アラーム ログ相関の実装

このモジュールでは、アラーム ログ相関の設定に関する概念とタスクについて説明します。アラーム ログ相関は、さまざまなアプリケーションおよびシステム サーバで生成されたメッセージのグループ化機能とフィルタリング機能、およびルータ上のルート メッセージの分離機能を含めるように、システム ロギングを拡張します。

- アラーム ログ相関の実装 （1 ページ）

アラーム ログ相関の実装

アラーム ログ相関は、さまざまなアプリケーションおよびシステム サーバで生成されたメッセージのグループ化機能とフィルタリング機能、およびルータ上のルート メッセージの分離機能を含めるように、システム ロギングを拡張します。このモジュールでは、アラーム ログ相関の設定とアラーム ログのモニタリングに関連する概念とタスクについて説明します。

アラーム ログ相関の実装に関する前提条件

適切なタスク ID を含むタスク グループに関連付けられているユーザ グループに属している必要があります。このコマンド リファレンスには、各コマンドに必要なタスク ID が含まれます。ユーザ グループの割り当てが原因でコマンドを使用できないと考えられる場合、AAA 管理者に連絡してください。

アラーム ログ相関の実装に関する情報

アラーム ロギングおよびデバッグ イベント管理システム

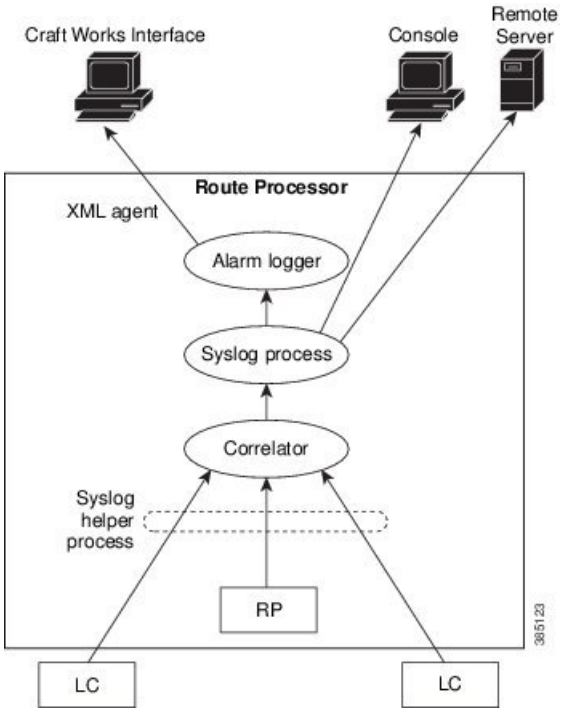
Cisco IOS XR ソフトウェアのアラーム ロギングおよびデバッグ イベント管理システム（ALDEMS）は、システム サーバおよびアプリケーションから転送されるアラーム メッセージをモニタリングし、格納するために使用されます。また、ALDEMS は、単一の根本原因のために転送されたアラーム メッセージ同士を相関します。

ALDEMS は、Cisco IOS XR ソフトウェアの基本的なロギング機能およびモニタリング機能を拡大して、数百のラインカードと数千のインターフェイスを備える可能性のある高度に分散化されたシステムに必要とされるレベルのアラームとイベント管理を実現します。

Cisco IOS XR ソフトウェアは、システムのノード全体にロギング アプリケーションを分散することによって、この必要なレベルのアラーム管理およびイベント管理を実現します。

