Vol.2018-IOT-40 No.1
2018/3/5

大規模システムを対象としたセキュアな監視環境の自動構築及び柔軟な解析環境システムの試作

出羽 裕一[1,a)]　桝田 秀夫[2,b)]　森 真幸[2,c)]　永井 孝幸[2,d)]

概要：大規模システムの管理者の負担を軽減するため，統合監視ソフトウェアによる監視環境の自動構築と収集した監視情報に対して統計解析手法を適用できるシステムの試作を行った． 監視情報の収集では，情報の改ざんやなりすましを防止することも考慮し，電子証明書による暗号化を施した． また，大規模なシステムでの利用を想定し，登録する ICT 機器の数に依る性能評価を行った．

Automated Setup of Secure Monitoring System and Flexible Statistical Analysis Environment for Many Agents

Yuichi Deba[1,a)]　Hideo Masuda[2,b)]　Masayuki Mori[2,c)]　Takayuki Nagai[2,d)]

Abstract: In order to reduce the burden on administrators of large-scale systems, we have built a monitoring environment automatically using integrated analysis software and a prototype system that can apply statistical analysis methods to the collected monitoring information. In collecting the monitoring information, encryption with a certificate was given considering to prevent information falsification and spoofing. We also evaluated the performance based on the number of agents to be registered, assuming use in a large-scale system.

1. はじめに

近年，仮想化技術の発展や ICT (Information and Communication Technology) 機器の低コスト化により，IoT(Internet of Things) やクラウドサービスに代表されるような，多数の機器が相互に通信を行って提供されるサービスが一般的になってきた．これにより，サービスの内容を柔軟に変更したり，故障などに備えた冗長性が確保されたシステムの構築が容易になった．

しかし，このようなシステムは，膨大な数の ICT 機器などの稼働状況をリアルタイムで把握する必要があるため，管理及び運用のコストが増加していると考えられる．セキュリティ企業 ForeScout 社が 2017 年に行った調査では，従業員 2500 人以上の組織における IoT デバイスセキュリティの実態について，IT 及び事業部門の責任者 603 人の 82%がネットワークに接続されているデバイスすべてを把握していないことが報告された [1]．

このような複雑なシステムを管理及び運用する負担を軽減するため，企業者や教育研究機関では統合監視ソフトウェアが用いられてる．統合監視ソフトウェアは，対象となるシステムの監視情報を監視サーバにリアルタイムに集約し，管理しやすい形式で出力することが可能となるツールである．また，統合監視ソフトウェアなどによりシステムから収集した ICT 機器の稼働状況のデータを元に，将来起こりうる障害やリソース消費を予測するアルゴリズムの開発が様々な研究機関で行われている．

東北大学工学研究科の立見ら [2] は，対象となるサーバのプロセス情報から将来のリソース消費予測を立て，適切なハードウェア資源の割り振りに役立てようとした．シス

[1] 京都工芸繊維大学 工芸科学研究科 情報工学専攻
Graduate School of Information Science, Kyoto Institute of Technology
[2] 京都工芸繊維大学 情報科学センター
Center for Information Science, Kyoto Institute of Technology
[a)] y-deba16@dsm.cis.kit.ac.jp
[b)] h-masuda@kit.ac.jp
[c)] morim@kit.ac.jp
[d)] nagai@kit.ac.jp

テム障害の解析では，北陸先端科学技術大学院大学の坂下ら [3] が，システム構成を把握していなくても，隠れマルコフモデルとベイズ推定を用いることで，機器が出力したログファイルを解析し，障害の原因を推測するアルゴリズムを提唱している．

しかし，統合監視ソフトウェアが監視情報を取得するためには，各 ICT 機器に対して個別に設定や監視用ソフトウェアのインストールが必要になる場合がある．このことから，複雑なシステム全体を監視下に置くだけで管理者の負担が大きくなり，収集したデータを用いた解析まで手が回らない場合がある．もちろん，解析手法を実際のシステムで動作させる際には，提案されたアルゴリズムを実装し，対象となるシステムに適応させる必要がある．これには高度なプログラミング知識が要求され，用いるデータも大量となるため，解析結果の検討も難しいのが実状である．

