Vol.2015-DPS-162 No.32
Vol.2015-CSEC-68 No.32
2015/3/5

IaaS パブリッククラウドにおけるコンプライアンス自動化のためのイベント監視と分析

渡邊裕治[†1]　　水谷正慶[†1]　浦本直彦[†1]

IaaS パブリッククラウドでシステムを運用することにより、様々な管理・運用プロセスを自動化し、システムの所有コスト、管理コストを削減しようとする流れが主流となってきている。一方で、物理環境と仮想環境との違い、マルチテナント環境などが障害となり、PCI や HIPAA 等のコンプライアンス対応の自動化には多くの課題が存在する。本論文では、コンプライアンス自動化のために必要なログ管理要件のうち、IaaS パブリッククラウド上のシステムに特徴的な要件に焦点を当て、それら要件を満たすためのアプローチを提案する。特に、大規模な IaaS パブリッククラウド上でユーザシステムのログ管理、それに基づくセキュリティ監視・監査を効率化するためのクラウド管理サービスとログ分析アプローチを提案する。

Event Monitoring and Analytics for Compliance Automation on IaaS Public Cloud

YUJI WATANABE[†1]　MASAYOSHI MIZUTANI[†1]
NAOHIKO URAMOTO[†1]

Hosting enterprise systems on IaaS (Infrastructure-as-a-Service) cloud gains popularity for reducing the cost of asset and operation by automating IT management process. Automating compliance operation on a cloud environment, however, has a lot of design issues requires cloud-specific consideration such as virtualization and multi-tenancy. In this paper, we address the requirements and issues for compliance automation specifically on IaaS public cloud, and propose cloud management service which provides automation and operational efficiency in security audit and monitoring for the IaaS users.

1. はじめに

IaaS パブリッククラウドでシステムを運用することにより、様々な管理・運用プロセスを自動化し、システムの所有コスト、管理コストを削減しようとする流れが主流となってきている。また、自動化に伴い DevOps や Auto-Scaling に代表されるような従来環境では困難だったシステムの柔軟な運用を可能にする。

これまで、Off-premise,On-premise をとわず、システム管理・運用を効率化するための様々な工夫が行われてきた。特に、パッチ管理、セキュリティ診断、ログ管理など、コンプライアンスに関する要件は、それを満足するために大きな運用コストが発生することが多いことから、運用自動化の効果が大きい領域である。

代表的な自動化項目としては、

- システムの運用状況をモニタリング : システムが事前に決められた状態で稼動しているか常時監視し、何らかの異常が検出された際にはそれに基づき必要な改善処置を立ち上げる。
- システムの状態をチェック : システムが健全な状態を維持するために、セキュリティ・ヘルスチェックを定期的に行い、必要なパッチを適用する。
- システムの利用が適正であるか : ID 管理やアクセス制御を行い、そのログを管理・保全する。

などがある。

本論文では、特にログの管理・保全に関する自動化に焦点を絞って、IaaS パブリッククラウド上で自動化を設計する際に考慮すべき課題を整理しする。

2. クラウド環境におけるログ管理の特徴

本章ではクラウド環境に特徴的なログ管理の課題を議論する。コンプライアンスからの要件から、不正な改ざんを防ぐためにログをリモートサーバに回収する必要がある。

(1) 物理環境と仮想環境の違い

仮想化されている。従って、VM 自体の管理と物理サーバの管理に加え、Hypervisor 以下の層の管理が必要となる。システム管理要件によっては、システムの運用全体にわたった要件がかされる場合がある。代表的なものとしてログ要件がある。VM に対する運用履歴が求められる場合、VM 自体から取得されるログとは別に、Hypervisor を含めた VM を運用するクラウド基盤のログが必要となる。例えば、図 1 中の VM（太枠）がログ要件は、それを管理する Hypervisor

†1 日本アイ・ビー・エム株式会社　東京基礎研究所
IBM Research – Tokyo, IBM Japan Ltd.