図 1 : ALDEMS コンポーネント通信 （2 ページ）に、ALDEMS を構成するコンポーネント間の関係を示します。

図 *1 : ALDEMS* コンポーネント通信



コリレータ

コリレータは、ルータ上のノード全体に分散されたシステムロギング（syslog）ヘルパープロセスからメッセージを受け取り、syslog メッセージを syslog プロセスに転送します。ロギング相関ルールが設定されている場合は、コリレータはルールで指定されているメッセージと一致するメッセージを検索して、そのメッセージをキャプチャします。コリレータは、一致を検出すると、ルールに指定されているタイムアウト間隔に対応するタイマーを開始させます。コリレータは、タイマーの期限が切れるまで、ルール内のメッセージとの一致を検索し続けます。根本原因メッセージを受信した場合は、相関が実行されます。受信しなかった場合は、キャプチャされたメッセージがすべてsyslogに転送されます。相関が実行された場合、相関メッセージはロギング相関バッファに保存されます。相関メッセージの各セットには、コリレータにより相関 ID がタグ付けされます。

システム ロギング プロセス

アラーム ロガーは、ルータに転送されるシステム ロギング メッセージの最終宛先です。アラーム ロガーには、ロギング イベント バッファ内のアラーム メッセージが保存されます。ロギング イベント バッファは循環バッファであるため、いっぱいになるとバッファ内の最も古いメッセージが上書きされます。

アラーム ロガー

アラーム ロガーは、ルータに転送されるシステム ロギング メッセージの最終宛先です。アラーム ロガーには、ロギング イベント バッファ内のアラーム メッセージが保存されます。ロギング イベント バッファは循環バッファであるため、いっぱいになるとバッファ内の最も古いメッセージが上書きされます。

（注） アラームは、ロギング イベント バッファ内で優先順位付けされます。アラーム レコードを上書きする必要がある場合は、ロギング イベント バッファは、最初に非バイステート アラーム、次に CLEAR ステートのバイステート アラーム、最後に SET ステートのバイステート アラームの順序でメッセージを上書きしていきます。

SET ステートのバイステート アラームにより発行されたメッセージでテーブルがいっぱいになると、（着信時刻ではなくメッセージのタイム スタンプ基準で）一番古いバイステート アラームがその他のメッセージよりも先に上書きされます。したがって、メモリ消費量が要件内に収まるように、ロギング イベント バッファおよびロギング相関バッファのバッファ サイズを調整する必要があります。

テーブルフル アラームは、ロギング イベント バッファが一巡するたびに生成されます。しきい値超過通知は、ロギング イベント バッファが容量のしきい値に到達するたびに生成されます。

ロギング イベント バッファに保存されたメッセージに対してクライアントからクエリーを実行して、特定の条件に一致するレコードを特定できます。アラーム ロギング メカニズムにより、各アラーム メッセージには連番で一意の ID が割り当てられます。

アラーム ログ相関の設定

必要に応じてアラーム ログ相関を設定するには、この項の設定タスクを実行します。

ロギング相関ルールの設定

ロギング相関を使用して、システム パフォーマンスに影響を及ぼすイベントの最上位ルート メッセージを分離できます。相関ルールが設定されている場合、セカンダリ（非根本原因）メッセージを生成する共通ルート イベントを分離して syslog に送信することで、セカンダリ メッセージを抑制できます。オペレータは、ロギング コリレータ バッファから相関メッセージをすべて取得して、発生した相関イベントを表示できます。相関ルールをルータ全体に適用した場合、メッセージのコンテキストまたはロケーション設定にかかわらず、設定されたルールの原因値に一致するメッセージだけで相関が発生します。相関ルールを特定のコンテキスト

またはロケーションのセットに適用した場合、ルールで設定されている原因値にするメッセージ、およびこれらのコンテキストまたはロケーションのいずれか1つに一致するメッセージだけで相関が発生します。

相関ルールが設定され適用された場合、コリレータにより、ルールの指定に従ってメッセージの一致が検索されます。タイムアウトは、一致が見つかったらメッセージ検索の時間間隔を指定するように設定できます。タイムアウトは、コリレータが相関ルールで指定されたアラームメッセージをキャプチャしたときに開始されます。