さらには，組織内の規定された情報セキュリティに関する要求事項を満たすためには，不要な ICT 機器の登録の除外や，通信途中の監視情報の改ざんの防止といった対策が必要であり，サブネット単位のような画一的なルールに基づいた統合監視ソフトウェアの運用自体が困難な場合がある．

そこで本研究では，複雑化するシステムの管理及び運用の負担を減らすことを目的として，セキュリティを考慮した，統合監視ソフトウェアによる監視環境の自動構築機能と，Web アプリケーションによる統計解析手法の試験及び開発環境の試作を行う．自動的に ICT 機器を監視対象とすることで，登録作業の容易化や登録漏れの軽減により管理者の負担を軽減することが可能になる．また，統計解析手法の試験時の負担を軽減するため，解析結果をグラフィカルに出力できる機能も実装する．

2. 要求

技術の進歩によりネットワークの複雑化と大規模化が進むと，システム管理者が対象となるシステムを監視する際に重要な情報を得るのが難しくなる．そのため，統合監視ソフトウェアにより情報を取得し，データベースに自動で蓄積させるなど，管理者の負担を抑えたシステムが求められる．以下にシステムに求められる要求として，監視環境の自動構築と統計解析の実行支援について述べる．

2.1 環境の自動構築

大規模なコンピュータシステムのすべての機器を監視下におくための環境を 1 台 1 台構築するのは困難である．そのため，すべての監視対象に自動で環境を設定する仕組みが必要になる．ただ，監視対象が増えるということはすべてが盗聴や改ざんなどされずに通信できているか，不要な対象の登録を拒否できているかが重要になる．このため，登録する必要がある監視対象を区別し，やりとりする情報を暗号化することで，セキュリティを確保する必要がある．

2.2 解析の実行支援

管理者が望む統計解析を実行したい場合，大規模なプログラムを作成する必要があり，負担が大きくなる．そのため，その解析が容易にプログラムで記述できるなど，負担を軽減することが可能なシステムが求められる．

その一方で，複雑な統計解析はパラメータの調整や，プログラムの記述などの必要があり実施が難しくなる．そこで，管理者が統計解析を適用したいとき，複雑な操作をすることなく GUI (Graphical User Interface) による操作及び，結果の表示も可能なシステムが求められる．

3. システムの方針

本章では，第 2 章の各節で述べたシステムの要求を満たすための方針について述べる．

3.1 環境の自動構築

監視対象とするサーバから，自動で監視情報を取得するために統合監視ソフトウェアの Zabbix[4] を利用する．Zabbix は，大規模システムの管理及び運用のため開発されているオープンソースの統合監視ソフトウェアである．Zabbix によるシステム監視を行うためには，まず監視対象とするサーバに Zabbix Agent (以下 ZA)，監視情報を集約するサーバに Zabbix Server (以下 ZS) をそれぞれインストールする．その後，ZA を ZS に監視対象サーバとして登録する必要がある．

Zabbix は監視情報の自動取得だけでなく，Zabbix がインストールされたサーバに対して様々なコマンドを実行するリモートコマンド機能や，ZA が起動したことを検知するアクティブチェック機能を有している．これらの機能に加えて，Zabbix 3.0[5] から，監視情報をやり取りする通信を暗号化する機能が追加された．

また，オープンソースであるため，有志による API の開発が盛んであり，ZA の登録などの処理をスクリプトによって実行することができる．

本研究では，Zabbix の機能を Zabbix API によって利用し，セキュアな監視環境の自動構築を行う．具体的には，アクティブチェック要求を送信した ZA の登録処理を，Zabbix API によって自動化し，高速に監視環境の構築を行う．また，監視情報の自動収集と暗号化は，Zabbix の機能を用いることで，確実に監視に必要な情報を，ZS に集約させる．

