 DOI: https://doi.org/10.12349/iser.v6i5.5699

A Real-Time Processing Method of Cybersecurity Log Data Based on EFK Architecture

Chunxia Xu

Siemens (China) Co., Ltd. Suzhou Branch, Suzhou, Jiangsu, 215000, China

Abstract

Facing the real-time processing demands of massive cybersecurity logs in the era of Internet of Everything, the traditional ELK (Elasticsearch, Logstash, and Kibana) architecture has issues such as high resource occupancy of the Logstash component and limited data processing throughput. Thus, it is hard to meet the requirements of low-latency and high-concurrency log analysis. This paper proposes an EFK architecture (Elasticsearch, Flink, Kafka and Kibana) based on Flink streaming computation, which reconstructs the log processing pipeline and optimizes system scalability to achieve efficient collection, real-time analysis, and accurate threat detection for cybersecurity logs. Experimental results show that the proposed solution exhibits significant advantages over the traditional ELK architecture in simulated high-concurrency scenarios with tens of millions of logs per second.

Keywords

Cybersecurity; ELK architecture; Flink streaming computation; EFK architecture; Threat detection

基于 EFK 的网络安全日志数据的实时处理方法

许春霞

西门子（中国）有限公司苏州分公司，中国 · 江苏 苏州 215000

摘　要

面对万物互联时代的海量网络安全日志的实时处理需求，传统ELK（Elasticsearch、Logstash和Kibana）日志架构因Logstash组件资源占用率高、数据处理吞吐量受限等问题，难以满足低延迟、高并发的日志分析要求。本文提出一种基于Flink流式计算的EFK架构（Elasticsearch、Flink、Kafka和Kibana），通过重构日志处理链路与优化系统扩展性，实现网络安全日志的高效采集、实时分析与精准威胁检测。实验结果表明，在模拟每秒千万级日志的高并发场景中，本方案较传统ELK架构呈现显著优势。

关键词

网络安全；ELK机构；Flink流计算；EFK架构；威胁检测

1 引言

随着万物互联时代的到来，互联网已成为现代社会不可或缺的基础设施，深刻改变了人们的生活、工作和社会运行方式。然而，网络规模的扩大与复杂性的提升，使得网络安全威胁呈现出多样化、隐蔽化和智能化的特征。根据网络安全公司 Orange Cyberdefense 报告数据显示，2023 年全球勒索攻击受害者数量同比增长超过 40%，高级持续性威胁（APT）、勒索软件攻击和漏洞利用等新型攻击手段对政府、企业和个人造成了严重的经济损失与隐私泄露风险 [1]。在此背景下，构建高效可靠的网络安全防御体系已成为学术界与工业界的共同目标。

网络安全日志作为网络活动与安全事件的关键记录载体，是威胁检测、入侵溯源和态势感知的核心数据来源。传统的日志管理系统（如 Syslog、SNMP 等）虽然在基础数据采集层面发挥了重要作用，但在应对海量异构日志（如网络流量日志、主机行为日志、应用层日志等）的实时处理、高效存储和深度分析方面存在显著不足。一方面，随着云计算、物联网（IoT）和 5G 技术的普及，网络环境产生的日志规模呈指数级增长，传统集中式日志存储架构面临吞吐量瓶颈与存储成本压力；另一方面，基于规则或签名的传统检测方法难以有效识别新型攻击模式，导致误报率（ Positive）和漏报率（ Negative）居高不下 [2]。此外，日志数据中隐含的上下文关联性与时序特征尚未被充分挖掘，限制了威胁情报的自动化生成能力。

本文针对上述挑战，提出一种基于 EFK 架构的网络安全日志系统设计方案。通过引入 Flink 流式计算引擎重构数据处理链路，实现日志采集、解析和分析的端到端低延迟流水线；结合动态水平扩展机制与数据顺序性保障算法，显著

【作者简介】许春霞（1981–），女，中国山东菏泽人，硕士，工程师，从事计算机网络安全与工业互联网研究。

提升系统吞吐量与复杂攻击模式的实时检测精度。方案创新性体现在：1）利用 Flink 状态管理与时间窗口机制挖掘日志时序关联性，增强上下文感知能力；2）通过 Kafka 解耦日志采集与处理模块，优化资源利用率。为大规模网络安全日志的实时监控与智能分析提供了高效、可扩展的技术支撑。

