障害が発生したサーバのSSHの接続ユーザ数が最も多いチケットの通知による発行から調査までの時間の短縮

月森 陽太[1]　平尾 真斗[2]　串田 高幸[1]

概要：東京工科大学コンピュータサイエンス学部の研究室である Cloud and Distributed Systems Laboratory(以後，CDSL とする) では，ESXi がインストールされている 10 台のサーバが稼働していることを確認するために監視システムを導入している．監視システムがアラートを作成すると，その内容をもとにチケットが作成される．1 次チケットの担当者として割り当てられた学生は，調査を行う．課題は，1 次チケットが発行されてから 1 次チケットの担当者が 1 次調査を開始するまでに時間がかかることである．理由は，調査が行われていない 1 次チケットのうち，どのチケットから調査をすべきなのかについて明記したルールがなく，発行日時が古いチケットの調査が遅れることがあったためである．基礎実験では，1 次チケットが発行されてから 1 次チケットの担当者が 1 次調査を開始するまでの時間を計測した．1 次チケットは全 28 件あり，調査が開始された 1 次チケットは 26 件である．そのうち，0 分から 1440 分のチケットが 5 件，1440 分から 2880 分のチケットが 1 件，2880 件から 4320 分のチケットが 2 件，4320 分から 5760 分のチケットが 2 件，5760 分から 7200 分のチケットが 1 件，7200 分から 8640 分のチケットが 0 件，8640 分から 10080 分のチケットが 2 件，10080 分以上が経過しているチケットが 13 件である．1 次調査の開始までに 1 週間以上が経過していたチケットがあり，調査が迅速に行われていないといえる．提案の対象は，ICMP パケットによるサーバとの疎通確認に失敗したことを示すアラートと，ノードのメトリクスが異常であることを示すアラートのチケットである．提案では，アラートが通知された監視対象であるサーバに対して ICMP パケットによる疎通確認を行い，返答がない場合は「疎通不可」，返答がある場合は「疎通可能」の 2 種類のラベルのいずれかをチケットに付与する．同時に，1 次チケットが発行された時刻におけるサーバの SSH の接続ユーザ数をチケットのスコアとする．ESXi をハイパーバイザーとするサーバからアラートが通知されている場合は，ESXi 上の仮想マシンにインストールされている Ubuntu Server の SSH の接続ユーザ数をそれぞれ計測し，合計した値をチケットのスコアとする．その後，スコアが最も高いチケットを 1 次チケットの担当者に通知する．スコアが同じチケットがある場合は，ラベルにもとづいて並び替えられる．

1. はじめに

背景

アプリケーションやサービスを提供するテクノロジー企業は，コミュニケーションやタスク管理，決済，エンターテインメントまであらゆるものを効率化し利便性を追求する一方で，予期せぬ出来事や計画外の中断によって IT(Information Technology) サービスの品質が低下するインシデントが数多く発生し，インシデントの予防と対応の効率化の重要性が高まっている [1]．インシデントとは，サービスの標準的な運用に含まれない，提供されるサービスの品質やレベルの低下を引き起こす，または引き起こす可能性のある事象である [2]．

監視システムや顧客から報告されたインシデントの追跡，トラブルシューティング，解決するためのアクションを記録するデータベースとして，チケットシステムが使用される [3]．

監視システムは，取得したメトリクスがあらかじめ指定したしきい値を満たしたとき，アラートを作成し E メールやチャットツールに通知する [4]．作成されたアラートからチケットが発行されると，オンコールエンジニアは文書化された手順にもとづき障害の調査を行う．さらなる調査が必要な場合に，より高度なスキルと知識を持つ 2 次または 3 次の担当者にエスカレーションされる [5]．

オンコール対応は，多くの運用チームやエンジニアリン

[1] 東京工科大学コンピュータサイエンス学部
クラウド・分散システム研究室
〒 192–0982 東京都八王子市片倉町 1404-1
[2] 東京工科大学大学院バイオ・情報メディア研究科
コンピュータサイエンス専攻
クラウド・分散システム研究室
〒 192–0982 東京都八王子市片倉町 1404-1

© 2025 Cloud and Distributed Systems Laboratory

グチームにとって，サービスの信頼性と可用性を維持するために不可欠な義務である [6].

チケット管理のツールの 1 つに，オープンソースソフトウェアである Redmine がある．1 つのタスクをチケットの単位で管理することで，管理者がプロジェクトの進捗状況や対応履歴を認識することが可能になる [7, 8].

インフラストラクチャやアプリケーションの監視のツールの 1 つに Prometheus がある．Prometheus は CPU 使用率やメモリ使用量，ディスク使用量，ネットワーク統計のメトリクスを監視し，システムの健全性の詳細なビューを提供する [9]．また，Prometheus のアラート管理コンポーネントである Alertmanager は，Prometheus から送信されるアラートの重複排除やグループ化，また E メールやチャットツールへの通知を行う [10].

東京工科大学コンピュータサイエンス学部の研究室である Cloud and Distributed Systems Laboratory(以後，CDSL とする) では，10 台の ESXi がインストールされているサーバを運用し，ハイパーバイザーとして Broadcom 社の VMware 製品である VMware ESXi が導入されている．10 台のうち 7 台は，CDSL に所属している学生が自由に実験や作業を行ったり，アプリケーションを稼働させる目的で Ubuntu Server がインストールされた仮想マシンを作成し，使用することができる．残りの 2 台は，DNS サーバや DHCP サーバ，Jump サーバを運用する目的で使用され，1 台は，外部に Web サイトを公開する目的で使用される．10 台のサーバのほかにも，Router や Switching Hub，NAS，Archive Server がある．

