LifeKeeperのI/Oフェンシング機能「Quorum/Witness」

クラウド環境でも高レベルのスプリットブレイン対策

2022/08/31

サイオステクノロジー株式会社
BC&CS Service Line

もくじ

- スプリットブレインとは？
- SCSI Reservation
- Quorum/Witness
- Quorum Check
  - majorityモード
  - tcp_remoteモード
  - storageモード
- Witness Check
  - remote_verifyモード
  - storageモード
  - none/offモード
- IPMI STONITH

スプリットブレインとは？

スプリットブレインとは?

- HAクラスターシステムにおいてハードウェアやコミュニケーションパス（ノード間通信）の障害によりシステムが分断され、1つのサービスが複数のノードで同時に起動してしまうこと

スプリットブレインが発生すると・・・

➢ サービス障害

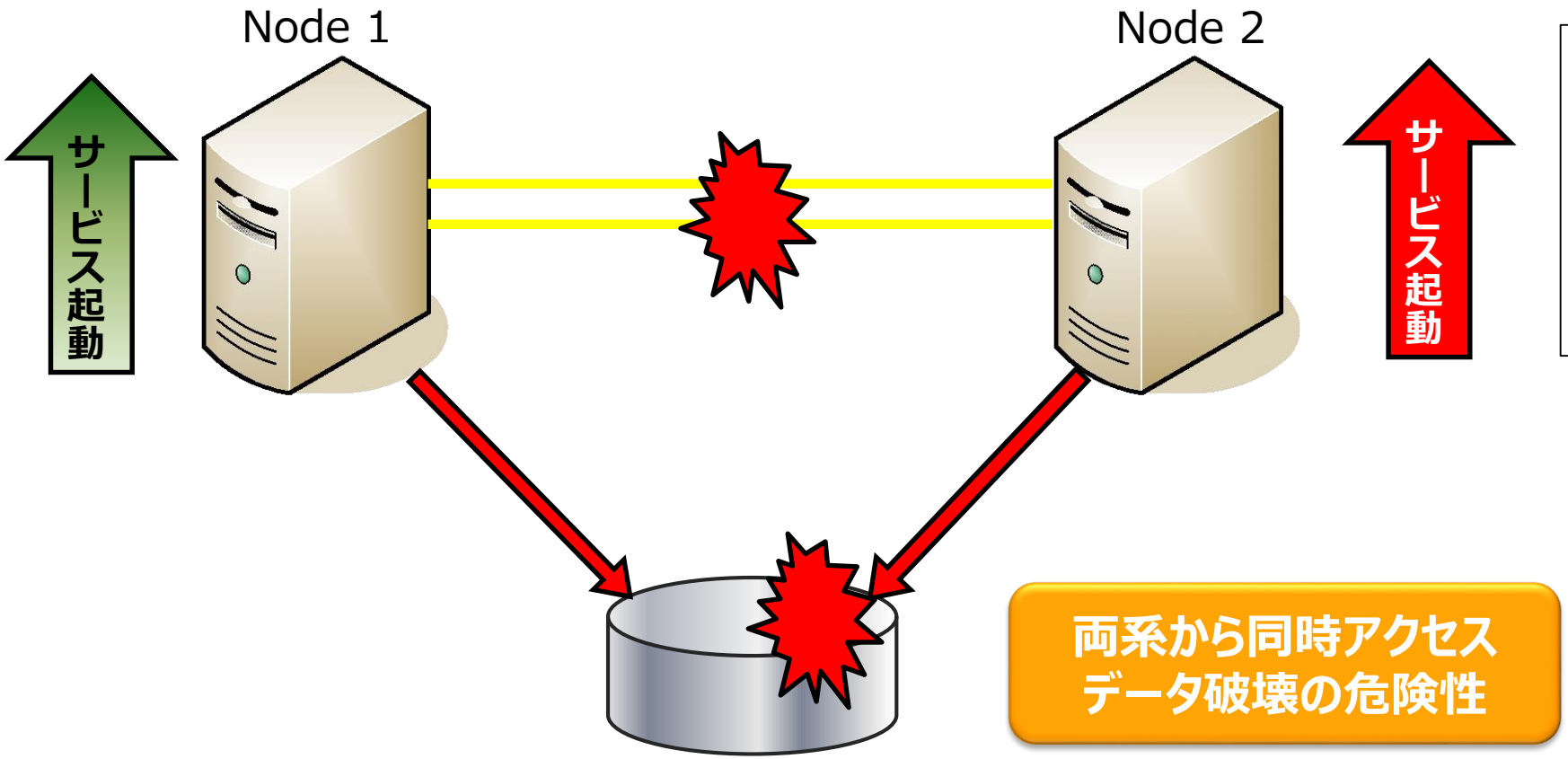
複数のノードでサービスが（重複して）起動しているため、サービスにアクセスできない場合がある。

➢ データ破壊

複数のノードから共有ディスクに書き込みが入ってしまうため、データの不整合や書き込み競合によりデータが破壊されてしまう可能性があります。

スプリットブレインとは？

- お互いが相手を認識できていないと、スタンバイ側はアクティブ側がダウンしていると認識してサービスを起動させてしまいます。
- コミュニケーションパス（ハートビート）が切れないようにすることがベストですが、万が一のときにスプリットブレインを抑止する方法（I/Oフェンシング）が３つあります。



■処理の流れ

1. コミュニケーションパスが全断
2. Standbyノードがフェイルオーバ処理を開始
3. 両ノードが同時にリソースを起動した状態となってしまう。
4. 共有ストレージへの同時アクセスによるデータ破壊の危険がある。