2 相关技术

2.1 ELK 日志解决方案

ELK 是 Elastic 公司研发的一套完整的日志收集、分析和展示的企业级解决方案。起初，ELK 日志管理系统解决方案由 Elasticsearch(ES)、Logstash 与 Kibana 核心组件组成。它通过 Logstash 接受日志数据，并经由 Logstash 进行解析、清洗和结构化处理，转换成统一格式后发送到 ES 中存储和索引。ES 作为一个分布式搜索引擎，可以对这些结构化日志进行高效的搜索和聚合分析。最后，用户通过 Kibana 访问 ES，进行日志查询、分析和可视化展示，帮助实时监控系统状态、排查故障或进行业务分析 [3]。其各个组件的主要功能职责与优缺点如表 1 所示。

表 1 ELK 的功能与优缺点

组件	核心功能	优点	缺点
ES	分布式存储与索引； 实时全文搜索与分析； 支持聚合计算	毫秒级搜索； 横向扩展性强； 容错机制完善； 支持结构化与非结构化数据	集群运维复杂； 吞吐量受硬件限制； 冷数据存储成本高
Logstash	数据采集与输出； 数据清洗与转换	插件丰富； 灵活的数据处理能力； 支持复杂过滤逻辑	资源消耗高； 实时性略低； 配置复杂度高
Kibana	数据可视化； 交互式数据探索； 索引管理与监控	直观的可视化操作； 支持告警与自动化任务	大规模数据渲染性差； 高级功能需学习查询语法

2.2 Kafka

随着互联网数据规模的爆发式增长，ELK 架构在高并发场景下面临因流量过载引发处理延迟或数据丢失的重大挑战。为此，分布式消息队列 Kafka 被引入 ELK 日志架构中。它可缓存海量日志数据，下游处理组件（如 Logstash）可根据自身处理速率从 Kafka 中按需消费数据，避免系统过载，形成流量削峰机制。

Kafka 由 Producer（生产者）、Topic（主题）、Partition（分区）、Consumer（消费者）和 Broker 核心组件组成 [4]。生产者可以将数据发布到指定的主题，而消费者可以订阅这些主题并消费其中的数据；主题是一个逻辑概念，一种消息类型可以看作同一个主题；同一主题下的数据可以分为多个分区，使得消息被并行消费，从而提高 Kafka 的吞吐率；Broker 是 Kafka 的物理服务实例，负责存储和处理消息。某一个主题下的分区可以存储在不同的 broker 中，以实现数据备份保存。

因 Kafka 的流量削峰机制，Kafka 与 ES、Logstash、Kibana 协同构建出高可靠的日志管理解决方案，在一定程度上实现了从数据采集、缓冲、处理到可视化分析的全链路闭环 [5]。

2.3 Flink

Flink 的前身为 Stratosphere 项目，由德国柏林理工大学于 2008 年启动，目的是为大规模数据分析构建新一代高性能引擎。2014 年，该项目正式更名为 Flink 并完成孵化，成为 Apache 软件基金会旗下的开源分布式流处理引擎。Flink 秉承“流处理优先”的设计哲学，以有状态计算为基石，将批处理视为有限流数据的特例，从而在编程接口（如 DataStream API）、执行引擎与资源管理层面上构建了流批一体的分布式实时数据处理框架。其分布式架构采用 Master-Slave 模式，主要由 JobManager（作业管理器）、TaskManager（任务管理器）和 ResourceManager（资源管理器）组件构成 [6]。

作业管理器（Master）在 Flink 中承担着核心的调度与管理职责。它负责接收客户端提交的作业，并将其转换为物理执行图，再将各个任务分发至任务管理器执行。作业管理器会向任务管理器请求所需的插槽（Slot），以确保作业能够分配到足够的计算资源。同时，它还负责容错机制的协调，管理检查点（Checkpoint）的创建与恢复，以保障作业在发生故障时能够实现一致性语义（如精确一次处理）。此外，作业管理器还持续监控各个任务的执行状态，并在任务失败时进行重新调度，以维持作业的稳定运行。

任务管理器（Slave）是 Flink 集群中负责实际任务执行的工作节点。每任务管理器包含多个任务插槽，用于实现资源隔离并支持任务的并行处理。在作业运行过程中，任务管理器不仅承担着具体计算，还通过网络与其他任务管理器进行数据交换，以完成跨节点的数据传输。此外，任务管理器还会将任务的运行状态和相关统计信息实时反馈给作业管理器，用于整体作业的监控与调度。

资源管理器是 Flink 集群中负责资源统一管理与协调的组件。它主要负责管理任务管理器的插槽资源，并根据作业管理器的请求分配空闲插槽，以满足作业的运行需求。同时，资源管理器还具备资源回收能力，能够终止空闲的任务管理

