Vol.2021-CLE-33 No.23
2021/3/26

クラウドを利用したMoodleのログ解析環境の実装

浜元 信州[1,a)]　横山 重俊[1,2]　竹房 あつ子[2]　合田 憲人[2]

概要：LMS が大学に普及し，全学統一の Moodle や大学をまたいだ LMS なども実現し，LMS ログデータは膨大なものとなりつつある．このような大量のデータを扱う上では，クラウドの利用が有効だが，LMS ログは個人情報に関わるデータが多く，クラウド上で利用するには，匿名化や仮名化などの処理が必要なケースも多い．本論文では，NII の提供する学認クラウドオンデマンド構築サービスを利用し，組織内のサーバとクラウド環境を組み合わせたログ解析環境の実装を行った．この環境では，LMS のログは，クラウド上に仮名化を施した状態で保存し，解析時には，仮名化を解除プロキシを利用し自動解除する．本環境を群馬大学 Moodle に適用した事例を示す．

キーワード：学認クラウド, Moodle, ログ解析環境

Implemention of Log Analysis Environment for Moodle using IaaS Cloud Service.

HAMAMOTO, Nobukuni[1,a)]　YOKOYAMA, Shigetoshi[1,2]　TAKEFUSA, Atsuko[2]　AIDA, Kento[2]

Abstract: As LMSs become more and more popular in universities, Moodle and cross-university LMSs have been implemented, and LMS log data are becoming huge. In order to handle such a large amount of data, it is effective to use cloud computing. However, since LMS log data contain many personal information, it is often necessary to anonymize or pseudonymize the data in order to use them on cloud computing. In this paper, we implemented a log analysis environment that combines an organization's internal server and a cloud environment by using the GakuNin-cloud on-demand construction service provided by NII. In this environment, LMS logs are stored in the cloud with pseudonymization, and the pseudonymization is automatically removed by using a proxy when analyzing the logs. We will describe a case study of applying this environment to Moodle at Gunma University.

Keywords: Gaku-Nin Cloud, Moodle, Log Analysis enviroment

1. はじめに

現在 LMS は多くの大学に普及しており，研究室のような小規模のものだけでなく，大学全体で利用するもの，または，国立情報学研究所の提供する学認 LMS のような全国で利用する大規模なものも運用されている [1], [2]．大規模な LMS には LMS 利用者が生成する大量のログデータが蓄積される．例えば，ウエブサービスの提供に必要な LAMP (Linux, Apache, MySQL, PHP) 環境のログや LMS の活動ログなど様々なログがある．また LMS 自身を動かす計算機リソースのログとして，CPU，メモリ，ストレージ利用率などの監視ログも発生する．これらのログは LMS の安定稼動にための障害解析に不可欠であるほか，このログを解析することにより，教育効果を測定し，向上に役立てることができるのではないかと期待されている [3]．

大規模な LMS では大量のログデータが出力されるため，相当の計算資源や保存容量が必要になる．例えば，解析時

[1] 群馬大学総合情報メディアセンター
Library and Information Technology Center, Gunma University, 4-2 Aramaki-machi, Gunma 371–8510, Japan
[2] 国立情報学研究所
National Institute of Informatics
[a)] n.hamamoto@gunma-u.ac.jp

に必要な計算資源や，長期に渡るデータ保存に必要なリソースがあるが，このようなリソースの確保には，クラウド資源を活用することが有用になると考えられる．クラウド資源では，オブジェクトストレージのような安価ストレージや，時間課金のサーバなどを確保することができるからである．

著者らはネットワークログの解析を目的としてクラウドとオンプレミスを利用したログ解析環境を提案した [4]．さらに，この環境を Moodle のログ解析にも適用し，その有効性を示した [5]．一般的なサーバのログ解析では，ネットワークログやサーバログ等のサーバ動作基盤のログを見るだけだが，LMS の場合は LMS 上の活動ログとサーバ動作基盤のログを合わせてみることが有効である．このため，クラウド上に 1 つのログ解析環境を構築し，複数のログを関連付けて解析できるログ解析環境を構築した．さらに，ログのうち一部のフィールドは仮名化（暗号化）されて保存されるようにした．しかしながら，本環境ではログの閲覧の際に，仮名化を自動的に解除することができないという点が課題となっていた．

