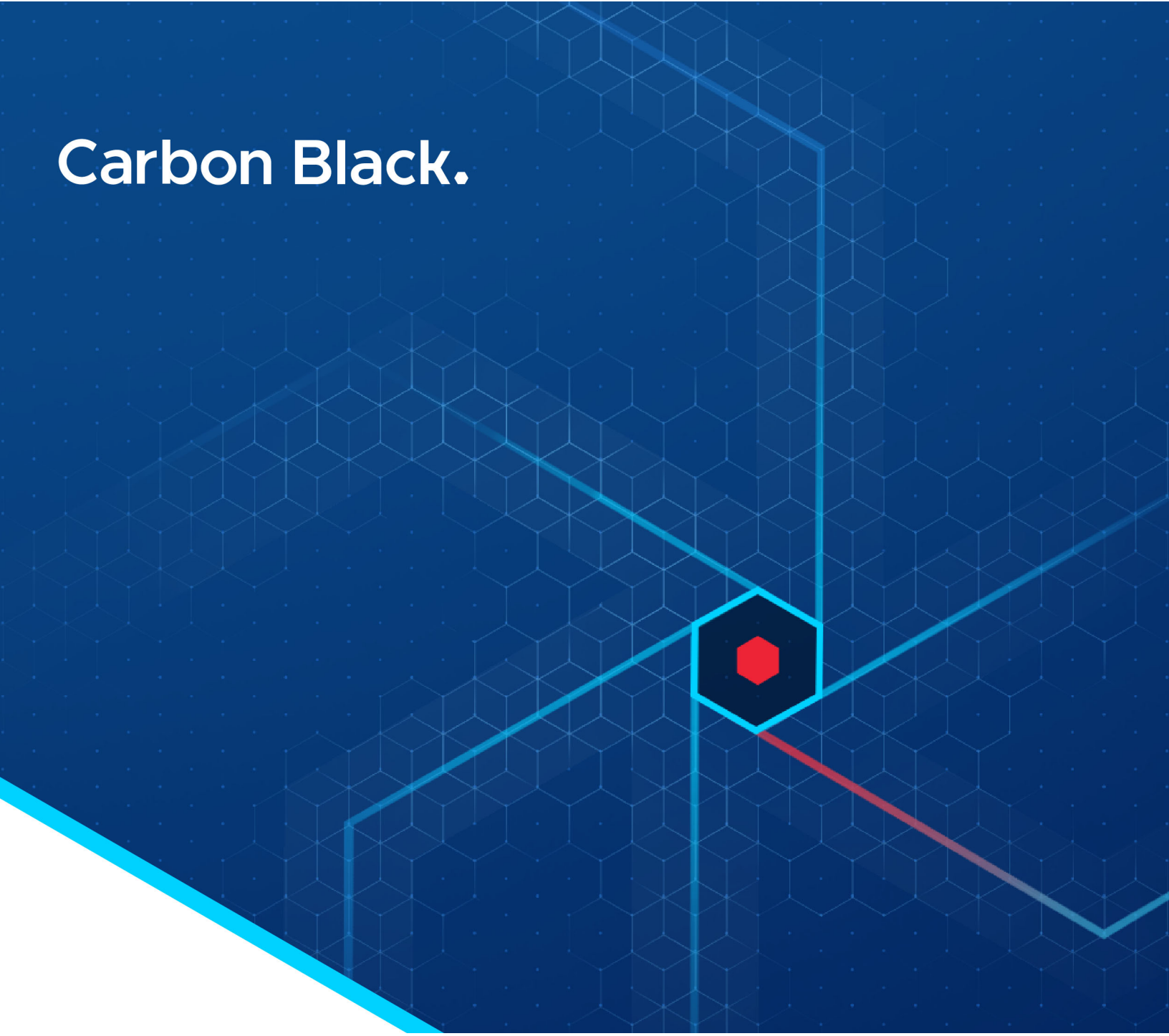
Carbon Black.



Cb Response サーバー / クラスター管理ガイド

サーバーのバージョン :6.2.3

ドキュメント日付 : 2018 年 8 月

著作権表示

Carbon Black, Inc.
1100 Winter Street, Waltham, MA 02451 USA
電話 : 617.393.7400
FAX: 617.393.7499
E メール : support@carbonblack.com
Web: http://www.carbonblack.com

Cb Response サーバー / クラスター管理ガイド
製品バージョン : 6.2.3
ドキュメント改定日付 : 10 31,2018

始める前に

この序文では、『*Cb Response 6.2.3 サーバー / クラスター管理ガイド*』について簡単に紹介します。

セクション

トピック	ページ
本文書の対象範囲	6
その他のドキュメント	6
コミュニティ リソース	7
サポートへのお問い合わせ	8

本文書の対象範囲

本文書では、Cb Response サーバーおよび Cb Response クラスターの管理方法について説明します。本ガイドの内容の要約を次の表に示します。

	章	説明
1	サーバーの概要	Cb Response サーバーのテクノロジ スタック、デーモン、構成、およびログについて概説します。
2	Cb Response サーバーのインストール	新しい Cb Response サーバーのインストールと初期化の方法、および既存の Cb Response サーバーのアップグレード、トラブルシューティング、アンインストールの方法について説明します。
3	サーバーのバックアップと復元	バックアップと復元のさまざまな手順を実行する方法について説明します。
4	ポートとプロトコル	さまざまなサーバー通信に使用されるポートおよびプロトコルの詳細情報が表にまとめられています。
5	Cb Response クラスターのインストール	Cb Response クラスターについて概説するとともに、クラスターの構成方法、既存のクラスターにミニオンを追加する方法、クラスターからミニオン ノードを削除する方法、およびクラスター ノードをアップグレードする方法について説明します。
6	非ルート ユーザーとして CBCLUSTER を使用する	非ルート ユーザーとして CBCLUSTER コマンドを使用する方法について説明します。

その他のドキュメント

このガイドで取り上げられていないタスクに関するドキュメントや、テクニカル サポート ソリューション用のナレッジ ベースとして保管されているドキュメントをお探しの場合は、Carbon Black User eXchange Web サイト (https://community.carbonblack.com) をご覧ください。ドキュメントには、新規リリース版のビルドごとに更新されるものもあれば、マイナー バージョンまたはメジャー バージョンの変更時にのみ更新されるものもあります。このサイトにあるドキュメントは次のとおりです。

- 『Cb Response リリース ノート』– 新機能と変更された機能、このリリースで解決された問題と全般的な改善点、および既知の問題と制限事項に関する情報が記載されています。また、サーバーをインストールする前の必須または推奨の準備手順についても記載されています。
- 『Cb Response 運用環境の要件 (OER)』– Cb Response サーバーの展開におけるパフォーマンスおよびスケーラビリティの考慮事項について説明しています。以前のリリースでは、『Server Sizing Guide』という書名でした。
- 『Cb Response Server Configuration Guide (cb.conf)』– Cb Response サーバー設定ファイル (`cb.conf`) のオプション、概要、パラメーターなどについて説明しています。

- 『Cb Response サーバー / クラスター管理ガイド』– (本文書) Cb Response サーバー / クラスターのインストール、管理、バックアップ、復元などの方法を説明しています。このガイドは、オンプレミスにインストールされた Cb Response のみを対象にしています。
- 『Cb Response ユーザー ガイド』– Cb Response 製品とその全機能の使い方、および管理タスクの実施方法を説明しています。
- 『Cb Response Unified View User Guide』– Cb Response 統合ビューのインストール方法および管理方法について説明しています。
- 『Cb Response Integration Guide』– Cb Protection、EMET、VDI、SSO などのさまざまなツールと Cb Response との統合を行う管理者向けの情報が記載されています。
- 『Cb Response API』– Cb Response REST API のドキュメントは、https://developer.carbonblack.com/reference/enterprise-response にあります。REST API へのアクセスを容易にする Python モジュールについてのドキュメントは、https://cbapi.readthedocs.io にあります。
- *『Cb Response Connectors』* – サードパーティ製品と Cb Response サーバーの通信を実現するには、コネクターが必要です。さまざまな Carbon Black コネクターの取り付け方法、構成方法、および管理方法を説明したドキュメントは、以下のアドレスに用意されています。https://developer.carbonblack.com/guide/enterprise-response/#connectors.

コミュニティ リソース

Carbon Black User eXchange の Web サイト (https://community.carbonblack.com) では、Carbon Black のユーザー、社員、およびパートナーが共有する情報にアクセスできます。このサイトでは、Carbon Black 全製品のユーザー向け情報の閲覧やコミュニティへの参加が可能です。

このリソースにログインすると、次のことができます。

- 他のユーザーに質問したり、他のユーザーの質問に回答したりする。
- " 投票 " によって製品アイデアのステータスを格上げする。
- 最新のユーザー ドキュメントをダウンロードする。
- Carbon Black 開発者コミュニティに参加し、アイデアや解決策を投稿したり、他のユーザーの投稿について話し合ったりする。
- Carbon Black 製品で利用可能なトレーニング リソースを見る。

User eXchange にアクセスするにはログイン アカウントが必要です。アカウントが必要な場合は、テクニカル サポートの担当者にお問い合わせください。

サポートへのお問い合わせ

Carbon Black テクニカル サポートでは、複数の方法でお客様からのお問い合わせを受け付けています。

テクニカル サポートへの問い合わせ方法
Carbon Black User eXchange: https://community.carbonblack.com
E メール : support@carbonblack.com
電話 : 877.248.9098
Fax: 617.393.7499

問題の報告

テクニカル サポートに電話または E メールで連絡する際は、サポート担当者に以下の情報を提供してください。

必要な情報	説明
連絡先	名前、会社名、電話番号、E メール アドレス
製品バージョン	製品名およびバージョン番号
ハードウェア構成	製品を実行するサーバーまたはコンピューターのハードウェア構成 (プロセッサー、メモリ、および RAM)
ドキュメント バージョン	ドキュメントに関する問題の場合は、使用しているドキュメントのバージョンを指定してください。ドキュメントの日付とバージョンは表紙に記載されています。分量が多いドキュメントの場合は、「著作権表示」セクションの後に記載されています。
問題	問題の原因となったアクション、返されたエラー メッセージ、その他の該当する出力
問題の深刻度	重大、深刻、マイナー、または改善

内容

著作権表示 2

始める前に 5
本文書の対象範囲 6
その他のドキュメント 6
コミュニティ リソース 7
サポートへのお問い合わせ 8
問題の報告 8

1 サーバーの概要 2
サーバーの概要 3
サーバーの構成 6
サーバー ログ 7
ログの概要 7
トラブルシューティング 7
Cb Response コンソール インターフェイスでのエラー 7
チェックインしていないセンサー 7
すべてが正常に機能していることの確認 8

2 Cb Response サーバーのインストール 9
概要 10
ファイアウォールと接続の要件 10
新規 Cb Response サーバーのインストールと初期化 11
Cb Response サーバーのアップグレード 21
cbupgrade の使用 22
サーバーのアップグレードと新しいセンサー バージョン 22
イベント データに対する複数ボリュームのサポート 23
命名規則 23
新しいデータ ディレクトリの使用 24
パーティションの作成 24
アクティブな読み取り専用ディレクトリ 25
パーティションの削除 25
オンザフライによるディスク容量の拡張 26
サーバーのトラブルシューティング 26
Cb Response サーバーのアンインストール 28
Cb Response サーバーの削除 29

3 サーバーのバックアップと復元 31
概要 32
復元先のサーバー 32
バックアップ / 復元スクリプト 32
バックアップ 33
構成バックアップ 33

データ バックアップ 36
復元 37
失敗したミニオン クラスターの復元 37
構成の復元 38
データの復元 41

4 ポートとプロトコル 43

5 Cb Response クラスターのインストール 47
概要 48
クラスターのアーキテクチャ 48
クラスターの動作 50
Cb Response クラスターの構成 52
センサーのインストールと検証 54
ベスト プラクティス 55
既存のクラスターへのミニオンの追加 55
既存のクラスターからのミニオンの削除 56
ベスト プラクティス 56
読み取り専用ミニオン 56
ミニオンの削除 57
クラスター ノードのアップグレード 58
クラスター ノードの手動アップグレード 59

6 非ルート ユーザーとして CBCLUSTER を使用する 61
概要 62
必要なユーザー権限 62
ユーザーの定義 64

タスク リスト

タスクの実行手順 ..

このセクションでは、このガイドでステップ バイ ステップのタスクとして説明されている手順の一覧を示します。一部のワンステップのタスクや、文章による説明の方が理解しやすいタスクは、ここには記載されていません。お探しのタスクが一覧にない場合は、メイン内容 ページを参照してください。

symlink された場所をイベント ストレージに使用する : 24
クラスター ノードをアップグレードする : 58
クラスターのミニオン サーバーを手動でアップグレードする : 59
クラスターのミニオンを削除する : 57
クラスター構成を設定する : 52
クラスター化したサーバーをアップグレードする : 21
サーバーをアンインストールする (Cb Response RPM のみ): 29
サーバーをアンインストールする (CB Response と関連の RPM): 30
サーバーをインストールし、それ以外は動作しているクラスター上の失敗したミニオンでデータを復元する : 37
スタンドアロンのサーバーをアップグレードする : 21
センサーをインストールし、Cb Response で表示されることを確認する : 54
データ バックアップを実行する : 36
データの復元を実行する : 42
ミニオンの読み取り専用のマークを解除する : 57
ミニオンを読み取り専用としてマークする : 57
構成の復元を実行する : 39
構成バックアップを実行する : 33
既存のクラスターにミニオンを追加する : 55
新しいサーバーをインストールおよび初期化する : 11

第 1 章

サーバーの概要

この章では、Cb Response サーバーのテクノロジ スタック、デーモン、構成、およびログについて概説します。

セクション

トピック	ページ
サーバーの概要	13
サーバーの構成	16
サーバー ログ	17

サーバーの概要

このセクションでは、Cb Response サーバーのテクノロジ スタックについて説明します。以下の表に示すとおり、Cb Response サーバーには 5 つの重要なデーモンがあります。

デーモン	説明
`cb-nginx`	内部デーモンへの HTTP リバース プロキシとして使用します。
`cb-coreservices`	(Python、Gunicorn) HTTP トランザクションを対象とした、データ以外のすべてのアプリケーション ロジック。
`cb-datastore`	(Java/Jetty) イベント ログやバイナリ ファイルなどのすべての入力データ。
`cb-solr`	(Java/Jetty) プライマリ データ ストアである Apache Solr。
`cb-postgres`	従来型のリレーショナル データベース。
`cb-sensorservices`	センサーのチェックイン、登録、アップグレードなど、データ以外のすべてのセンサー要求を処理します。

`nginx` は、パブリックなソケットを備えた唯一のデーモンです。他のデーモンは、デフォルトの IP アドレスである `127.0.0.1` を使用して Cb Response サーバーにバインドされています。これらのデーモンにアクセスするには、ローカルでのアクセスとするか、 `nginx` リバース プロキシを使用する以外にありません。

`nginx` は、`tcp/80` と `tcp/443` の使用権限を持っており、以下の表に示すように、URL の接頭辞に基づいて、`coreservices`、`cb-datastore`、`cb-sensorservices`、または Cb Response の web `root:q` にリダイレクトされます。

nginx	リダイレクト先
`/`	`/var/www/cb/`
`/api/*`	`tcp/5000` 上の `coreservices`
`/sensor/*`	`tcp/6500` および `6501` 上の `sensorservices`
`/data/*`	`tcp/9000` 上の `cb-datastore`

注意

`/api/*` は **coreservices** で処理されます。

`/api/` を接頭辞とするすべての URL は、Cb Response コンソール インターフェイスおよび REST クライアントで使用します。

`/sensor/*` は **sensorservices** で処理されます。

`/sensor/` を接頭辞とするすべての URL は、データをプッシュするセンサーで使用します。これらの URL は相互に分離されていて、nginx サーバーの独立したインスタンスを `tcp/443` にバインドできるようになっています。このバインドは、内部ネットワークの外部にあるセンサーのパブリック /DMZ インターフェイス上で実現できます。このようなセンサーとして、出張中の社員が使用しているノートパソコン上のセンサーや、社員の自宅にある作業用ノートパソコン上のセンサーなどが考えられます。この状態で、外部からは `/api/` インターフェイスにアクセスできないようになっています。`nginx` の構成の簡単な変更で、これらの URL を分離できます。以下のファイルにその例が示されています。