3.2 解析の実装支援

本研究では統計解析向けの開発実行環境である R[6] を用いる．R は，必要な関数や統計モデルを豊富に有しているため，複雑な統計解析も他の言語と比較して容易に実装

することが可能である．また，解析結果をグラフ化するための描画機能も有しているため，解析結果の視認性を向上させることも可能である．

一方で，システムの管理者が簡単に種々の統計解析を実行するため，本研究では，Web ブラウザ上から操作できる Web アプリケーションとして実装する．また，R の Web アプリケーション実装用パッケージである shiny[7] を利用する．shiny を用いることで，GUI パーツを利用してインタラクティブに統計解析を行うことが可能になる．

4. システムの概要

本章では，前章で述べた要求を満たすシステムの概要について述べる．図 1 は提案システムの全体像である．

4.1 セキュアな監視環境

本節では，提案システムの概要について述べる．

監視情報は主に図 1 中の，ZS と ZA の間でやりとりされる．このとき，ZA は正しい ZS に対して通信をしているかどうか，ZS は監視情報が途中で改ざんされていないことを確認するため，認証局から発行された証明書を用いて監視情報を暗号化する．このような Zabbix によるセキュアな統合監視のための環境は，スクリプトの実行により自動で構築される．

4.2 環境の自動構築

Zabbix によるセキュアな監視環境を構築するために，主に ZA が持つ，アクティブチェック機能を用いる．この機能は，ZA が起動した時，設定された ZS に対して自身の情報を送信し，受け取った ZS は設定されたルールと，受け取った情報を照合して，ZA の登録など条件に合致したアクションを実行する．

ZS と ZA でセキュアな通信を行うために，ZA に登録が必要な情報は以下のとおりである．

- ZS の IP アドレスかホスト名
- ZS に自身を識別させるためのホスト名
- 通信に使用する証明書と照合するための認証局

また，本研究では ZA の自動登録のため，これらに加えて次の項目を設定する．

- ZS からのリモートコマンド実行許可フラグ
- ホストを区別するためのメタデータ

メタデータは監視には使用しないパラメータであり，これを用いることで，サーバのフィルタリングやクラスタリングが可能になる．

この情報は ZA のインストール終了直後に登録することで，ZA が起動した際，アクティブチェック要求の送信時に ZS に送信される情報を操作することができる．

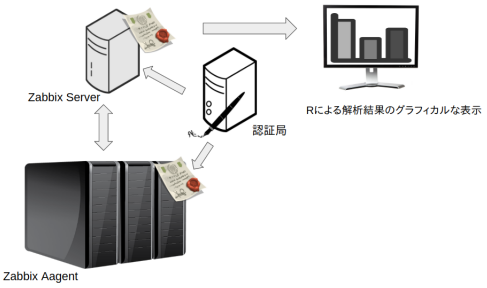

図 1　提案システムの全体図

4.3 統計解析アプリケーション

ZS に収集された監視情報に対して，管理者は種々の統計解析を実行する．実行するためのプログラムは，shiny パッケージを利用して，R 言語で Web アプリケーションとして実装する．GUI による操作が可能になっており，解析結果も，Web ブラウザ上にグラフィカルに表示される．また，予め使用頻度が高いと考えられる統計解析手法を選択可能にし，管理者が自由に使用できる機能を持たせる．

加えて，同じプログラムで新しい解析手法が実装できるよう，R 言語の開発実行環境が組み込まれており，管理者は新しい手法を自ら実装することが可能である．

5. 試作システム

本章では，4 章で述べたシステムを実際に試作したシステムについて述べる．

5.1 監視環境

Zabbix を用いてセキュアな監視環境の自動構築を行う時，ZA の登録に必要な時間と，登録する ZA の数との関係を調査するため，次のようなシステムを試作した．