本研究では，LMS のログ解析環境の改善のため，仮名化自動解除プロキシを実装して機密性に応じた仮名化が適宜行えるようにした．LMS には Moodle を採用し，Moodle および関連するミドルウェアのログと，動作基盤のログを収集，解析する．ロク解析環境は，国立情報学研究所が提供している「学認クラウドオンデマンド構築サービス」を用いて Docker コンテナベースで構築した．本サービスでは Prometheus を用いたリソース監視を行っており，それを動作基盤ログとして活用した．

2. Moodle のログ

本研究では，LMS として Moodle を取り上げて，そのログ解析環境を構築する．Moodle は PHP で書かれた Web サーバ上で動作するソフトウエアである．動作には，Web サーバソフトウエアの他にデータベースソフトウエアを利用している．多くの場合稼働用の OS としては Linux が採用されている．これらをあわせて LAMP（Linux, Apache, MySQL, PHP) 環境と呼ばれることも多く，本研究でもこの環境を利用した．これらの環境からは，各ソフトウエアの動作状況を記録したログが出力される．また，Moodle のみならず，Web サービスの運用時には CPU 利用率，メモリ利用率，ディスク利用率などのリソース監視を行うことも多い．さらに Moodle 自身が Moodle を利用するユーザの活動をログとして記録している．

以上をまとめると，Moodle の運用や学習ログ解析に関連するログは表 1 のようになる．今回はログの種類として，Moodle 本体に標準機能からの追加を行なうことや，クライアント側のログを収集することは考慮しないこととし，標準的なログを検討する．

動作基盤の CPU, メモリ, ディスク, ネットワーク利用率などのリソース監視を行うためには，サーバ本体ではなく外部から定期的にリソースを取得するのが一般的である．このための代表的なソフトウエアは Zabbix, Sensu, Prometheus 等がある．オンプレミスや IaaS 環境などでは，サーバの外部にこれらのサービスを提供する監視サーバを立ち上げて監視を行う．リソース監視で取得される内容には，個人情報を含むことはなく，機密性は高くない．保存形式は通常は監視サーバのデータベースを利用することが多い．

OS やミドルウエアのログに関しては，Web，PHP，認証，データベースなどの各サービスを提供するソフトウエアが，各サーバ上にファイル出力している．PHP，データベースや OS のログには，標準では，機密性の高い情報が含まれることはない．

Web サーバのログには接続元アドレスや Web ブラウザの情報が含まれる．この情報だけで個人を特定できるわけではないが，ユーザ端末の情報があるため，機密性は中とした．また，認証ログには，認証に利用したユーザ ID が含まれ，個人を特定できるが，ログイン・ログアウトの情報のみのため，機密性は中とした．認証には LDAP や Shibboleth 等が使われることが多いが，どちらも各サーバ上のファイルで保存されることが一般的である．

最後に，学習ログだが，Moodle の場合は，Moodle 自身が Moodle を利用するユーザの活動をログとして記録している．これはデータベース上の `logstore_standard_log` テーブルに記録される．ここには，ユーザの学習活動に関わる情報が含まれるが，特に，ユーザ ID やコース ID については個人や授業のコースを特定できる．課題提出等の活動記録も含まれることから，機密性は高とした．

3. Moodle ログ解析環境

上記のように Moodle には様々なログが出力されるが，図 1 の左図のように，従来は様々なログサーバにログが分散されて保存されることが多かった．これらのログは互いに関連していることが多く，例えば，サーバ負荷の原因が Moodle 上での一斉クイズの実施だったなどということがあるが，これは，動作基盤のリソース監視ログと，学習ログを組み合わせて得られる知見である．本論文では，このような複数ログを関連させて解析する環境を構築する．

3.1 ネットワークログ解析環境

ログ解析環境は，Moodle だけでなくサーバやネットワークのログなどでも同様に必要である．障害対応の場合には Zabbix 等の監視ログで十分な場合もあるが，予兆検知やより深い解析のためには複数のログを関連させて原因を追究することが必須となっている．また，セキュリティ対応も考えると，近年はインシデントが長期化していることか