CDSL では，サーバやアプリケーションに障害が発生した場合に，迅速に調査や対応を行うことができるように対応する担当者が時間交代制で決められている．担当時間の内訳は，月曜日から金曜日の 9 時 00 分から 10 時 30 分，10 時 30 分から 12 時 30 分，12 時 30 分から 15 時 00 分，15 時 00 分から 17 時 30 分である．障害は監視システムによって検出され，同時にアラートの作成と通知を行う．また障害の調査と対応はチケットシステムで管理されている．CDSL の監視システムとチケットシステムの構成を図 1 に示す．図中の Mint は，CDSL で運用している 10 台のサーバのうちの 1 台の名称である．Archive Server は，使用されなくなった仮想マシンにインストールされている Ubuntu Server のデータをアーカイブする目的でのみ使用されるサーバの名称である．監視システム内には，ハイパーバイザーである VMware ESXi や，仮想マシンにインストールされている Ubuntu Server 上で動作するアプリケーションの外形監視の結果をメトリクスとして取得する Blackbox exporter や，ESXi 上のホストや Ubuntu Server のメトリクスを取得する vmware exporter，稼働中の Docker コンテナや Kubernetes Pod のメトリクスを取得する cAdvisor が配置されている．また Archive Server にインストールされている OS 上には，OS のメトリクスを取得する Node exporter が配置されている．取得したメトリクスは，Prometheus に送信される．Prometheus は，取得したメトリクスが，あらかじめ指定したしきい値を満たしたときにアラートを作成する．作成されたアラートは Alertmanager によって集約され，Ticket Creator に送信される．Ticket Creator は，受信したアラートの名前，アラートが作成された時刻，インスタンス名，1 次チケットの担当者を入力した 1 次チケットを Redmine 上に作成する．

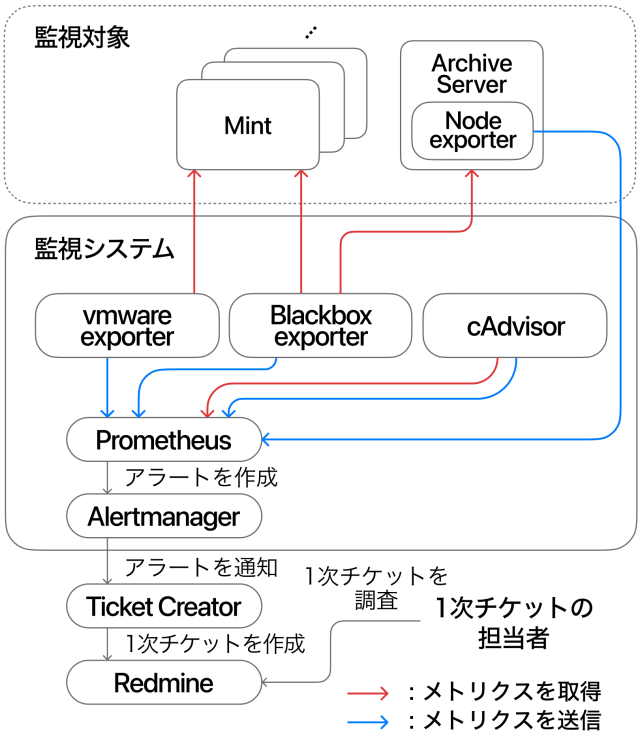

図 1　監視システムと Ticket Creator の構成

次に CDSL で行っている調査と対応の流れを図 2 に示す．1 次チケットの担当者は，Runbook とよばれる調査の手順を記載した手順書をもとにした調査を行い，調査が終了したら，あらかじめ決められたレビューの担当者が Runbook に記載されている調査項目に沿って調査が実施されているか確認し，調査が十分であると判断した場合は調査を完了とする．一方で，不十分であると判断した場合は，1 次チケットの担当者に再度調査を行うように指示し，1 次チケットの担当者は調査を行う．調査が完了したら，対応を行う．対応が終了したら，レビューの担当者が 1 次チケットの担当者が行った対応が有効であるか判断し，有効であると判断した場合は対応が完了したとして，1 次調査を終了する．一方で，対応できない，あるいは対応が不十分であると判断した場合は，1 次チケットの担当者に 2 次チケットの作成を依頼する．1 次チケットの担当者は 2 次チケットを作成する．2 次チケットの担当者は，調査の

© 2025 Cloud and Distributed Systems Laboratory

手順を記載したテンプレートをもとにした調査を行い，調査が終了したら，レビューの担当者がテンプレートに記載されている調査項目に沿って調査が実施されているか確認し，調査が十分であると判断した場合は調査を完了とする．一方で，調査が不十分であると判断した場合は，2 次チケットの担当者に再度調査を行うように指示し，2 次チケットの担当者は調査を行う．調査が完了したら，対応を行う．対応が終了したら，レビューの担当者が 2 次チケットの担当者が行った対応が有効であるか判断し，有効であると判断した場合は対応が完了したとして，2 次調査を終了する．一方で，対応ができない，あるいは対応が不十分であると判断した場合は，2 次チケットの担当者に 3 次チケットの作成を依頼する．2 次チケットの担当者は 3 次チケットを作成する．3 次チケットの担当者は調査を行い，調査が終了したらレビューを依頼する．レビューの担当者が調査が十分であると判断した場合は調査を完了とし，3 次チケットの担当者は次に対応を行う．一方で，レビューの担当者が根本原因の特定ができていないと判断した場合は，代替となるサーバやアプリケーションを選定し，それが利用可能であれば，3 次調査を終了する．3 次チケットの担当者の対応が終了したらレビューを依頼し，レビューの担当者が 3 次チケットの担当者が行った対応が有効であると判断した場合は対応が完了したとし，3 次調査を終了する．一方で，対応ができない，あるいは対応が不十分であると判断した場合は，代替となるサーバやアプリケーションを選定し，それが利用可能であれば，3 次調査を終了する．