5.1.1 仮想サーバの自動作成及び自動設定環境

KVM (Kernel-based Virtual Machine) によって仮想的に作成したサーバに，ZS のインストールと，証明書を照合するための認証局を設置する (以下，監視サーバと称す)．インストールした ZS は 3.0LTS の最新版 (2018̇01̇2̇9 現在) である，3.0.14 バージョンを使用した．監視サーバは実験用 VLAN 内に設置し，静的プライベートアドレスとして，192.168.100.1 を割り当てた．監視サーバには，ZA を登録するためのスクリプトが保存されており，そのスクリプトを実行するためのアクティブチェックルールが登録されている．スクリプトの詳細は，5.1.2 で述べる．監視サーバの環境を表 1，監視サーバが稼働する物理サーバの環境を表 2 に示す．

一方，ZA は ZS とは別の物理サーバ上で作成した仮想サーバにインストールする．これは，物理サーバのリソースの競合を防ぐためである．ZA 環境が構築されたサーバ (以下，被監視サーバと称す) を同時に複数起動させるため，Vagrant[8] と Ansible[9] を利用した，仮想サーバの自動作

表 1　ZS 用仮想サーバの環境

OS	CentOS 7.4
CPU	Intel CPU Corei7 2.7GHz x4
Memory	1GB
ZS	3.0.14

表 2　ZS 用物理サーバの環境

OS	CentOS 7.4
CPU	Intel CPU Xeon 2.7GHz x24
Memory	96GB
KVM	2.9.0

表 3　被監視サーバの環境

OS	CentOS 7.4
CPU	Intel CPU Corei7 2.8GHz x1
Memory	500MB
ZA	3.0.14

表 4　ZA 用物理サーバの環境

OS	CentOS 7.4
CPU	Intel CPU Xeon 2.8GHz x16
Memory	58GB
Vagrant	2.0.1
Ansible	2.4.2.0
KVM	3.2.0

成及び自動設定環境を構築した．具体的には，Vagrant で作成された仮想サーバに対して，Ansible で ZA のインストール，4.2 で述べた設定項目の反映，暗号化通信のための証明書の配布までを自動で行う．被監視サーバは監視サーバと同じ VLAN に設置し，Vagrant の DHCP 機能によって 192.168.0.0/16 のアドレスが振られる．被監視サーバの環境を表 3，被監視サーバが可動する物理サーバの環境を表 4 に示す．

この環境により，被監視サーバのセットアップが終了し，最終的に ZA デーモンが起動すると，アクティブチェック要求が監視サーバに送信され，監視対象サーバとして監視サーバに自動で登録される．

5.1.2　ZA 登録スクリプト

被監視サーバを監視サーバに登録するスクリプトは，まず被監視サーバの ZA に関する情報を編集し，編集した情報を基に監視サーバに登録する．具体的な処理のシーケンス図を図 2 に示す．

ZA の設定変更は，Zabbix への組み込みも考慮して，SSH などの一般的なプロトコルではなく，Zabbix のプロトコルに従った．具体的には，Zabbix が監視情報を取得する際などに利用される zabbix_get コマンドを用いて，シェルスクリプトの実行を行った．

監視サーバへの登録は，ZabbixAPI の 1 つである pyzabbix[10] を利用した．登録処理に Zabbix 本来の処理を使用しなかったのは，Zabbix3.0LTS の段階で，監視情報の通信を暗号化するための設定が自動で行えないためである．