`/etc/cb/nginx/conf.d/cb-multihome.conf.example`

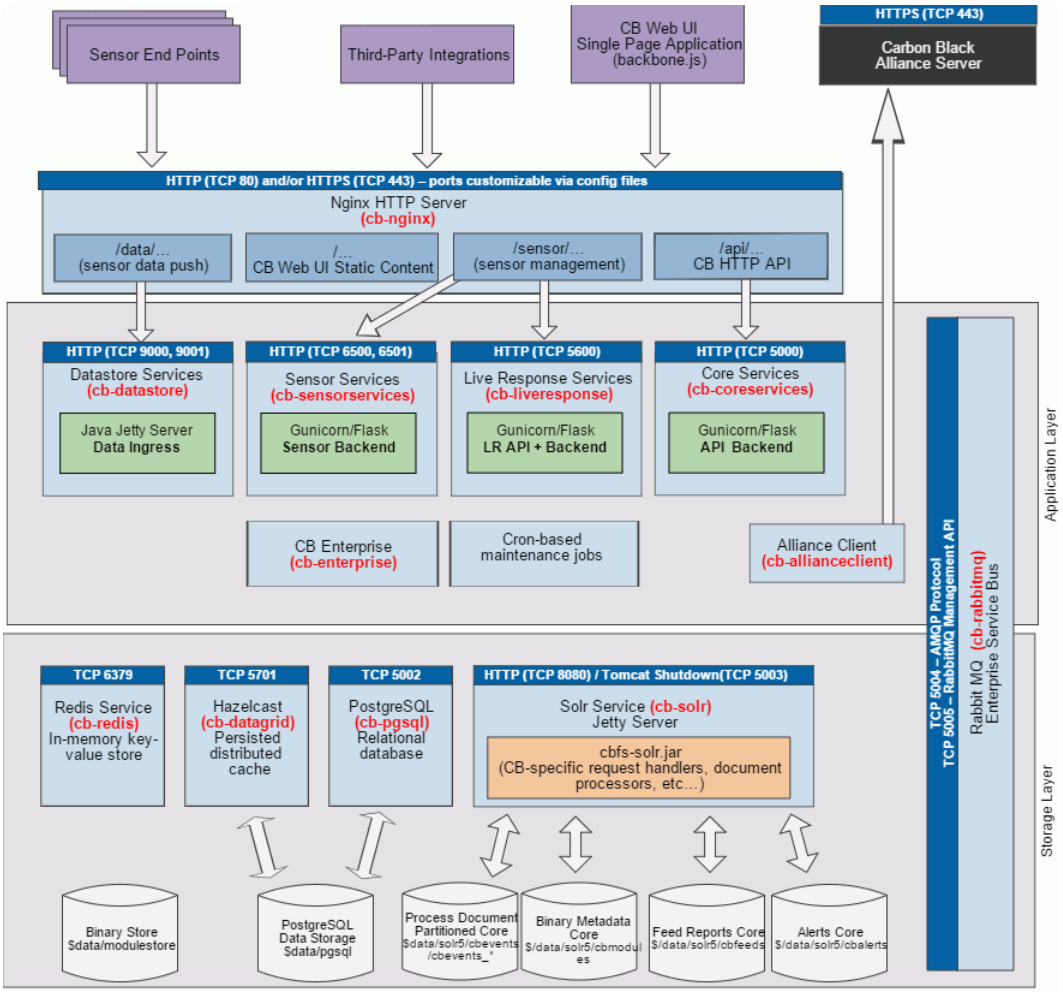
注意

クラスター化したセットアップでは、リッスン ポートの構成が異なります。詳細については、クラスター専用のドキュメントを参照してください。

一般的に、センサーは登録処理を経たうえで、`nginx` を使用して `sensorservices` にチェックインします。チェックインした後でデータを取得したセンサーは、`nginx` を使用して `cb-datastore` にイベント ログを送信します。

`cb-datastore` では、データが数分間キャッシュされた後、互いに関連するデータの集合が `cb-solr` に送信されます。

以下の図は、Cb Response サーバーのアーキテクチャを概略的に示したものです。



サーバーの構成

デーモンの構成データは通常、静的であるため、一般的な Linux の規則に従ったフラット ファイルに保存されます。動的な実行時構成データは PostgreSQL に保存されます。このデータを構成するには Cb Response コンソールを使用します。

主な構成はそのほとんどが、Cb Response サーバーをインストールした後、`cbinit` スクリプトを実行することで設定されます。`cbinit` では、静的構成ファイルと PostgreSQL の両方を組み合わせて初期設定が構成されます。Cb Response サーバーのインストールと構成の詳細については、第 2 章 「Cb Response サーバーのインストール」を参照してください。

以下の表は、主な静的構成ファイルを示したものです。

静的構成ファイル	説明
`/etc/cb/cb.conf`	Cb Response の主な全社規模設定。
`/etc/cb/solr5/core_conf`	このディレクトリには、Apache Solr の設定が保存されます。それぞれのサブディレクトリに、Solr コアの構成が個別に保存されます。
`/var/cb/data/pgsql/postgresql.conf`	PostgreSQL の設定。
`/etc/cb/coreservices-logger.conf`	Python の `coreservices` デーモン向けログ設定の構成。
`/etc/cb/sensorservices-logger.conf`	Python の `sensorservices` デーモン向けログ設定の構成。
`/etc/cb/cb-datastore/*`	Java の `cb-datastore` デーモン向けログ設定の構成。
`/etc/cb/nginx/conf.d/cb.conf`	リスナーにバインドする IP アドレスとポートを定義するための nginx サーバー設定。
`/etc/sysconfig/iptables`	サーバーのファイアウォール向けの標準的な iptables 構成 (CentOS 6 向け)。

以下の表は、副次的な静的構成ファイルを示したものです。

静的構成ファイル	説明
`/etc/cb/allianceclient-logger.conf`	Cb Response Alliance クライアント デーモン向けログ設定の構成。
`/etc/cb/nginx/cb-nginx.conf`	Cb Response に固有の標準的な nginx サーバー構成。
`/etc/cb/solr5/solr.in.sh`	Solr 用 Jetty サーブレット設定。

サーバー ログ

ログの概要

Cb Response サーバーではログ ファイルを多用します。これらのファイルには、システムが正常に機能していることを示す記録とエラー条件下で発生したアクティビティの記録が収められています。

以下の表は、Cb Response サーバー上の重要なログ ファイルをまとめたものです。

ログ	説明
`/var/log/cb/nginx/access.log` (および `error.log`)	すべてのセンサーと API トラフィックについての `nginx` の HTTP アクセスとエラーのログ。
`/var/log/cb/coreservices/debug.log`	API トラフィックのアプリケーション ロジック。
`/var/log/cb/sensorservices/debug.log`	センサー トラフィックのアプリケーション ロジック。
`/var/log/cb/datastore/debug.log`	受信したセンサー データのキャッシュ。
`/var/log/cb/solr/debug.log`	センサー データの保存、インデックス作成、およびクエリ。

トラブルシューティング

Cb Response コンソール インターフェイスでのエラー

ログ ファイル `/var/log/cb/coreservices/debug.log` で、Python のスタック トレースを詳しく確認します。詳細については、Carbon Black のテクニカル サポート担当者までお問い合わせください。

チェックインしていないセンサー

チェックインしていないセンサーが見つかった場合は、ログ ファイル `/var/log/cb/nginx/access.log` で、該当のホストからの要求を確認します。以下に例を示します。

```
164.230.214.13 - - [20/Apr/2017:20:04:52 +0000(3.811)] "POST /
sensor/checkin/35998 HTTP/1.1" 502 166 "-" "sensors.vibrant-
pies.my.carbonblack.io" ">170.16.20.21:6501" "-" "-"
```

この出力例では、いくつかのフィールドが赤色で強調表示されていますが、これらは問題を診断するうえで有用なフィールドです。

- チェックイン フィールドの後に表示されているのはセンサー ID です。上記の例では、**35998** となっています。
- HTTP/1.1 という文字列に続くフィールドに表示されているのは、実際のエラー コードです。上記の例では、**502** がエラー コードです。

- クラスター化された環境では、一部の要求によってクラスター内の別のミニオンに呼び出しがプロキシされます。このミニオンのアドレスは、">" という文字の後に表示されます (上記の例では **170.16.20.21**)。

チェックイン エラー、または接頭辞が "/sensor/" であるその他すべての呼び出しに対するエラーが検出された場合は、以下のログでエラーの詳細を確認します。

```
/var/log/cb/sensorservices/debug.log
```

接頭辞が "/data/" である要求についてのエラーが検出された場合は、以下のログを確認します。

```
/var/log/cb/datastore/debug.log
```

クラスター環境では、nginx のエラー ログ エントリで参照されているノード上のログ ファイルを確認する必要があります。

代替手段

センサーがチェックインしていないにもかかわらず `access.log` にエントリがない場合は、 `error.log` を確認します。

センサーの SSL クライアント証明書 (`/etc/cb/certs`) に認証局 (CA) の署名がなく、`/etc/cb/nginx/conf.d/cb.conf` での構成も行われていない場合、`nginx` により要求が拒否されます。

error.log にエントリがない場合は、『Cb Response ユーザー ガイド』の 「センサーのトラブルシューティング」というセクションにある説明に従ってセンサーの通信ステータスを確認します。

すべてが正常に機能していることの確認

`nginx` の `access.log` に HTTP 応答コードとして '`200`' が記録されていることを確認します。このコード `200` が記録されていれば、通信が正常に機能しています。

`/var/log/cb/solr/debug.log` で、 `/update` ハンドラーに対する要求があるかどうかを確認することもできます。以下に例を示します。

```
INFO:[cbevents] webapp=/solr path=/update
params={wt=javabin&version=2} {add=[a2247de5-fa3b-7b80-0000-
000000000001(1448549632617480192), b6879e21-eff8-dbad-0000-
000000000001 (1448549632695074816), 3abdd29a-018b-3037-0000-
000000000001 (1448549632700317696), f405c732-b898-bedd-0000-
000000000001 (1448549632705560576), ... (203 adds)]} 0 20248
```

この例の中に表示されている "`203 adds`" は、1 つ以上のイベントで更新されたプロセスの数を示しています。これらの要求は、実稼働システム上で円滑、高速で実行できます。

第 2 章

Cb Response サーバーのインストール

この章では、新しい Cb Response サーバーのインストールと初期化の方法、および既存の Cb Response サーバーのアップグレード、トラブルシューティング、アンインストールの方法について説明します。

セクション

トピック	ページ
概要	20
新規 Cb Response サーバーのインストールと初期化	21
Cb Response サーバーのアップグレード	31
イベント データに対する複数ボリュームのサポート	33
サーバーのトラブルシューティング	36
Cb Response サーバーのアンインストール	38

概要

この章では、Cb Response サーバーのインストール手順を説明します。ここで取り上げるのは新規インストールとサーバーのアップグレードです。ダウンロード速度が妥当であれば、すべてのプロセスを 10 分ほどで完了できます。

別冊の『Cb Response Server Operating Environment Requirements guide』には、Cb Response サーバーに必要なハードウェアおよびソフトウェアについてのガイドラインが記載されています。インストール作業を開始する前に、使用環境がこれらの要件を満たしている必要があります。このガイドは Carbon Black User eXchange に用意されています。

Cb Response サーバーをインストールする主な手順は以下のとおりです。

1. Carbon Black から RPM を入手してインストールします。この RPM は Cb Response サーバーをインストールするものではありません。Yum リポジトリをセットアップし、SSL クライアント証明書をインストールするツールです。この証明書を使用することで、完全版の Cb Response サーバーをダウンロードしてインストールできます。
2. Cb Response サーバーをインストールします。この手順には、`yum install` コマンドの実行と `cbinit` 構成スクリプトの実行の 2 つのプロセスがあります。`yum install` コマンドを実行すると、Cb Response サーバーがダウンロードされます。

`cbinit` の詳細については、Carbon Black User eXchange にある 『Automating cbinit』を参照してください。

サーバーのインストールが完了すると、監視対象とするエンドポイントにセンサーをインストールすることができます。センサーのインストール手順およびアップグレード手順については、『Cb Response ユーザー ガイド』の「センサーの管理」という章を参照してください。

ファイアウォールと接続の要件

以下の表にあるシナリオの Cb Response サーバー システムでは、送信 TCP ポートを通じたインターネット接続が必要です。

シナリオ	説明	アドレス
Cb Response Yum リポジトリ	RPM インストーラーで Yum リポジトリをセットアップします。	yum.carbonblack.com:443 yum.distro.carbonblack.io:443
Cb Response Alliance サーバーおよび Cb Threat Intel	Alliance サーバーと Cb Threat Intel は脅威インテリジェンスを提供し、Cb Threat Intel パートナーを通じてエンドポイントでファイルを詳しく解析できるようにします。 すべての脅威インテリジェンス データを確認するには、両方のアドレスが必要です。	api.alliance.carbonblack.com:443 threatintel.bit9.com:443
CentOS Yum リポジトリ	Cb Response サーバーのインストールで標準のパッケージをダウンロードするために使用した標準の CentOS Yum リポジトリ サーバー	mirror.centos.org:80

新規 Cb Response サーバーのインストールと初期化

このセクションでは、新規 Cb Response サーバーのインストールと初期化の手順について説明します。インストールと構成のプロセス全体を通じて、ルートレベルの権限が必要です。インストールと初期化のコマンドを入力するには、`su` または `sudo` を使用します。

警告

このセクションで取り上げる手順は、新規インストールのみを対象としています。Cb Response サーバーを既にインストール済みの場合は、**これらの手順を実行しないでください**。その場合は 「サーバーのアップグレードと新しいセンサー バージョン」(32 ページ) を参照してください。

既存のサーバーに対して新規インストールの手順を使用すると、すべてのデータが失われる可能性が高くなります。センサーから収集した構成データやイベント データも同様です。

新しいサーバーをインストールおよび初期化する :

1. Cb Response サーバーのインストール先とするホスト マシンが、Carbon Black の担当者から受け取った『Cb Response Server Operating Environment Requirements guide』で指定されているハードウェアとソフトウェアの要件を満たしていることを確認します。
2. サーバーが、「ファイアウォールと接続の要件」(21 ページ) で指定されているインターネット接続を備えていることを確認します。

3. Carbon Black テクニカル サポートに問い合わせ、Cb Response サーバーのインストール RPM を入手します。
4. 以下の手順で RPM をインストールします。
   a. 入手したお客様固有の RPM を使用して、以下のコマンドを実行します。
      `sudo rpm -ivh carbon-black-release-1.0.3-1-<customername>.x86_64.rpm`
   b. (オプション) Cb Response [cb] Yum リポジトリが正しく構成されたことを確認します。以下のコマンドを実行すると、Cb Response に関する新しい Yum リポジトリ エントリの内容を確認できます。
      `cat /etc/yum.repos.d/CarbonBlack.repo.`

```
[root@cb-enterprise-testing ~]# cat /etc/yum.repos.d/
CarbonBlack.repo

[CarbonBlack]
name=CarbonBlack
baseurl=https://yum.distro.carbonblack.io/enterprise/stable/
$releasever/$basearch/
gpgcheck=1
enabled=1
metadata_expire=60
sslverify=1
sslclientcert=/etc/cb/certs/carbonblack-alliance-client.crt
sslclientkey=/etc/cb/certs/carbonblack-alliance-client.key
```

   c. (オプション) 以下のディレクトリに Cb Response SSL の証明書とキーがあることを確認します。
      `/etc/cb/certs/`
5. 以下の手順で Cb Response サーバーをインストールします。
   a. 使用しているコンピューターの日時が正確に設定されていることを確認します。この日時の設定が正しくないと、Yum のダウンロードで必要となる SSL ネゴシエーションで問題が発生することがあります。
   b. 以下のコマンドを実行します。
      `sudo yum install cb-enterprise`

```
[bsmith@localhost yum.repos.d]$ sudo yum install cb-enterprise
```

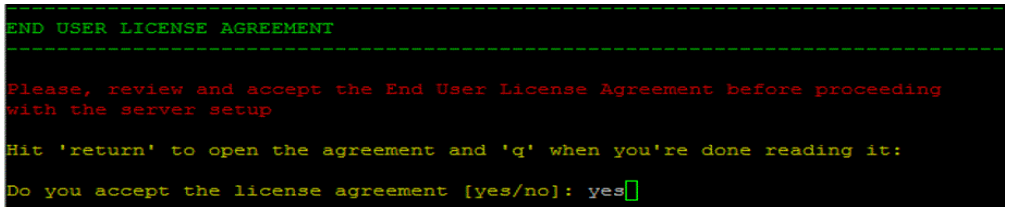
   c. CentOS GPG キーのインストールを指示された場合はそれをインストールします。
   d. 送信側ファイアウォール例外を必要とする環境では、「ファイアウォールと接続の要件」(21 ページ) に記述されている例外に従っていることを確認します。http://mirror.centos.org の baseurl が有効になるように `/etc/yum.repos.d/CentOS-Base.repo` を更新することも必要です。

      注意 : Yum では Web プロキシを使用できます。なお、NTLM 認証の Web プロキシとともに Yum を使用する方法について、Carbon Black は知識を持ち合わせていません。

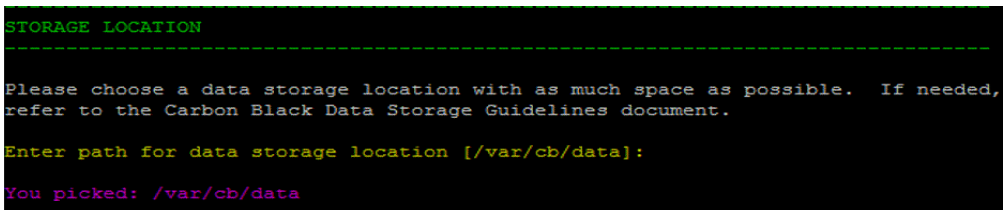
6. Cb Response サーバーのインストールが完了したら、その初期化と構成に進みます。
   a. 以下のコマンドを実行します。