Vol.2015-DPS-162 No.32
Vol.2015-CSEC-68 No.32
2015/3/5

にも求められる場合がある。

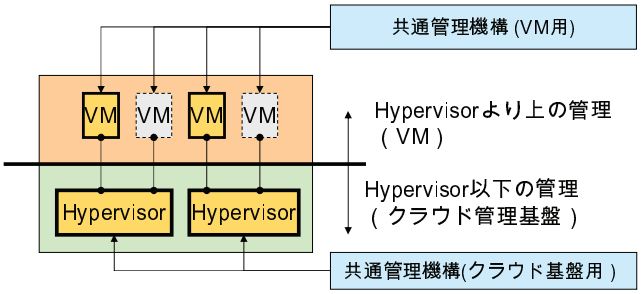


図 1　仮想環境に求められる管理要件の範囲

(2) マルチテナント環境の Isolation

Public クラウドにおいては、複数の Tenant の VM が共通のクラウド基板上で運用される。各 Tenant の VM から取得されるログは、それぞれ Tenant 毎に隔離される必要がある。一方で、クラウド上の運用自動化の仕組みを Tenant 毎ではなく共通に利用可能にすることが運用コストの削減のために求められる。

(3) 大規模分散環境

クラウドの基盤は地理的にも世界中に分散して配置される。クラウド基盤が巨大になるにつれ、発生したログを滞りなく回収するために効率的なログサーバの配置が必要となる。中央に配置されたサーバに一元的にログを回収する場合、ログ転送時の欠落を防ぐために、クラウド基盤の拠点間の必要なるネットワーク負荷と、ログサーバの受信スループットが課題となる。ログの回収を各拠点で個別に行う場合、受信処理の負荷も分散されるが、ログを利用する立場で考えると、分散配置されたログに効率的にアクセスする手段が必要となる。

(4) リアルタイム性

VM でイベントが発生したらそのログをすぐにログサーバに回収することが望ましい。その理由は、

- できるだけ回収したログに対して分析を行い、不正・異常な状態を検出し、必要なアクションを取る。
- ログを VM 上に置くことで VM 上の特権ユーザがログに対して不正な改ざんを行うことを防ぐ

がある。回収されたログは、ログに含まれるメッセージを解析し、レポジトリに保管され、その後の検索や分析に必要なインデックスが付与され、即時に分析処理が行われる。大量に発生するイベントを滞りなく回収・処理するためには、時間当たりに発生するイベントの総量に応じて、ログサーバに十分なデータ処理能力が求められる。