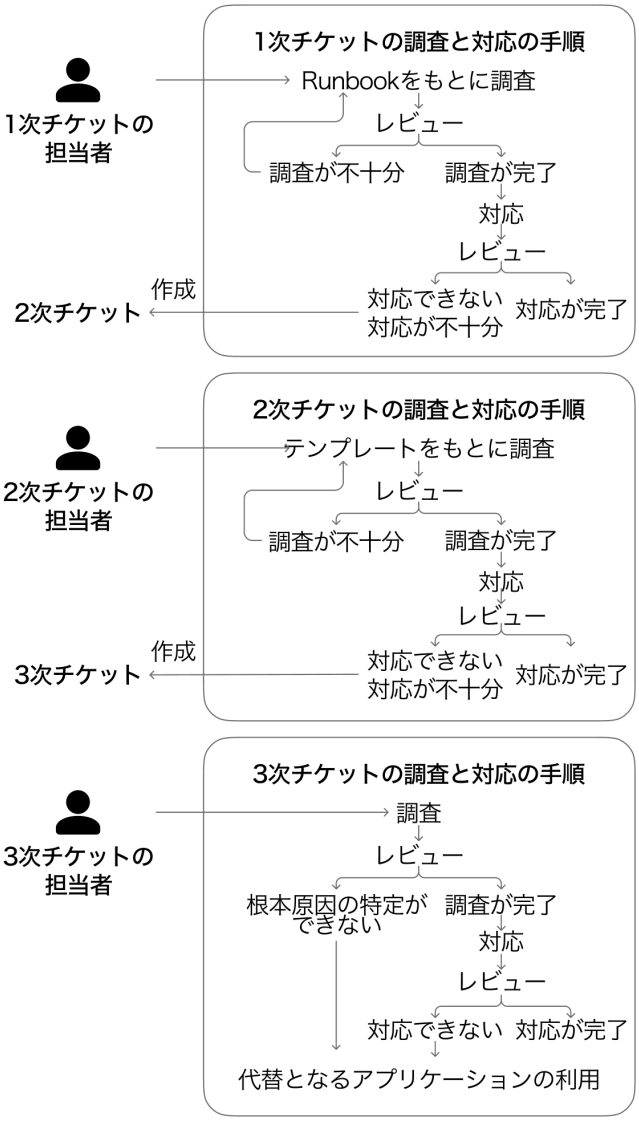

図 2　CDSL で行っている調査と対応の流れ

課題

課題は，1 次チケットが発行されてから 1 次チケットの担当者が調査を開始するまでに時間がかかることである．そのため，1 次チケットが発行されてから調査または対応が完了し，チケットのステータスが完了に変更されるまでの時間が増加し，障害が発生してから復旧するまでに時間がかかる．特に影響を受けるユーザ数が多い障害は，調査や対応を迅速に行う必要がある．原因は，1 次チケットの担当者が担当時間内に通知されたすべての 1 次チケットの調査を行うことができなかった場合，残った 1 次チケットのうち，どのチケットから調査をすべきなのかについて明記したルールがなかった．そのため．1 次チケットが発行されてから 1 次調査までに時間がかかっているチケットがあった．

図 3 に，1 次チケット発行から 2 次調査終了までに行った作業の日時を示す．CDSL で行っている調査と対応の作業において，1 次チケットの「[Alert] Internal Archive Server ICMP-Check」が，2025 年 6 月 17 日 15 時 13 分に発行された．1 次調査は 2025 年 6 月 20 日 11 時 26 分に開始され，2025 年 6 月 20 日 12 時 35 分に終了した．2 次チケットは 2025 年 6 月 20 日 12 時 43 分に発行された．2 次調査は 2025 年 6 月 23 日 11 時 44 分に開始され，2025 年 6 月 23 日 12 時 17 分に終了した．これらの時間のうち，1 次チケットが発行されてから 1 次調査が開始されるまで，2 日と 20 時間 12 分が経過している．

基礎実験

基礎実験では，CDSL で行っている調査と対応の作業において，1 次チケットの発行日時と 1 次調査を開始した日時の差分を分布として示す．基礎実験の対象となるデータは，2025 年 6 月 12 日 0 時 00 分から 2025 年 7 月 4 日 0 時 00 分の期間で発行された 28 件の 1 次チケットであり，そのうち 26 件は 1 次調査が開始しており，2 件は 1 次調査が開始していない．図 4 は，1 次調査が開始された 26 件の 1 次チケットについて，それぞれ 1 次チケット発行から 1 次調査を開始するまでにかかった時間の分布である．この図では，1440 分 (1 日) 単位の区間と，10080 分 (1 週間) 以上の区間でそれぞれ経過したチケットの件数を示す．