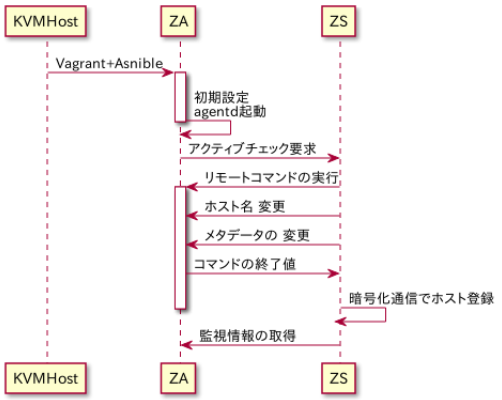

図 2　登録スクリプトのシーケンス図

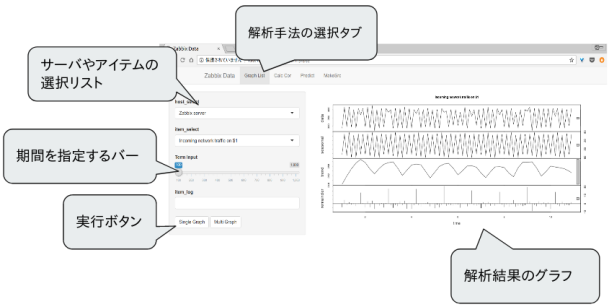

図 3　統計解析を実行する操作画面

pyzabbix によって，被監視サーバの IP アドレス，識別名，監視すべきリソースなどの情報と共に，監視情報を暗号化するためのフラグ情報を加えて登録する．

このような処理によって，被監視サーバの ZA が起動した際，自動的に監視サーバに登録されるようになり，結果としてセキュアな監視環境の構築が可能になる．

5.2　統計解析アプリケーション

Zabbix で収集した監視情報を R で参照することを検証するため，統計解析アプリケーションの試作を行った．Zabbix は収集した監視情報を MySQL[11] で保存できるため，R から MySQL データベースにアクセスする RMySQL[12] パッケージを使用した．実装した統計解析手法は以下のとおりである．

- 時系列データのトレンド抽出
- 相関係数の算出による類似データの検索
- 自己回帰モデルを用いた予測値の算出

試作した操作画面の例を図 3 に示す．図 3 のとおり，統計解析を実行する時は，選択リストによる解析対象ホストの選択や，解析に使用するデータの期間を指定するために，スライドバーを使用することが可能である．また，タブによって統計解析を切り替えることが可能である．

実装された統計解析の他，ユーザが自分で解析手法を実装できる環境として，次の機能を実装した．

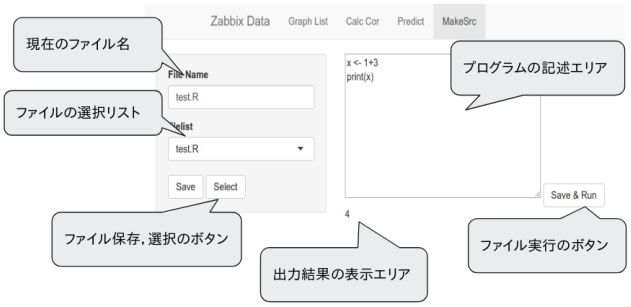

図 4　統計解析を実装する操作画面

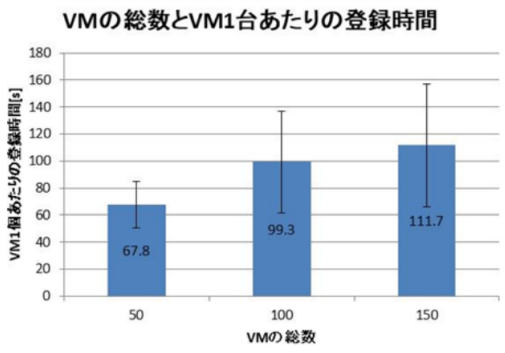

図 5　1 台あたりの処理時間

- R 言語のコーディング及び実行環境
- 作成したプログラムのファイル保存
- 保存された R プログラムの実行

試作した実装画面を図 4 に示す．図 4 のとおり，画面右側のテキストエリア内で記述された R プログラムを実行することが可能である．また，画面左側の選択リストから過去に作成したプログラムを読み出すことも可能である．