      ```
      sudo /usr/share/cb/cbinit
      ```

   b. **Entrer** キーを押して EULA を表示します。その内容を確認した後、「`q`」を入力し、続いて 「`yes`」と入力します。

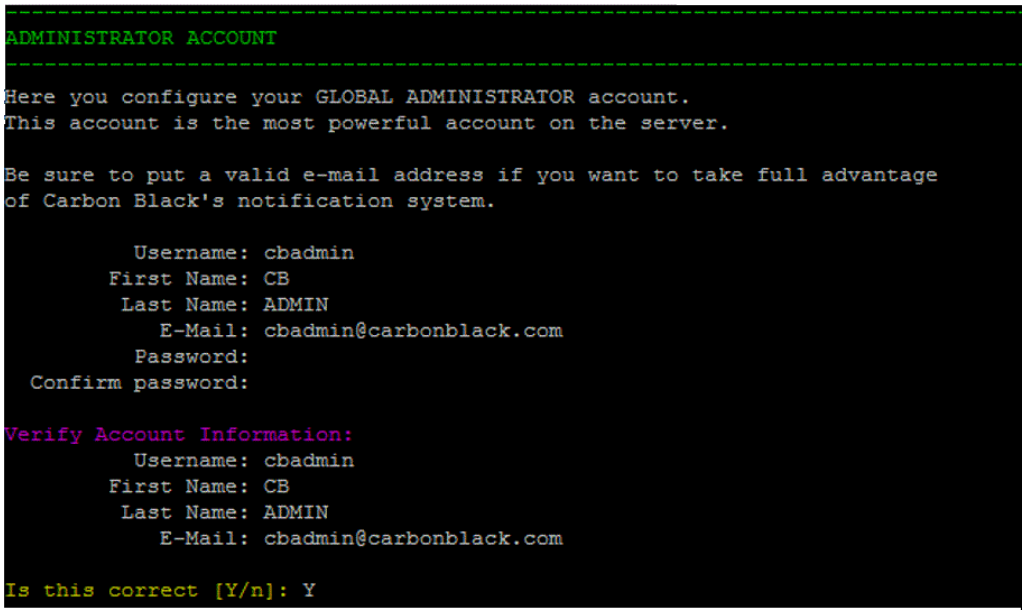


   c. データを保存するストレージの場所を選択して **Enter** キーを押します。



      注意：『Cb Response Operating Environment Requirements guide』に従い、プライマリ データストアはデフォルトで `/var/cb/data` にマッピングされます。この推奨事項に従ってストレージを構成していない場合は、Carbon Black のサポートまたはプロフェッショナル サービスを利用して、現在のファイル システムのマッピング (`df -h`) を確認します。ディスクの構成が正しくない場合や不十分な場合、Cb Response は正しく機能しません。

   d. 初期の管理者アカウントで Cb Response にログインし、その構成を開始します。**Username**、**First Name**、**Last Name**、**E-Mail**、**Password**、**Confirm Password** の各値を次のように入力します。



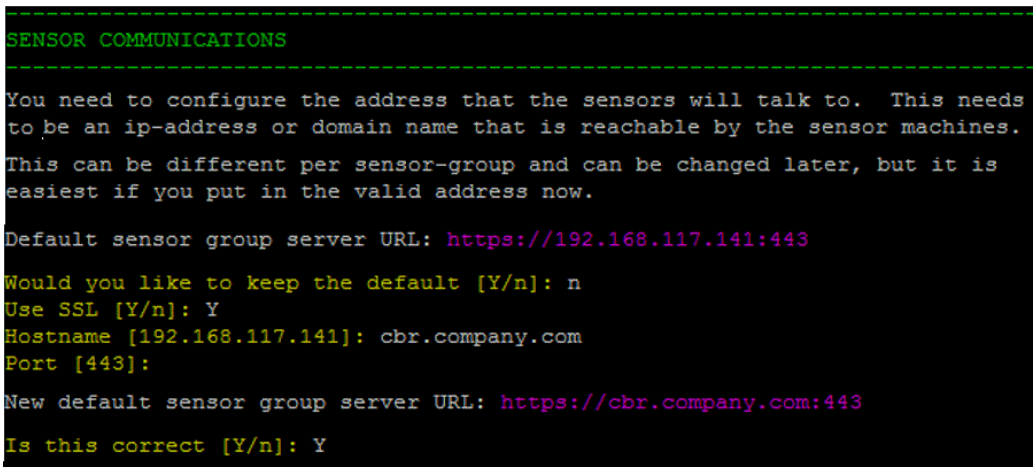
e. **Enter** キーを押してから 「y」を入力し、作成したアカウント情報を確認します。

f. **Sensor Communication** セクションで、Cb Response サーバーとの通信でセンサーが使用するアドレスを定義します。

```
Would you like to keep the default [Y/n]:n
Use SSL [Y/n]:Y
Hostname [192.168.117.141]:cbr.company.com
Port [443]:return

If the Verify Account Information looks correct, Y
```

注意 : サーバーの IP アドレスには、デフォルトの SSL ポート 433 を介してアクセスします。この IP アドレスを参照する DNS レコードを使用することをお勧めします。

Carbon Black サポートまたはプロフェッショナル サービスの支援を受けながら、Cb Response サーバーでサポートされている外部接続オプションを確実に理解してください。



g. システムからのすべての指示事項を確認し、各自のセキュリティ ポリシーに従って共有設定を構成します。この推奨設定を以下で説明します。Cb Response コンソールにアクセスし、右上隅の **[< ユーザー名 >] > [Sharing Settings (共有設定)]** の順に選択すれば、これらの設定をいつでも変更できます。

- `Do you want to enable communication with the Carbon Black Alliance?- Y`

  この設定により、Cb Threat Intel および Cb Threat Intel パートナーの広範なネットワークから得られる最新の脅威インテリジェンスでプログラムを補強できます。

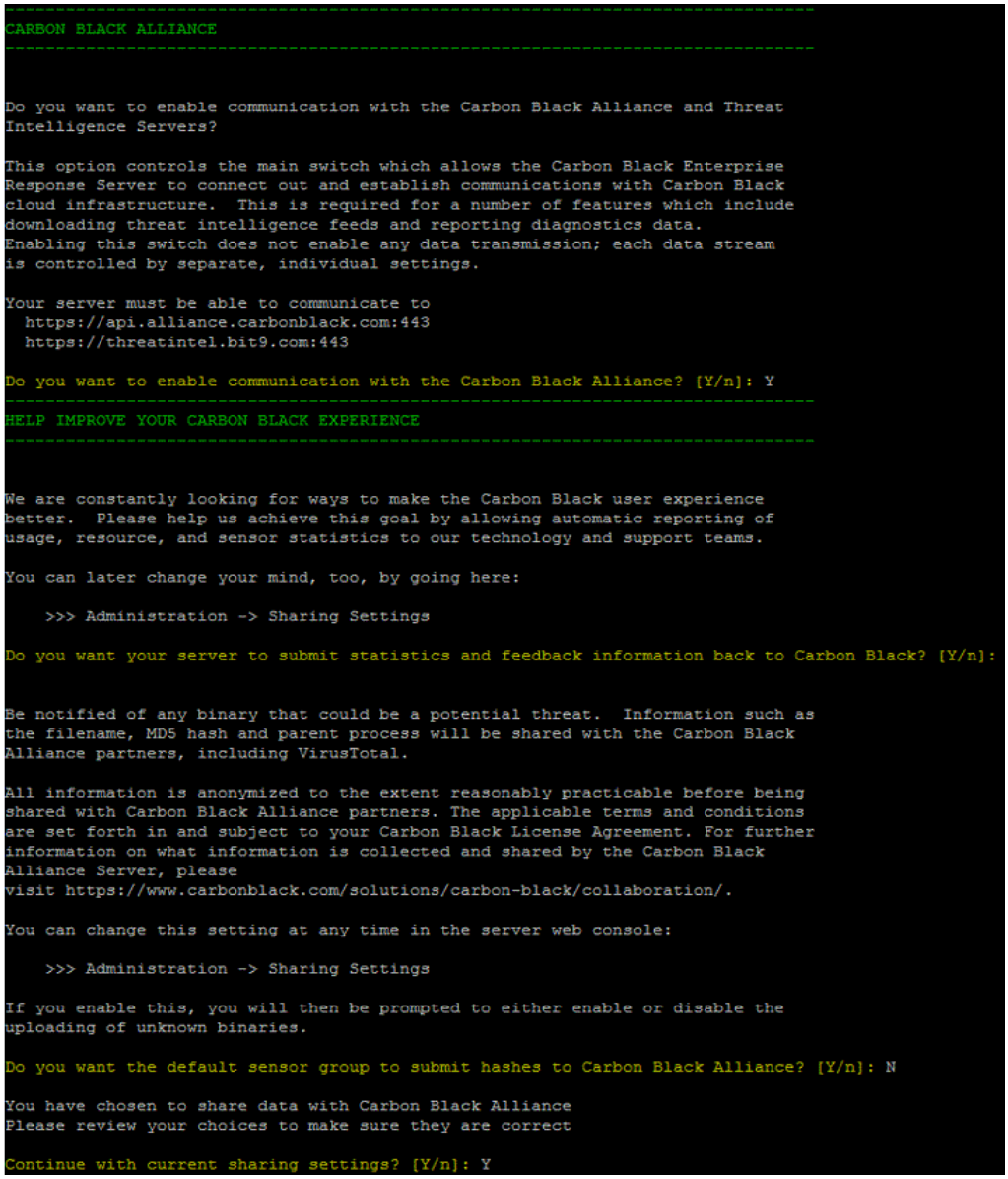
- `Do you want your server to submit statistics and feedback information to Carbon Black?- Y`

  この設定により、サーバーから Cb Response に正常性の統計情報を送信できます。Carbon Black サポートとプロフェッショナル サービスでは、これらの統計情報を使用して、割り当てられているサーバーが当社のアプリケーションの下でどのように動作しているかを判断できます。

- `Do you want the default sensor group to submit hashes to Carbon Black Alliance?- N`

  Cb Response とのハッシュの共有に関する詳細については、『Cb Response ユーザー ガイド』の 「脅威インテリジェンス フィード」という章を参照してください。

- Continue with current sharing settings?- Y



h. **SSL 証明書**のセクションは入力が自動化されているので、ユーザーによる入力は不要です。

注意 : これらの証明書は、各自の組織が発行した証明書に置き換えることができます。詳しい方法については、Carbon Black サポートまたはプロフェッショナル サービスにお問い合わせください。

以下のスクリプトを実行し、暗号化した証明書をバックアップとして作成します。障害復旧の際には、原本と正確に同一の証明書が不可欠です。

```
/usr/share/cb/cbssl backup --out <backup_file_name>
```

```
------------------------------------------------------------------------------
SECURITY - SSL CERTIFICATE GENERATION
------------------------------------------------------------------------------

Generating self-signed HTTPS Server certificate...

Generating self-signed HTTPS Sensor CA certificate...


Carbon Black Enterprise Response Server uses a SSL certificate to establish
secure communications between sensors and the server.

Should the certificate and/or its private key be lost, sensors will no longer
be able to communicate with the server.

We recommend backing up the SSL certificate files at this time by running:

  /usr/share/cb/cbssl backup --out <backup_file_name>

IMPORTANT: Backup file must be securely stored.  Anyone with access to the
information contained in that file will be able to compromise the security
of sensor-server communications and potentially compromise the security of
the computers on which the sensors run.


Continue [return]: []
```

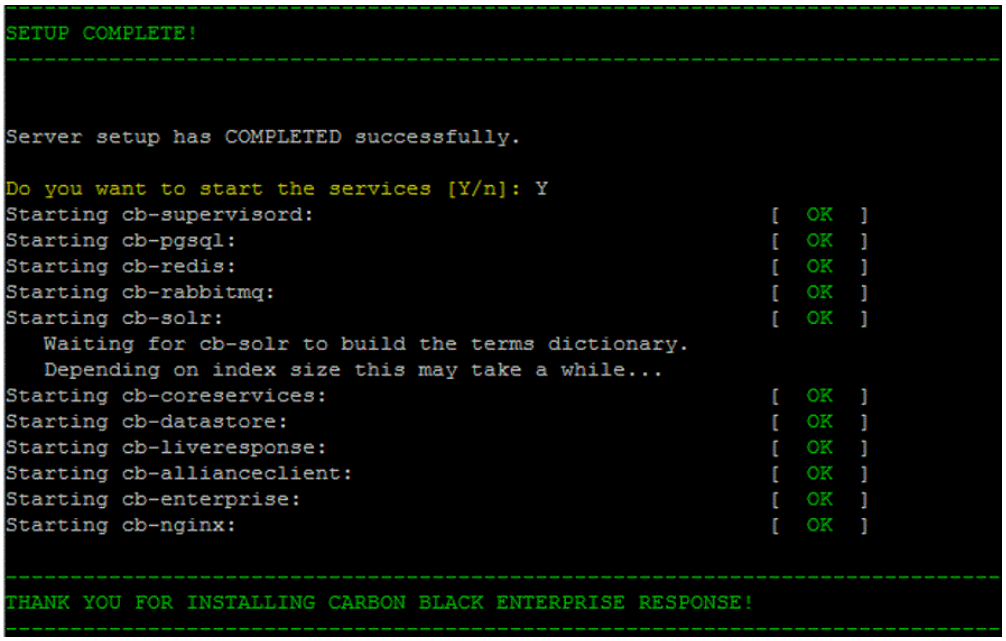
i. **IP テーブル**のセクションで Y と回答します。この指定により、サーバーの IP テーブルにあるポート 433 が開きます。

```
------------------------------------------------------------------------------
SECURITY - IPTABLES CONFIGURATION
------------------------------------------------------------------------------

Carbon Black Enterprise Response Server listens on a number of TCP/IP ports. If
iptables firewall is running on the host machine, iptables must be configured
to allow incoming connections on these ports.

To get a list iptables rules that need to be added to current host
configuration, you can run '/usr/share/cb/cbcheck iptables -l' at any time and
apply the rules manually.  Alternatively, server setup and configuration
tools can take over management of iptables configuration and apply updates
whenever they are needed.

Would you like Carbon Black Enterprise Response Server to manage iptables [Y/n]: Y
Applying iptables rules:
   -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 443 -j ACCEPT

iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]
```

j. **POSTGRESQL データベースのセットアップ**のセクションは入力が自動化されているので、ユーザーによる入力は不要です。

```
--------------------------------------------------------------------------------
SETTING UP POSTGRESQL DATABASE
--------------------------------------------------------------------------------

Initializing Carbon Black Server PostgreSQL Instance...

The files belonging to this database system will be owned by user "cb".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.UTF-8".
The default text search configuration will be set to "english".

Data page checksums are disabled.

creating directory /var/cb/data/pgsql ... ok
creating subdirectories ... ok
selecting default max_connections ... 100
selecting default shared_buffers ... 128MB
creating configuration files ... ok
creating template1 database in /var/cb/data/pgsql/base/1 ... ok
initializing pg_authid ... ok
setting password ... ok
initializing dependencies ... ok
creating system views ... ok
loading system objects' descriptions ... ok
creating collations ... ok
creating conversions ... ok
creating dictionaries ... ok
setting privileges on built-in objects ... ok
creating information schema ... ok
loading PL/pgSQL server-side language ... ok
vacuuming database template1 ... ok
copying template1 to template0 ... ok
copying template1 to postgres ... ok
syncing data to disk ... ok

Success. You can now start the database server using:

    /usr/pgsql-9.3/bin/postgres -D /var/cb/data/pgsql
or
    /usr/pgsql-9.3/bin/pg_ctl -D /var/cb/data/pgsql -l logfile start

waiting for server to start.... done
server started
Creating alliance model DB schema...
Creating core model DB schema...
waiting for server to shut down.... done
server stopped
```

k. **セットアップの完了**のセクションで 「Y」を入力するとサービスが開始されます。



注意 : センサーとサーバーとの通信が適切に機能していることを確認する :

1.Google Chrome を開き、次の URL を指定して、使用しているサーバーにアクセスします。

`https://<your_cber_server_url>`

2. センサーをダウンロードして、エンドポイントにインストールします。

センサーのインストールおよび管理に関する詳細については、『Cb Response ユーザー ガイド』の 「センサーの管理」という章を参照してください。

7. ファイアウォールを構成していない場合は、ここで構成します。ファイアウォールを構成するにはいくつもの方法があります。以下は、CentOS 6 の一例です。

   a. `cbinit` スクリプトで iptables を管理できるようにしていない場合はポート 443 を開きます。

```
[bsmith@localhost yum.repos.d]$ sudo vim /etc/sysconfig/iptables
     # システムによって構成されたファイアウォールのファイアウォール構成
     # このファイルを手動でカスタマイズすることはお勧めできません。
     *filter
     :INPUT ACCEPT [0:0]
     :FORWARD ACCEPT [0:0]
     :OUTPUT ACCEPT [0:0]
     -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
     -A INPUT -p icmp -j ACCEPT
     -A INPUT -i lo -j ACCEPT
     -A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
     -A INPUT -j REJECT --reject-with icmp-host-prohibited
     -A FORWARD -j REJECT --reject-with icmp-host-prohibited
     # Carbon Black の IP テーブルへの新規追加
     -A INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
     COMMIT
```

   b. (オプション) ポート 80 を開き、安全ではないチャネルを通じて Web インターフェイスとセンサーとの通信を使用できるようにします。この手順は必須ではなく、調査またはトラブルシューティングでのみ実施することをお勧めします。ポート 80 を介して Web インターフェイスに接続すると、ポート 443 にリダイレクトされます。