© 2025 Cloud and Distributed Systems Laboratory

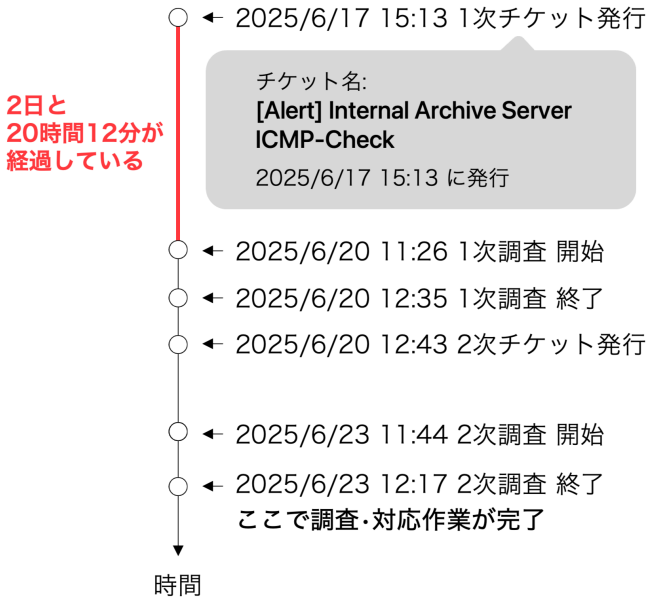

図 3 1 次チケット発行から 2 次調査終了までに行った作業の日時

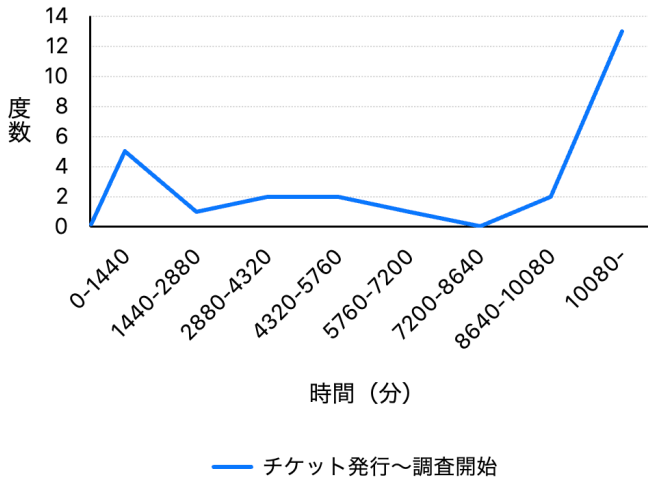

図 4 すべての 1 次チケット発行から 1 次調査を開始するまでにかかった時間の分布

1 次チケットが発行されてから 1 次チケットの担当者が 1 次調査を開始するまでの時間が 0 分から 1440 分のチケットが 5 件，1440 分から 2880 分のチケットが 1 件，2880 件から 4320 分のチケットが 2 件，4320 分か 5760 分のチケットが 2 件，5760 分か 7200 分のチケットが 1 件，7200 分か 8640 分のチケットが 0 件，8640 分か 10080 分のチケットが 2 件である．10080 分以上が経過しているチケットが 13 件である．障害が発生したら迅速に調査や対応を行う必要があるが，基礎実験の結果から，1 次調査の開始までに 1 週間以上が経過していたチケットがあることが分かり，迅速に調査を行うことができていないといえる．

各章の概要

第 2 章の関連研究では，関連する既存研究について述べる．第 3 章の提案では，課題を解決する提案とユースケース・シナリオについて述べる．第 4 章の実装では，提案をもとに作成したソフトウェアの実装方法について述べる．第 5 章の評価実験では，実験環境について述べる．第 6 章の議論では，提案の議論を述べる．第 7 章では，全体のまとめを述べる．

2. 関連研究

IT サービスの運用管理におけるシフト計画とスケジューリングに関する研究がある．この研究では，労力と人員の推定，スケジューリング，シフトの計画と割り当てを統合したフレームワークを作成し，IT 運用におけるリソースの最大利用率を確保することでコストを最小化することを提案している．その結果，シフト計画と人員配置の最適化により，Service Level Agreement(SLA) 内で IT サービスの効率的な運用が保証されることが示された．一方で，インシデントチケットの発行から一次調査までの特定のプロセスに焦点を当てた詳細な分析や，そのための具体的なメカニズムは提供されていない点に改善の余地がある [11].

クラウドにおける AI ワークロードのインシデント診断と報告を効率化する研究がある．この研究では，顧客中心の自動インシデント診断システムである AidAI を提案している．AidAI は，過去のオンコール経験から内部知識ベースを構築し，人間の専門家の推論プロセスを模倣してインシデントを診断する．その結果，Microsoft の実際のインシデント記録を用いて，既存の手法を上回る診断能力を示した．一方で，このシステムはインシデントの根本原因の診断は行うものの，インシデントに対する具体的な緩和策は提供していない [12].

重要なプロセスおよびプロジェクト活動の監視のためのエスカレーション管理に関する研究がある．この研究では，個々のユーザの視点から情報フローとプロセス活動の時間制約を監視することを目指し，プロセス参加者の観測された行動に自動的に適応する通知およびエスカレーションメカニズムを提案している．その結果，ユーザ固有のモデルデータと既存の経験に基づいてユーザをサポートすることが可能となった．一方で，エスカレーション戦略の適用に関するフィードバックは提供されるものの，インシデント対応における実際に短縮された時間の評価は明確に示されていない [13].

3. 提案

提案では，発行された 1 次チケットのうち，調査を迅速に開始すべき 1 次チケットを判断し，1 次チケットの担当者に通知する．これは，調査を迅速に開始すべき 1 次チケットが発行されてから 1 次調査を開始するまでの時間を，他

© 2025 Cloud and Distributed Systems Laboratory

のチケットより優先して短縮することを目的としている．提案の対象とするアラートは，ICMP パケットによるサーバとの疎通確認に失敗したことを示すアラートと，ノードのメトリクスが異常であることを示すアラートである．迅速な調査が必要な障害は，影響を受ける人数が多いことを条件とする．例えば，停止したサーバ内の ESXi 上に配置されている仮想マシンにインストールされている OS 上で動作するアプリケーションを稼働させることができず，アプリケーションの開発者は更新の適用やメンテナンスをすることができない．またアプリケーションの利用者に対してサービスの提供が停止する．図 5 に提案の概要を示す．監視対象は複数のサーバであり，それぞれにエージェント