6. 評価

6.1 監視環境の自動登録

この評価では，被監視サーバを監視サーバに登録するために必要な時間と，被監視サーバの数の関係を調査する．評価環境は 5.1 節で述べたものである．

評価方法は，Vagrant と Ansible で自動的に作成された被監視サーバが，監視サーバ内にあるスクリプトによって自動登録され，Zabbix によって監視情報を取得できるまでの処理時間を計測する．具体的には，ZA から送信されたアクティブチェック要求を受け取ったことを示す trapper got ログと，ZA に対して監視情報を取得する関数を実行したことを示す get_value_agent ログを用いる．このログを用いて，被監視サーバの数を，50，100，150 と増加させた時，1 台の被監視サーバを登録するために必要な時間と，最終的に全被監視サーバが監視可能となるまでの時間を計測する．計測は，それぞれ 3 回ずつ行い，その平均を算出した．

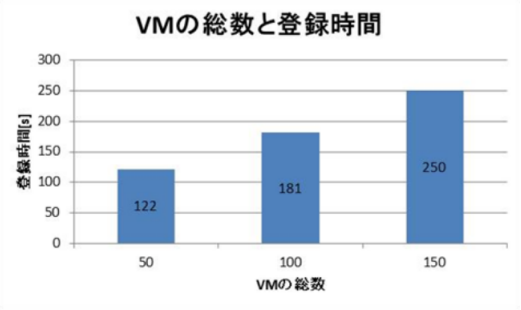

図 6　全体の処理時間

評価結果を図 5，6 に示す．

6.2 統計解析の処理時間の測定

この評価では，試作した統計解析アプリケーションにおいて解析結果が得られるまでの時間の測定を行う．実験を行った環境を 5 に示す．

プログラムの起動時から，解析結果の表示までの処理を次のように分解する．

(1) プログラムの起動
(2) パッケージの読み込み
(3) GUI の描画
(4) データの取得期間や対象となるアイテムの選択
(5) 統計解析の実行開始
(6) データベースとの接続とデータの取得
(7) 統計解析
(8) グラフ表示

この処理中で，計測の対象を処理 5 から処理 8 までとした．各統計解析に対して次のような評価を行う．

時系列解析と AR モデルによる予測

監視情報の取得期間に Zabbix の取得単位であるクロックを使用し，200 から 1000 クロックまでを 200 クロックごとにそれぞれ 5 回計測し，その平均値を処理時間とする．監視対象のアイテムとして各クロックごとに異なるものを使用する．

相関係数による比較

監視情報の取得期間を 100 クロック，出力結果を 3 項目として監視対象のアイテムの数を 60 から 20 アイテム毎に 120 まで変化させそれぞれ 5 回計測し，その平均値を処理時間とする．監視対象のアイテムとしては各計測ごとに異なるものを使用する．

6.3 評価する監視アイテム

本システムの評価において，各統計解析の対象としたアイテムを表 6 に示す．表中のアイテムに対して各統計解析をそれぞれ 1 回ずつ行い，処理時間の平均値を取得する．

本システムにおいて，処理時間の計測には R に実装されている proc.time 関数を用いた．この関数は実行時に，R プログラムの実行開始時間を起点とした時刻を返すもので

表 5　ZS 環境

Hypervisor	KVM (Kernel-based Virtual Machine)
OS	CentOS 6.6
CPU	AMD Opteron(tm) Processor 8222 3.0 GHz
vCPU	2.0 GHz (8core)
Disk	10 GB
Memory	1 GB
kernel	2.6.32-504.el6.x86_64

表 6　評価対象のアイテム

対象のホスト	対象のアイテム
Zabbix server	Incoming network traffic on eth1
Zabbix server	Outgoing network traffic on eth1
Zabbix server	Processor load (15 min average per core)
Zabbix server	Available memory
Zabbix server	Zabbix history write cache, % free

表 7　時系列解析の処理時間 [s]