設定例

次に、ロギング相関ルールを設定して適用する例を示します。この例では、タイムアウトは60000 ミリ秒として設定されています。

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# logging correlator rule rule1 type stateful
RP/0/RP0/CPU0:Router(config-corr-rule-st)# timeout 60000
RP/0/RP0/CPU0:Router(config)# logging correlator apply-rule rule1
RP/0/RP0/CPU0:Router(config-corr-apply-rule)# all-of-router
or
RP/0/RP0/CPU0:Router(config-corr-apply-rule)# location 0/1/CPU0
or
RP/0/RP0/CPU0:Router(config-corr-apply-rule)# context HundredGigE_0_0_1_0
RP/0/RP0/CPU0:Router(config)# commit
```

ロギング相関ルール セットの設定

ロギング相関ルール セットを設定し、複数の相関ルールを組み込むことができます。

設定例

次に、複数の相関ルール用にロギング相関ルールセットを設定して適用する例を示します。ロギング相関ルール セットは、ルータ全体または特定のコンテキストまたは場所に適用できます。

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# logging correlator ruleset ruleset1
RP/0/RP0/CPU0:Router(config-corr-ruleset)# rulename stateful_rule1
RP/0/RP0/CPU0:Router(config-corr-ruleset)# rulename stateful_rule2
RP/0/RP0/CPU0:Router(config)# logging correlator apply ruleset ruleset1
RP/0/RP0/CPU0:Router(config-corr-apply-rule)# all-of-router
or
RP/0/RP0/CPU0:Router(config-corr-apply-rule)# location 0/2/CPU0
or
RP/0/RP0/CPU0:Router(config-corr-apply-rule)# context HundredGigE_0_0_1_0
RP/0/RP0/CPU0:Router(config)# commit
```

根本原因アラームと非根本原因アラームの相関

根本原因メッセージは、相関ルールに設定された最初のメッセージ（カテゴリ、グループ、およびコードの3つが設定されたもの）により定義されます。根本原因メッセージは、必ずsyslogプロセスに転送されます。根本原因を 1 つ以上の非根本原因アラームと相関させ、それらをルールの一部として設定することができます。

設定例

次の例では、根本原因を1つ以上の非根本原因アラームと相関させ、それらをルールに設定する方法を示します。

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# logging correlator rule rule_stateful type stateful
RP/0/RP0/CPU0:Router(config-corr-rule-st)# rootcause CAT_BI_1 GROUP_BI_1 CODE_BI_1
RP/0/RP0/CPU0:Router(config-corr-rule-st)# nonrootcause
RP/0/RP0/CPU0:Router(config-corr-rule-st-nonrc)# alarm CAT_BI_2 GROUP_BI_2 CODE_BI_2
RP/0/RP0/CPU0:Router(config)# commit
```

階層的な相関ルール フラグの設定

階層的な相関は、1つのアラームがあるルールの根本原因であり、かつ別のルールの非根本原因でもある場合、およびアラームが生成され、結果として両方のルールに関連する正常な相関となった場合に発生します。非根本原因アラームに起こったことが、相関根本原因アラームの動作を決定します。これらの階層に関連するステートフル動作を制御する必要があるケース、および非バイステートアラームの再配置および再発行などのフラグを実装する必要があるケースがあります。階層的な相関および相関フラグの詳細については、を参照してください。階層的な相関 （10 ページ）

設定例

次に、階層的な相関ルールのフラグを設定する例を示します。

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# logging correlator rule rule_nonstateful type nonstateful
RP/0/RP0/CPU0:Router(config-corr-rule-st)# reissue-nonbistate
RP/0/RP0/CPU0:Router(config-corr-rule-st)# reparent
RP/0/RP0/CPU0:Router(config)# commit
RP/0/RP0/CPU0:Router# show logging correlator rule all (optional)
```