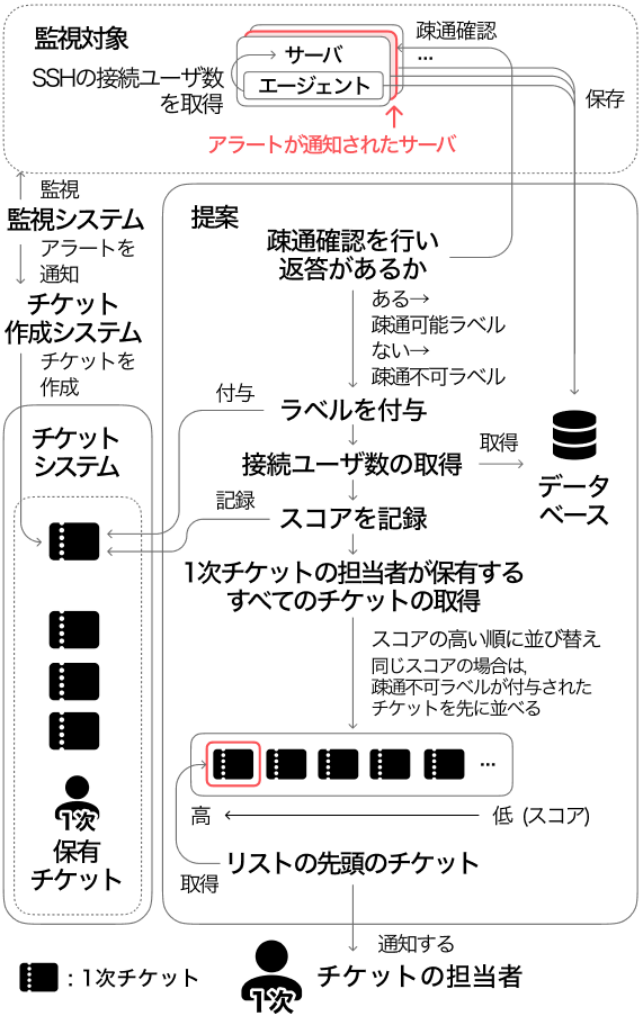

図 5　提案の概要

が配置されている．監視システムは，取得したメトリクスが，あらかじめ指定したしきい値を満たした時にアラートを作成し，チケット作成システムに通知する．チケット作成システムは，受信したアラートのアラート名，アラートが作成された時刻，インスタンス名，1 次チケットの担当者を入力した 1 次チケットをチケットシステム上に作成する．データベースは，エージェントが取得した SSH の接続ユーザ数を保存している．アラートが通知された監視対象であるサーバに対して ICMP(Internet Control Message Protocol) Echo Request を送信し，ICMP Echo Reply がない場合は「疎通不可」，ある場合は「疎通可能」のラベルをチケットに付与する．監視対象のサーバにあらかじめ配置されているエージェントは，サーバの OS の SSH(Secure Shell) の接続ユーザ数を計測しデータベースに保存する．データベースから 1 次チケットが発行された時刻の SSH の接続ユーザ数を取得し，チケットのスコアとする．ESXi をハイパーバイザーとしているサーバからアラートが通知されている場合は，ESXi 上の Ubuntu Server の SSH の接続ユーザ数をそれぞれの Ubuntu Server ごとに計測し，合計した値をチケットのスコアとする．その後，1 次チケットの担当者が保有しているすべてのチケットは，スコアの高い順に並びかえられる．このとき，スコアが同じである場合は，ラベルにもとづいて並び替えられる．具体的には同じスコアのチケットのうち，「疎通不可」ラベルが付与されたチケットが先に並び，「疎通可能」ラベルが付与されたチケットが後に並ぶ．この並び替えられたリストの先頭にあるチケットが 1 次チケットの担当者に通知され．1 次チケットの担当者は通知されたチケットの調査を開始する．1 次チケットの担当者が通知されたチケットより低いスコアのチケットの調査を行っていた場合は，そのチケットの調査を中断し，通知されたチケットの調査を開始する．一方で，1 次チケットの担当者が通知されたチケットよりも高いスコアのチケットの調査を行っていた場合は，そのチケットの調査を継続する．

ユースケース・シナリオ

ユースケース・シナリオとして，CDSL の環境で障害が発生し通知されたアラートから Ticket Creator が作成した 1 次チケットにおける，1 次調査を想定する．図 6 に提案適用後のユースケース・シナリオを示す．Blackbox exporter は外形監視を行い，サーバにインストールされている ESXi や，仮想マシンにインストールされている Ubuntu Server 上で動作するアプリケーションを監視している．Blackbox exporter が，CDSL のサーバである Mint 内の Ubuntu Server に対して ICMP Echo Request を送信し，ICMP Echo Reply を受信できるかどうかの結果をメトリクスとして監視システムに送信する．監視システムが作成したアラートは，Ticket Creator に通知される．Ticket Creator は，提案ソフトウェアにアラートを通知する．提案ソフトウェアは．アラートが通知された監視対象である Mint に対して ping コマンドを実行し，その出力結果に応じて「疎通不可」または「疎通可能」のラベルをチケットに付与する．また，データベースに格納されている SSH の接続ユーザ数をスコアとし，チケットに記録する．その後，1 次チケットの担当者が保有しているすべてのチケットは，

© 2025 Cloud and Distributed Systems Laboratory

スコアの高い順に並びかえられる．このときスコアが同じである場合は，「疎通不可」ラベルが付与されたチケットが先に並び，「疎通可能」ラベルが付与されたチケットが後に並ぶ．この並び替えられたリストの先頭にあるチケットの URL を，1 次チケットの担当者に Slack で通知する．1 次チケットの担当者は通知されたチケットの調査を開始する．