8. https://< 使用しているサーバーのアドレス >/ で Cb Response サーバーの Web インターフェイスにログインし、`cbinit` スクリプトでセットアップしたユーザー名とパスワードを使用します。

注意

このリリースでサポートされているブラウザーは Google Chrome のみです。正式にはサポートされていませんが、当社内のテストでは Firefox、Opera、IE10 以降でも動作しています。ただし、IE ブラウザーは互換モード以外のモードにする必要があります。ブラウザーと同じサブネットにあるサーバーには自動的にそのモードで接続されます。

Cb Response サーバーをインストールし、構成して初期化すると、自己署名証明書を使用してポート 443 上で Web インターフェイスから Cb Response サーバーにアクセスできます。ポート 80 上で HTTP を介して Web インターフェイスにアクセスしようとすると、ポート 443 にリダイレクトされます。

テスト環境などでは、この次の手順として、センサーを 1 つ以上ダウンロードしてインストールし、データの収集を開始します。センサーのインストールについては、『Cb Response ユーザー ガイド』の「センサーの管理」という章を参照してください。

Cb Response サーバーのアップグレード

サーバーをアップグレードする場合は、以下のような条件によってその手順が異なります。

- アップグレードの対象がスタンドアロンのサーバーであるかクラスター化したサーバーであるか。
- サーバーの新しいバージョンをインストールした後に移行するのがデータベース スキーマであるか Cb Threat Intel のフィード データであるか。

これらの手順では、SSH またはコンソールにより、root 権限を使用してサーバーとミニオンにアクセスする必要があります。

スタンドアロンのサーバーをアップグレードする :

1. 目的のサーバー上で、次のように Cb Response の各サービスを停止します。
   `sudo service cb-enterprise stop`
2. (オプション) メタデータとパッケージの Yum キャッシュを消去します。
   `yum clean all`
3. 次のように Cb Response の各サービスを更新します。
   `sudo yum upgrade cb-enterprise`
4. 次のように Cb Response の各サービスを再起動します。
   `sudo service cb-enterprise start`

クラスター化したサーバーをアップグレードする :

1. マスター サーバー上で `Cb` のインストール先ディレクトリ (デフォルトでは `/usr/share/cb`) に移動し、次のように Cb Response の各サービスを停止します。
   `sudo cbcluster stop`
2. (オプション) メタデータとパッケージの Yum キャッシュを消去します。
   `yum clean all`
3. マスター サーバーとミニオン サーバーの各ノードで、Cb Response の各サービスを更新します。
   `sudo yum update cb-enterprise`
4. cbupgrade ユーティリティを実行します (詳細については、以下のセクションを参照)。
   `/usr/share/cb/cbupgrade`
5. 目的のマスター サーバー上で、次のように Cb Response の各サービスを再起動します。
   `sudo cbcluster start`

警告

6.1 より前のバージョンの Cb Response では、cb サービス アカウントのホーム ディレクトリ (デフォルトで /var/cb) を変更することができました。サーバーのインストールは、このアカウントの支援に基づいて行われます。このバージョンではこの機能がサポートされていないため、サーバーをアップグレードすると、nginx エラーが発生します。

cb サービス アカウントのホーム ディレクトリは、/var/cb でなければなりません。アップグレードでは、このディレクトリに nginx の実行時構成ファイルが配置されます。そのため、cb サービスのホーム ディレクトリを変更すると、実行時構成ファイルが見つけられなくなります。/etc/password でホーム ディレクトリを編集した場合は、アップグレードの前に /var/cb に戻してください。

cbupgrade の使用

Cb Response サーバーのアップグレードでは、`yum update cb-enterprise` を実行した後、`cbupgrade` というユーティリティを使用してデータベース スキーマまたは Cb Threat Intel のフィード データを移行することが必要になる場合があります。cb-enterprise サービスを開始しようとすると、この要件が通知されます。クラスター化したサーバー構成では、すべてのノードで `cbupgrade` ツールを実行してからクラスターを再起動する必要があります。クラスター環境でこのユーティリティを実行するときは、Cb Response サービスを起動するかどうかの問い合わせに対して必ず `No` と回答してください。クラスター化したサーバーを起動するには `cbcluster` を使用する必要があります。

`cbupgrade` オプションの説明については、「クラスター ノードのアップグレード」(58 ページ) を参照してください。

サーバーのアップグレードと新しいセンサー バージョン

Cb Response サーバーのリリースには、通常、1 つ以上のオペレーティング システムの更新されたセンサー バージョンが含まれています。サーバーとセンサーは別々にアップグレードできます。またセンサーはすべて一度にアップグレードするのではなく、センサー グループごとにアップグレードできます。

新規バージョンのセンサーが存在する場合は、既存のセンサー インストール環境に即座に新規センサーを展開するのか、まずサーバーのアップデートのみインストールするのかを決定してください。Carbon Black は、ネットワークおよびサーバーのパフォーマンスに対する容認できない影響を回避するために、センサーを段階的にアップグレードすることを推奨します。また環境内のすべてのセンサーを誤って一度にアップグレードすることを防ぐために、サーバーのアップグレード前に、センサー グループのアップグレード ポリシーを確認することを強く推奨します。センサー グループのアップグレード ポリシーの詳細については、『Cb Response ユーザー ガイド』の「センサー グループ」セクションを参照してください。

[Upgrade Policy (アップグレード ポリシー)] の設定

[Sensors (センサー)] ページの [Create or Edit Group (グループの作成または編集)] パネルの [アップグレード ポリシー] セクションには、Windows、OS X、Linux プラットフォームで、グループ内のインストールされたセンサーのアップグレード ポリシーを設定できるオプションが含まれています。アップグレードポリシーのオプションは次のとおりです。• **[No automatic updates (自動更新なし)]** – センサーをアップグレードする時期を手動で指定できます。• **[Automatically install the latest version (最新バージョンを自動的にインストール)]** – センサーが最新バージョンに自動的にアップグレードされます。• **[Automatically install a specific version (特定のバージョンを自動的にインストール)]** – グループ内のすべてのセンサーに特定のバージョンをインストールします。このオプションでは、すべてのセンサーが選択したバージョンに統一されます。ドロップダウン リストを使用してバージョン番号を選択します。特定のバージョンのアップグレード ポリシーを選択できると、センサーの特定のバージョンをテストまたは詳しく検査する場合に便利です。

イベント データに対する複数ボリュームのサポート

このセクションでは、最新のリリースにアップグレードした後で、既存の Cb Response の展開にストレージをさらに追加する方法について説明します。具体的には、cbevents コア用の Solr データ ディレクトリを複数追加する方法について説明します。これらのディレクトリは、マウント ポイントとして新しいストレージ アレイに追加することができるため、ディスク容量の追加を容易に行うことができます。構成を行うことも容易です。基本的に、ディスク容量がさらに必要な場合は、新しいボリュームをアタッチし、それを Solr データ ディレクトリにマウントします。サーバーは新しいボリュームの使用を自動的に開始します。

命名規則

Solr では、以下のいずれかがディレクトリ名の接頭辞になっている場合は、新しい `cbevents` ディレクトリ (マウント ポイント) が使用されます。

```
cbevents*
```

または

```
_cbevents*.
```

注意

(接尾辞のない) cbevents ディレクトリは、デフォルトのディレクトリですが、元のデータ パーティション上に残しておく必要はありません。このディレクトリは必要に応じて削除できます。

次に示すのは、有効な複数ボリューム構成の具体例です。

```
[root@ip-172-31-14-184 solr5]# df
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/xvda1      32895856 10341996  20860168  34% /
tmpfs           31389104        0  31389104   0% /dev/shm
/dev/xvdb      206293688 28033360 167758184  15% /data
/dev/xvdf      206293688 35809668 159981876  19% /data/solr5/cbevents2
/dev/xvdg      226936188 63976332 151409136  30% /data/solr5/cbevents3
```

この例では、デフォルトのデータ ドライブは `/dev/xvdb` にマウントされ、/data は `cb.conf` 内のデータ ルートとして構成されます。また、2 つのボリュームが追加され、それぞれ `/data/solr5/cbevents2` と `/data/solr5/cbevents3` にマウントされます。

警告

cb-enterprise が再起動されると、正しい `user:group` が割り当てられます。ライブ サーバー上にマウント ポイントを作成した場合は、Cb Response サーバーに割り当てられているユーザーに対し、マウントされたディレクトリへの書き込み権限を必ず付与してください。付与しないと、新しいマウント ポイントがシステムに認識されません。

このほか、cbevents ストレージを拡張する手段として、symlink を使用することもできます。手順は以下のとおりです。

symlink された場所をイベント ストレージに使用する :

1. `/data2` など、ファイル システムの別の場所にマウント ポイントを作成します。
2. マウントされたディレクトリを参照している `solr5` ディレクトリ内の `cbevents*` ディレクトリへの symlink を作成します。以下に例を示します。

   ```
   ln -s /data2 /var/cb/data/solr5/cbevents2
   ```

3. マウントされたディレクトリ (`/data2`) に対する書き込み権限が Cb Response ユーザーに付与されていることを確認します。

新しいデータ ディレクトリの使用

このセクションでは、新しいデータ ディレクトリに関するパーティションの作成と削除について説明します。

パーティションの作成

次回パーティション作成時 (デフォルトでは 3 日間隔) または現在のデータ ディスクの空き容量がなくなりそうになった場合には、新しいデータ ディレクトリが使用されます。パーティション作成のタイミングと新しいイベント パーティションの配置場所を計算する際、サーバーでは簡単なヒューリスティックが使用されます。

1. パーティションの作成時には、空き容量を最大限に活用した新しいパーティションが `cbevents*` ディレクトリに作成されます。

2. 現在のデータ ボリュームの使用量が 95% を超え、その他に存在するパーティションの空き容量が 5% を超えている場合は、サーバーによって直ちにパーティションが作成されます。

   このしきい値は、以下の構成パラメーターを使用して制御することができます。

   `SolrTimePartitioningFreeSpaceThresholdPerc`

- ルール 1 により、複数の新しいボリュームがバランスよく使用されます。古いデータが有効期間を超過すると (削除されると)、一部のパーティションが解放されます。これにより、空き容量を最適に使用できるようになります。
- ルール 2 により、`cbevents*` ディレクトリが多数存在する場合に、断片化されたディスク容量が効率的に使用されます。たとえば、ボリュームが 5 つあり、それぞれの空き容量が 20% だとします。この場合、どのボリュームでも 3 日分のパーティションを確保できない可能性があります。そのようなときは、引き続きいずれか 1 つのパーティションを (使用可能な最大容量まで) 使用し、その後で次のパーティションを使用することになります。そのため、サーバーの各パーティションが、最終的にさらに小さくなることが考えられます。ただし、これはまれなケースです。

アクティブな読み取り専用ディレクトリ

`cbevents*` を接頭辞に持つ `cbevents` ディレクトリはいずれも、cbevent パーティションを新規作成する際に使用されます。

`_cbevents*` を接頭辞に持つ `cbevents` ディレクトリはいずれも、読み取り専用として使用されます。これらのディレクトリを使用すると既存のパーティションをロードすることができますが、新規のパーティションは作成されません。古いボリュームを破棄する場合にはこの方法を用いることができます。古いパーティションは最終的に、時間に基づいて削除されます。接頭辞 `_cbevents` は、" コールド " パーティション (必要な場合にのみロードされる古いパーティション) 専用のディレクトリにも使用されます。

パーティションの削除

パーティションの削除は、ディスク容量、時間、または使用できるパーティションの最大数に基づいて実行されます。

ディスク容量に基づいて削除が実行される場合、削除アルゴリズムではディスクの空き容量全体を考慮します。たとえば、100 GB のボリュームが 3 つ存在し、空き容量がそれぞれ 30 GB であるとします。この場合、空き容量は全部で 90 GB、ディスク容量全体は 300 GB となります。イベント データの合計サイズは、3 つのボリュームすべてのインデックス サイズを合算したものです (メイン データ ボリュームにはストア ファイルや他のデータも含まれていることがあるため、イベント データの合計サイズは 210 GB 未満である場合もあります)。

以下は、ボリュームが複数存在する場合に (`cb.conf` 内の) 現在の削除しきい値がどのように解釈されるかを示したものです。

- `MaxEventStoreSizeInPercent` - 全イベント コアの合計サイズが (全ボリューム上の) ディスク容量全体の所定の割合を超過した場合に最も古いパーティションが削除されます。

- `MaxEventStoreSizeInMB` - (全ボリューム上の) イベント ストアの合計サイズが所定のしきい値を超過した場合に最も古いパーティションが削除されます。
- `MinAvailableSizeInMB` - (全ボリューム上の) 全空きディスク容量が所定のしきい値を下回った場合に最も古いパーティションが削除されます。

オンザフライによるディスク容量の拡張

ディスク容量はオンザフライで追加することが可能です。その際に、Cb Response サーバーを再起動する必要はありません。新しいパーティションが作成されると自動的に新しいディレクトリが使用されるため、サーバーのダウンタイムは発生しません。前述したように、必ずそのディレクトリに対する適切な読み取り / 書き込み権限を cb ユーザーに付与してください。

サーバーのトラブルシューティング

以下の表は、`/var/log/cb` にある Cb Response サーバーのログをまとめたものです。これらのログは、コンポーネントごとに別々のサブディレクトリに配置されています。Cb Response サーバー ログ :

コンポーネント	説明
allianceclient	アライアンス クライアントは Cb Response Alliance サーバーと通信します。
audit	禁止、センサーの分離、および Live Response の各アクティビティを記録します。`cb.conf` で `EnableExtendedApiAuditLogging` が有効な場合、このディレクトリには、ユーザーがコンソールで生成する API 呼び出しに基づいたユーザー アクティビティのログ ファイルも含まれます。
cbfs	Cb Response の初期バージョンでデータストア エンジンの場所となっていましたが、バージョン 5.0.0 以降は使用されなくなりました。
cbfs-http	第 2 世代の Java データストア エンジンのログ ファイルを収めています。
cli	サーバーのコンソール レベルで使用する Cb Response サービス コマンドに関連するイベントを収めています。
coreservices	Web API を介して、Web インターフェイスとセンサーの両方が各種機能にアクセスできるようにします。インターフェイスの問題のほとんどすべては、coreservices のログ エントリとして記録されています。
sensorservices	センサーの登録とチェックインのエントリポイントとなるものです。センサーの接続について不具合が生じた場合は、ここで問題点を調べます。
datastore	コア イベント データの処理および受信するセンサー データの管理に使用します。
enterprise	Cb Response サービスのイベント ログ記録に使用します。

コンポーネント	説明
job-runner	Cb Response サーバーでは、cron ジョブを使用して、スケジュールされたさまざまなメンテナンスやデータ トリミングなどのタスクを実現しています。
liveresponse	Cb Live Response セッション関連イベントの保持に使用します。
nginx	Cb Response サーバーでリバース プロキシと SSL ターミネーションを実装するポイント。
notifications	フィードとウォッチリストに関連する syslog 出力の場所。
pgsql	Cb Response サーバーでは、Postgres SQL を使用して管理データを保存します。センサーから収集したイベント データは Postgres に保存されません。
rabbitmq	Cb Response サーバーの rabbitmq コンポーネントのログ記録場所。
redis	Cb Response サーバーの redis コンポーネントのログ記録場所。
services	Cb Response サーバーの開始 / 停止サービスのログ記録場所。
solr	データのインデックス作成と保存に使用します。
supervisord	supervisord プロセス ユーティリティは、サーバーのさまざまなコンポーネントやサービスの間における起動およびシャットダウンの依存関係の処理など、Cb Response サーバー プロセスの管理に使用されます。

以下の表は、`/usr/share/cb` にあるスクリプトをまとめたものです。このディレクトリにあるスクリプトは大半が診断スクリプトで、この表にも診断に使用されるスクリプトのみが記載されています。

コンポーネント	説明
cbbanning	Cb Response サーバーの禁止機能の管理を支援します。使用できるコマンドの一覧を取得するには、このコマンドを次のように実行します。 `cbbanning commands`
cbstats	このユーティリティは、Cb Response サーバーで収集された統計情報にアクセスできるようにします。
cbsyslog	Cb Response の通知 syslog 出力をテストするためのインターフェイスを提供します。
cbpost	このユーティリティを使用して Alliance サーバーにファイルを送信します。多くの場合、Carbon Black テクニカル サポートとの情報交換の際に使用します。
py_runtime_info	実行中の Cb Response プロセスのスタック トレース、プロセス メモリ マップ、および開いているファイル ディスクリプターを示す実行時レポートを生成します。