3. クラウド上のログ管理サービス

クラウド上の VM を利用するユーザが VM に関するログの取り扱いを自動化するために利用するのが本節で提案するログ管理サービスである。

3.1 ログ管理サービスの機能

ログ管理サービスの機能を図 2 に示す。

(1) サービス管理

クラウド上に多数の VM を保持する Tenant は、以下のサー

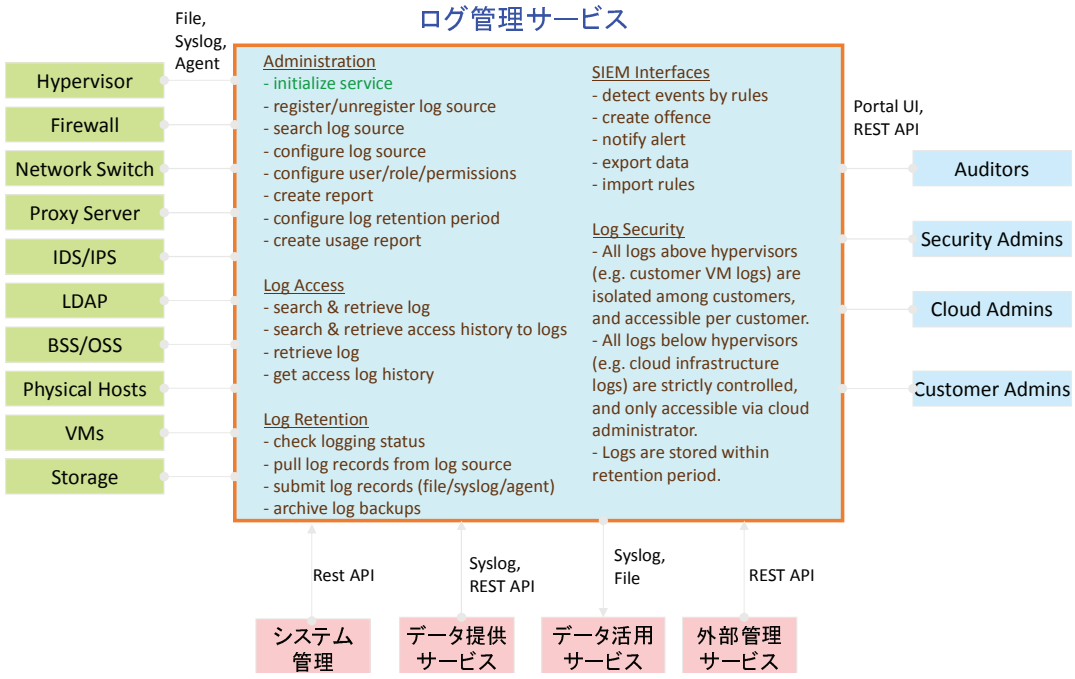


図 2 ログ管理サービス

拠点間ネットワークの負荷は低減され、ログサーバのログ

Vol.2015-DPS-162 No.32
Vol.2015-CSEC-68 No.32
2015/3/5

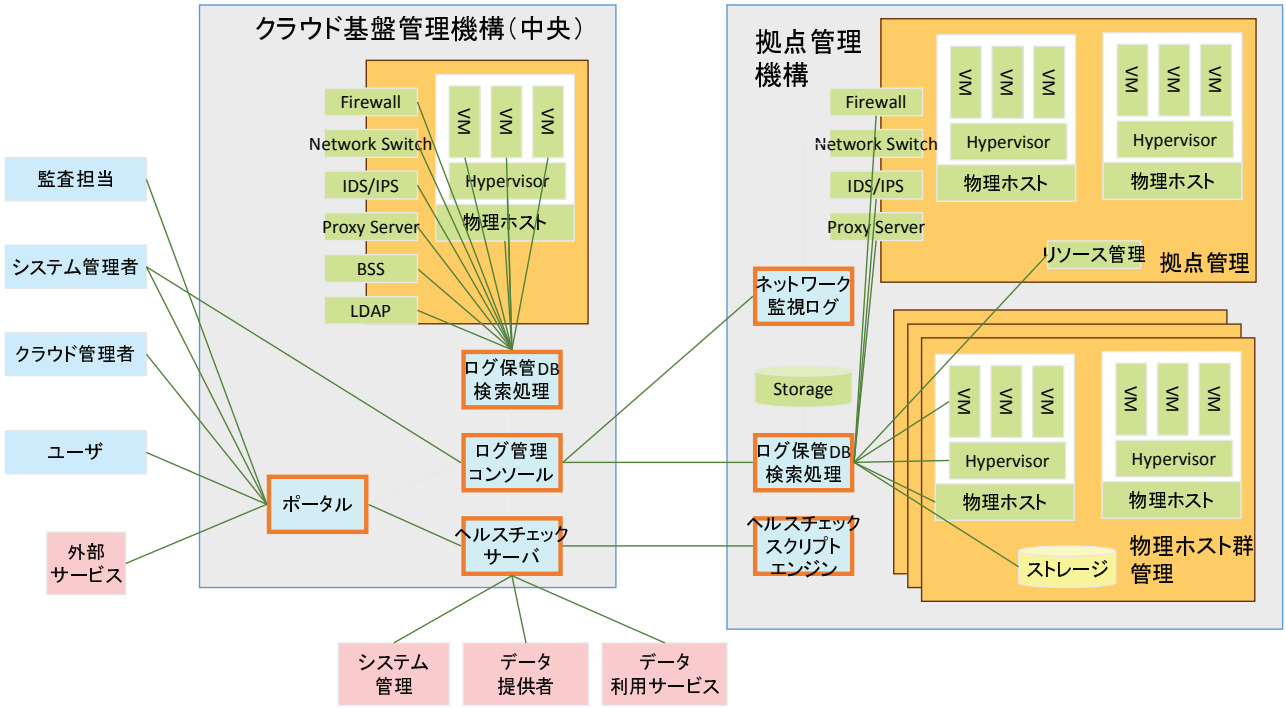


図 3　クラウド管理サービスのクラウド内の展開

ビスの初期化を行う。

- ログを収集する元となる VM をリソース管理基盤と連携して探索し、ログ管理対象として登録する。ログ収集元に対してはログの送信のための初期設定（例：Syslog の送信レベル・送信先ホストの設定）を行う。OS 毎、バージョン毎に異なる取得対象のセキュリティログや特権ユーザ監視のためのログを自動的に検出し設定する。
- VM プロビジョニング機構と連携し、VM の初期設定時にログ管理初期設定も自動的に適用する。
- ログにアクセスできるユーザ・ロール・権限を設定する。
- ログの収集を開始し、収集したログからサマリーデータを含むレポートの自動生成を開始する。
- ログ保全期間を設定する。
- ログを誰がいつ利用したか利用レポートを生成する。

(2) **ログへのアクセス**

ユーザの収集したログに対するアクセスを可能にする。

- 回収したログを構文解析してログからパラメータを抽出し、データベースに格納し、インデックスを構築する。
- ユーザからのログの検索・抽出・ダウンロードに対する API を提供する。

(3) **ログの保全**

ログを正しく回収し一定期間保管する。