クロック	1 回目	2 回目	3 回目	4 回目	5 回目	平均値
200	0.058	0.059	0.060	0.059	0.059	0.059
400	0.070	0.073	0.075	0.070	0.069	0.071
600	0.079	0.080	0.080	0.083	0.085	0.081
800	0.082	0.084	0.092	0.085	0.087	0.086
1000	0.093	0.106	0.099	0.120	0.106	0.105

表 8　相関係数による比較の処理時間 [s]

アイテム数	1 回目	2 回目	3 回目	4 回目	5 回目	平均値
60	1.358	1.348	1.361	1.381	1.367	1.363
80	2.069	2.049	2.042	2.044	2.060	2.053
100	2.502	2.429	2.530	2.368	2.153	2.397
120	3.012	2.763	2.775	2.858	3.070	2.897

表 9　AR モデルによる予測値の処理時間 [s]

クロック	1 回目	2 回目	3 回目	4 回目	5 回目	平均値
200	0.049	0.054	0.062	0.061	0.056	0.056
400	0.057	0.053	0.065	0.055	0.066	0.059
600	0.066	0.063	0.067	0.065	0.076	0.067
800	0.069	0.068	0.085	0.063	0.074	0.072
1000	0.077	0.074	0.075	0.080	0.073	0.076

あり，計測したい処理の開始時刻と終了時刻の差を取ることで処理に必要な時間を計測した．得られる結果の単位はすべて秒である．得られた結果を表 7，8，9 に示す．

7. 考察

本章では提案システムを実際に利用した際に考えられる利点や問題点，今後の発展性について述べる．

7.1 自動構築の処理時間

本節では，監視環境の自動構築にかかる処理時間について考察する．図 6 より，被監視サーバの登録処理に，Zabbix API を用いたスクリプトを使用することにより，従来の，Web ダッシュボード上での操作に比べて，高速に処理を行えることがわかる．

一方で，図 5 より，VM1 台あたりの登録に必要な時間の標準偏差が大きい事がわかる．これは，Zabbix がアクティブチェック要求を受け付けた後，すでに登録した被監視サーバに対して監視情報の要求処理を行っているため，アクティブチェック要求の受け取りが遅れた被監視サーバの登録に時間がかかっているためであると考えられる．

以上のことから，Zabbix API を用いた登録作業の自動化によって，高速に監視環境の構築が行えることが可能になる．

7.2 監視環境のセキュリティ

本節では，システムの管理者が実際に提案システムを使用して監視環境の構築を行った際の影響について考察する．提案システムにより，監視対象のサーバを被監視サーバとして登録するのみでなく，やりとりされる通信情報の暗号化も自動で行う．これにより，通信の途中で監視情報が改ざんされ，システムの管理に支障を来す要因を削減することが可能になる．また，被監視サーバを登録する際のアクションを制御することも可能であるため，幅広いシステムの監視に適応することが可能である．

通信の暗号化についての今後の展望として，認証方式を変更することが考えられる．試作システムでは，暗号化に使用する証明書は，サーバ証明書であった．この場合，被監視サーバが正しい監視サーバと通信していることは証明できるが，監視サーバが正しい被監視サーバと通信しているかどうかは証明できない．この問題を解決するため，暗号化に使用する証明書をサーバ証明書からクライアント証明書に変更する手法が考えられる．これにより，被監視サーバとして適切でないサーバを，自動登録の範囲外とすることが可能になる．

一方，予想される問題として，通信情報の暗号化に使用する証明書の管理の難しさが挙げられる．証明書を用いた通信の暗号化では，公的な認証機関から発行された証明書が必須である．また，証明書には有効期限が存在するため，各サーバが有する証明書を適切な時期にアップデートする必要が生じる．

7.3 統計解析アプリケーション

本節では，提案した統計解析アプリケーションを実際に利用した際に考えられる利点や，問題点について述べる．

試作システムでは，GUI の操作で種々の統計解析手法が利用できるため，プログラミングの経験が少ないユーザでも，目的の解析結果を得られやすいと考えられる．一方で，統計解析手法を管理者が実装することが可能であるため，より対象のシステムに特化した解析が可能になると考えら