種類	ソース	内容	機密性	標準保存場所	型式
学習ログ	Moodle ソフトウエア	標準イベントログ	高	Moodle サーバ	DB
ミドルウエア	Web, PHP	アクセスログ，エラーログ	中	Moodle サーバ	File
	DB	動作ログ	低	DB サーバ	File
	LDAP,Shibboleth	認証ログ	中	Moodle サーバ	File
OS	OS	動作ログ	低	サーバ	File
動作基盤	オンプレ・IaaS	CPU, メモリ,Disk,NW 使用量	低	監視サーバ	DB

表 1 Moodle に関連するログの種類

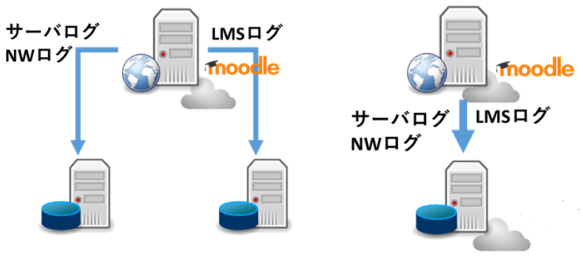

図 1 統合ログ解析環境

ら，長期間のログをため込んで端末特定などの追跡に使うこともある．上記のような背景から，クラウドを利用したログ解析環境として以下の構築要件が提案されている [4]．

要件 (1) 多様なログを集約し，数年，数か月にわたるログの長期保存ができること．

要件 (2) ログの分析によりインシデント検知できること．

要件 (3) 外部 SOC などからの情報を元に，迅速に端末が特定できること．

要件 (4) 多くの機関で利用可能であること．

また，大量の計算資源を要する場合にクラウドを利用する場合には，クラウドストレージなどに保存するデータが管理会社へ漏洩する可能性を極力防ぐ必要がある．このため以下の要件を挙げている．

要件 (a) クラウド上へ機密情報を保存する際には，漏洩対策を施すこと．

3.2 Moodle への適用

この環境は LMS の障害検知等では要件が変わらないことから，我々は Moodle に適用した環境を構築した [5]．この環境の概要を図 2 に示す．Moodle のログや Zabbix のリソース監視ログは，ログ転送サーバでの暗号化を行った後，クラウド上のデータベースに保存される．このデータベースを利用したログの解析には Kibana を利用できるようしている．

クラウド上のストレージを利用することで要件 (1) に示したような長期保存を可能とする．またインシデント，転送サーバを介してログを 1 か所に保存し全文検索が可能な Elaticsearch を利用することで，ログの相関分析が行いやすい環境とし，要件 (2) のインシデント検知を行いやすくしている．一方で，ネットワークログの時と違い，端末特定に必ずしも複数のログを組み合わせる必要がないことから，要件 (3) に関しては，特別な工夫を行う必要はなく満たすことができる．

要件 (4) については，学認クラウドオンデマンド構築サービスを利用して，多くの機関で利用できる実装を実現することを目指している．本サービスでは，Docker コンテナを利用し，パブリック／プライベートクラウドの別によらない統一した手順を提供できることから，実装結果の汎用性が高くなる．また，本サービスでは，Literate Computing for Reproducible Infrastructure (LC4RI) [8], [9] という形で提案されている Jupyter Notebook[10] を利用したサーバ構築・管理が実施できるようしている．これにより，構築管理手順の内容が分かるように共有することで，各機関での個々の事情も反映した手順を作成できるようにしている．本研究でも，学認クラウドオンデマンド構築サービスを前提とした Docker コンテナを利用した実装を行なうことで，多くの機関が利用可能な実装とする．

なお，学認クラウドオンデマンド構築サービスでは，実例としてアプリケーションテンプレートを提供しており，現在でも LMS 構築のテンプレートを利用すると，Moodle を構築することができるようになっている．