ロギング抑制ルールの設定

アラームロギング抑止機能を使用すると、抑止するアラームのタイプを指定するロギング抑止ルールを定義することで、アラームのロギングを抑制できます。ロギング抑制ルールでは、すべてのタイプのアラーム、または特定のメッセージカテゴリ、グループ名、およびメッセージコードを持つアラームを指定できます。ロギング抑制ルールは、ルータ上のすべてのロケーションから発生するアラームに対して適用するか、または特定のノードから発生するアラームに対して適用できます。

設定例

次の例に、ロギング抑制ルールの設定方法を示します。

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# logging suppress rule infobistate
RP/0/RP0/CPU0:Router(config-suppr-rule)# alarm MBGL COMMIT SUCCEEDED
RP/0/RP0/CPU0:Router(config)# logging suppress apply rule infobistate
RP/0/RP0/CPU0:Router(config-suppr-apply-rule)# all-of-router
RP/0/RP0/CPU0:Router(config)# commit
```

ロギング イベント バッファ設定の変更

アラーム ロガーには、ロギング イベント バッファ内のアラーム メッセージが保存されます。ロギング イベント バッファは、バッファがいっぱいになったときに最も古いメッセージを上書きします。ロギング イベント バッファの設定は、ネットワークのパフォーマンスに影響するユーザアクティビティ、ネットワーク イベント、システム設定イベントの変更、またはネットワーク モニタリング要件の変更に対応して調整できます。適切な設定は、システムの設定および要件に応じて異なります。しきい値超過通知は、ロギング イベント バッファが容量のしきい値に到達するたびに生成されます。

設定例

次の例は、ロギング イベント バッファのサイズ、しきい値、およびアラーム フィルタの設定を示しています。

```
RP/0/RP0/CPU0:Router# configure terminal
RP/0/RP0/CPU0:Router(config)# logging events buffer-size 50000
RP/0/RP0/CPU0:Router(config)# logging events threshold 85
RP/0/RP0/CPU0:Router(config)# logging events level warnings
RP/0/RP0/CPU0:Router(config)# commit
```

ロギング相関バッファ設定の変更

相関が実行された場合、相関メッセージはロギング相関バッファに保存されます。ロギング相関バッファのサイズは、予想される着信相関メッセージを収容できるように調整できます。レコードを指定してバッファからレコードを削除したり、バッファにあるレコードをすべてクリアしたりできます。

設定例

次の例では、相関バッファのサイズを設定し、バッファからレコードを削除します。

```
RP/0/RP0/CPU0:Router# configure terminal
RP/0/RP0/CPU0:Router(config)# logging correlator buffer-size 100000
RP/0/RP0/CPU0:Router(config)# exit
RP/0/RP0/CPU0:Router# clear logging correlator delete 48 49 50 (optional)
RP/0/RP0/CPU0:Router# clear logging correlator delete all-in-buffer (optional)
```

バイステート アラームのアラーム ソース ロケーション表示フィールドのイネーブル化

バイステート アラームは、システムのハードウェアに関連付けられている状態の変化によって生成されます。バイステート アラーム メッセージの形式は、syslog メッセージに似ています。オプションで、出力に実際のアラーム ソースのロケーションが含まれるように設定できます。このアラーム ソースは、アラームをロギングしたプロセスとは異なる場合があります。バイステート アラームの詳細については、を参照してください。 バイステート アラーム (9 ページ)

設定例

次の例に、バイステート アラームのアラーム ソース ロケーション表示フィールドをイネーブルにする方法を示します。

```
RP/0/RP0/CPU0:Router# configure
RP/0/RP0/CPU0:Router(config)# logging events display-location
RP/0/RP0/CPU0:Router(config)# commit
```

SNMP 相関ルールの設定

大規模システムでは、定期的な間隔で出力される多数の SNMP トラップに遭遇する状況になる可能性があります。これらのトラップは、Cisco IOS XR によるトラップの処理時間を延長させます。また、追加のトラップはトラブルシューティングを遅くし、モニタリング システムおよびオペレータの作業負荷を増大します。SNMP アラーム相関は、既存の syslog コリレータから相関機能の一般的な部分を抽出するのに役立ちます。相関ルールを設定して、SNMP トラップの相関ルールを定義し、特定のトラップ宛先に適用することができます。

設定例

次に、SNMP トラップの相関ルールを設定および適用する例を示します。SNMP コリレータのバッファ サイズも 600 バイトに設定されています。バッファ サイズのデフォルト値は 64KB です。