コンポーネント	説明
cbfeed_scrubber	既存の Solr ドキュメント上でフィード タグをクリーンアップします。
cbinit	Cb Response サーバーをインストールする際、初期設定を組み合わせて構成するのに使用します。
cbdiag	ログや構成などの詳しいトラブルシューティング情報を gzip アーカイブにダンプします。このファイルはオフラインで分析できるほか、サポート依頼とともに Carbon Black に提出することもできます。
sql_stats	さまざまな SQL データベース統計の出力を収めています。多くの場合、トラブルシューティングで使用します。
cbsolr	データのインデックス作成と保存に使用します。
cbget	このユーティリティは、Alliance サーバーにあるファイルのダウンロードや一覧表示に使用します。多くの場合、Carbon Black テクニカル サポートとの情報交換の際に使用します。
sensor_report	Cb Response サーバーと通信している各センサーのステータスを示すレポートを生成します。必要に応じ、IT サポート担当者による警戒が必要な特定のセンサーの識別に使用できます。

cbcheck	Cb Response サーバーをインストールする作業のトラブルシューティングを支援します。使用できるコマンドの一覧を取得するには、このコマンドを次のように実行します。
	`cbcheck commands`
	特定のコマンドの詳しい情報を表示するには、このコマンドを次のように実行します。
	`cbcheck <command> -h`

cbcluster	クラスターの管理に使用します (診断ツールではありません)。
cb_rabbitmq-server.sh	これはシステム ユーティリティであるため、手動では実行しないでください。
cbrabbitmqctl	Cb Response の rabbitmq サービスにアクセスできるようにするコマンドライン インターフェイスです。
pgsql_diag.sh	CBER Postgres データベースに関する診断情報を出力します。
cbpasswd	ユーザーのパスワードをリセットします。ルートとしてのみ実行できます。

Cb Response サーバーのアンインストール

このセクションでは、RHEL/CentOS から Cb Response サーバーをアンインストールする手順について説明します。

以下の Cb Response パッケージを削除する必要があります。

- `carbon-black-release`
- `cb-datagrid`

- cb-datastore
- cb-enterprise
- cb-solr
- cb-swagger
- cbui
- libselinux-cb-python
- python-cb-coreservices
- python-cb-response-venv

Cb Response サーバーの削除

Yum ユーティリティを使用して、2 つの方法のいずれかで Cb Response サーバーをアンインストールできます。1 つは Cb Response サーバーを構成するパッケージを削除または消去する方法で、もう 1 つは Cb Response パッケージとそれが依存している他のすべてのサードパーティ製パッケージを削除する方法です。以下では、これら 2 つの方法について説明します。

サーバーをアンインストールする (Cb Response RPM のみ):

1. 以下のいずれかのコマンドを使用して、Cb Response の各サービスを停止します。

   ```
   sudo /usr/share/cb/cbcluster stop
   ```

   または

   ```
   sudo service cb-enterprise stop
   ```

2. 以下の Yum ユーティリティ コマンドを使用して、上記の Cb Response パッケージを削除します。

   ```
   sudo yum remove <package1> <package2> <packageN>
   ```

3. 以下の各 `cb` ディレクトリを手動で削除します。

   - `/var/www/cb/`
   - `/var/run/cb/`
   - `/var/log/cb/`
   - `/var/lib/cb/`
   - `/var/cb/`
   - `/usr/share/cb/`
   - `/etc/cb/`

Yum ユーティリティを使用して、Cb Response パッケージとそれが依存している他のすべてのサードパーティ製パッケージを削除することもできます。

警告

この手順を実行すると、他のソフトウェア アプリケーションでも必要なパッケージが削除されることがあります。この点に注意してこの手順を実行してください。

サーバーをアンインストールする (CB Response と関連の RPM):

1. 以下のいずれかのコマンドを使用して、Cb Response の各サービスを停止します。

   ```
   sudo /usr/share/cb/cbcluster stop
   ```

   または

   ```
   sudo service cb-enterprise stop
   ```

2. 以下のコマンドを使用して `yum.conf` ファイルにアクセスします。

   ```
   vi /etc/yum.conf
   ```

3. 以下の行を `yum.conf` に追加します。

   ```
   clean_requirements_on_remove=1
   ```

4. 以下のように入力します。

   ```
   yum erase cb-enterprise
   ```

5. 以下のように入力します。

   ```
   yum remove carbon-black-release
   ```

6. 以下の各 `cb` ディレクトリを手動で削除します。

   - `/var/www/cb/`
   - `/var/run/cb/`
   - `/var/log/cb/`
   - `/var/lib/cb/`
   - `/var/cb/`
   - `/usr/share/cb/`
   - `/etc/cb/`

第 3 章

サーバーのバックアップと復元

この章では、バックアップと復元のさまざまな手順を実行する方法について説明します。

セクション

トピック	ページ
概要	42
バックアップ	43
復元	47

概要

この章では、Cb Response サーバーのバックアップ手順と復元手順について説明します。ここで説明する手順は、重大な障害が発生した場合のデータ損失を最小限に抑えることを目的としたものです。

バックアップ ファイルおよびバックアップ データは、日常業務で使用されているものとは別のサーバーに保存する必要があります。

本文書で説明するすべての手順はコマンド プロンプトで実行し、ルートレベルのアクセス権が必要です。

備考

- この章では全体を通して、`/var/cb` がデフォルトのインストール データ パスであるとの前提で説明を進めます。別の場所にデータストアのルートを設定している場合は、文書内で説明されている場所をその場所に置き換えてください。
- ネットワーク　インテグレーションの設定は完全にはバックアップされません。ブリッジやコネクターをインストールする必要がある場合は、構成を復元する前に、復元先の新しいサーバーにインストールしておく必要があります。ネットワーク インテグレーションを構成する項目は、この章で説明する手順全体を通じて復元されます。

復元先のサーバー

すべての復元方法について、新しいサーバーは同じ数のマスターとミニオンのシステム構成でインストールされ、それぞれが各システムに同じ数の Solr データ シャーディングを導入していて、同じホスト名と IP アドレスを使用するように構成されていることが前提となります。

また、バックアップするシステムと復元するシステムでは、どちらも同じサーバーのバージョンを使用する必要があります。

新しいシステムのインストールについては、「Cb Response サーバーのインストール」(19 ページ) を参照してください。

すべてのインストール手順は、`cbinit` (クラスター化したシステムの場合は `cbcluster add-node` も) の実行を伴います。この作業は、システムの復元を実行する前に完了しておく必要があります。

バックアップ / 復元スクリプト

Carbon Black Developer Community には、以下で説明するバックアップ タスクおよび復元タスクの一部を処理できるオープンソースのスクリプトが用意されています。これらを使用すれば入力する時間を節約することができます。このスクリプトの使用方法については、本書では説明しません。このスクリプトについては、Github (https://github.com/cbcommunity/cb-administration-scripts/tree/master/backup_restore) で提供されているスクリプトについてのドキュメントをお読みください。

バックアップ

このセクションでは、バックアップと復元の方法として次の 2 つを取り上げます。

- 構成のみのバックアップは、システムのすべての構成とセンサーのメタデータが対象となります。このバックアップはストレージ サイズを最小限に抑えることを目的としたもので、その実行中に重大なサーバー障害が発生した場合でも迅速に稼働状態に復帰できます。センサーが送信するデータ (イベントまたはバイナリ データ) のほか、実行中のサーバーから送信されるフィード情報やアラート情報はバックアップの対象にはなりません。構成バックアップの手順は、バックアップと復元を行う 2 つの方法のいずれにおいても必須です。詳細については、「構成バックアップ」(43 ページ) を参照してください。
- データ バックアップは構成バックアップの延長であり、サーバーが格納するすべてのデータが対象となります。このようなデータをバックアップすることで、システムの完全な復旧を保証します。バックアップの対象が完全なデータセットとなるため、サーバー外のストレージ要件がはるかに大きくなります。また、大量のデータを新しいシステムにコピーおよび展開する必要があるため、復元の実行にかかる時間も長くなる傾向があります。

  注意

  この手順では、バックアップ データのストレージ場所の例として `/cbdata/_backup/ServerName` を使用します。データ バックアップも実行する場合は、バックアップを格納する正しい場所を確認し、必要なサイズに注意してください。

  別途記載がない限り、マスターおよびミニオンの各システムですべてのコマンドを実行する必要があります。すべてのスタンドアロン サーバーですべての手順を実行します。詳細については、「データ バックアップ」(46 ページ) を参照してください。

構成バックアップ

構成バックアップでは、システムの復元に必要なファイルとデータだけがバックアップされます。動作中のセンサーは含まれますが、記録されたデータは含まれません。この手順はすべてのバックアップ方法で必須ですが、これ自体が独立した手順として完結しています。データの収集と復元を最も迅速に実行できる方法であり、最小限のディスク容量しか消費しません。構成バックアップは、動作状態の Cb Response で実行できます。

構成バックアップを実行する :

1. バックアップの場所にディレクトリを変更します。

   ```
   cd /cbdata/_backup/ServerName
   ```

2. 次のコマンドを実行して、構成ファイルをバックアップします。

   a. hosts ファイル :

   ```
   tar -P --selinux -cvf cbhosts.tar /etc/hosts
   ```

   b. yum ファイル :

   ```
   tar -P --selinux -cvf cbyum.tar /etc/yum.repos.d
   ```

c. IP テーブル :

```
tar -P --selinux -cvf cbiptables.tar /etc/sysconfig/iptableS
```

d. SSH 構成および SSH キー :

```
tar -P --selinux -cvf cbssh.tar /etc/ssh/
```

e. Cb Response 構成 :

```
tar -P --selinux -cvf cbconfig.tar /etc/cb/
```

f. rsyslog 構成 :

```
tar -P --selinux -cvf cbrsyslog.tar /etc/rsyslog.conf
```

g. rsyslog.d 構成 :

```
tar -P --selinux -cvf cbrsyslogd.tar /etc/rsyslog.d/
```

h. RabbitMQ cookie:

```
tar -P --selinux -cvf cbrabbitmqcookie.tar /var/cb/
.erlang.cookie
```

i. RabbitMQ ノード構成 :

```
tar -P --selinux -cvf cbrabbitmqnode.tar /var/cb/data/
rabbitmq
```

j. (オプション) SSH 認証キー :

注意 : この手順は、クラスター環境内のシステム間で信頼できるキーを使用している場合のみ実行してください。

```
tar -P --selinux -cvf cbrootauthkeys.tar /root/.ssh/
authorized_keys
```

k. (マスターのみ) Syslog CEF テンプレート :

```
tar -P --selinux -cvf cbceftemp.tar /usr/share/cb/
syslog_templates
```

l. (オプション - マスターのみ) Cb インストーラーのバックアップ :

注意 : この手順は、追加バージョンのセンサーを手動でインストールしている場合のみ実行してください。

```
tar -P --selinux -cvf cbinstallers.tar /usr/share/cb/
coreservices/installers/
```

m. (オプション - マスターのみ) カスタムの syslog テンプレート :

注意 : 次の各手順は、 `/user/share/cb/syslog_templates` に保存されていないカスタムの syslog テンプレートを使用している場合のみ実行してください。

i. 以下のように検索して、使用中のカスタムの syslog テンプレート パスを特定します。

`SyslogTemplate=` の任意のインスタンスについて `/etc/cb/cb.conf` ファイルを検索

以下に例を示します。

`WatchlistSyslogTemplateBinary`

および

`FeedIngressSyslogTemplateBinary`

ii. 「=」の後のファイル パスを記録しておきます。
以下に例を示します。

```
WatchlistSyslogTemplateBinary=/var/custom/syslog/
watchlist_binary_custom.template
```

この例のパスは、`/var/custom/syslog` になります

iii. 次のコマンドを使用して特定された各カスタム パスに対して tar を実行します。

```
tar -P --selinux -cvf syslog_custom1.tar /var/custom/syslog
```

3. (マスターのみ) Postgres データベースのバックアップを実行します。

   注意 : この手順は、構成のみのバックアップを実行する場合に限り実行します。それ以外の場合はこの手順をスキップし、で説明されている Postgres データベースのバックアップ手順を実行してください。「データ バックアップ」(46 ページ)

   a. 構成のバックアップを実行します。

```
pg_dump -C -Fp -f psqldump_config.sql cb -p 5002 \
--exclude-table-data=allianceclient_comm_history \
--exclude-table-data=allianceclient_uploads \
--exclude-table-data=allianceclient_pending_uploads \
--exclude-table-data=banning_sensor_counts \
--exclude-table-data=binary_status \
--exclude-table-data=cb_useractivity \
--exclude-table-
data=detect_dashboard_average_alert_resolution_history \
--exclude-table-data=detect_dashboard_binary_dwell_history \
--exclude-table-data=detect_dashboard_host_hygiene_history \
--exclude-table-data=investigations \
--exclude-table-data=maintenance_job_history \
--exclude-table-data=moduleinfo_events \
--exclude-table-data=mutex_watchlist_searcher \
--exclude-table-data=sensor_activity \
--exclude-table-data=sensor_comm_failures \
--exclude-table-data=sensor_driver_diagnostics \
--exclude-table-data=sensor_event_diagnostics \
--exclude-table-data=sensor_licensing_counts \
--exclude-table-data=sensor_queued_data_stats \
--exclude-table-data=sensor_resource_statuses \
--exclude-table-data=server_storage_stats \
--exclude-table-data=storefiles \
--exclude-table-data=tagged_events
```

   b. ユーザーとグループのバックアップを実行します。

```
pg_dumpall -p 5002 --roles-only -f psqlroles.sql
```

4. バックアップの場所をリモートの場所にコピーします。

データ バックアップ

データ バックアップでは、サーバーに格納されているすべての記録済みデータを取得します。このバックアップでは、動作しているシステムの完全な復元を実行する必要があります。このセクションの手順を実行する前に、「構成バックアップ」(43 ページ) を完了しておく必要があります。システム / クラスターに保持されているデータの量によっては、このバックアップで取得したデータが非常に大きくなる可能性があります。完全バックアップを実行する場合は、Cb Response を停止状態にする必要があります。

データ バックアップを実行する :

1. 次のコマンドを実行して、すべての Cb Response サービスを停止します。
   a. クラスター化したサーバー環境では、マスターでのみ次のコマンドを実行します。
      ```
      /usr/share/cb/cbcluster stop
      ```
   b. スタンドアロンのサーバー環境では、次のコマンドを実行します。
      ```
      service cb-enterprise stop
      ```
2. バックアップの場所にディレクトリを変更します。
   ```
   cd /cbdata/_backup/ServerName
   ```
3. 次のコマンドを実行して、Solr データベースをバックアップします。
   ```
   tar -P --selinux -cvf cbsolr.tar /var/cb/data/solr5/
   ```
4. モジュール ストアのバックアップを実行します。
   ```
   tar -P --selinux -cvf cbmodulestore.tar /var/cb/data/modulestore/
   ```
5. (マスターのみ) Postgres データベースのバックアップを実行します。
   a. Postgres サービスを開始します。
      ```
      service cb-pgsql start
      ```
   b. Postgres データベースのバックアップを実行します。
      ```
      pg_dump -C -Fp -f psqldump_full.sql cb -p 5002
      ```
   c. (これは 「構成の復元」(48 ページ) と重複する手順です) ユーザーとグループのバックアップを実行します。
      ```
      pg_dumpall -p 5002 --roles-only -f psqlroles.sql
      ```
   d. Postgres サービスを停止します。
      ```
      service cb-pgsql stop
      ```
6. 次のコマンドを実行して、Cb Response サーバーを開始します。
   a. クラスター化したサーバー環境では、マスターでのみ次のコマンドを実行します。
      ```
      /usr/share/cb/cbcluster start
      ```
   b. スタンドアロンのサーバー環境では、次のコマンドを実行します。
      ```
      service cb-enterprise start
      ```
7. バックアップの場所をリモートの場所にコピーします。

復元

システムを復元するには、以下の前提条件を満たしている必要があります。

- フレッシュな Cb Response サーバー (または古いスナップショット) がバックアップ ファイルの復元先として利用できること。
- マスターとミニオンについて、サーバーが同じ構成であること (クラスター化したサーバー環境の場合)。
- 同じホスト名と IP アドレスが使用されていること。
- 新しいサーバーは、バックアップを作成したサーバーと同じサーバー バージョンでインストールする必要があります (例 : v6.2.2)。

インストールの完了に詳細な構成項目が必要な場合は、「構成バックアップ」(43 ページ) で生成される次の各ファイルを使用して、バックアップ サーバーから構成を取得します。

- 各システムで使用する IP アドレス。 `cbhosts.tar`: `/etc/hosts` ファイルに記述されています。
- 使用するシステム数とマスター / ミニオンの割り当て。 `cbconfig.tar`: `/etc/cb/cbcluster.conf` ファイルに記述されています。

新しいシステムのインストールについては、「Cb Response サーバーのインストール」(19 ページ) を参照してください。サーバーの復元を実行する前に、`cbinit` (クラスター化したサーバー環境の場合は `cbcluster add-node` も) の実行を含むすべてのインストール手順を完了しておく必要があります。 `cbinit` の実行時に選択した構成項目は、構成ファイルの復元が完了すると上書きされます。

失敗したミニオン クラスターの復元

このセクションで説明する手順を実行すると、フレッシュなサーバーをセットアップし、失敗した 1 つ以上のミニオンをそれ以外は動作しているクラスターに復元できます。

この手順を実行するには、以下の前提条件を満たしている必要があります。

- (少なくとも) マスター サーバーが引き続き動作していて、失敗したミニオンのバックアップ データが利用可能であること。
- 復元するミニオンのインストール先サーバーが、クラスター内の、Carbon Black ソフトウェアがインストールされていないその他のサーバーと同じベース レベルのオペレーティング システムを実行していること。

サーバーをインストールし、それ以外は動作しているクラスター上の失敗したミニオンでデータを復元する

1. クラスターを停止します。

   ```
   /usr/share/cb/cbcluster stop
   ```

2. Postgres サービスを開始します。

   ```
   service cb-pgsql start
   ```

3. Postgres に対して以下のクエリを実行し、欠落しているミニオンの `node_id` を特定します。

   ```
   psql -p 5002 cb -c "select * from
   cluster_node_sensor_addresses;"
   ```