器实例，从而释放计算资源并提升集群整体利用率。在资源紧张或任务规模扩展的情况下，资源管理器还可以与底层资源平台协同，动态申请并启动新的任务管理器实例，实现计算资源的弹性扩展。

Flink 作为第三代流处理引擎的标杆，凭借其融合低延迟、高吞吐、容错机制与强一致性的核心特性，在实时计算领域占据领先地位。

3 基于 EFK 的数据实时处理方法

通过以上技术分析，ELK 日志管理系统面临着显著瓶颈：资源占用率高、吞吐性能不足、实时响应滞后等问题，难以满足现代工业场景对高可靠性、实时处理及海量日志承载的严苛要求。针对上述缺陷，本文设计了一种基于 EFK 架构的增强型实时日志处理方案，通过优化数据采集、管道缓冲与异常恢复机制，实现日志处理全链路的可靠性提升与毫秒级延迟。EFK 整体系统架构如图 1 所示。

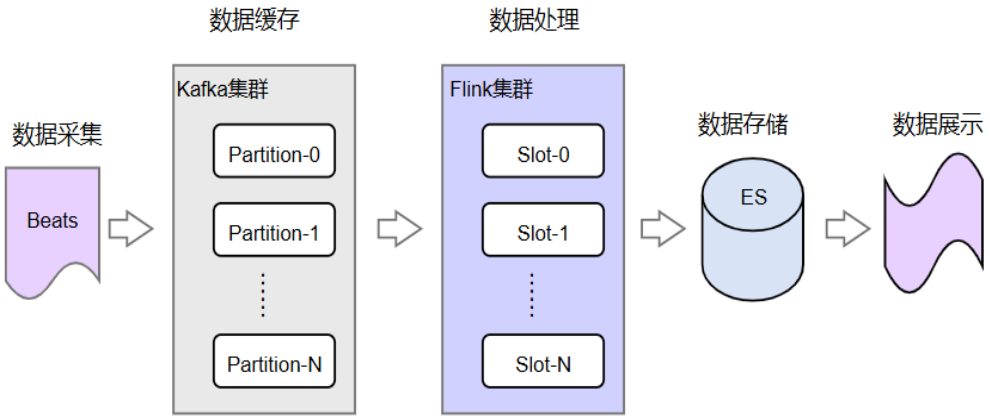

图 1 EFK 系统架构

EFK 系统架构的主要设计思想与优化策略：

1）数据采集层的轻量化重构

采用 Beats 家族作为统一采集代理，有效替代资源占用较高的 Logstash，从而提升系统整体性能与运行效率，同时提供 ACL 安全策略与 TLS 加密传输能力，更好地满足现代日志管理的要求。

2）消息缓冲层的可靠性增强设计

引入具备解耦、限流削峰等特性的高吞吐量分布式发布 - 订阅消息系统 Kafka，以增强系统的数据处理能力与可靠性：

· 多副本机制（ISR 集合）保障数据持久化。

· 动态分区策略配合智能负载均衡算法，实现每秒百万级消息处理能力。

· 基于时间窗口的 retention 策略与分层存储架构，平衡存储成本与数据回溯需求。

3）流处理层的实时计算优化

采用 Flink 流引擎实现亚秒级延迟处理与事件顺序保证：

· 水印机制（Watermark）与 AllowedLateness 策略协同，构建事件时间处理体系，解决事件乱序问题 [7]。

· 状态后端优化实现 TB 级状态数据高效管理，支持精确一次语义。

· 动态反压机制（Backpressure）与弹性扩缩容策略联动，应对流量洪峰。

4）全链路协同优化机制

Kafka 分区数与 Flink 并行度的动态映射算法，实现处理资源的最优分配。

基于 PID 控制器的批量写入阈值自适应调整，在 ES 写入阶段维持最佳 bulk size。

全链路监控指标体系（吞吐量、延迟、错误率）的实时可视化。

4 实验与分析

本文通过搭建仿真环境，对基于 EFK 架构的网络安全日志实时数据处理方案进行了系统性的可行性验证。所采集的数据来源于本公司网络安全运维平台、工控监控系统以及实验室多个工作站的真实网络日志数据，涵盖了设备运行状态、访问行为和安全事件等多个维度数据，具有代表性和复杂性。

本实验采用 Java 编程语言与 IntelliJ IDEA Community Edition 2022 开发工具，在 Dell EMC PowerEdge R440 服务器上构建虚拟化集群环境。通过搭建 3 台 Flink 和 3 台 Kafka 虚拟机构建数据处理集群。使用消息中间件 Kafka 进行数据解耦分流，Flink 负责日志的实时计算、聚合、过滤和匹配等操作。Flink 的节点 1 运行 JobManager 进程，负责协调作业在各个节点上的分布式执行。其他节点负责执行 Taskmanager 任务。考虑到发挥并行处理的最大作用，这里设置 Kafka 各主题的分区数量均等于 Flink 集群中数据处理程序的并行度的数量。例如，在 Kafka 主题 A 中，分区数量设置为 3，则把 Flink 处理程序的并行度设置为 3[8]。集群各节系统配置信息如表 2 所示。