- ログ収集元からログが正しく到着しているか検査する。
- 到着したログをデータベースに保持し一定期間検索可能にする。
- データベース中のログのバックアップを一定期間保持する。

(4) **セキュリティイベント分析**

収集したログに基づいてコンプライアンスの自動化に必要な処理を行う。

- ルールに基づいてログを分析し、不正なアクセスや異常な振る舞いパターンを検出し、事象毎に事前に設定した重み付けを行い必要に応じて管理者に警告を通知する。
- ルールに基づいて外部サービスに対して連携の必要なるログの発生を検出し、そのデータを転送する。
- 上記ルールの設計と開発を支援する。

(5) **ログへのアクセス制御**

回収したログをテナント毎に（論理的または物理的に）データベースに格納する。リソース管理機構と連携して、ログ取得元の VM がどの Tenant のものかを判定して、格納先を決める。クラウド基盤ログに対するアクセスはクラウド管理者のみがアクセスできる。

3.2 ログ管理サービスのクラウド内展開

クラウド基盤に対して上記ログ管理サービスの展開する方法を図 3 に示す。クラウド基盤全体を管理する中央のクラウド基盤管理機構と、拠点毎に配置される拠点管理機構が存在する。

- ユーザに対する一元的なアクセスを提供するため、ポータル・ヘルスチェックサーバ・ログ管理コンソールはクラウド基盤管理機構に配置する。

Vol.2015-DPS-162 No.32
Vol.2015-CSEC-68 No.32
2015/3/5

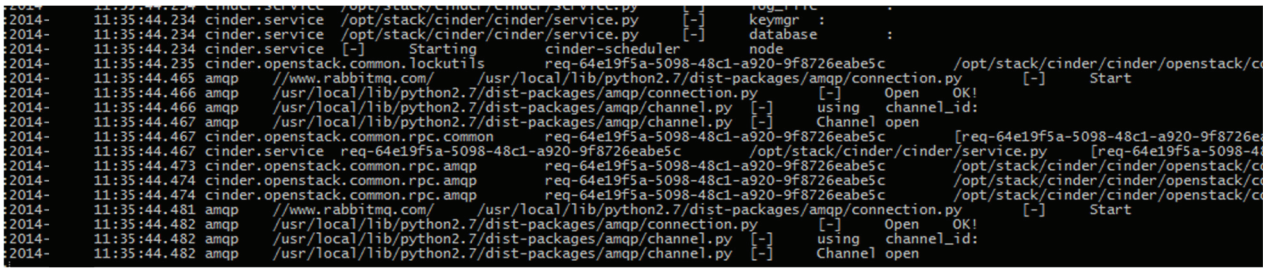


図 4　クラウド基盤のログ

- ログデータの広域転送を最小限にするため、ログを回収・保管する DB・ネットワーク監視機能は各拠点管理機構に配置する。

ログの検索はログ取得対象のホストに応じた拠点に対して要求を行う。複数拠点間をまたがる相関分析を行う際には、データ検索・集約を可能な限り拠点レベルで行い、それを中央に伝送して最終の集約処理を行う。