以上のように，統合化されたログ解析環境は実現されてはいるが，検索システムの Kibana でログを解析する際に，クラウド上の仮名化されたログの仮名化を解除できないという課題があった．このため，Kibana 上でログを解析した後，必要な情報は別途 Notebook 上の手順を実行して仮名化を解除する必要がある．また，リソース監視に Zabbix を利用していたが，学認クラウドオンデマンド構築サービス（OCS）で標準の Prometheus を活用し，構成を簡略化した．

図 3 は，本研究で考えるログ解析環境である．従来の環境との違いは，検索システム Kibana の前段に仮名再識別 Proxy が配置されていることと，Zabbix の代わりに Prometheus があることである．以下ではこの 2 点の詳細を述べる．

3.3 仮名再識別

仮名再識別 Proxy は，ログ DB 上で仮名化された再識別を行うための Proxy サーバである．再識別の説明の前に，本環境でのログの仮名化について述べる．本環境では，ロ

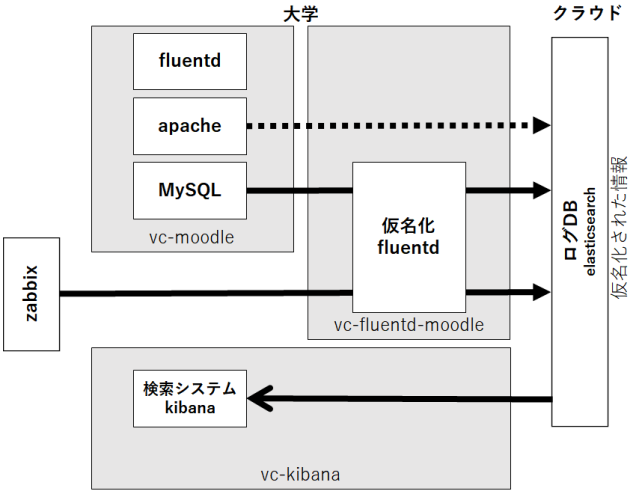

図 2　従来のログ解析環境の概要図

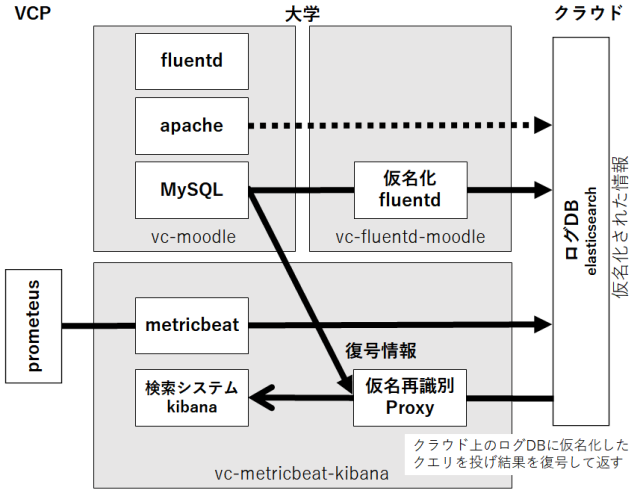

図 3　ログ解析環境（本研究）の概要図

グの仮名化のために Fluentd を利用している．ここでは，MySQL，Apache，Prometheus 等のログをフィールドに分け，クラウド上の Elasticsearch に格納する．Moodle に関連して発生するログは表 1 で示した通りだが，全てのログの機密性が高いわけではなく，学習ログのような一部のログだけが機密性が高い．学習ログの中身は表 2 に示しているが，この中身全ての機密性が高いわけではない．このため，本環境での Fluentd での仮名化・匿名化の実装として，フィールド単位で仮名化・匿名化を行うかどうかを指定できるようにしている．実際には，以下の処理を選択して行えるようにしている [4].

