.261Z",
 "@metadata": {
   "beat": "filebeat",
   "type": "_doc",
   "version": "7.9.1"
 },
 "message": "2021-02-08 09:06:27 [redacted] {\"vsite_name\": \"None\",\"count_num\":1,\"country\":\"CN\",\"reason\":0,\"def_ip\":\"\",\"policy_id\":2359297,\"dmac\":\"\",\"attack_type\"
 "input": {
   "type": "log"
 },
 "source": "waf",
 "ecs": {
   "version": "1.5.0"
 },
 "host": {
   "name": "[redacted]"
 },
 "agent": {
   "ephemeral_id": "d56fdfe7-70f1-4406-87a1-ca071e44dbf0",
   "id": "65dcdb70-3836-4901-8877-754628c83d4b",
   "name": [redacted],
   "type": "filebeat",
   "version": "7.9.1",
   "hostname": [redacted]
 },
 "log": {
   "offset": 34057,
   "file": {
     "path": "[redacted]waflog_2021-02-08-09.log"
```


图 5：Topic Message

表 1

字段名	类型	解释
time	timestamp	日志产生时间
sip	text	源地址
action	text	策略动作（放行/丢弃）
domain	text	域名
site_name	text	站点名称
dip	text	目的地址
dport	text	目的端口
request	text	http 请求
rule_name	text	策略名称
asset	text	资产名称
device	text	设备

t	action	prevent
t	asset	Nginx
t	attack.country	China
t	characters	7,38,1,27,11,/../../../../../../../../../etc/passwd|||||
t	dev_ip	1 2
t	device	
t	dip	1
t	domain	www.glsc.com.cn
#	dport	443
t	event	路径穿越攻击
t	kafka_topic	waf
t	method	GET
t	protocol_type	HTTPS
t	proxy_info	None
t	request.header.Accept	text/html,application/xhtml xml,application/xml;q=0.9,*/*;q=0.8
t	request.header.Accept-Encoding	gzip, deflate
t	request.header.Accept-Language	en-US,en;q=0.9,zh-CN;q=0.8,zh;q=0.7
t	request.header.Connection	Keep-Alive
t	request.header.Cookie	rememberMe=test
t	request.header.Host	www.glsc.com.cn
t	request.header.Range	bytes=0-102400
t	request.header.Referer	https://www.baidu.com/
t	request.header.User-Agent	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:6.0) Gecko/20100101 Firefox/6.0
t	risk	高
t	rule_name	password_access

图 6：JSON Syslog

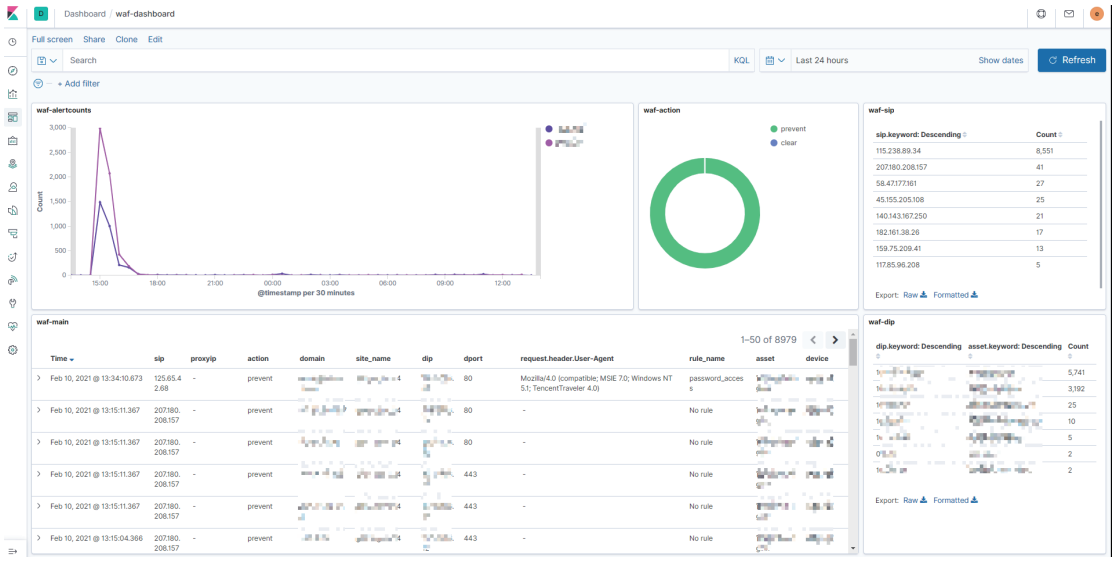


图 7：Kibana Dashboard

的进行扩展。Elasticsearch 属于非关系型数据库的一种，本质上存储的数据就是 JSON 格式的文档，这里存储的就是 JSON 格式的 Syslog 数据，Elasticsearch 对 JSON 文档中的每一个字段都会构建一个对应的倒排索引。

(2) Kibana

Kibana 则提供了一个友好的 Web 界面，可以搜索、查看、操作存储在 Elasticsearch 索引中的数据，通过使用 Kibana 能对处理后的数据进行可视化的展示。在将 Elasticsearch Index 全部汇总至 Kibana Index 后，便可以绘制仪表盘，对告警日志进行集中展示，如图 7。