4. クラウドログの基盤ログの分析

クラウド基盤か取得したログに基づいてシステムの異常を検出するためのアプローチを示す。この実験はクラウド基盤として Open Stack を想定し、基盤コンポーネントが生成したログを回収してサンプルデータとして分析を行った。

(1)　（事前準備 1）正常系ログ系列の記録

まずは正常時に吐き出されるログを収集する。この例では、OpenStack の環境を構成し、その上で新しい VM の生成を複数回行い、その際に生じるログをログサーバに集約した（**Error! Reference source not found.**）。

(2)　（事前準備 2）正常系ログ系列のパターンを抽出

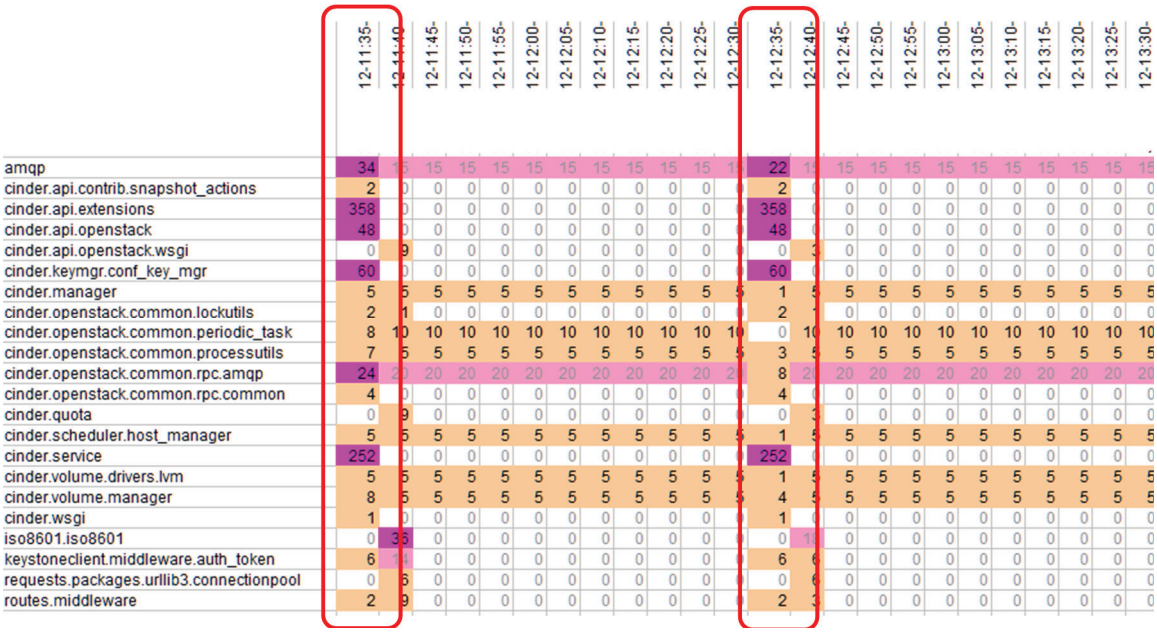
正常系のログ系列を縦軸をログを出力したコンポーネント名、横軸を時間窓で可視化する（図 5）。すると、大別してあるコンポーネントから定常的に吐き出されているログ

- 何らかのイベントの発生に伴い多数のコンポーネントから一斉に吐き出されるログ

が存在することがわかる。

ここでさらに時間解像度をあげると、ほぼ同じログ発生パターンが見られる（図 6）。また、各パターンの詳細（**Error! Reference source not found.**）を見ると、それらは異なる VM 生成によって発生したログ系列であることがわかる。実際にそれらログ系列を比較すると殆ど差がないことがわかる。従ってこれは「VM 生成イベント」を特徴付けるログ生成パターンとして登録する。