4. `node_id` が欠落しているミニオンと一致する行を `cluster_node_sensor_address` テーブルから削除します。

   ```
   psql -p 5002 cb -c "delete from cluster_node_sensor_addresses
   where node_id = 2;"
   ```

   `2` は欠落しているミニオンの番号です

5. Postgres サービスを停止します。

   ```
   service cb-pgsql stop
   ```

6. `/etc/cb/cluster.conf` ファイルを次のように編集します。

   a. ファイルの `[Cluster]` セクションで、 `NodeCount` および `NextSlaveAutoInc` の各行から `N` を減算します。`N` は、復元対象となる失敗したミニオンの数です。

   b. 失敗したミニオンごとに `[SlaveN]` セクションを削除します。

   c. ファイルを保存します。

7. 通常のインストールの一部とするミニオンを追加するため、

   ```
   /usr/share/cb/cbcluster add-node
   ```

   を実行し、適切な IP アドレス/ホスト名とシャーディング情報を指定します。

8. マスターにより、ミニオンに `cb-enterprise` がインストールされ、通常動作が可能になります。

9. バックアップ プロセスで生成されたバックアップ データをコピーし、次のセクションの復元手順に従います。

10. クラスターを開始します。

    ```
    /usr/share/cb/cbcluster start
    ```

構成の復元

このセクションでは、構成ファイルとデータ ファイルを復元する手順を説明します。復元は、「バックアップ」(43 ページ) のバックアップ手順で実行された内容と一致している必要があります。

この手順では、バックアップ データのストレージ場所の例として `/cddata/_backup/ServerName` を使用します。データ バックアップも実行する場合は、バックアップを格納する正しい場所を確認し、必要なサイズに注意してください。

この手順を実行するには、以下の前提条件を満たしている必要があります。

- クラスター化したサーバー環境では、別途記載がない限り、以下の各セクションのすべてのコマンドがマスター サーバーおよびミニオン サーバーに対して実行されていること。
- スタンドアロンのサーバー環境では、以下の各セクションのすべてのコマンドが実行されていること。
- すべての復元作業において、Cb Response が停止状態になっていること。

構成の復元を実行する :

1. 次のコマンドを実行して、Cb Response を停止します。
   a. (マスターのみ) クラスター化したサーバー環境では、次のコマンドを実行します。
      ```
      /usr/share/cb/cbcluster stop
      ```
   b. スタンドアロンのサーバー環境では、次のコマンドを実行します。
      ```
      service cb-enterprise stop
      ```
2. 保存されている ssh キーをすべて削除します (ファイルは存在しない場合があります)。
   ```
   rm /root/.ssh/known_hosts
   ```
3. バックアップの場所にディレクトリを変更します。
   ```
   cd /cbdata/_backup/ServerName
   ```
4. 次のように構成ファイルを復元します。
   a. hosts ファイル :
      ```
      tar -P -xvf cbhosts.tar
      ```
   b. yum ファイル :
      ```
      tar -P -xvf cbyum.tar
      ```
   c. IP テーブル ファイル :
      ```
      tar -P -xvf cbiptables.tar
      ```
   d. SSH キー :
      ```
      tar -P -xvf cbssh.tar
      ```
   e. Cb Response 構成 :
      ```
      tar -P -xvf cbconfig.tar
      ```
      i. server.token を消去します。
      ```
      rm /etc/cb/server.token
      ```
      ii. 新しいサーバー トークンを取得します。
      ```
      /usr/share/cb/virtualenv/bin/python -c "from
      cb.alliance.token_manager import SetupServerToken;
      SetupServerToken().set_server_token('/etc/cb/server.token')"
      ```
   f. rsyslog の構成 :
      ```
      tar -P -xvf cbrsyslog.tar
      ```
   g. rsyslog.d の構成 :
      ```
      tar -P -xvf cbrsyslogd.tar
      ```
   h. Rabbitmq cookie の構成 :
      ```
      tar -P -xvf cbrabbitmqcookie.tar
      ```
   i. Rabbitmq ノードの構成 :
      ```
      tar -P -xvf cbrabbitmqnode.tar
      ```
   j. (オプション) SSH 認証キー :
      ```
      tar -P -xvf cbrootauthkeys.tar
      ```
   k. (マスターのみ) Syslog CEF テンプレート :
      ```
      tar -P -xvf cbceftemp.tar
      ```

l. (オプション - マスターのみ) Cb インストーラーのバックアップ :

```
tar -P -xvf cbinstallers.tar
```

m. (オプション - マスターのみ) カスタムの syslog テンプレート - カスタムのテンプレート TAR ごと :

```
tar -P -xvf syslog_custom1.tar
```

5. (マスターのみ) Postgres データベースの復元 :

注意 : 完全なデータの復元を実行する場合は、この手順をスキップします。

a. Postgres データベースを開始します。

```
service cb-pgsql start
```

b. 古いデータベースをドロップします。

```
dropdb cb -p 5002
```

c. ロールの復元 :

注意 : アカウントが作成済みであることを示すエラーが発生します。

```
psql template1 -p 5002 -f psqlroles.sql
```

d. データ ダンプを復元します。

```
psql template1 -p 5002 -f psqldump_config.sql
```

e. データベースのシーケンス値を復元します。以下のステートメントをコピーして、サーバー上の `psqlcbvalues` というファイルに貼り付けます。

```
SELECT
pg_catalog.setval('allianceclient_comm_history_id_seq', 1,
false);
SELECT
pg_catalog.setval('allianceclient_pending_uploads_id_seq',
1, false);
SELECT pg_catalog.setval('allianceclient_uploads_id_seq', 1,
false);
SELECT pg_catalog.setval('cb_useractivity_id_seq', 1,
false);
SELECT
pg_catalog.setval('detect_dashboard_average_alert_resolution
_history_id_seq', 1, false);
SELECT
pg_catalog.setval('detect_dashboard_binary_dwell_history_id_
seq', 1, false);
SELECT
pg_catalog.setval('detect_dashboard_host_hygiene_history_id_
seq', 1, false);
SELECT pg_catalog.setval('investigations_id_seq', 1, false);
SELECT pg_catalog.setval('maintenance_job_history_id_seq',
1, false);
SELECT pg_catalog.setval('moduleinfo_events_id_seq', 1,
false);
SELECT pg_catalog.setval('sensor_comm_failures_id_seq', 1,
false);
SELECT pg_catalog.setval('sensor_driver_diagnostics_id_seq',
1, false);
SELECT pg_catalog.setval('sensor_event_diagnostics_id_seq',
1, false);
SELECT pg_catalog.setval('sensor_licensing_counts_id_seq',
1, false);
```

```
SELECT pg_catalog.setval('sensor_queued_data_stats_id_seq',
1, false);
SELECT pg_catalog.setval('sensor_resource_statuses_id_seq',
1, false);
SELECT pg_catalog.setval('server_storage_stats_id_seq', 1,
false);
SELECT pg_catalog.setval('tagged_events_id_seq', 1, false);
```

f. `psqlcbvalues` を復元します。

```
psql cb -p 5002 -f psqlcbvalues
```

g. デフォルト調査を再構築します。

```
psql cb -p 5002 -c "INSERT INTO investigations VALUES
('1','Default Investigation',to_timestamp((select value from
cb_settings where key='ServerInstallTime'),'YYYY-MM-DD
hh24:mi:ss'),NULL,to_timestamp((select value from
cb_settings where key='ServerInstallTime'),'YYYY-MM-DD
hh24:mi:ss'),'Automatically Created at Installation Time');"
```

h. 以前のクエリベースのフィードをすべて削除します。

```
psql cb -p 5002 -c "delete from watchlist_entries where
group_id <> '-1';"
```

i. Postgres データベースを停止します。

```
service cb-pgsql stop
```

6. 次のように Cb Response の各サービスを開始します。

注意 : データの復元を実行する場合は、この手順をスキップします。

a. (マスターのみ) クラスター化したサーバー環境では、次のコマンドを実行します。

```
/usr/share/cb/cbcluster start
```

b. スタンドアロンのサーバー環境では、次のコマンドを実行します。

```
service cb-enterprise start
```

データの復元

このセクションでは、構成ファイルとデータ ファイルを復元する手順を説明します。復元は、「バックアップ」(43 ページ) のバックアップ手順で実行された内容と一致している必要があります。

この手順では、バックアップ データのストレージ場所の例として `/cddata/_backup/ServerName` を使用します。データ バックアップも実行する場合は、バックアップを格納する正しい場所を確認し、必要なサイズに注意してください。

この手順を実行するには、以下の前提条件を満たしている必要があります。

- データの復元を実行する前に、「構成の復元」(48 ページ) の手順が実行されていること。
- クラスター化したサーバー環境では、別途記載がない限り、以下の各セクションのすべてのコマンドがマスター サーバーおよびミニオン サーバーに対して実行されていること。
- スタンドアロンのサーバー環境では、以下の各セクションのすべてのコマンドが実行されていること。

データの復元を実行する :

1. 「構成の復元」(48 ページ) を実行した後で、Cb Response サーバーが停止状態になっていることを確認します。
2. Solr データを削除して復元します。

   ```
   rm -rf /var/cb/data/solr5/cbevents/0/data
   tar -P -xvf cbsolr.tar
   ```
3. 次のコマンドを実行して、モジュール ストアを復元します。

   ```
   rm -rf /var/cb/data/modulestore/*
   tar -P -xvf cbmodulestore.tar
   ```
4. (マスターのみ) 次のコマンドを実行して、Postgres データベースを復元します。
   a. Postgres サービスを開始します。

      ```
      service cb-pgsql start
      ```
   b. 古いデータベースをドロップします。

      dropdb cb -p 5002
   c. ロールの復元

      注意 : アカウントが作成済みであることを示すエラーが発生します。

      ```
      psql template1 -p 5002 -f psqlroles.sql
      ```
   d. Postgres データを復元します。

      ```
      psql template1 -p 5002 -f psqldump_full.sql
      ```
   e. Postgres サービスを停止します。

      ```
      service cb-pgsql stop
      ```
5. 次のように Cb Response の各サービスを開始します。
   a. (マスターのみ) クラスター化したサーバー環境では、次のコマンドを実行します。

      ```
      /usr/share/cb/cbcluster start
      ```
   b. スタンドアロンのサーバー環境では、次のコマンドを実行します。

      ```
      service cb-enterprise start
      ```

第 4 章

ポートとプロトコル

この章には、さまざまなサーバー通信に使用されるポートおよびプロトコルに関する情報が記載されています。

注意

いずれのサーバーについても基本的な IP アドレスが変わる可能性があります。特定の IP アドレスを示すのではなく、DNS 名を使用するようにファイアウォールを構成してください。

通信	ポート	プロトコル	コメント
管理ステーションから Cb Response サーバー	TCP 22	SSH	管理ステーションは、システム管理者が Cb Response サーバーに SSH で接続して、必要な管理タスクやトラブルシューティングを行うためのマシンです。
管理ステーションから Cb Response サーバー	TCP 443	HTTPS (構成可能)	
センサーから Cb Response サーバー	TCP 443	HTTPS (構成可能)	N/A

通信	ポート	プロトコル	コメント
マスター Cb Response サーバーからミニオン Cb Response サーバー	TCP 22	SSH	cb.conf ファイル内の MinionApiPort では、HTTPS ポート構成を設定できます。
	TCP 443	HTTPS (構成可能)	
	TCP 4369	RabbitMQ	
	TCP 5701	datagrid	
	TCP 6379	REDIS	
	TCP 6500	sensorservices	
	TCP 6501	sensorservices	
	TCP 8080	SOLR	
	TCP 9000	CB データストア	
	TCP 25004	RabbitMQ	
ミニオン Cb Response サーバーからマスター Cb Response サーバー	TCP 4369	RabbitMQ	N/A
	TCP 5002	POSTGRES	
	TCP 5600	liveresponse	
	TCP 5701	datagrid	
	TCP 6379	REDIS	
	TCP 6500	sensorservices	
	TCP 6501	sensorservices	
	TCP 8080	SOLR	
	TCP 25004	RabbitMQ	
ミニオン Cb Response サーバーからミニオン Cb Response サーバー	TCP 4369	RabbitMQ	N/A
	TCP 5701	datagrid	
	TCP 6500	sensorservices	
	TCP 6501	sensorservices	
	TCP 8080	SOLR	
	TCP 25004	RabbitMQ	

通信	ポート	プロトコル	コメント
Cb Response サーバーから CB Alliance サーバー	TCP 443	HTTPS	Cb Alliance サーバー通信を受け入れ可能な URL については、次のいずれかを参照してください。 • api.alliance.carbonblack.com Cb Alliance サーバーをポイントしています。背後にある単一の IP は、さまざまな理由によって時間の経過とともに変わります。 • api2.alliance.carbonblack.com Cb Alliance サーバーをポイントしています。背後にある単一の IP は api.alliance.carbonblack.com と異なる場合もあれば同じである場合もあります。さらに、さまざまな理由によって時間の経過とともに変わる可能性があります。 これらのサーバーの背後にある IP は変わる可能性があることに注意してください。
Cb Response サーバーから Cb Threat Intel	TCP 443	HTTPS	Cb Threat Intel 通信を受け入れる URL については、threatintel.bit9.com を参照してください。この URL は背後に複数の Elastic IP があり、" 次世代型 " Cb Threat Intel インフラストラクチャをポイントしています。この URL の背後にある IP は変わる可能性があることに注意してください。

通信	ポート	プロトコル	コメント
Cb Response サーバーから YUM リポジトリ	TCP 443	HTTPS	このタイプの通信における API については、yum.distro.carbonblack.io を参照してください。
	TCP 80	HTTP	このタイプの通信における API については、mirror.centos.org やその他の有効なリポジトリを参照してください。 基本の CentOS パッケージをインストールする際、Cb Response ではデフォルトの CentOS 構成が使用されます。 YUM リポジトリのインストールまたは更新をサポートする通信では、最初に mirror.centos.org が利用されますが、そのホストは、現在のミラー サーバー リストを特定するためだけに使用されます。そのうえで、実際のパッケージをダウンロードする際には、使用可能なミラー サーバーのいずれかが選択されます。 この CentOS のデフォルトの動作に問題がある場合は、CentOS 構成を変更するようにシステム管理者に依頼してください。

第 5 章

Cb Response クラスターのインストール

この章では、Cb Response クラスターについて概説するとともに、クラスターの構成方法、既存のクラスターにミニオンを追加する方法、クラスターからミニオン ノードを削除する方法、およびクラスター ノードをアップグレードする方法について説明します。

注意

一部のフォルダーやスクリプトでは「スレーブ」という用語を使用しています。これは 「ミニオン」と同じ意味です。

セクション

トピック	ページ
概要	58
Cb Response クラスターの構成	62
既存のクラスターへのミニオンの追加	65
既存のクラスターからのミニオンの削除	66
クラスター ノードのアップグレード	68

概要

Cb Response クラスターとは特定の役割を実行するサーバーのグループであり、単一の Cb Response インスタンスとして機能します。クラスターは、単一のスタンドアロン サーバーでは対応できない多数のセンサーやデータに対応する目的で使用します。この章では、Cb Response クラスターのセットアップについて詳しく説明します。

Cb Response は、さまざまな速度で流れる大量のデータを収集します。データの量と速度が増加すると、単一のサーバーで対応できる限界に到達してしまいます。増大するデータの量と速度に対応できるよう規模を拡大するためには、インフラストラクチャを水平方向に拡張する必要があります。つまり、インフラストラクチャにサーバーを追加するということです。目的とするセンサーの数やアクティビティ レベル、データの保持量が特定のしきい値を超過すると単一の Cb Response サーバー インスタンスのパフォーマンスに悪影響を及ぼすため、Cb Response サーバーのクラスターを構成する必要があります。

水平方向のスケーリング (規模の拡大) は、Cb Response がデータを格納するために使用するコア コンポーネントの Solr によって使用されます。Solr は、分散インデックスやそのようなインデックスに対する分散クエリを使用するビッグデータ ソリューションです。大量かつ高速なインデックスに対するクエリのパフォーマンスが向上します。最適なスケーリング パフォーマンスに対応できるように、Cb Response のインフラストラクチャはこの概念を中心としてモデル化されています。

サーバーのパフォーマンスおよびハードウェアの要件に関する詳細については、『Cb Response Operating Environment Requirements (OER)』ドキュメントを参照してください。

クラスターのアーキテクチャ

Cb Response クラスターの各サーバーは、次の 2 種類のロールに分けられます。

- マスター – ヘッドノードとも呼ばれます
- ミニオン – インデックス ノードとも呼ばれます

各クラスターのメンバーシップ ロールは、それぞれが特定の機能を実行し、その集合として単一の Cb Response インスタンスを構成します。またそれぞれの基本レベルに応じて、すべてのクラスター ノードは、各メンバーシップ ロールを実行するために構成された特定の機能と内部コンポーネントを持つ Cb Response サーバーとして機能します。

次の表は、各ロールに応じた Cb Response サービスと特定のコンポーネントをまとめたものです。

サービス / コンポーネント	スタンドアロン	マスター	ミニオン
cb-nginx	あり	あり	あり
cb-datastore	あり	あり	あり
cb-coreservices	あり	あり	あり