I/Oフェンシング

■ スプリットブレインシンドローム発生時のファイルシステムを保護するための仕組み（I/Oフェンシング）として以下の3つがあります

SCSI Reservation ※Linux版のみ

- 物理ストレージにおける最高レベルの保護の仕組み
- ストレージによっては利用できない
- DataReplication構成、NASを使う場合には利用できない

セットで使用

Quorum Check

- 「自ノードがリソースを起動すべきか否か」を、クラスター全体で多数決を採って合意するための仕組み
- 自発的なリソース停止により、スプリットブレインを回避

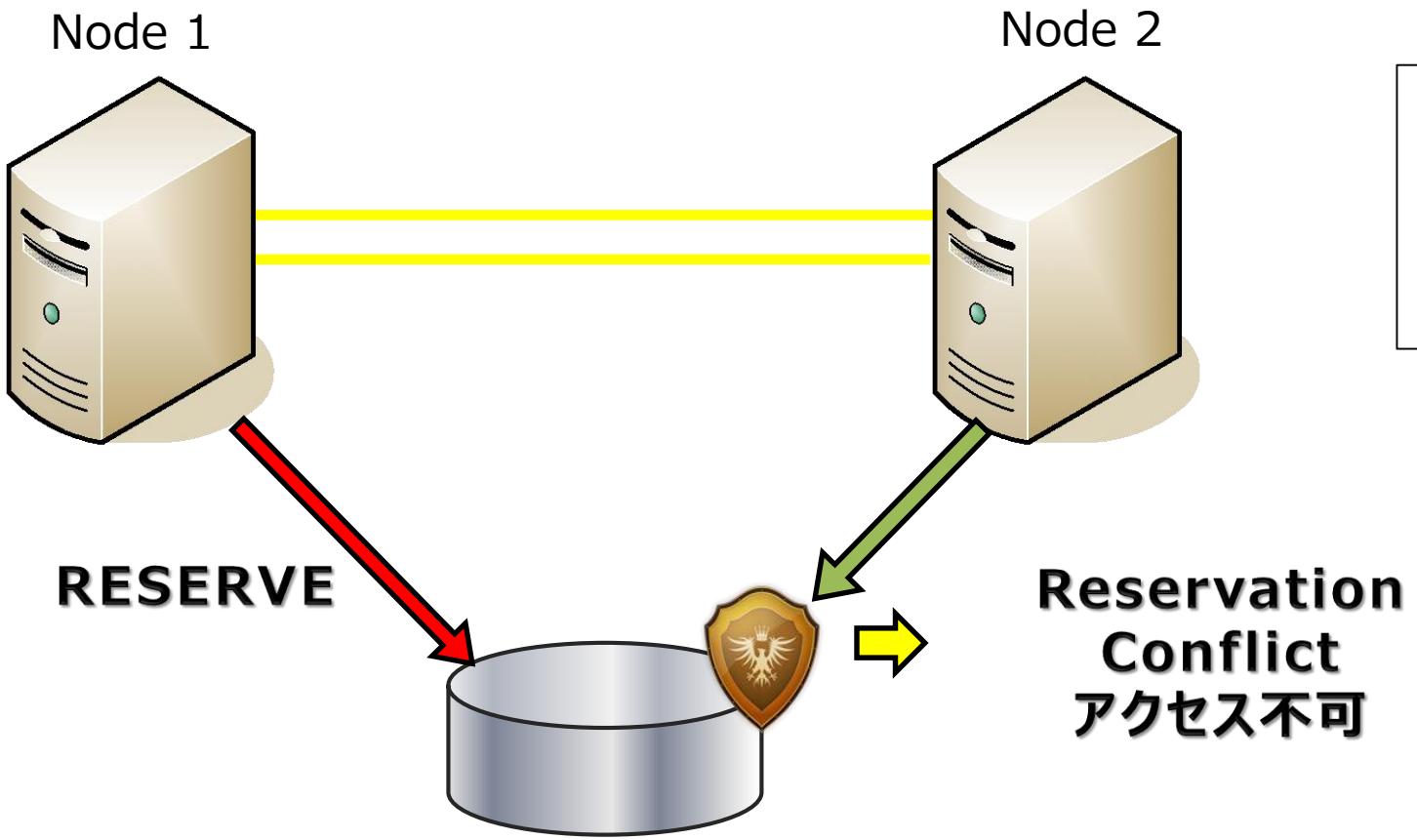
Witness Check

- あるノードの死活状態を、第三者ノード（デバイス）に問い合わせて再確認する仕組み
- ノードの死活状態を再確認

SCSI Reservation
※Linux版のみ

SCSI Reservation　※Linux版のみ

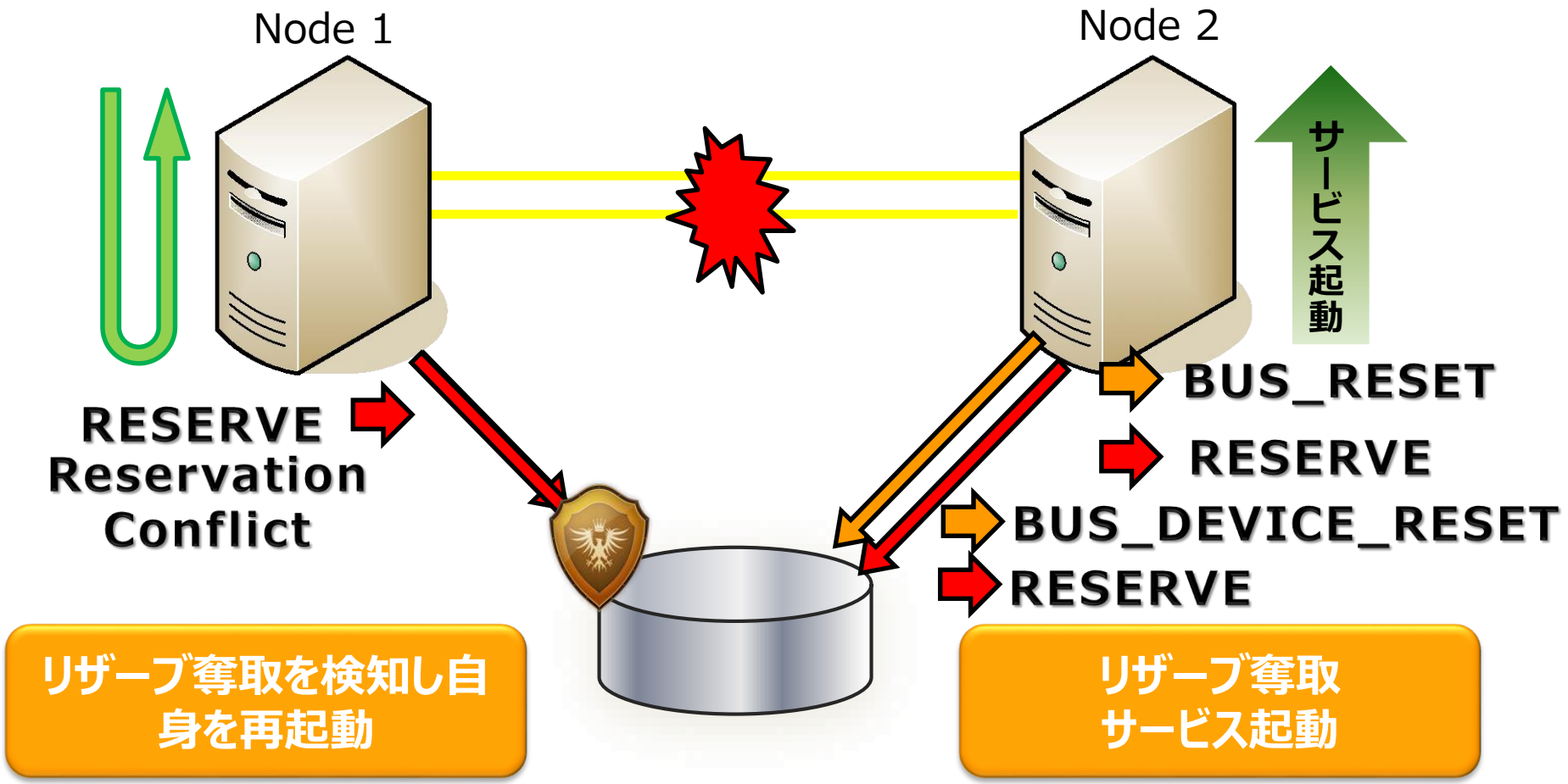
もし待機ノードがアクセスしても



■処理の流れ
1. Node-1 (Active) が共有ディスクをRESERVEしている。
2. Node-2 (Standby)が共有ディスク領域にアクセスを試みたとしても、Reservation Conflictが発生
3. Reservation Conflictにより、LUそのものへのアクセスが拒絶されます。

SCSI Reservation ※Linux版のみ

コミュニケーションパス断線時



■処理の流れ

1. Node1 がディスクをRESERVEして、サービスを起動している状態
2. Node1とNode2の間でハートビート障害
3. コミュニケーションパス全断を検知。スプリットブレインになる状態
4. Node2からBUS RESETを実行
5. Node1のRESERVEが消滅
6. Node2がRESERVEを実行
7. Node2がリザーブ奪取 サービス起動
8. Node1がRESERVEを試行するが、Reservation Conflictとなり、その結果リザーブ奪取を検知し、自身を再起動

Quorum/WitnessによるI/Oフェンシング

- 共有ストレージが使えるオンプレミス環境は、SCSI Reservationが有効なI/Oフェンシング方法です。

- 一方、物理的な共有ストレージが使えないクラウド環境…特に**AWS・Azure・Google Cloudといったパブリッククラウド環境**では、SCSI Reservationに代わるI/Oフェンシング方式として「Quorum/Witness」が標準機能として用意されており、**使用が推奨されています**。*1*2

*1：Quorum/Witnessは無償でお使いいただけますが、お使いの際には別途お申し込みが必要です。
*2：Windows版ではv8.9.0から当機能が提供されました。