図 5　ログの出力元とログ発生時間との相関



	12-11:35-	12-11:40-	12-11:45-	12-11:50-	12-11:55-	12-12:00-	12-12:05-	12-12:10-	12-12:15-	12-12:20-	12-12:25-	12-12:30-	12-12:35-	12-12:40-	12-12:45-	12-12:50-	12-12:55-	12-13:00-	12-13:05-	12-13:10-	12-13:15-	12-13:20-	12-13:25-	12-13:30-
amqp	34	15	15	15	15	15	15	15	15	15	15	15	22	15	15	15	15	15	15	15	15	15	15	15
cinder.api.contrib.snapshot_actions	2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
cinder.api.extensions	358	0	0	0	0	0	0	0	0	0	0	0	358	0	0	0	0	0	0	0	0	0	0	0
cinder.api.openstack	48	0	0	0	0	0	0	0	0	0	0	0	48	0	0	0	0	0	0	0	0	0	0	0
cinder.api.openstack.wsgi	0	9	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0
cinder.keymgr.conf_key_mgr	60	0	0	0	0	0	0	0	0	0	0	0	60	0	0	0	0	0	0	0	0	0	0	0
cinder.manager	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5
cinder.openstack.common.lockutils	2	1	0	0	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0
cinder.openstack.common.periodic_task	8	10	10	10	10	10	10	10	10	10	10	10	0	10	10	10	10	10	10	10	10	10	10	10
cinder.openstack.common.processutils	7	5	5	5	5	5	5	5	5	5	5	5	3	5	5	5	5	5	5	5	5	5	5	5
cinder.openstack.common.rpc.amqp	24	20	20	20	20	20	20	20	20	20	20	20	8	20	20	20	20	20	20	20	20	20	20	20
cinder.openstack.common.rpc.common	4	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0
cinder.quota	0	9	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0
cinder.scheduler.host_manager	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5
cinder.service	252	0	0	0	0	0	0	0	0	0	0	0	252	0	0	0	0	0	0	0	0	0	0	0
cinder.volume.drivers.lvm	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5
cinder.volume.manager	8	5	5	5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	5
cinder.wsgi	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
iso8601.iso8601	0	36	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0
keystoneclient.middleware.auth_token	6	4	0	0	0	0	0	0	0	0	0	0	6	6	0	0	0	0	0	0	0	0	0	0
requests.packages.urllib3.connectionpool	0	6	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0
routes.middleware	2	9	0	0	0	0	0	0	0	0	0	0	2	3	0	0	0	0	0	0	0	0	0	0

図 6　同一パターンの系列の出現

(3)　**（ログ計測）クラウド基盤のログを計測し各パターンの類似マッチングによりログを分類**

実際のクラウドの運用でクラウド基盤からログを収集する。実際の運用では、VM 生成単体の事象だけでなく、他にも様々な事象が同時並行的に生じうる。そのため、前述のパターンがそのまま出現することはまれである。そのため、ログ系列に対して、上記のパターンが一定期間内に時系列的に同順序で出現する箇所を類似系列のマッチングを行うことによって特定する。

(4)　**異常ログの検査**

実際に計測されたログから以下の二つを削除する

・パターンにマッチして分類された部分ログ系列

・定常的に各コンポーネントから出力される部分ログ系列

これにより、パターンに合致しない非定常なログ系列のみが抽出され、そこからさらなる問題分析を行う。

このアプローチの利点は、

- 多数のコンポーネントから構成されるクラウド基盤においてどうログが発生するのかを事前に把握する必要がない。
- 各コンポーネントから出力されるログのフォーマットについて最小限の事前知識のみで分析が行える。用いている情報はログの出力元とその時刻だけである。
- 大量のログからパターンに合致する部分ログ系列を自動分類するとともに、パターンに合致しない異常系列を抽出することができるため、ログに基づく問題判別・監査を効率化できる。
- 実際には、各時刻にどんな事象が起こったかはクラウド基盤で把握できるため、それと相関することで精度を高めることが可能である。
- ログフォーマットの構造分析の自動化 1)と組み合わせることによりさらなる精度向上を見込める。