```
RP/0/RP0/CPU0:Router# configure terminal
RP/0/RP0/CPU0:Router(config)# snmp-server correlator buffer-size 600 (optional)
RP/0/RP0/CPU0:Router(config)# snmp-server correlator rule test rootcause A varbind A1
value regex RA1  nonrootcause  trap B varbind B1 index regex RB1
RP/0/RP0/CPU0:Router(config)# snmp-server correlator apply rule test host ipv4 address
1.2.3.4
RP/0/RP0/CPU0:Router(config)# commit
```

SNMP 相関ルールセットの設定

SNMP 相関ルール セットを設定し、複数の SNMP 相関ルールを組み込むことができます。

設定例

次に、複数のルールを 1 つのグループにグループ化できるルールセットを設定する例を示します。指定したグループをホストのセットまたはすべてのホストに適用できます。

```
RP/0/RP0/CPU0:Router# configure terminal
RP/0/RP0/CPU0:Router(config)# snmp-server correlator ruleset rule1 rulename rule2
RP/0/RP0/CPU0:Router(config)# snmp-server correlator apply ruleset rule1 host ipv4 address
 1.2.3.4
RP/0/RP0/CPU0:Router(config)# commit
```

アラーム ロギング相関の詳細

アラーム ロギング相関を使用して、システム パフォーマンスに影響を及ぼすイベントの最上位ルート メッセージを分離できます。たとえば、ラインカードの活性挿抜（OIR）を示す元のメッセージが分離され、根本原因メッセージのみ表示され、同じイベントに関連するすべての後続メッセージが相関になる場合があります。相関ルールが設定されている場合、セカンダリ（非根本原因）メッセージを生成する共通ルート イベントを分離して syslog に送信することで、セカンダリ メッセージを抑制できます。オペレータは、ロギング コリレータ バッファから相関メッセージをすべて取得して、発生した相関イベントを表示できます。

相関ルール

相関ルールを設定して、システム アラームを生成する可能性のあるルート メッセージを分離できます。相関ルールは、アラームロギングおよびデバッグイベント管理システム（ALDEMS）に、不要なメッセージの蓄積に起因する不要なストレスを与えないようにします。各相関ルールはメッセージIDに依存し、メッセージカテゴリ、メッセージグループ名、メッセージコードで構成されます。コリレータプロセスは、メッセージをスキャンしてメッセージの発生を検出します。コリレータがルートメッセージを受信すると、コリレータはそのメッセージをロギング コリレータ バッファに保存し、さらにRPのsyslogプロセスに転送します。その後、syslogプロセスにより、そのルート メッセージはロギング イベント バッファ内のアラーム ロガーに転送され、保存されます。また、ネットワーク デバイスの構成に応じて、ルート メッセージが syslog プロセスからコンソール、リモート ターミナル、リモート サーバ、障害管理システム、および簡易ネットワーク管理プロトコル（SNMP）エージェントなどの宛先に転送される場合もあります。同一の条件に一致する後続のメッセージ（別に発生したルート メッセージを含む）は、ロギング相関バッファに保存され、ルータの syslog プロセスに転送されます。

メッセージが複数の相関ルールに一致する場合、一致したルールすべてが適用され、そのメッセージはロギング コリレータ バッファ内の一致する相関キューすべての一部になります。次のメッセージ フィールドを使用して、ロギング相関ルールのメッセージが定義されます。

- メッセージ カテゴリ
- メッセージ グループ
- メッセージ コード