次章からはQuorum/Witness機能についてご説明致します。

Quorum/ Witness

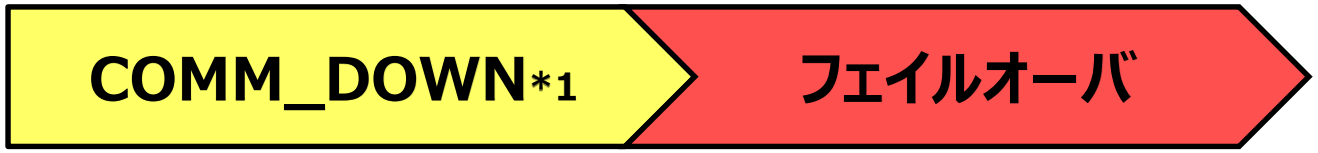
Quorum/Witnessの概念

- Quorum = 定足数（多数決）

- Witness = 証人、立会人

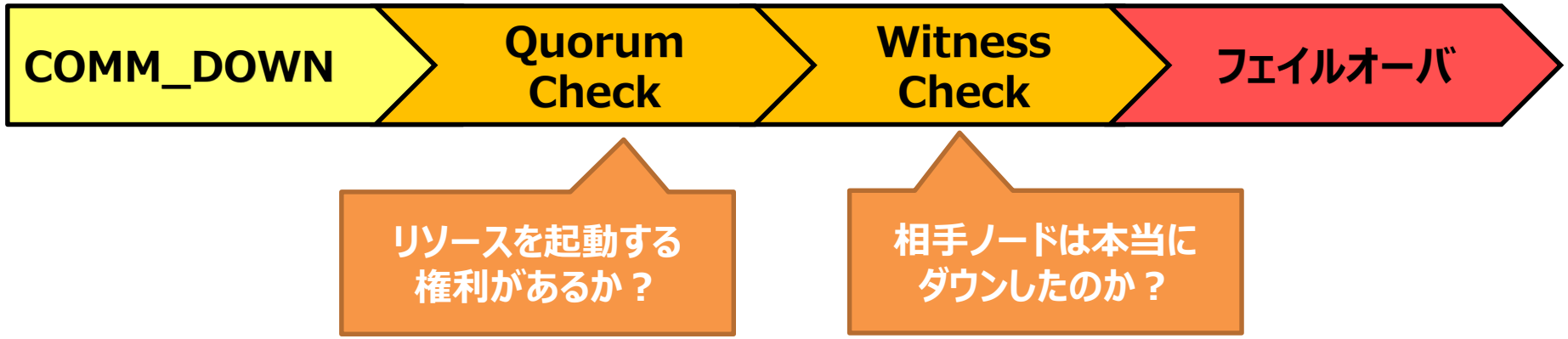
LifeKeeperのフェイルオーバー処理の概念

Quorum/Witenssを使っていない場合



*1：COMM_DOWNとは、コミュニケーションパス全断のイベントを指します。

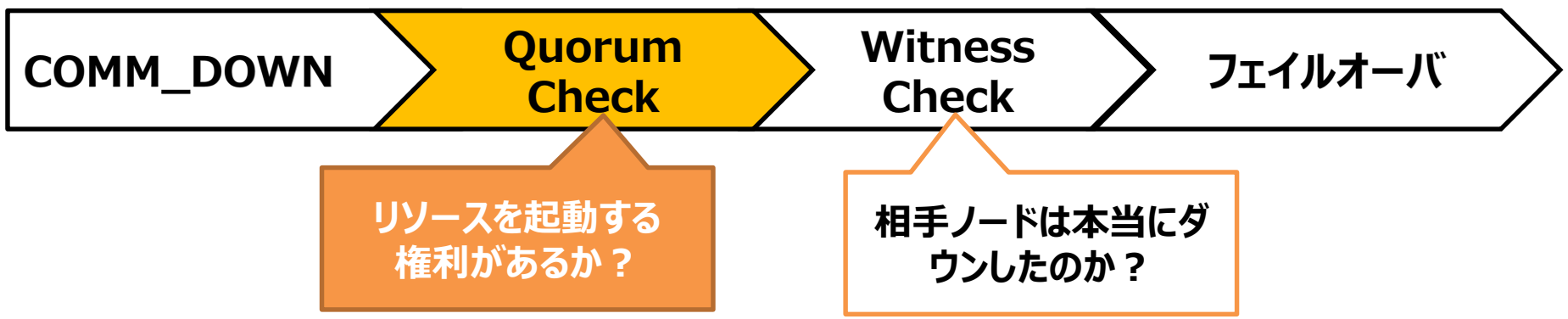
Quorum/Witnessを使っている場合



【ポイント】

Quorum/Witnessを使うことで、コミュニケーションパス全断時に、本当に待機系ノードが起動すべきか？を判断するための確認処理が増えることで、フェイルオーバーの信頼性を高めることができます。

Quorum Check



Quorum Checkで設定可能なモード

Quorum Checkモード	特徴
majority （デフォルト）	• クラスタを構成するノードのうち過半数と通信可能かどうかで、サービス起動・停止の判断を行う（多数決）。 • 過半数と通信できるならQuorumを持っている。*1 • ※当モードはデフォルトであり、よく使われるモードです。
tcp_remote ※Linux版のみ	• 事前に指定した宛先ホストのうち、過半数に到達可能かどうかで、サービス起動・停止の判断を行う。 • 過半数に到達可能ならQuorumを持っている。 • ※当モードは、majorityモードで必要となるWitnessサーバーを用意できない場合に使われることが一般的ですが、結果的にWitnessチェックを行うのならWitenssサーバーが必要になり、それ以外の方式であればSTONITHを検討する必要があるので、お勧めしていません。
storage	• クラスター内のすべてのノードからアクセスできる共有ストレージを用いた合意システムで、共有ストレージを介してノードの情報交換を行います。共有ストレージにアクセスできることで、Quorumチェック成功と判断します。別途、共有ストレージが必要です。 • なお、Quorumモードにstorageを選んだ場合、後述のWitnessモードもstorageを選択しなければなりません。 • ※当モードはAWS環境で使われることが増えています。