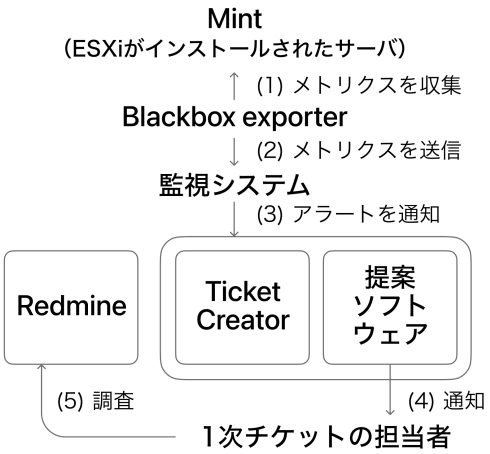

図 6　ユースケース・シナリオ

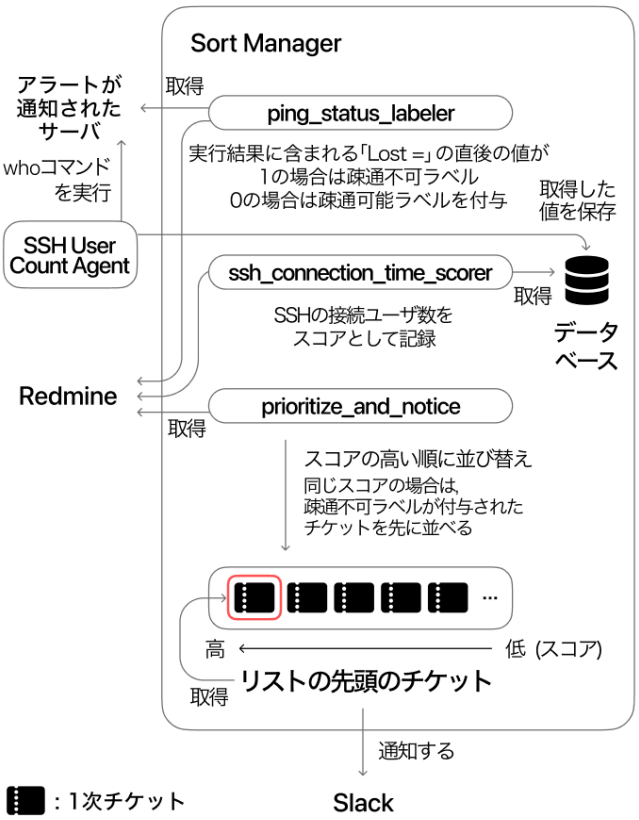

図 7　実装の全体構成

4. 実装

本提案を実装したソフトウェア (以後，提案ソフトウェアとする) である Sort Manager は，Python 3.12.3 で作成する．提案ソフトウェアは，あらかじめ監視対象のすべてのサーバの Ubuntu Server 上で SSH User Count Agent が動作していることを前提とする．図 7 に，実装の全体構成を示す．監視対象は複数のサーバであり，それぞれにエージェントである SSH User Count Agent が配置されている．Prometheus は，取得したメトリクスが，あらかじめ指定したしきい値を満たしたときにアラートを作成する．作成されたアラートは Alertmanager によって集約され，Ticket Creator に送信される．Ticket Creator は，受信したアラートのアラート名，アラートが作成された時刻，インスタンス名，1 次チケットの担当者を入力した 1 次チケットを Redmine 上に作成する．データベースは，SSH User Count Agent が取得した SSH の接続ユーザ数を保存している．Sort Manager は，関数である ping_status_labeler と ssh_connection_time_scorer, prioritize_and_notify を順番に実行する．

SSH User Count Agent

監視対象のすべてのサーバの OS 上で，ホスト名と SSH の接続ユーザ数を ssh_connection_time_scorer に送信する bash によるシェルスクリプトである．このシェルスクリプトは systemd によって 1 分おきに動作する．ホスト名は hostname -I コマンドで取得する．SSH の接続ユーザ数は，who コマンドの出力にあるユーザ ID の中で重複を除いたユーザ ID の数をカウントすることで取得する．

ping_status_labeler

Sort Manager 内の関数であり，アラートが通知された監視対象であるサーバに対して ping コマンドを実行し，その結果にもとづいてチケットにラベルを付与する．ping コマンドとは，ICMP を利用して指定したホストに ICMP Echo Request を送信し，ICMP Echo Reply を取得し，その結果を出力するコマンドである．ping コマンドの実行結果のうち，「Lost = 」の直後の値が 1 の場合は疎通確認の返答の受信に失敗したと判断し，「疎通不可」のラベルを付与する．「Lost = 」の直後の値が 0 の場合は疎通確認の返答の受信に成功したと判断し，「疎通可能」のラベルをチケットに付与する．

ssh_connection_time_scorer

Sort Manager 内の関数であり，ping_status_labeler で付与されたラベルと，SSH User Count Agent から送信されるホスト名と SSH の接続ユーザ数にもとづき，チケットにスコアを記録する．チケットに付与されたラベルが「疎通不可」の場合，疎通ができなくなる直前に送信された SSH

© 2025 Cloud and Distributed Systems Laboratory

の接続ユーザ数をスコアとしてチケットに記録する．チケットに付与されたラベルが「疎通可能」の場合，チケットが発行された時刻の SSH の接続ユーザ数をスコアとしてチケットに記録する．

prioritize_and_notify

Sort Manager 内の関数であり，1 次チケットの担当者が保有しているすべてのチケットを Redmine から取得し，ラベルとスコアにもとづいて並び替えを行う．すべてのチケットをスコアの高い順に並びかえる．このとき，スコアが同じである場合は，ラベルにもとづいて並び替えられる．具体的には同じスコアのチケットのうち，「疎通不可」ラベルが付与されたチケットが先に並び，または「疎通可能」ラベルが付与されたチケットが後に並ぶ．この並び替えられたリストの先頭にあるチケットの URL を，1 次チケットの担当者に Slack で通知する．

5. 評価実験

評価実験では，あらかじめ 4 人の 1 次チケットの担当者を指定し，それぞれの 1 次チケットが発行されてから 1 次調査を開始するまでの時間の削減率を，提案の適用前と適用後で評価する．評価実験の期間は 1 ヶ月とし，提案の適用前と適用後でそれぞれ 15 日間で時間を計測する．提案の適用前のデータは，基礎実験で収集した 4 人の 1 次チケットの担当者のそれぞれ 1 次チケットが発行されてから 1 次調査を開始するまでの時間を使用する．1 次チケットの発行日時は，Redmine にチケットが登録された日時とする．1 次調査を開始する時刻は，1 次チケットの担当者が 1 次調査を開始する直前に，指定した Google スプレッドシートに 1 次調査の開始時刻として記録した日時を使用する．

実験環境

図 8 に実験環境を示す．監視対象は ESXi がインストールされた 10 台のサーバや，Archive Server であり，それぞれに SSH User Count Agent が配置されている．Blackbox exporter は，ESXi や Ubuntu Server 上で動作するアプリケーションの外形監視の結果をメトリクスとして取得する．vmware exporter は ESXi 上のホストや Ubuntu Server のメトリクスを取得する．Archive Server に配置されている Node exporter は，OS のメトリクスを取得する．ESXi がインストールされた 10 台のサーバのうち，7 台は Mint, Jasmine, Rose, Lotus, Violet, Plum, Lily であり，CDSL に所属している学生が自由に実験や作業を行うことができる．残りの 2 台は Makaron と Canele であり，DNS サーバや DHCP サーバ，Jump サーバを運用する目的で使用され，1 台は Iris であり，外部に Web サイトを公開する目的で使用されている．Iris では CDSL に所属する学生が研究成果を記述したレポートであるテクニカルレポートを検索する Web サイト「doktor」を公開している．Archive Server は，使用されなくなった Ubuntu Server のデータをアーカイブする目的で使用されるサーバである．Google スプレッドシートは，1 次チケットの担当者が 1 次調査の開始時刻を記入するために使用され．実験の前にあらかじめ作成されている．