という点にある。また、2)の技術を利用することにより、クラウド基盤のログを秘匿したままユーザに特定パターンが発生したことを通知するサービスが実現できる。

5.　まとめと今後の課題

本論文では、マルチテナントの IaaS パブリッククラウドにおけるコンプライアンス自動化、特にログ管理の自動化の課題とアプローチを提案した。またクラウド基盤から収集したログ分析による問題判別・監査の効率化のアプローチを提案した。クラウド上のシステム管理自動化サービスとして、ログの収集と分析をユーザに利用可能にすることによって、非クラウド環境ではコスト的にも機能的にも実現することのできなかったログの利用環境が整いつつある。その利用を可能にするためにも、クラウド環境において特徴的な課題を現実的に解決していくことが重要となる。

クラウド基盤ログの分析は、分析の道筋は見えてきたものの、その異常検出・問題判別のためのステップの多くはユーザに依存する。この分析ステップの自動化することは今後の課題である。

参考文献

1) M. Mizutani, “Incremental Mining of System Log Format”, IEEE International Conference on Services Computing(SCC) 2013. Santa Clara, June 28-July 2, 2013

2) 渡邊裕治、水谷正慶、クラウドのログ管理における秘匿型イベント検出, 暗号と情報セキュリティシンポジウム(SCIS) 2013.

Vol.2015-DPS-162 No.32
Vol.2015-CSEC-68 No.32
2015/3/5

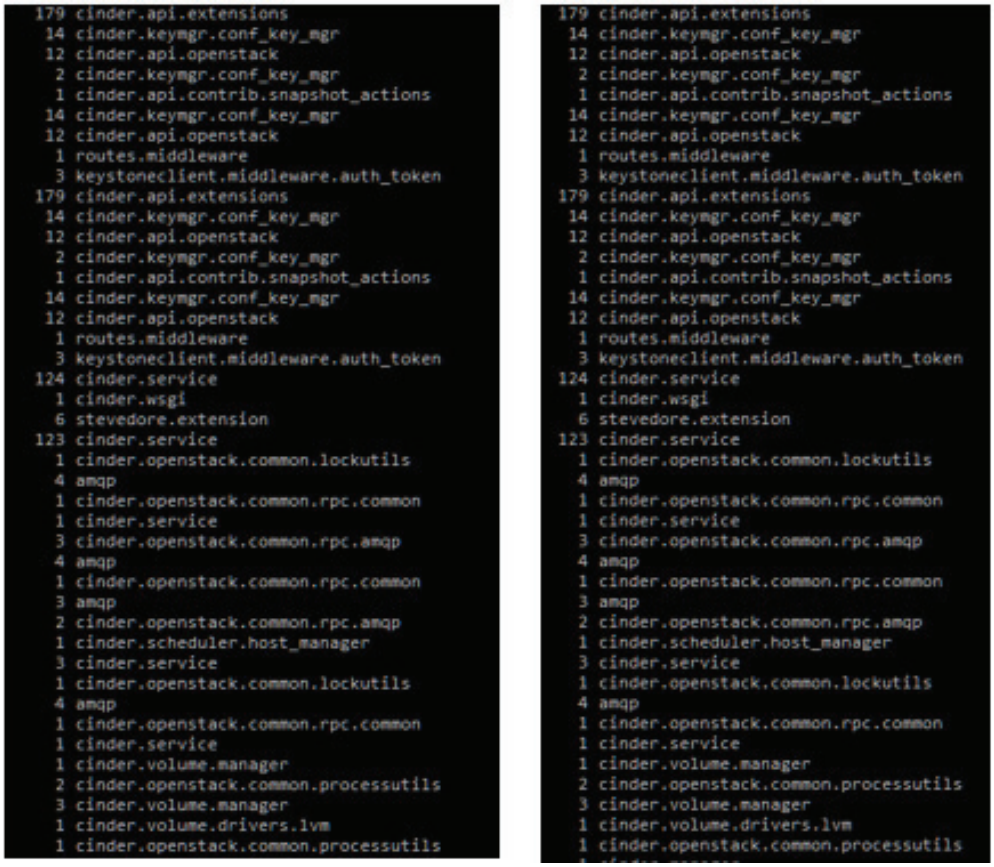


図 7　異なる VM を生成した際のログ系列の比較