*1：「Quorumを持っている」＝自ノードはリソースを起動する権利がある

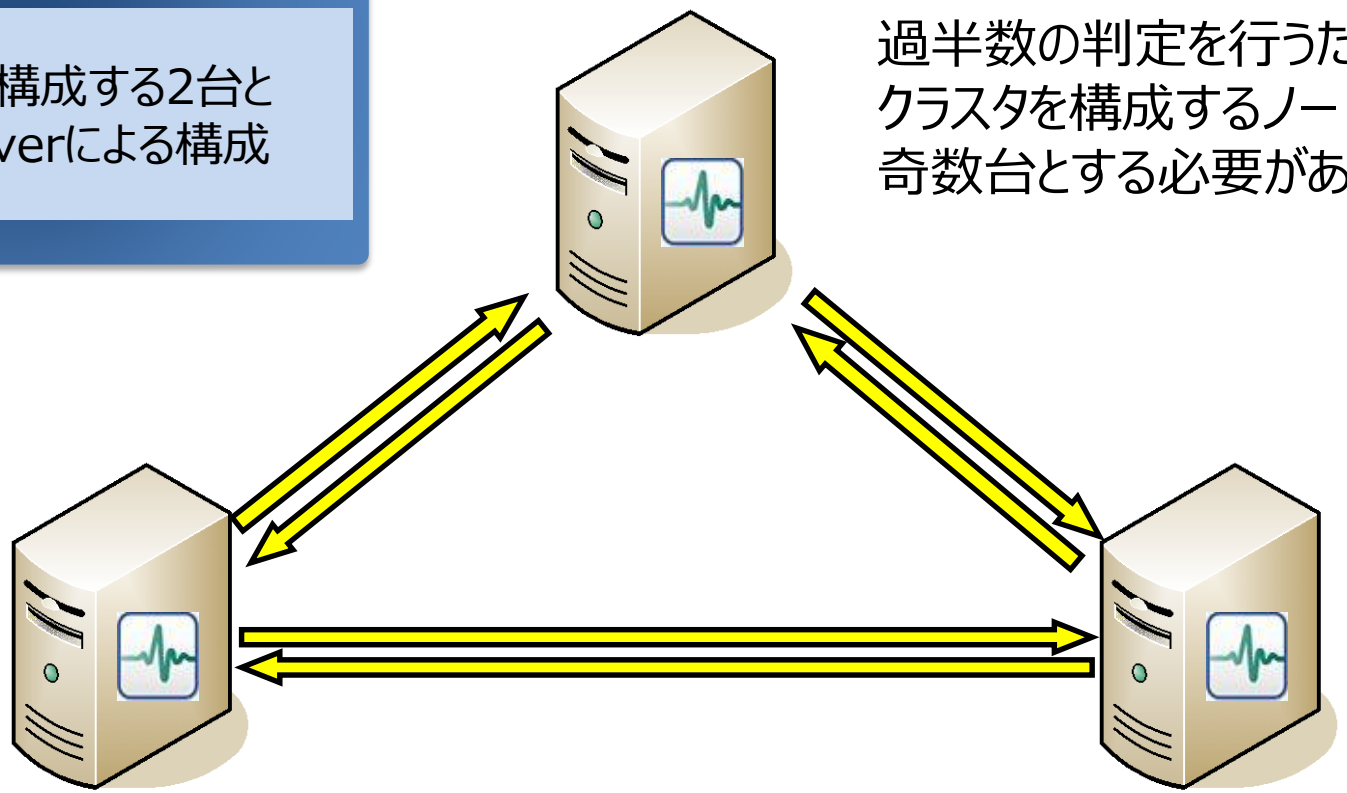
詳細はオンラインマニュアルをご参照ください。
・Linux版：https://docs.us.sios.com/spslinux/9.6.1/ja/topic/quorum-witness
・Windows版：https://docs.us.sios.com/sps/8.9.0/ja/topic/lifekeeper-quorum