(1) 暗号化
(2) マスキング（特定の文字で置き換える）
(3) ハッシュ化
(4) 消去

これらのうち，暗号化は再識別することが可能で「仮名化」に当たるが，他は一度処理を行うと再識別することはできない「匿名化」にあたる．機密性の程度と，保存先の信頼度に応じて仮名化か匿名化を選ぶことになる．

Kibana でのログの再識別の方法として，Kibana からログ DB である Elasticsearch への検索をかける際に，仮名再識別 Proxy を経由する方法を検討した．Kibana から Elasticsearch への検索は，Elasticsearch で実装されている REST API を利用しているため，HTTP サーバソフトウエアで Web Proxy を実装することで再識別が可能である．つまり，平文のクエリを仮名化して Elasticsearch に送信するとともに，仮名化されたレスポンスを再識別して Kibana 側に返せばよい．

3.4　リソース監視

リソース監視では，学認クラウドオンデマンド構築サービスの Prometheus を利用する．本サービスの制御ソフトウェアが配備されたノードに保存された Prometheus のデータは metricbeat を利用して取得する．図 4 に設定のサンプルを示す．

```
- module: prometheus
  period: 10s
  hosts: ["http://[ip.addr.of.vcc]:9090"]
  metrics_path: '/federate'
  query:
    'match[]': '{instance="[ip.addr.of.instance]:18083"}'
  metrics_filters:
    include: ["container_cpu_*",
    "container_memory_*",
    "http_*","container_network_*"]
    exclude: ["container_memory_failures_total"]
```

図 4　Prometheus 設定例

4.　実装例

ログ解析環境実装に向けての例として，本学 Moodle サーバのログの仮名化解除についての検討と実装例を述べる．

本環境の概略はすでに述べた図 3 と同様となる．今回は，試験実装ということもあり，実際のクラウドを利用せず，群馬大学内のサーバで完結する構成とし，仮名再識別 Proxy 部分の実装を中心に検討を行なった．図 3 上の各サーバは，MySQL, Apache の搭載されている vc-moodle は vmware での仮想サーバとして実装した．Kibana, 仮名再識別 Proxy の搭載される vc-metricbeat-kibana は，それぞれ vmware 上の Docker コンテナとして実装した．コンテナ環境の構築には学認オンデマンド構築サービスの vmware イメージを利用した．また，仮名化に利用する vc-log-mdl 上の Fluentd は，vc-moodle 上に同梱する形とした．

今回仮名化の対象としたログは Moodle の学習ログである．学習ログの各項目は表 2 に示している．この項目の中で，機密性が高いと判断した項目は，ユーザ ID の載る `userid` フィールド，テキストによる課題内容などがみえる

こともある other フィールド，Moodle のコースを特定できる courseid フィールド等がある．今回はサンプル実装として courseid フィールドについて検討する．このフィールドは Fluentd で仮名化しても良いのだが，既に ID という形で仮名化されているため，今回は Fluentd を使わずに仮名化されている ID から実名への変換を行なうことを検討した．ただし，後述する技術的な理由により，ID そのものではなく，ID の前に固定文字列（CIDTRF-) をつける変換を実施した．ID から実名への変換は Moodle の MySQL にある course テーブルに記載されている．ID から実名の変換を Proxy から Moodle 上の SQL サーバに逐次問い合わせする形で実装することとした． なお，ユーザ ID に関しても同様に user テーブルを参照する形で実装できる．

field	値の例
id	
eventname	\mod_assign\event \assessable_submitted
component	mod_assign
action	submitted
target	assessable
objecttable	assign_submission
objectid	746036
crud	u
edulevel	2
contextid	183484
contextlevel	70
contextinstanceid	101254
userid	17209
courseid	2307
relateduserid	-
anonymous	false
other	a:1{s:19: submission_editable;b:1;}
timecreated	1613019209
origin	web
ip	11.22.33.44
realuserid	-

表 2 logstore_standard_log テーブルのフィールド名と値の例．なお一部の値は現実性を損なわない範囲で架空の値としている．

Kibana から Proxy 経由でのクエリの流れを図 5 に，Proxy から Kibana へのレスポンスの流れを 6 に示す．Kibana では，通常は Elasticsearch をデータベースとして指定するが，ここでは仮名解除 Proxy を指定しておく．クエリは以下の流れとなる．

(1) Kibana から JSON 型式でにクエリを POST する．
(2) Proxy で"courseid": "授業名"の部分を抜き出す．
(3) MySQL に問い合わせて，course テーブルから ID を算出する．
(4) クエリを変換し"courseid": "CIDTRF-コース ID"と変換する．
(5) ログ DB に変換されたクエリをなげる．