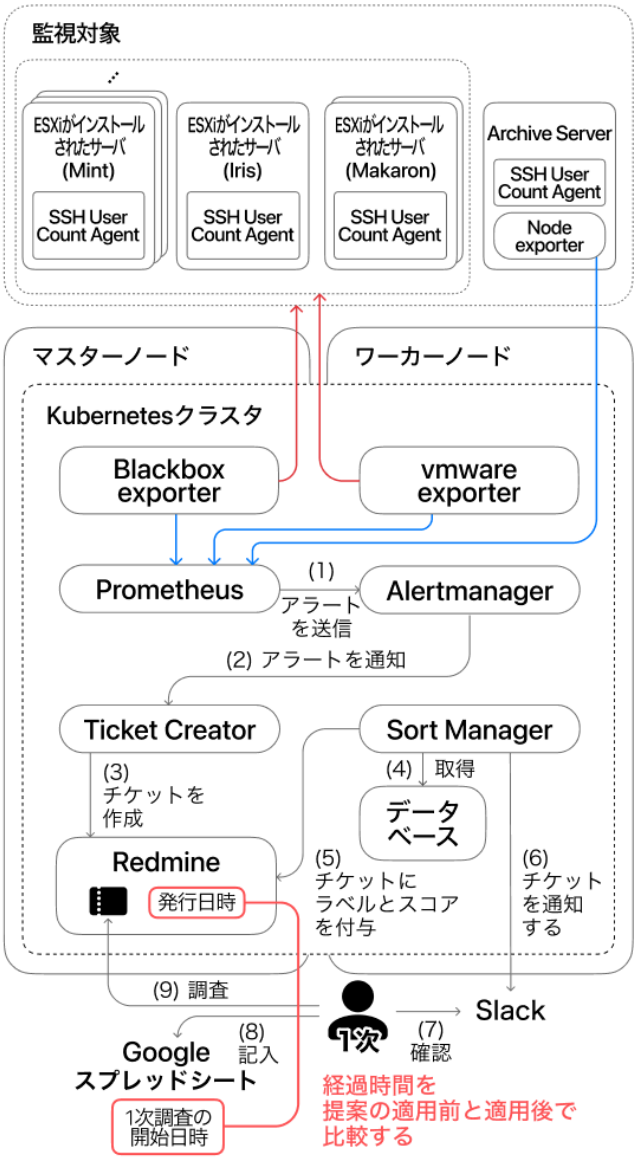

図 8 実験環境

Kubernetes クラスタはマスターノード 1 台とワーカーノード 1 台の構成であり，Kubernetes クラスタ上には Blackbox exporter, vmware exporter, Prometheus, Alertmanager, Ticket Creator, Redmine, 提案ソフトウェアである Sort Manager を配置する．Blackbox exporter は ESXi がインストールされたサーバや Archive Server の外形監視の結果をメトリクスとして取得し，vmware exporter はサーバにインストールされた ESXi 上のホストや仮想マシンにインストールされている OS のメトリクスを取

© 2025 Cloud and Distributed Systems Laboratory

得，Node Exporter は Archive Server 内の OS のメトリクスを取得する．Blackbox exporter や vmware exporter，Node exporter は取得したメトリクスを Prometheus に送信する．Prometheus は，送信されたメトリクスがあらかじめ指定したしきい値を満たしたときにアラートを作成し，Alertmanager に送信する．Alertmanager はアラートを Ticket Creator に通知する．Ticket Creator は，通知されたアラートのアラート名，アラートが作成された時刻，インスタンス名，1 次チケットの担当者を入力した 1 次チケットを Redmine 上に作成する．データベースは，SSH User Count Agent が取得した SSH の接続ユーザ数を保存している．Sort Manager は，Redmine 上のチケットにラベルとスコアを付与し，調査を迅速に開始すべき 1 次チケットを特定した後，そのチケットの URL を Slack で 1 次チケットの担当者に通知する．1 次チケットの担当者は，通知された 1 次チケットを Redmine で確認し，その 1 次チケットの名前，URL，1 次調査の開始時間を Google スプレッドシートに記入してから，1 次調査を開始する．1 次チケットの発行日時と 1 次調査の開始日時から，1 次チケットが発行されてから 1 次調査の開始までの経過時間を，提案の適用前と適用後で比較する．以下に，Kubernetes クラスタのマスターノードとワーカーノードとして機能するそれぞれの仮想マシンの構成情報を示す．

- Kubernetes クラスタのマスターノードとして機能する仮想マシンの構成情報
  - OS: Ubuntu Server 24.04.2 LTS
  - vCPU: 4 [Core]
  - RAM: 8 [GB]
  - SSD: 40 [GB]
- Kubernetes クラスタのワーカーノードとして機能する仮想マシンの構成情報
  - OS: Ubuntu Server 24.04.2 LTS
  - vCPU: 4 [Core]
  - RAM: 8 [GB]
  - SSD: 40 [GB]

6. 議論

提案では，障害によって影響を受ける人数を，アラートが通知された監視対象であるサーバの SSH の接続ユーザ数のみで取得している．一方で，障害によって影響を受ける人数は，サーバを使用するユーザが関係する外的なイベントによって変動するため，改善の余地がある．例えば CDSL では，テクニカルレポートの提出や卒業論文の提出が終わっていない学生は，提出が終わっている学生と比較して実験でサーバを使用する機会が多くなるが，障害によって実験ができないことで影響を受ける．解決策として，テクニカルレポートの提出が終わっていない学生が使用する Ubuntu Server の SSH の接続ユーザ数で並び替えを行ったあと，テクニカルレポートの提出が終わっていない学生の Ubuntu Server がインストールされている仮想マシンが配置されているサーバで通知されたアラートのチケットのみを抜粋し，リストの先頭に配置する．テクニカルレポートの提出が終わっていない学生を特定する方法は，あらかじめ Ubuntu Server にログインするユーザ ID と学生の氏名の紐付けを記録しておき，CDSL の Web サイトで公開されるテクニカルレポートの著者名を抽出し，テクニカルレポートの提出が終わった学生のユーザ ID と照合することで特定する．

提案では，アラートが通知された監視対象であるサーバの SSH の接続ユーザ数を，障害によって影響を受ける人数として取得している．一方で，アラートが通知された日時には SSH の接続を行っていなかったが，その前まで複数のユーザが SSH の接続を行っていた場合，その接続ユーザ数をカウントしていないため 0 になる．解決策として，直近のアラートが解決された通知の日時から，1 次チケットが発行された日時までの SSH の接続ユーザ数を取得し，これをその期間の分単位の時間で除算した値をスコアとする．この解決策は，以前にアラートの通知があり，それが解決されていることを前提としている．このアラートが解決された通知の日時を起点とする期間にする理由は，取得する期間を監視対象の Ubuntu Server ごとに揃えるためである．また分単位の時間で除算する理由は，ログイン日時の記録を表示する last コマンドの最小粒度が分単位だからである．