（二）处理引擎

处理引擎需要实现日志检索、情报查询和黑名单写入三方面的功能，Java 或 Python 均能较为容易的实现上述功能的代码编写，这里采用 Python 语言来编写处理引擎。

另一方面，由于金融行业变更操作规范要求，一般不在日间执行变更操作，故处理引擎并不适合实时运行，因此将处理引擎设定为夜间运行是比较好的做法。处理引擎汇总前一天告警日志中的所有源地址，去重后在威胁情报源查询，再将恶意地址写入黑名单列表，最后由边界防护设备自动读取黑名单，下发拦截策略。

1、日志检索

官方提供了 Python 用于 ElasticSearch 数据检索的专用代码包，只需引入 ElasticSearch 包，便可方便的操作 ElasticSearch 存储的数据。这部分的主要代码逻辑就是从 ElasticSearch 中读取前一天所有安全设备 Index 的所有日志数据，提取源地址字段的值并进行去重，最终作为要发往威胁情源报进行查询的地址。

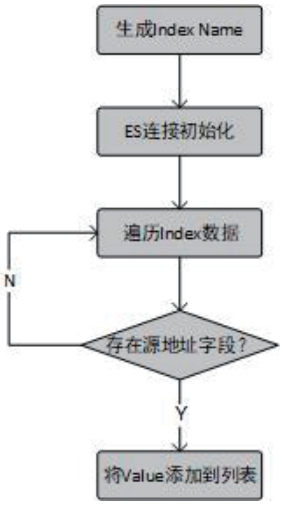


图 8：检索逻辑

2、情报查询

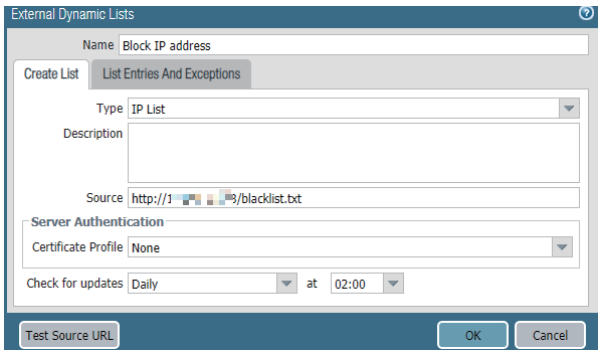
情报查询需要连接第三方威胁情报源 API，这里采用微步在线情报社区提供的在线云 API。微步云 API 提供了 IP 分析、IP 信誉、域名分析等基础功能接口，支持 HTTP 方式对这些接口进行调用，由于微步在线对一个地址是否为恶意自有的一套判定机制，这里直接采用微步情报的判定结果，scene/ip_reputation 接口的 is_malicious 响应字段表示该地址是否为恶意。可以将接口返回的 is_malicious（是否为恶意 IP）、confidence_level（可信度）、severity（严重级别）等数据与地址列表合并形成一个 Dataframe 格式的数据，写入查询结果文件，如图 9。

```
sip,is_malicious,confidence_level,severity
116.68.96.245,True,high,medium
58.205.224.201,False,medium,info
125.44.247.80,True,high,medium
117.242.208.76,True,high,medium
173.225.107.22,True,low,low
111.7.96.162,True,high,medium
236.254.58.40,False,high,info
36.99.136.136,True,medium,low
192.241.208.143,True,high,high
14.116.203.102,True,high,high
175.215.202.131,True,high,low
198.45.19.113,True,low,low
58.248.193.251,True,high,medium
192.241.223.35,True,high,medium
128.14.133.58,True,high,medium
192.241.217.239,True,high,medium
```

图 9：检索结果

3、黑名单写入

处理引擎的黑名单写入需要先过滤出 is_malicious 为 Ture 的行，再将这些行的源地址，写入一个文本文件，这也在处理引擎实现。之后边界防火墙便可以读取写有恶意地址的文本文件，以 Paloalto 防火墙为例，Paloalto 防火墙的外部动态列表功能，支持从一个 URL 读取地址列表，写入自身的地址对象，并且可以规定读取的时间和频率。为了配合处理引擎的运行时间，通常设定在处理引擎生成每日的封禁地址列表之后进行读取，随后便可创建一条默认拦截策略，源地址设置为外部动态列表对象。对于不支持类似外部动态列表功能的防火墙，也可以使用脚本方式，通过命令行将这些恶意地址批量刷入配置文件，以达到相同的功能。



四、总结

本次技术实践对传统安全运维中依赖人工执行恶意地址封禁这一典型场景进行了优化与改进，首先利用 Elastic Stack 开源平台对不同品牌类型的安全设备告警日志实现了统一管理，相较于使用某一厂商的特定统一日志管理系统，使用开源组件具有更好的灵活性与可维护性。在统一日志管理的基础上，本次技术实践还引入了威胁情报，可以更好地对恶意地址从各维度进行分析，最终结合威胁情报判定结果，对恶意地址实施自动化封禁，实现统一日志平台、威胁情报、安全设备之间的多方联动。

随着网络攻击的日益增多，也催生了更多种类的安全设备、安全技术或安全名词，对于金融企业来说，如何将不同的安全产品纳入统一管理，避免产生安全孤岛，真正实现多方联动，是安全建设的重点也是难点。国联证券将对此进行更多的探索，提高安全运营水平。