サービス / コンポーネント	スタンドアロン	マスター	ミニオン
cb-sensorservices	あり	あり	あり
cb-liveresponse	あり	あり	いいえ
cb-allianceclient	あり	あり	あり
cb-datagrid	あり	あり	あり
cb-enterprise	あり	あり	あり
- 脅威インテリジェンス	あり	あり	いいえ
- 統計情報	あり	あり	あり
cb-redis	あり	あり	あり
cb-pgsql	あり	あり	いいえ
cb-solr	あり	あり	あり
- プロセス データ (cbevents コア シャーディング)	あり	いいえ	あり
- バイナリ情報データ (cbmodules コア)	あり	あり	あり *
- フィード データ (cbfeeds コア)	あり	あり	いいえ
cb-rabbitmq	あり	あり	あり
バイナリ (modulestore)	あり	いいえ	あり

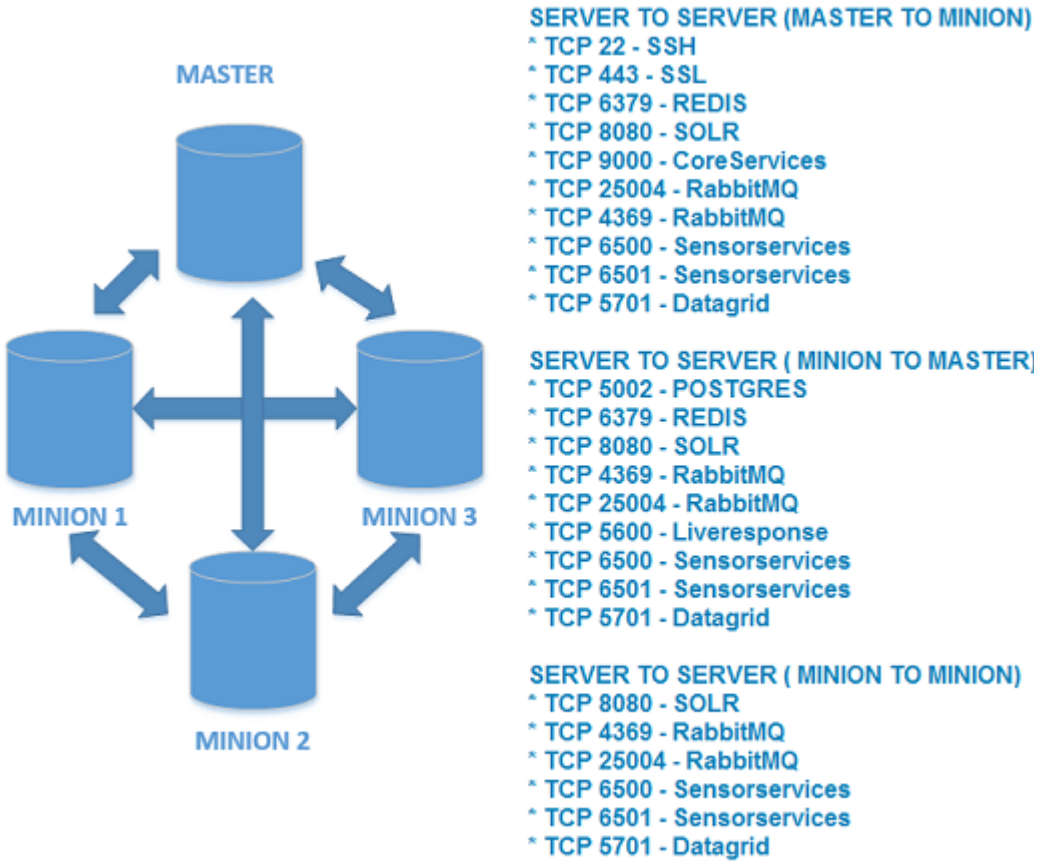
* マスターによってレプリケーション

Solr のクラスター化は、単一の Solr コアをシャーディングと呼ばれる複数のコアに分割することで実行されます。シャーディングは 0 から始まる整数値で識別され、個別の管理および分散クエリについて、各インデックス ノード (ミニオン) に均等に分散されます。

小規模なクラスター環境の場合はマスターがインデックス サーバーのロールも実行したり、スタンドアロン サーバーと同様にすべてのロールを実行したりする場合もあります。ただし、センサーの数が 60,000 を超えたり、ミニオンが 4 基構成されたりしている大規模な組織では、ミニオンの処理を実行しない専用のマスターをクラスター内に構成する必要があります。

Cb Response がクラスター化されている場合、アプリケーションが単一のインスタンスとして動作できるように内部的なノード間通信を発生させる必要があります。ほとんどの内部コンポーネントは、他のコンポーネントやノードに標準の呼び出しを実行することで、このような容量が分散された環境で動作することができます。

次の図は、ノード間のクラスター通信を示したものです。



RabbitMQ を単一のシステムとして正常に機能させるには、同様にクラスター化する必要があります。RabbitMQ は、プロセス、アプリケーション、サーバー間でデータ (またはメッセージ) を交換するために使用するアプリケーション メッセージ バスです。RabbitMQ クラスターを作成すると、内部 (サーバー プロセスおよびアプリケーション) やノード間 (ノード対ノード) でメッセージを交換したりキューしたりできるため、単一の Cb Response インスタンスとして適切に動作できるようになります。

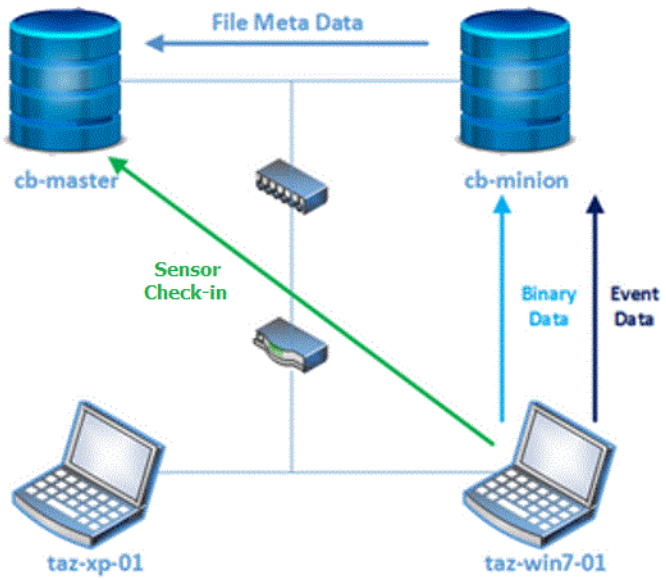
クラスターの動作

Cb Response クラスターは、スタンドアロンのインスタンスと同じように複数サーバーの環境で動作できなければなりません。そのため、各ロールが Cb Response インスタンスの特定の側面に対する役割を担う必要があります。

マスターにはユーザー インターフェイスが用意されており、API やほとんどの統合に対する主要なフロントエンドとして機能します。ユーザーが Web コンソールを操作すると、標準の API エンドポイント呼び出しがマスターに対して実行され、適切なバックエンドのストレージ場所に対してクエリが実行されます。プロセス データに対するクエリが実行された場合、マスターはクエリをミニオンに分散し、集約された結果を生成します。ただし、バイナリ、脅威インテリジェンス、またはアラートの検索が実行された場合は、ローカルの Solr コアに対してのみクエリが実行されます。

マスターには PostgreSQL データベースのインスタンスも含まれます。これは、ほとんどのアプリケーション固有の構成と、センサーの管理に使用する特定の状態情報を格納するデータベースです。マスターはセンサーの構成と通信を管理する役割を担っており、Cb Live Response の機能が含まれます。センサーの管理にはセンサーの状態情報だけでなく、ミニオンの状態に関するメトリックも必要です。これにより、ミニオンの帯域幅を独立して分散し、割り当て済みのセンサーがデータを送信できるようになります。

ミニオンは、センサー データの主要な収集ポイントとして機能します。ミニオンにデータを送信するようにマスターからセンサーに指示が出されると、そのミニオンはデータを受信し、保存用のデータ収集を開始します。ミニオンはすべてのプロセス関連のデータをそれぞれの cbevents Solr コアに格納します。ここでは、他のクラスターとは独立してインデックスが管理されています。このデータは、マスターによって分散クエリが実行された場合に取得されます。バイナリ データはマスターに転送され、保存および管理されます。この処理は、各センサーの一意のバイナリごとに 1 回のみ実行されます。バイナリ メタデータとは異なり、バイナリのコピーはミニオンにローカルに格納されます。コピーの送信と格納が実行されるのは、各 Cb Response クラスターまたは各 Cb Response インスタンスにつき 1 回のみです。バイナリ格納の分散方法は、そのバイナリの送信先となるセンサーによって異なります。このデータ フローは、次の図のようになります。



Cb Response クラスターの構成

このセクションでは、マスター ノードと任意の数のミニオン ノードからなる Cb Response クラスターを構成する方法について説明します。このプロセスでは、初期状態ではクライアント ソフトウェアが構成されていない 2 つの汎用 Cb Response サーバーをインストールします。このプロセスの目的は、`cb-master` をマスター ノードとして持ち、`cb-minion` をミニオン ノードとして持つ Cb Response クラスターを作成することにあります。

最初に `cbinit` を実行する必要があります。この cbinit はマスター ノードで実行されます。

ノードがマスターであり、かつクラスター内にノードが 4 つ以上ある場合は、マスター ノードでイベントが保存されないように以下のパラメーターを追加します。

```
--no-solr-events
```

注意

以下の手順は、マスター ノードに Cb Response RPM がインストールされており、かつ `yum install cb-enterprise` コマンドが実行済みであることを前提としています。ミニオンは単純な CentOS の汎用インストールです。詳細については、「新規 Cb Response サーバーのインストールと初期化」(21 ページ) を参照してください。

クラスター構成を設定する :

1. マスター ノードで、`cbinit` コマンドを発行します。このとき、各自のインストール環境に応じて、適切な `--no-solr-events` フラグを指定します。この例では、マスター ノードでイベントが保存されないように指定します。

   ```
   /usr/share/cb/cbinit --no-solr-events
   ```

   以下の Cb Response クラスター管理コマンドライン ツール オプションを使用して、クラスターの構成を開始します。
   `[root@cb-master~]# /usr/share/cb/cbcluster`
   使用方法 :`cbcluster COMMAND [CMD OPTIONS]`
   使用可能なコマンド :
   `help` – ヘルプ画面を表示します
   `start` – クラスターを開始します
   `stop` – クラスターを停止します
   `status` – クラスターの実行ステータスを取得します
   `add-node` – クラスターにミニオン ノードを追加します
   `change-node` – 既存のクラスター ノードのパラメーターを変更します
   `remove-node` – クラスターからミニオン ノードを削除します

2. `cb-master` から以下のコマンドを実行して、クラスターの構成を開始します (「ベスト プラクティス」(65 ページ) を参照してください)。

   ```
   [root@cb-master~]# /usr/share/cb/cbcluster add-node
   ```

3. 求めに応じて次の情報を入力します。
   - リモート ノードのホスト名または IP アドレスを求められたら、ミニオン ノードになるサーバーの IP アドレスを入力します。例 :172.xx.xxx.xxx。
   - ミニオン ノードになるサーバーのパスワードを求められたら、そのサーバーのルート パスワードを入力します。Cb Response のパスワードは入力しないでください。

   各ミニオン上に Cb Response ソフトウェアがインストールされます。追加された各ノードでは、マスターで使用される Yum リポジトリの構成が使用されます。
4. ミニオン ノードの構成がすべて完了したら、以下のように入力してクラスター サービスを開始します。

```
[root@cb-master~]# /usr/share/cb/cbcluster start
```

5. 結果は、以下のように構成ファイルで確認できます。

```
[root@cb-master~]# cd /etc/cb/
[root@cb-master~]# less cluster.conf

#####################################################################
#############
#
# /etc/cb/cluster.conf:
# This file contains Cb Response server cluster configuration,
# which includes the list of participating nodes and Solr shards
present on
#     every one of those nodes.
#
# NOTE:The contents of this file are being managed by
#  /usr/share/cb/cbcluster command line tool and any changes
made here may
# be overwritten next time that tool is used.
#
#####################################################################
############

[Cluster]
NodeCount=2
NextSlaveAutoInc=2

[Master]
Host=172.16.100.110
HasEvents=False
User=root

[Slave1] Host=172.16.100.111
HasEvents=True
User=root
```

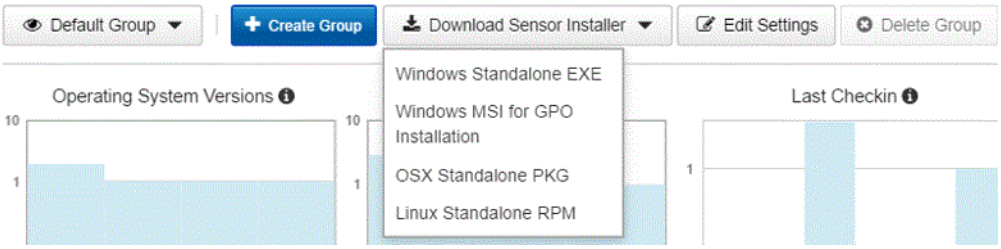
6. Cb Response にログインします。
7. コンソールの右上隅に表示されている **[< ユーザー名 >] > [Settings (設定)]** の順に選択します。
8. [Settings (設定)] パネルで、[Server Nodes (サーバー ノード)] を選択し、クラスターのサーバー ノードを表示します。

センサーのインストールと検証

次の手順ではクラスター ノードにセンサーを追加し、その後、インストールしたセンサーがマスター ノードの Cb Response コンソールに表示されることを確認します。いずれのセンサーもマスターへのレポートを行うため、プロセス イベントとバイナリ データをレポートするためのノードが各センサーに割り当てられます。

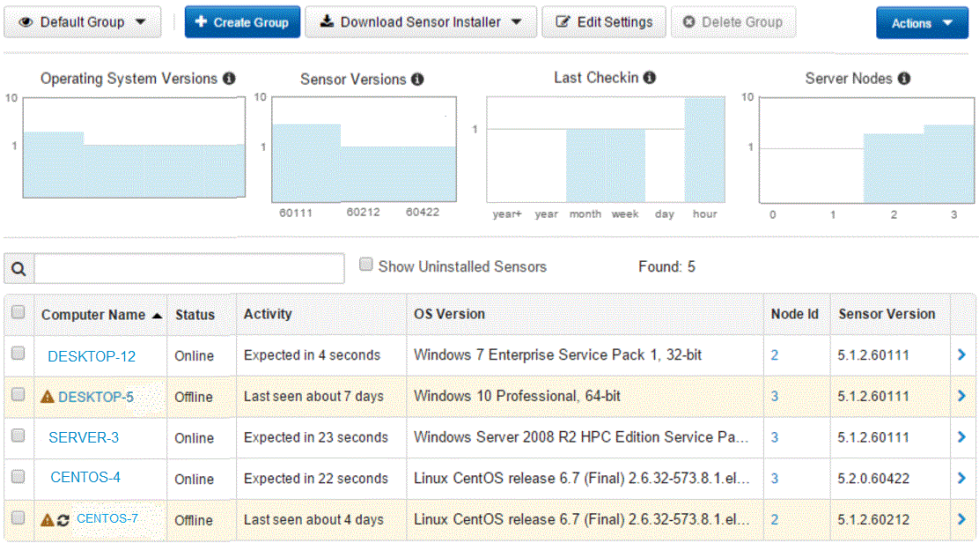
センサーをインストールし、Cb Response で表示されることを確認する :

1. Cb Response にログインします。
2. 左側のナビゲーション メニューから、[Sensors (センサー)] を選択します。
3. **[Download Installer (インストーラーのダウンロード)]** を選択し、インストールするセンサーを選択します。



4. クラスター ノードにセンサーをインストールします。センサーのインストール方法については、『Cb Response ユーザー ガイド』の 「センサーのインストール」の章を参照してください。

5. すべてのセンサーをクラスター ノードにインストールしたら、左側のナビゲーション メニューの **[Sensors (センサー)]** に再度移動して、インストールしたセンサーがマスター ノードの Cb Response コンソールに表示されることを確認します。



ベスト プラクティス

Cb Response クラスターの構成時は、『Cb Response Operating Environment Requirements　guide』を参照することをお勧めします。このガイドでは、Cb Response の展開におけるパフォーマンスとスケーラビリティの考慮事項が説明されています。

既存のクラスターへのミニオンの追加

既存のクラスターにミニオンを追加する :

1. マスターとミニオンが、同じバージョンの Cb Response 上で構成されていることを確認します。
2. マスター サーバーにログインし、以下のように入力してクラスター サービスを停止します。

   ```
   /usr/share/cb/cbcluster stop
   ```

3. 以下のように入力してステータスを確認し、クラスター サービスが停止されていることを確認します。

   ```
   /usr/share/cb/cbcluster status
   ```

4. クラスターにノードを追加する次のコマンドを実行します。

   ```
   /usr/share/cb/cbcluster add-node
   ```

5. マスター サーバーで以下のように入力し、クラスターを開始します。

   ```
   /usr/share/cb/cbcluster start
   ```

既存のクラスターからのミニオンの削除

不要なミニオンをクラスターから削除すると、クラスター全体のサイズが縮小され、クラスターの展開が簡素化されます。これにより、ネットワークのオーバーヘッドが軽減されるため、クラスターのパフォーマンスが向上します。