レスポンスは以下の流れとなる．

(1) ログ DB からレスポンスを proxy が受け取る．
(2) Proxy は"CIDTRF-コース ID"に対応する部分を抜き取り，コース ID を特定する．
(3) MySQL に問い合わせて，course テーブルからコース名を算出する．
(4) Kibana に対して変換されたレスポンスを返す．

以上の変換に関わる実装は Web サーバソフトウエアの Apache の ext_filter モジュールを利用して実装した [12]．本モジュールでは，標準入力から取得したリクエストボディを加工して標準出力に出力すれば良く，実装にはどの様なプログラミング言語を利用しても構わない．本実装では，速度を犠牲にはするが，簡便な Perl を利用して上記の変換を実装した．

今回 Fluentd では ID の前に固定文字列を追加している．この理由は，レスポンスの (2) の段階で，コース ID に該当するものを特定するためである．問い合わせの際には，courseid フィールドに対する問い合わせ部分をみればよいのだが，レスポンスは必ずしも courseid フィールドだけに含まれるわけではないため，コース ID を表す数字であることを明示的にわかる形にしておく必要があるためである．（例えばグラフの凡例項目などは courseid ではなく key という形で返される）

なお，MySQL への問い合わせを毎回行うと遅くなるため，Redis を利用したキャッシュサーバもコンテナの形で実装した．体感的にはこの実装により，ストレスなく検索できるようになった．

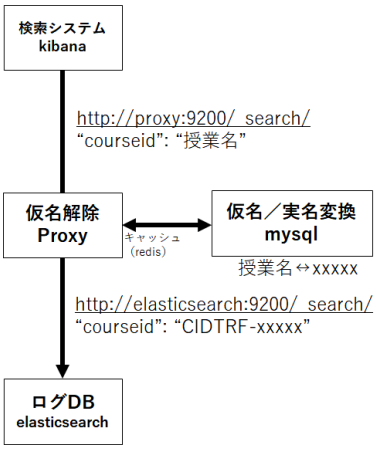

図 5 仮名再識別 (クエリ)

以上の実装を行った検索システム（Kibana) のスクリーンショットを図 7 に示す．比較として仮名化解除 Proxy を経由しない場合の検索システムの同検索条件のスクリーンショットを図 8 に示す．これら 2 つの図は，配色が違うが，検索条件は同一なので，各項目の値などは同一である．例えば，左上の 1774, 579 という数字は，指定した期間内で

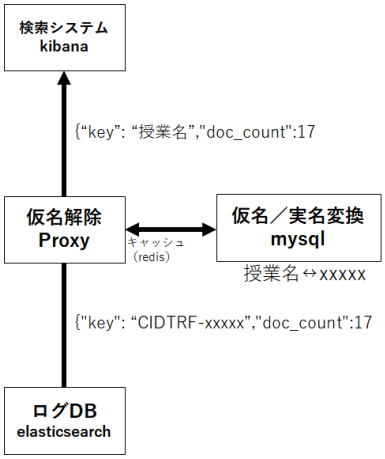

図 6　仮名再識別（レスポンス）

活動のあったユニークユーザ数，ユニークコース数をそれぞれ示しているが，同一のデータを利用した解析結果なので，2 つの図で同一の値である．一方で，右上のグラフと左下の表にあるコース名は図 7 では，コース名の実名が出ている．（ただし，本論文では公開制限のため一部マスクした）一方で，図 8 では，コース ID の数字が見える形となっている．

今回は Elasticsearch という全文検索可能なデータベースを利用している．しかしながら，今回は仮名再識別プロキシでは `courseid` 項目の変換は行うが，全文検索時には変換を行なわない．このため，全文検索時にはコース ID 項目は対象にならないという点が問題として残る．この解決は今後の課題となる．