表 2 Kafka 与 Flink 集群各节点系统配置信息

节点	操作系统	CPU	内存	硬盘
节点 1	Debian GNU/Linux 10	4 核	8G	500G
节点 2	Debian GNU/Linux 10	4 核	8G	500G
节点 3	Debian GNU/Linux 10	4 核	8G	500G

在硬件设备和数据处理程序一致的前提下，本文通过调整 Kafka 的分区数量和 Flink 的并行度，研究其对 Kafka 和 Flink 集群数据处理性能的影响。通过脚本生成不同规模的数据集，并在处理过程中不断调节 Kafka 分区和 Flink 的并行参数，以实现性能优化并验证集群的处理效率。数据处理结果对比如图 2 所示。

图中的横坐标表示 Kafka 分区与 Flink 并行度的大小，纵坐标表示一定数据量下的数据处理时间，单位是秒。对于每一种配置与数据集组合，均做了 5 次实验取平均值，从而得到了在每一种配置下的数据处理时间。经过实验发现，随着分区与并行度的增加，数据处理时间显著减少。这说明提高并行度能够有效提升系统处理性能，特别是在处理 100 万条与 150 万条数据时，优势显著。但当并行度从 6 提升至 8 时，系统处理效率的提升幅度明显减小，这说明单纯增加并行度并不能持续优化性能，必须与硬件资源形成科学匹配。对于 4 核 CPU 环境，建议采用黄金配置方案为：Flink 的并行度为 4，使其与物理核数一一对应，避免核内资源竞争；Kafka 分区设置为 4-6，使其略大于并行度以预留扩展空间。

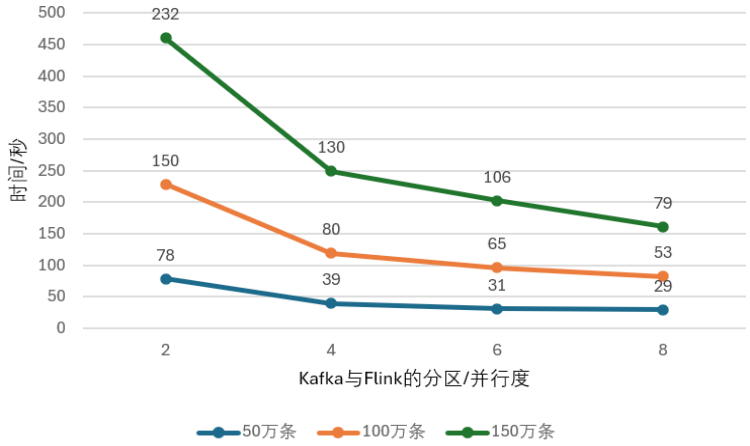

图 2 不同配置下的数据处理效能

5 结语

本文基于 Flink 构建的网络安全日志管理系统 EFK，通过流批一体架构与有状态计算引擎，实现了对海量异构日志的实时采集、高效处理与精准分析，有效解决了传统方案在高并发场景下的吞吐量不足、威胁响应滞后等问题。本系统设计方案不仅为《等保 2.0》《数据安全法》等合规要求提供了技术实现路径，更推动了安全运营从“规则驱动”到“智能感知”的范式跃迁。未来将进一步探索联邦学习框架下的威胁情报共享机制，打破安全数据孤岛，构建协同联动的主动防御生态。

参考文献

[1] Orange Cyberdefense. Security Navigator 2023.

[2] Sommer R, Paxson V. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. S&P 2010.

[3] 陈敬. 基于ELK的日志分析与异常检测系统的设计与实现(D). 西安电子科技大学, 2022

[4] 黄娅婷. 面向Kafka消息系统的性能优化方法研究(D). 桂林电子科技大学, 2023

[5] Karim M R, et al. Real-time Log Analysis using Elasticsearch. ICDCN 2021.

[6] Apache Flink®.https://flink.apache.org/.2024.

[7] 秦政,许利杰,等. 面向Apache Flink流式分析应用的高吞吐优化技术(J). 软件学报, 2024.

[8] 高立京,陈志敏,等. 基于Flink的空间科学卫星数据实时处理方法(J). 计算机仿真, 2023, 40(7): 26-31.