いずれのメッセージ フィールドでも、ワイルドカードを使用してより幅広いメッセージ セットをカバーできます。

根本原因メッセージ、ステートフル相関および非ステートフル相関を分離するために、ルールには 2 つのタイプの相関が設定されています。非ステートフル相関は、発生後に固定されます。抑制された非根本原因アラームがsyslogプロセスに転送されることはありません。非根本原因アラームはすべて相関バッファにバッファされた状態で残ります。ステートフル相関は、バイステート根本原因アラームがクリアされると、相関発生後に変更される場合があります。アラームがクリアされると、すべての相関された非根本原因アラームはsyslogに送信され、相関バッファからは削除されます。ステートフル相関は、疑われる根本原因がすでに存在しないにもかかわらず、存在し続けている非根本原因状態を検出する場合に役立ちます。

アラーム重大度とフィルタリング

フィルタ設定を使用して、重大度に基づいて情報を表示できます。アラーム フィルタ表示は、アラーム、レコード数、現在のログ サイズ、最大ログ サイズのレポートに使用される重大度の設定を示します。

アラームは、次の表に示されている重大度に応じてフィルタリングできます。

表 *1*:イベント ロギングのアラーム重大度

重大度	システムの状態
0	緊急
1	アラート
2	クリティカル
3	エラー
4	警告
5	通知
6	情報

バイステート アラーム

バイステート アラームは、アクティブから非アクティブへのインターフェイス ステートの変化、ライン カードの活性挿抜（OIR）、またはコンポーネントの温度の変化など、システム ハードウェアに関連するステート変更によって生成されます。デフォルトでは、バイステート アラーム イベントはロギング イベント バッファにレポートされます。情報メッセージおよびデバッグ メッセージはレポートされません。

Cisco IOS XR ソフトウェアには、アラームをリセットおよびクリアする機能があります。システムのアラームをモニタリングする必要のあるクライアントは、モニタリング対象のアラームの状態が変化したときに非同期通知を受信するためのアラーム ロギング メカニズムを登録できます。

バイステート アラーム通知により、次のことがわかります。

- 発信元 ID。発生またはクリアされるアラームの発信元であるリソースを一意に特定します。このリソースは、インターフェイス、ラインカード、または特定用途向け集積回路（ASIC）などです。発信元 ID は、ロケーション、ジョブ ID、メッセージ グループ、メッセージ コンテキストの一意の組み合わせです。

デフォルトでは、バイステート アラーム メッセージの一般形式はすべての syslog メッセージで同一です。

node-id:*timestamp* : *process-name* [*pid*] : *%category-group-severity-code* : *message-text*

次に、バイステート アラーム メッセージの例を示します。

```
LC/0/0/CPU0:Jan 15 21:39:11.325 2016:ifmgr[163]: %PKT_INFRA-LINEPRO
TO-5-UPDOWN : Line protocol on Interface HundredGigE 0/0/1/0, changed state to Down
```

メッセージのテキストには、アラームをロギングしたプロセスのロケーションが含まれます。この例では、アラームは HundredGigE インターフェイス 0/0/1/0 のライン プロトコルによってロギングされました。オプションで、出力に実際のアラーム ソースが含まれるように設定でき

ます。このアラーム ソースは、アラームをロギングしたプロセスとは異なる場合があります。これは、メッセージ テキストの前の追加表示フィールドに表示されます。

アラーム ソースのロケーションを表示した場合、一般形式は次のようになります。

node-id:*timestamp* : *process-name* [*pid*] : *%category-group-severity-code* : *source-location message-text*

次に、アラーム ソースのロケーションが表示されている場合の例を示します。