ベスト プラクティス

既存のクラスターからミニオンを削除する際は、あらかじめ以下のベスト プラクティスに即していることを確認してください。サイズを縮小したクラスターにすべてのエンドポイントをサポートするだけの処理能力があることを確認する場合は、『Cb Response Server Sizing (OER) Guide』を参照してください。

- 削除後に残ったミニオンにはすべてのデータを保存できるだけのディスク容量が必要です。
  - 各センサーの 1 日あたりのディスク使用量を計算したうえで、残ったミニオンですべてのデータを保持するために必要なディスク容量を再度計算します。以下は計算の具体例です。

    ミニオンが 6 台、エンドポイントが 40,000 あるクラスターをミニオンが 3 台のクラスターに縮小するとします。データの保持期間は 30 日です。このクラスターは、それぞれのミニオンにあるエンドポイントの数が 18,750 未満であるため、Cb Response v6.0 以降でサポートされます。

    各センサーの現在の 1 日あたりのディスク使用量を計算すると (クラスターの全データ ボリューム ストレージを現在の 1 日あたりの保持データ量とエンドポイント数で割ると) 20 MB という結果になりました。

    縮小後のストレージは全体として、20 MB * 40,000 (エンドポイント数) * 30 (日数) = 24 TB (ノードあたり 8 TB) 以上必要となります。

    今後センサーのアクティビティ増加に対応できるよう、算出した量に使用可能なディスク容量の 20 ～ 30% を上乗せします。
  - 数日間のコールド ストレージ用として追加のディスク容量を確保する必要があります。
- マスター クラスター ノードには、削除されたすべてのミニオンのバイナリ ファイルを保存するのに十分なディスク容量が必要です。バイナリ ファイルは、通常イベントほど多くのディスク容量を必要とするわけではありませんが、それでも考慮する必要があります。マスターに必要な追加のディスク容量を計算する場合は、Cb Response にログインして左側のナビゲーション メニューにある [Server Dashboard (サーバー ダッシュボード)] に移動し、ミニオンの [Storage statistics (ストレージ統計)] を確認します。削除するミニオンのバイナリのサイズをすべて合算すると、マスター ノード上で必要な追加の最大ディスク容量がわかります。
- 削除後に残ったミニオンには、(OER に従って) センサーの負荷に十分対応できる CPU 能力とメモリ容量が必要です。

読み取り専用ミニオン

既存のクラスターからミニオンを削除する場合は、あらかじめそれらのミニオンの状態を読み取り専用にしておく必要があります。読み取り専用のミニオンは、API やユーザー インターフェイスを介して検索できますが、センサーのチェック

インやイベント / データストアのプッシュには使用されません。ミニオンの状態が読み取り専用である間、バイナリ ファイルはマスターにコピーされます。

ミニオンにイベント データは追加されず、既存のデータは通常動作の場合と同様、定期的に削除されます。そのため、読み取り専用のミニオンは、保持期間が経過すると、完全に非アクティブの状態になります。指定した保持期間が経過すれば、読み取り専用のミニオンは安全に削除することができます。

読み取り専用のミニオンは、必要に応じていつでもアクティブなミニオンに戻すことができます。

ミニオンの削除

ミニオンを読み取り専用としてマークする手順、ミニオンの読み取り専用のマークを解除する手順、およびクラスターのミニオンを削除する手順は以下のとおりです。node_id　パラメーターが必要な場合は、該当ノードの　`/etc/cb/cluster.cfg` に記載されています。

ミニオンを読み取り専用としてマークする :

1. クラスターを停止します。
2. 削除する各ミニオンに対して、以下のコマンドを実行します。

   ```
   cbcluster change-node -N {node_id} -R True
   ```
3. (オプション) イベントが保存されないマスター ノードにイベントが保存されるようにする場合は、以下のコマンドを実行します。

   ```
   cbcluster change-node -E True
   ```
4. クラスターを開始します。

ミニオンの読み取り専用のマークを解除する :

1. クラスターを停止します。
2. 削除する各ミニオンに対して、以下のコマンドを実行します。

   ```
   cbcluster change-node -N {node_id} -R False
   ```
3. (オプション) マスター ノードをイベントが保存されない状態に戻す場合は、以下のコマンドを実行します。

   ```
   cbcluster change-node -E False
   ```
4. クラスターを開始します。

クラスターのミニオンを削除する :

1. クラスターを停止します。
2. 削除する各ミニオンに対して、以下のコマンドを実行します。

   ```
   cbcluster remove-node -N {node_id}
   ```
3. クラスターを開始します。

削除されたミニオンは、クラスターに属さなくなるため、シャットダウンしてプロビジョニング解除することができます。

備考

- 削除できるのは、読み取り専用のミニオンのみです。
- ミニオンは、ストアファイルがすべてマスター ノードにコピーされている場合にのみ削除できます。コピーされていないストアファイルがある場合はエラーが表示され、移動すべきファイルがあといくつ残っているかが明示されます。ミニオンの削除は、後で再度試行できます。

クラスター ノードのアップグレード

現在、クラスター ノードをアップグレードする方法には、以下の 2 つがあります。

- マスター　ノードをアップグレードした後、そのマスター　ノードで　`/usr/share/cb/cbcluster start` を実行します。変更された構成ファイルを示す RPMNEW ファイルがどのクラスター ノードにも存在しない場合、すべてのクラスター ノードがアップグレードされます。新しい RPMNEW ファイルが存在する場合は、`cbcluster start` で処理が停止し、手動でノードをアップグレードするように求めるメッセージが表示されます。
- `cbupgrade` を使用して各クラスター　ノードを手動でアップグレードします。新しい RPMNEW ファイルが存在する場合は、常にこの方法を実行する必要があります。

重要

アップグレード中に RPMNEW ファイルが検出された場合は、ノード上にある現在の構成ファイルを新しい構成情報によって調整した後、そのノードから RPMNEW ファイルを削除する必要があります。これにより、 `cbcluster start` が正常に完了します。

クラスター ノードをアップグレードする :

1. マスター　ノードにログインし、以下のように入力してクラスターを停止します。

   ```
   /usr/share/cb/cbcluster stop
   ```

2. クラスターが完全に停止したら、再起動します。

   ```
   /usr/share/cb/cbcluster start
   ```

   RPMNEW ファイルが存在しない場合、クラスター ノードは、それ以降の操作なしにアップグレードされます。新しい RPMNEW ファイルが存在する場合は、以下の手順を使用して、各ノードを手動でアップグレードする必要があります。

クラスター ノードの手動アップグレード

クラスター　ノードを手動でアップグレードする手順は、スタンドアロンの Cb Response サーバーをアップグレードする手順とほぼ同じです (「Cb Response サーバーのアップグレード」(31 ページ) を参照)。また、クラスター ノード間の通信が正しく行われるようにするために、その他の手順も実行する必要があります。

クラスターのミニオン サーバーを手動でアップグレードする :

1. マスター ノードをアップグレードした後、クラスター内のその他の各マシンにログインして cb-enterprise をアップグレードします。

   ```
   yum upgrade cb-enterprise
   ```

2. 各クラスター ノードにログインし、以下のように入力して、それぞれのデータ スキーマを最新バージョンにアップグレードします。

   ```
   /usr/share/cb/cbupgrade [--proceed-on-rpmnew] [--non-interactive]
   ```

   引数が指定されていない場合、変更された構成ファイルを示す Cb Response の RPMNEW ファイルが検出されると、このコマンドの実行が停止され、それらのファイルの名前がレポートされます。RPMNEW ファイルの内容をそれと同じ名前の既存の構成ファイルの内容と比較し、違いを手動で解決する必要があります。その後、RPMNEW ファイルを削除すると、cbupgrade の実行が完了します。

   `proceed-on-rpmnew` 引数が指定されている場合、cbupgrade の実行は完了しますが、アップグレード後に対処できるように、検出された RPMNEW ファイルが併せてレポートされます。この方法は、アップグレードを途中で停止することなく完了した後、構成ファイルを落ち着いて編集する必要がある場合、またはそれを望む場合に便利です。

   `--proceed-on-rpmnew` オプションを `--non-interactive` オプションと一緒に使用すると、アップグレードを続行するためにキーを押す必要はなく、またそれを求めるメッセージも表示されません。ただし、`non-interactive` オプションを使用せずにコマンドを実行した場合と異なり、cb-enterprise サービスは自動的に開始しません。

   警告

   cbupgrade プロセスの最後にサービスを開始するように求めるメッセージが表示されますが、サービスを開始しないでください。

3. ソフトウェアのバージョンが新しくなると、通信ポートの要件が変わることがあります。そのため、firewall の設定を更新する必要があるかどうかを、以下のようにして確認する必要があります。

   a. firewall を手動で管理する場合は、次のコマンドを実行して、追加する必要のあるルールを特定します。

      ```
      /usr/share/cb/cbcheck firewall -l
      ```

b. firewall の設定を自動的に調整するには、次のコマンドを実行して、ユーティリティ経由で各設定を適用します。

```
usr/share/cb/cbcheck firewall --apply
```

4. すべてのノードをアップグレードしたら (ただし開始しない)、マスター ノードにログイン後、次のコマンドを発行して、クラスター全体を一斉に開始します。

```
/usr/share/cb/cbcluster start
```

この手順はアップグレード後に実行することが重要です。これは、cbcluster ツールを使用してクラスターを開始すると、同期する必要がある構成ファイルがクラスター内のすべてのノードで再配布されるためです。ソフトウェアのバージョンを新しくすることで構成が変更される場合は、この手順を実行することによって各ミニオン ノードが確実に更新されます。

第 6 章 l

非ルート ユーザーとして CBCLUSTER を使用する

この章では、非ルート ユーザーとして CBCLUSTER コマンドを使用する方法について説明します。

セクション

トピック	ページ
概要	72
必要なユーザー権限	72
ユーザーの定義	74

概要

`cbcluster` は、マスター ノード上の Cb Response クラスターを管理および構成する目的で使用するコマンド ライン ユーティリティです。以下のオプションをサポートしています。

- `cbcluster start`
- `cbcluster stop`
- `cbcluster status`
- `cbcluster add-node`

これらのコマンドは *root* 権限で実行します。また、マスター ノード上で `/usr/share/cb/cbcluster` を起動するには *sudo* 権限が必要です。さらに、クラスターに新しいミニオン ノードを追加する目的で使用する `cbcluster add-node` コマンドでは、ミニオンに接続してクラスター構成用の一連のコマンドをリモートで実行することになります。

備考

- `cbcluster add-node` コマンドを実行する前に、マスターとミニオンが同じバージョンの Cb Response 上にあることを確認してください。バージョンが異なる場合は、エラー メッセージが表示されます。
- `cbcluster reshard` コマンドは、廃止予定でありサポートされていません。

Cb Response バージョン 5.1.1 では非ルート ユーザーをリモート ユーザーとして定義し、ミニオンとのやり取りやコマンドの実行に使用することができます。以前のリリースではクラスターにミニオン ノードを追加する場合、`cbcluster` ユーティリティを使用するには、対象のミニオン ノードでルート ユーザーを使用できることが必要でした。今回のリリースではこの要件が緩和されており、ミニオン ノードを非ルート ユーザーとして構成できるようになりました。

本章では非ルート ユーザーに対して必要な権限と、クラスター セットアップ時の使用方法について説明します。

必要なユーザー権限

ミニオン上のリモート　ユーザーに以下の権限を割り当てておかないと、`cbcluster` を起動し、非ルート ユーザーとしてミニオンに接続することはできません。

- ミニオン ノードに対する SSH アクセス権
- 以下に示すコマンドに対する sudo 権限。ユーザーは、NOPASSWD で実行できるように設定されている必要があります。

非ルート ユーザーで `cbcluster` コマンドを使用するには、以下のエントリを sudoers ファイルに追加します。

```
## Required sudo privileges on minion to run cbcluster add-node
Cmnd_Alias HOSTNAME = /bin/hostname
Cmnd_Alias CB_INIT = /usr/share/cb/cbinit
Cmnd_Alias YUM_INSTALL_CB = /usr/bin/yum install cb-enterprise -y
Cmnd_Alias YUM_INSTALL_RSYNC = /usr/bin/yum install rsync -y
Cmnd_Alias MKDIR_ETC_CB = /bin/mkdir /etc/cb --mode=755
Cmnd_Alias MKDIR_ETC_CB_CERTS = /bin/mkdir /etc/cb/certs --mode=755
Cmnd_Alias COPY_ALLIANCE_CRT = /usr/bin/rsync --remove-source-files -
-verbose /tmp/.cb_tmp/carbonblack-alliance-client.crt /etc/cb/certs/
carbonblack-alliance-client.crt
Cmnd_Alias COPY_SERVER_CRT = /usr/bin/rsync --remove-source-files --
verbose /tmp/.cb_tmp/cb-server.crt /etc/cb/certs/cb-server.crt
Cmnd_Alias COPY_CLIENT_CA_CRT = /usr/bin/rsync --remove-source-files
--verbose /tmp/.cb_tmp/cb-client-ca.crt /etc/cb/certs/cb-client-
ca.crt
Cmnd_Alias COPY_ALLIANCE_KEY = /usr/bin/rsync --remove-source-files -
-verbose /tmp/.cb_tmp/carbonblack-alliance-client.key /etc/cb/certs/
carbonblack-alliance-client.key
Cmnd_Alias COPY_SERVER_KEY = /usr/bin/rsync --remove-source-files --
verbose /tmp/.cb_tmp/cb-server.key /etc/cb/certs/cb-server.key
Cmnd_Alias COPY_CLIENT_CA_KEY = /usr/bin/rsync --remove-source-files
--verbose /tmp/.cb_tmp/cb-client-ca.key /etc/cb/certs/cb-client-
ca.key
Cmnd_Alias COPY_CB_REPO = /usr/bin/rsync --remove-source-files --
verbose /tmp/.cb_tmp/CarbonBlack.repo /etc/yum.repos.d/
CarbonBlack.repo
Cmnd_Alias COPY_CLUSTER_CONF = /usr/bin/rsync --remove-source-files -
-verbose /tmp/.cb_tmp/cluster.conf /etc/cb/cluster.conf
Cmnd_Alias COPY_ERLANG_COOKIE = /usr/bin/rsync --remove-source-files
--verbose /tmp/.cb_tmp/.erlang.cookie /var/cb/.erlang.cookie
Cmnd_Alias COPY_SERVER_LIC = /usr/bin/rsync --remove-source-files --
verbose /tmp/.cb_tmp/server.lic /etc/cb/server.lic
Cmnd_Alias COPY_SERVER_TOKEN = /usr/bin/rsync --remove-source-files -
-verbose /tmp/.cb_tmp/server.token /etc/cb/server.token
Cmnd_Alias CBCHECK_FIREWALL = /usr/share/cb/cbcheck firewall --apply
Cmnd_Alias CB_ENTERPRISE = /etc/init.d/cb-enterprise
Cmnd_Alias CAT_VERSION = /bin/cat /usr/share/cb/VERSION
Cmnd_Alias CBUPGRADE = /usr/share/cb/cbupgrade --non-interactive
Cmnd_Alias CBUPGRADE_CHECK = /usr/share/cb/cbupgrade --check

my_user        ALL=(ALL)  NOPASSWD:HOSTNAME, CB_INIT, YUM_INSTALL_CB,
YUM_INSTALL_RSYNC, MKDIR_ETC_CB, MKDIR_ETC_CB_CERTS,
COPY_ALLIANCE_CRT, COPY_SERVER_CRT, COPY_CLIENT_CA_CRT,
COPY_ALLIANCE_KEY, COPY_SERVER_KEY, COPY_CLIENT_CA_KEY, COPY_CB_REPO,
COPY_CLUSTER_CONF, COPY_ERLANG_COOKIE, COPY_SERVER_LIC,
COPY_SERVER_TOKEN, CBCHECK_FIREWALL, CB_ENTERPRISE, CAT_VERSION,
CBUPGRADE, CBUPGRADE_CHECK
```

これらのエントリをコピーして使用環境に貼り付けられるようにフォーマットしたバージョンについては、https://community.carbonblack.com/docs/DOC-5692 を参照してください。

`cbcluster` コマンドを実行すると最初に確認が行われ、必要な権限のいずれかが設定されていない場合は、不足している権限に関するメッセージが表示されます。

ユーザーの定義

> **注意**
> 競合が発生する可能性があるため、ユーザー名として “cb” を選択しないでください。

ルート以外のユーザーとして `cbcluster` を起動し、ミニオン ノードを構成する方法は以下の 2 種類です。

- まだクラスターに追加されていない新しいミニオンの場合は、以下に示す新しい add-node オプションを指定することで、add-node の実行時に新しいユーザーを設定することができます。

  `--user=<arg>`

  以下に例を示します。

  `$ /usr/share/cb/cbcluster add-node --hostname <my_host> --user <my_user>`

  これにより、 `<my_host>` に対するすべてのリモート アクションを `<my_user>` として実行するように `cbcluster` が設定されます。 `/etc/cb/cluster.conf` ファイルが更新され、key - value ペア "`User=<my_user>`" を追加した新しい設定が反映されます。
- 既存のミニオンの場合は、`/etc/cb/cluster.conf` ファイルを直接修正できます。ファイルをエディターで開き、各ミニオン ノードについて key-value ペア "`User=<my_user>`" を追加します。`<my_user>` の部分は、そのミニオンに接続するときに `cbcluster` で使用する実際のユーザー名に置き換えてください。