7. おわりに

CDSL では，ESXi がインストールされている 10 台のサーバが稼働していることを確認するために監視システムを導入している．監視システムがアラートを作成すると，その内容をもとにチケットが作成される．1 次チケットの担当者として割り当てられた学生は，調査を行う．課題は，1 次チケットが発行されてから 1 次チケットの担当者が 1 次調査を開始するまでに時間がかかることである．理由は，調査が行われていない 1 次チケットのうち，どのチケットから調査をすべきなのかについて明記したルールがなく，発行日時が古いチケットの調査が遅れることがあったためである．基礎実験では，1 次チケットが発行されてから 1 次チケットの担当者が 1 次調査を開始するまでの時間を計測した．1 次チケットは全 28 件あり，調査が開始された 1 次チケットは 26 件である．そのうち，0 分から 1440 分のチケットが 5 件，1440 分から 2880 分のチケットが 1 件，2880 件から 4320 分のチケットが 2 件，4320 分から 5760 分のチケットが 2 件，5760 分から 7200 分のチケットが 1 件，7200 分から 8640 分のチケットが 0 件，8640 分から 10080 分のチケットが 2 件，10080 分以上が経過しているチケットが 13 件である．1 次調査の開始までに 1 週間以上

© 2025 Cloud and Distributed Systems Laboratory

が経過していたチケットがあり，調査が迅速に行われていないといえる．提案の対象は，ICMP パケットによるサーバとの疎通確認に失敗したことを示すアラートと，ノードのメトリクスが異常であることを示すアラートのチケットである．提案では，アラートが通知された監視対象であるサーバに対して ICMP パケットによる疎通確認を行い，返答がない場合は「疎通不可」，返答がある場合は「疎通可能」の 2 種類のラベルのいずれかをチケットに付与する．同時に，1 次チケットが発行された時刻におけるサーバの SSH の接続ユーザ数をチケットのスコアとする．ESXi をハイパーバイザーとするサーバからアラートが通知されている場合は，ESXi 上の仮想マシンにインストールされている Ubuntu Server の SSH の接続ユーザ数をそれぞれ計測し，合計した値をチケットのスコアとする．その後，スコアが最も高いチケットを 1 次チケットの担当者に通知する．スコアが同じチケットがある場合は，ラベルにもとづいて並び替えられる．

謝辞 調査と対応の作業にご協力いただいた東京工科大学コンピュータサイエンス学部の有田海斗さん，岡田京太郎さん，坂井萌桜さん，佐藤健斗さん，山崎拓海さんに御礼申し上げます．

参考文献

[1] Arifiansyah, F.: Analyzing Systemic Failures in IT Incident Management: Insights from Post-Mortem Analysis, *Eduvest - Journal of Universal Studies*, Vol. 5, pp. 5522–5535 (online), DOI: 10.59188/eduvest.v5i5.51192 (2025).

[2] Amanullah Bashir, T. R. S.: Comparative Study on Incident Management, *International Journal of Applied Information Systems*, Vol. 3, No. 2, pp. 21–23 (online), DOI: http:/ijais12-450467 (2012).

[3] Salah, S., Maciá-Fernández, G., Díaz-Verdejo, J. E. and Sánchez-Casado, L.: A model for incident tickets correlation in network management, *Journal of Network and Systems Management*, Vol. 24, pp. 57–91 (2016).

[4] Vinnakota, K. and Kolla, M.: Creating Effective Alerts for Monitoring Distributed Systems.

[5] Huang, H., Jennings III, R. B., Ruan, Y., Sahoo, R. K., Sahu, S. and Shaikh, A.: PDA: A Tool for Automated Problem Determination., *LISA*, Vol. 7, pp. 1–14 (2007).

[6] SPADACCINI, A. and GULIANI, K.: Being an On-Call Engineer, *Operating Systems and Sysadmin*, p. 43.

[7] Yamaoka, H., Yamamoto, K., Nagai, T. and Masuda, H.: Case Study of Implementing an IT Service Desk Ticketing System at Small Computer Center, *Proceedings of the 2019 ACM SIGUCCS Annual Conference*, SIGUCCS '19, New York, NY, USA, Association for Computing Machinery, p. 140–144 (online), DOI: 10.1145/3347709.3347820 (2019).

[8] BRIGUGLIO PELLEGRINO, R. A., Xompero, M., Riva, M., Selmi, C., Urban, C., AZZAROLI, N., OGGIONI, L., SCALERA, M. A., RICCARDI, A., BALESTRA, A. et al.: Be social, be agile: team engagement with Redmine, SPIE, The International Society for Optical Engineering (2022).

[9] Panaite, D. C.: Evaluating the performance of eBPF-based security software in a virtualized 5G cluster, PhD Thesis, Politecnico di Torino (2024).

[10] Pai, K. and Srinivas, B.: Enhanced Visibility for Real-time Monitoring and Alerting in Kubernetes by Integrating Prometheus, Grafana, Loki, and Alerta, *International Journal of Scientific Research in Engineering and Management* (2024).

[11] Rai, V. and Chandak, P.: Shift Planning and Scheduling For IT Service Operations Management, (online), DOI: 10.1109/SYSCON.2015.7116824 (2015).

[12] Yang, Y., Deng, Y., Xiong, Y., Li, B., Xu, H. and Cheng, P.: AidAI: Automated Incident Diagnosis for AI Workloads in the Cloud, *arXiv preprint arXiv:2506.01481* (2025).

[13] Winkler, S., Sonnberger, H. and Bodendorf, F.: Escalation Management for the Monitoring of Critical Process and Project Activities, *2008 International Conference on Computational Intelligence for Modelling Control Automation*, pp. 889–894 (online), DOI: 10.1109/CIMCA.2008.141 (2008).

© 2025 Cloud and Distributed Systems Laboratory