れる．

統計解析アプリケーションの今後の展望として，実際にシステム管理に携わっている管理者に対してインタビューを行い，求められる頻度の高い監視項目に特化した，解析手法の実装を行うことが考えられる．また，統計解析手法の実装に関しても，使用頻度の高い処理のマクロ化や，GUI による操作を追加するなどにより，操作性が向上することが考えられる．

システムの問題点として，試作システムにおける相関係数の算出のような，複数の被監視サーバを横断する解析は，被監視サーバ数の増加により線形に処理時間が増加する．この問題の解決方法として，解析の対象とする被監視サーバを限定するなどして，処理時間を抑制することが考えられる．

8. おわりに

本論文では，統合監視ソフトウェア Zabbix と R 言語を用いて，大規模システムに対する，セキュアな監視環境の自動構築と，操作性の高い統計解析アプリケーションシステムを提案した．さらに，被監視サーバの数と必要な処理時間の計測と，統計解析アプリケーションの試作を行った．その結果，被監視サーバの数に伴い，環境の構築や統計解析による処理に必要な時間が増加するが，セキュアな監視環境が高速に構築され，統計解析アプリケーションによりシステム管理に有益な情報を得ることが可能であることを示した．また，将来的に被監視サーバの認証方式の変更により，監視環境の更なるセキュリティの向上に関する可能性も示した．今後の課題として，提案システムを実システムで稼働させ，より強固なセキュリティを確保した環境の自動構築や統計解析手法の開発実行環境の充実が必要であると考えられる．

参考文献

[1] ForeScount: IoT and OT Security Research Exposes Hidden Business Challenges, ForeScount (online), available from ⟨https://www.forescout.com/iot_forrester_study/⟩ (accessed 2018-01-22).
[2] 立見博史，菅谷至寛，阿曽弘具：プロセス情報を用いた計算機不可長期予測モデル，第 4 回情報科学フォーラム，pp. 13–15 (2005).
[3] 坂下幸徳，東条　敏，敷田幹文：障害原因解析における構成情報の統計的推論方式，情報処理学会論文誌，pp. 767–776 (2015).
[4] Zabbix SIA: Zabbix オフィシャル日本語サイト，Zabbix SIA（オンライン），入手先 ⟨http://www.zabbix.com/jp/⟩（参照 201-01-22）.
[5] Zabbix SIA: What's new in Zabbix 3.0.0, Zabbix SIA (online), available from ⟨https://www.zabbix.com/documentation/3.0/manual/introduction/whatsnew300⟩ (accessed 2018-01-22).
[6] The R fondation.: The R Project for Statistical Computing, The R Foundation. (online), available from ⟨https://www.r-project.org/⟩ (accessed 2016-2-9).
[7] RStudio, Inc.: Shiny by Rstudio, RStudio, Inc. (online), available from ⟨http://shiny.rstudio.com/⟩ (accessed 2017-01-22).
[8] Hashicorp: Vagrant Documentation, Hashicorp (online), available from ⟨https://www.vagrantup.com/docs/index.html⟩ (accessed 2018-01-23).
[9] Red Hat Inc.: AUTOMATION FOR EVERYONE, Red Hat Inc. (online), available from ⟨https://www.ansible.com/⟩ (accessed 2018-01-23).
[10] lukecyca: pyzabbix, lukecyca (online), available from ⟨https://github.com/lukecyca/pyzabbix⟩ (accessed 2018-01-22).
[11] Oracle Corporation and/or its affiliates: MySQL, Oracle Corporation and/or its affiliates (online), available from ⟨https://www-jp.mysql.com/⟩ (accessed 2016-2-9).
[12] Department of Biostatistics Vanderbilt University School of Medicine: RMySQL, Vanderbilt University School of Medicine (online), available from ⟨http://biostat.mc.vanderbilt.edu/wiki/Main/RMySQL⟩ (accessed 2018-01-22).