表 3 に，Proxy や Redis キャッシュの有無によるサンプルビューの表示時間を接続方法の各パタンについて 3 回計測した結果を示す．サンプルビューは 2 月 12 日の上位アクセス 100 コースを示した表である．接続方法の「Direct」は Proxy を介さず，仮名化も解除しない状態で表示した場合，「Proxy」は Proxy を経由して表示するが Redis キャッシュのない場合，「Proxy-Redis」は Redis キャッシュ付きの Proxy を経由する場合である．本測定から，Proxy を経由することにより表示時間は遅くなるものの，Redis キャッシュによって，高速化されていることが分かる．

接続方法	1st	2nd	3rd	Average
Direct	7.22	8.61	5.17	7.00
Proxy	17.34	14.12	18.57	16.68
Proxy-Redis	14.42	13.03	13.72	13.72

表 3　サンプルビューの表示時間 [s]

5. まとめと今後の課題

本論文では，クラウド上のログ解析環境を提案し，Moodle を利用したサンプル実装を行った．本環境では，Moodle 運用に関わる様々なログを一元管理できるとともに，ク

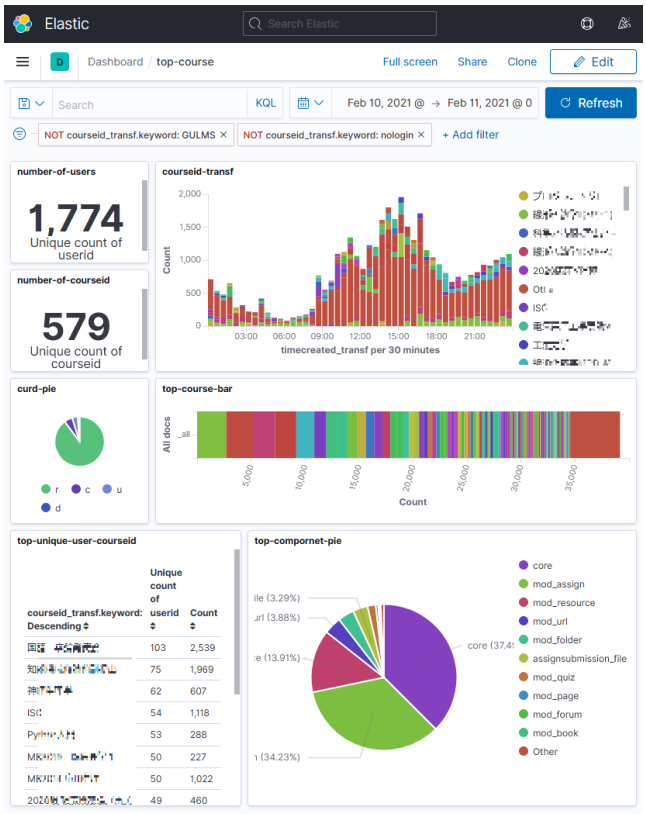

図 7　仮名解除プロキシを用いた場合の Kibana のスクリーンショット

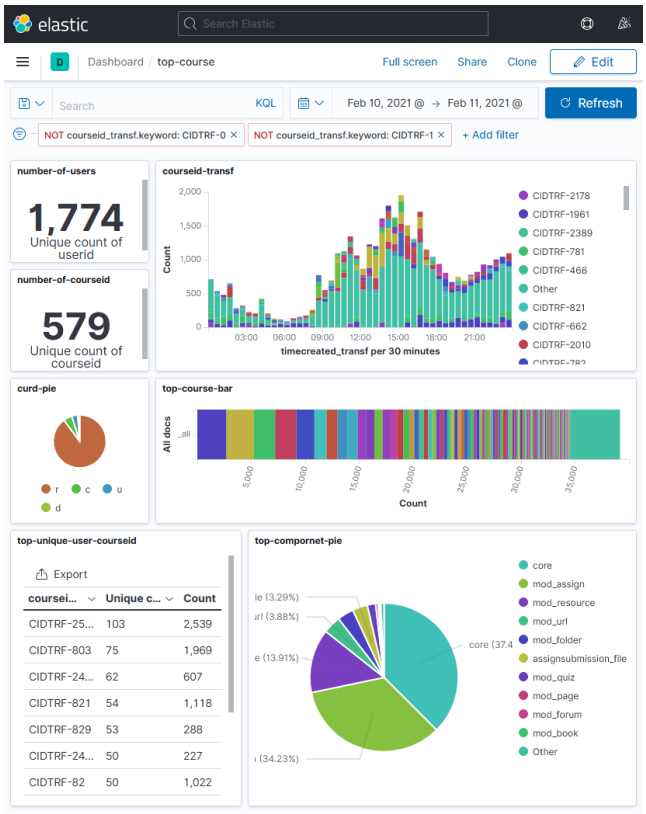

図 8　仮名解除プロキシを用いなかった場合の Kibana のスクリーンショット

ラウド上のログは仮名化，匿名化した形で保存できる．また，仮名化されて保存されたログに関しては，仮名化解除

Proxy を経由することで，ログ閲覧，解析時には仮名化解除した形で行なえることができるようにした．また，実装は学認クラウドオンデマンド構築サービスで利用できることを想定して Docker コンテナを利用した．

今回の検討とサンプル実装で，基本的な仮名化や仮名化解除の機能は実装できることが分かった．また，ログの統合や Prometheus を利用したリソース監視に関しても，技術的には過去の論文や今回検討した設定により実装可能である．今後は，これらをまとめて Jupyter Notebook などを利用してテンプレート化し，学認クラウドオンデマンド構築サービスで利用できる Moodle のログ解析環境として提供する方向での検討を進める予定である．

謝辞 本研究は 2020 年度国立情報学研究所公募型共同研究（20FA03）及び JSPS 科研費 (JP20K12088) の助成を受けたものです．本研究に関してご助言いただいた数理技研の小泉敦延様に感謝申し上げます．

参考文献