Quorum Check / majorityモード

Quorum Check / majorityモード

全体構成

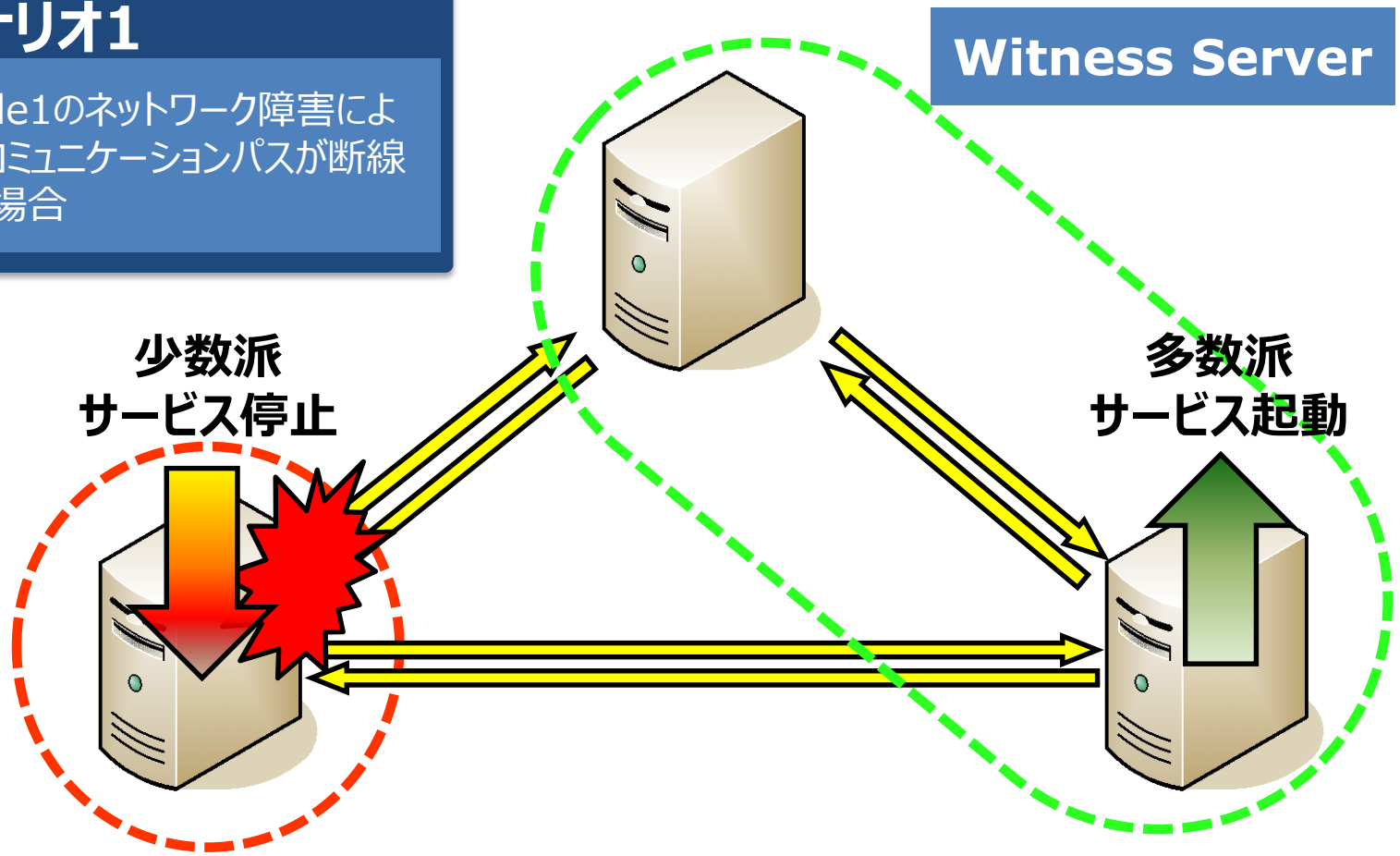
LifeKeeperを構成する2台と
Witness Serverによる構成



Quorum Check / majorityモード

シナリオ1

Node1のネットワーク障害により、コミュニケーションパスが断線した場合



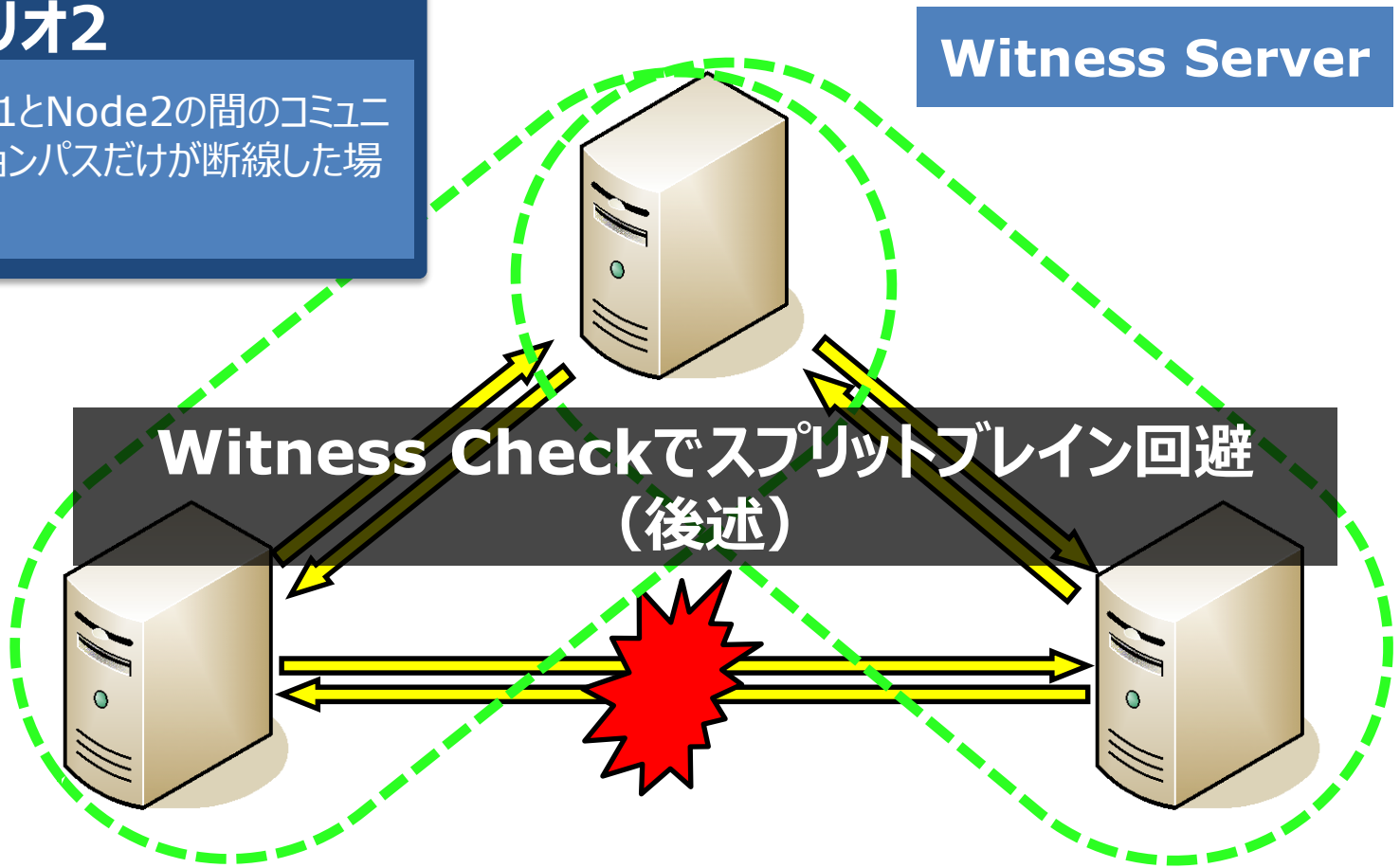
■処理の流れ

1. Node1からNode2, Witness Serverへのコミュニケーションパスが切れます。
2. この結果、クラスタが分断されます。
3. Node1が孤立していますが、対するNode2はWitness Serverと疎通可能です。