```
LC/0/0/CPU0:Jan 15 21:39:11.325 2016:ifmgr[163]: %PKT_INFRA-LINEPRO
TO-5-UPDOWN : interface HundredGigE 0/0/1/0: Line protocol on Interface HundredGigE
0/0/1/0, changed state to Down
```

階層的な相関

階層的な相関は、次の条件が満たされた場合に有効になります。

- 1 つのアラームが、あるルールの根本原因であり、かつ別のルールの非根本原因でもある場合。
- アラームが生成され、結果として両方のルールに関連する正常な相関となった場合。

次に、2 つの階層的な相関ルールの例を示します。

ルール 1	**カテゴリ**	**グループ**	**コード**
Root Cause 1	Cat 1	Group 1	Code 1
Non-root Cause 2	Cat 2	Group 2	Code 2
ルール 2			
Root Cause 2	Cat 2	Group 2	Code 2
Non-root Cause 3	Cat 3	Group 3	Code 3

Cause 1、2、3 に対して 3 つのアラームが生成され、すべてのアラームがそれぞれの相関タイムアウト期間内に着信した場合、階層的な相関は次のように出現します。

Cause 1 -> Cause 2 -> Cause 3

相関バッファには、2 つ（Cause 1 と Cause 2 に対して 1 個、Cause 2 と Cause 3 に対して 1 個）の異なる相関が示されます。ただし、階層的な関係は暗黙的に定義されます。

（注） アラームの再配置および再発行などのステートフル動作は、ステートフルとして定義されているルールの場合（つまり、相関が変化する可能性がある場合）にサポートされます。

コンテキスト相関フラグ

コンテキスト相関フラグを使用すると、「コンテキストごと」に相関が行われるかどうかを設定できます。

このフラグを使用すると、ルールが1つ以上のコンテキストに適用される場合のみ、動作が変化するようになります。このフラグは、ルータ全体またはロケーション ノード全体に対して適用されている場合は、有効になりません。

次に、コンテキスト相関動作のシナリオを示します。

- Rule 1 には、根本原因 A が含まれ、非根本原因が関連付けられている。
- Rule 1 にはコンテキスト相関フラグは設定されていない。
- Rule 1 はコンテキスト 1 および 2 に適用されている。

Rule 1 にコンテキスト相関フラグが設定されていない場合、コンテキスト 1 からアラーム A が生成され、コンテキスト 2 からアラーム B が生成されるシナリオでは、コンテキストのタイプにかかわらず、ルールが両方のコンテキストに適用されます。

Rule 1 にコンテキスト相関フラグが設定され、同じアラームが生成された場合、これらのアラームは、異なるコンテキストからのものであるとして、相関されません。

フラグが設定されていると、アラームが同じコンテキストから送信された場合に限り、コリレータはアラームをルールに照らして分析します。つまり、アラーム A がコンテキスト 1 から生成され、アラーム B がコンテキスト 2 から生成された場合、相関は行われません。

時間タイムアウト フラグ

根本原因タイムアウト（指定されている場合）は、特定のルールの根本原因アラームが着信する前に、非根本原因アラームが着信した状況の場合に使用する代替ルール タイムアウトです。通常、このタイムアウトは、根本原因アラームが着信する可能性が低く、そのため非根本原因アラームの保持がすぐに解除されることを想定して、より短いタイムアウトを設定する状況で使用されます。

再配置フラグ

再配置フラグは、非根本原因アラームの直接の根本原因がクリアされた場合に、階層的な相関においてその非根本原因アラームがどのように処理されるかを指定します。

次に、コンテキスト相関動作の例を示します。

- Rule 1 には、根本原因 A が含まれ、非根本原因が関連付けられている。
- Rule 1 にはコンテキスト相関フラグは設定されていない。
- Rule 1 はコンテキスト 1 および 2 に適用されている。

このシナリオでは、コンテキスト 1 から生成されたアラーム A とコンテキスト 2 から生成されたアラーム B が送信された場合、コンテキストにかかわらず相関が行われます。

Rule 1 にコンテキスト相関フラグが設定され、同じアラームが生成された場合、これらのアラームは、異なるコンテキストからのものであるため、相関されません。