[1] 大学 ICT 推進協議会，”2017 年度　高等教育機関における ICT の利活用に関する調査研究報告”，https://axies.jp/report/ict_survey/2017result/，(参照 2021-2-16).
[2] 国立情報学研究所，“学認 LMS”, https://lms.nii.ac.jp/, (参照 2021-2-16).
[3] 古川雅子，上田浩，浜元信州，中村素典，山地一禎 : 学認 LMS における標準規格に基づく教材配信及び学習履歴取得システム，情報処理学会研究報告, Vol. 2019-IOT-47，No. 13，pp1-4, 2019
[4] 浜元信州，横山重俊，竹房あつ子，合田憲人 : 端末特定のためのログ解析クラウド環境の構築，情報処理学会研究報告，Vol. 2019-IOT-44, No. 29, pp1–8 2019.
[5] 浜元信州，横山重俊，竹房あつ子，合田憲人，桑田喜隆，石坂徹 : クラウドを利用したログ解析環境の Moodle への適用，MoodleMoot Japan 2019 Proceedings pp24-31, 2019.
[6] 竹房あつ子，横山重俊，政谷好伸，丹生智也，佐賀一繁，長久勝，合田憲人 : SINET を活用したインタークラウド環境構築システムの開発，電子情報通信学会技術研究報告，CPSY2017-17, No. 153 pp7-12, 2017.
[7] Takefusa, A., Yokoyama, S., Masatani, Y., Tanjo, T., Saga, K., Nagaku, M., Aida, K.(2017) Virtual Cloud Service System for Building Effective Inter-Cloud Applications，Proc. IEEE CloudCom 2017，pp296 -303, 2017.
[8] 政谷好伸・谷沢智史・横山重俊・吉岡信和・合田憲人 : インフラ・コード化の実践における IPython notebook の適用，信学技報, 115(72), SC2015-6，27 - 32，（2015）.
[9] 長久勝, 政谷好伸, 谷沢智史, 中川晋吾, 合田憲人 : Literate Computing for Reproducible Infrastructure による研究・教育環境の構築と運用，2017 年度大学 ICT 推進協議会，（2017）.
[10] Project Jupyter, https://jupyter.org, (参照 2021-2-16).
[11] Fluentd Project，https://www.fluentd.org，(参照 2021-2-16).
[12] Apache モジュール mod_ext_filter, https://httpd.apache.org/docs/2.4/ja/mod/mod_ext_filter.html, (参照 2021-2-13).