4. したがって、Node1が少数派となり、QUORUM_LOSS_ACTIONを実行します。
5. Node2は多数派となりますので、通常通りフェイルオーバを実施し、リソースを起動します。

Quorum Check / majorityモード

シナリオ2

Node1とNode2の間のコミュニケーションパスだけが断線した場合



■処理の流れ

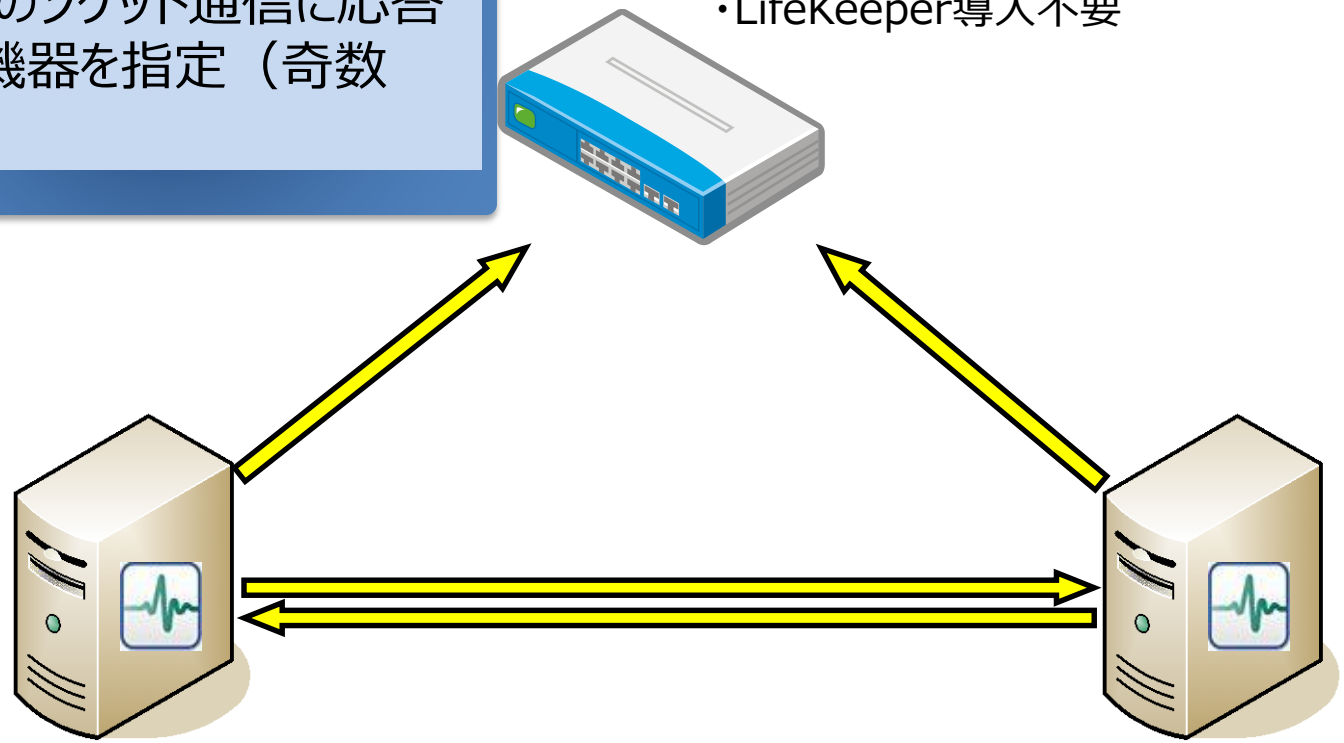
1. Node1とNode2の間のコミュニケーションパスが切れると、各ノードでQuorum Checkが行われます。
2. 結果として、両ノードともWitness Serverと疎通可能ですので、両ノードが同時に多数派となり「引き分け」状態になってしまいます。
3. 後に紹介する Witness Checkで、このような引き分け状況からのスプリットブレインを回避することができます。

Quorum Check / tcp_remoteモード
※Linux版のみ

tcp_remoteモード　※Linux版のみ

Quorum Host

任意のソケット通信に応答する機器を指定（奇数台）



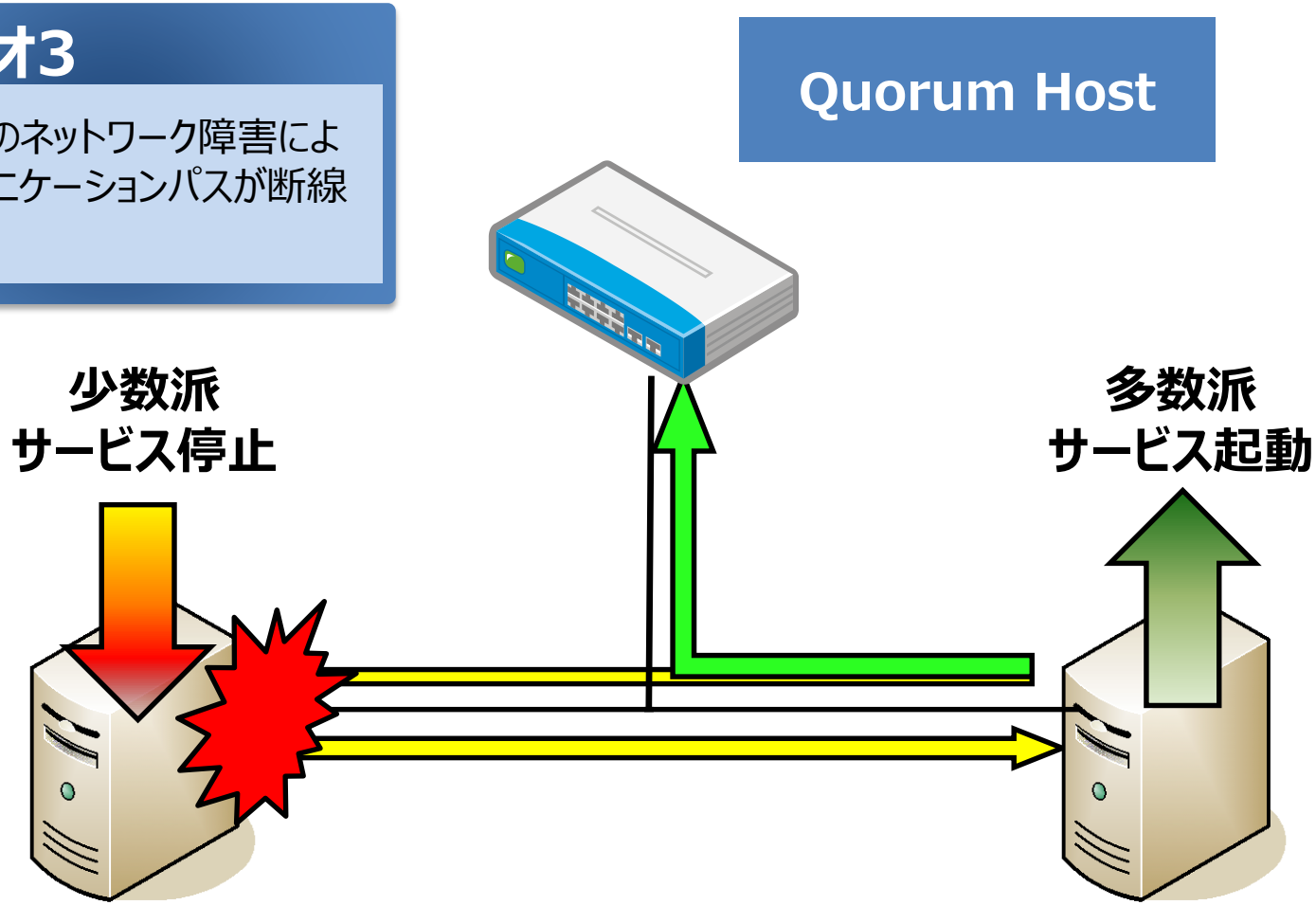
■処理の流れ

1. tcp_remoteモードでは、予め指定したホストへの接続可否で自身が多数派かどうかを判断します。QUORUM_HOSTSというパラメータで指定します。
2. QUORUM_HOSTSには、任意のノードを指定することができます。Socketを張っているだけなので、ルータやストレージの管理ポートなんかでもOKです。LifeKeeperがインストールされている必要はありません。
3. この特性のため、2ノード構成を維持したまま、Quorum方式を導入することができます。majorityモードで必要になるWitnessサーバーを立てられないケースに向いています。
4. 注意点として、QUORUM_HOSTSへのアクセス経路を、少なくとも1つのコミュニケーションパスと同一の経路にすることです。
5. そうしますと、仮にノードの障害に起因してコミュニケーションパスが断線したとして、障害が発生したノードの方がQUORUM_HOSTSに到達できない状態となります。これにより、障害ノードが少数派となり、自発的なリソース停止が行われます。
6. 左図は簡単に説明するためにQuorum Hostsを1台にしていますが、Quorum Hostsは奇数であれば複数台指定できます。複数台指定した場合、指定したうちの過半数に接続できれば多数派、過半数に接続できなければ少数派となります。Quorum Host自体の可用性に不安が残る場合には、複数台のQuorum Hostの指定をお勧めします。

tcp_remoteモード　※Linux版のみ

シナリオ3

Node1のネットワーク障害により、コミュニケーションパスが断線した場合



■処理の流れ

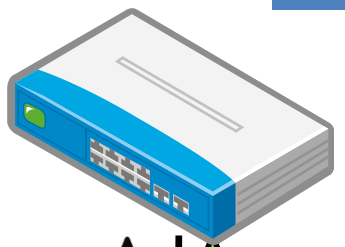
1. Node1が障害となり、ネットワーク経路が失われてコミュニケーションパスが断線。
2. Node1、Node2のそれぞれでQuorum Checkが行われます。
3. Node2は、Quorum Hostに到達可能なので多数派となり、サービスの起動を開始します。この図では1台ですので、1台中1台に接続できれば過半数を取ったことになります。
4. 対して、Node1はQuorum Hostへ到達できないので、少数派となり、QUORUM_LOSS_ACTIONを実行する、という流れになります。

tcp_remoteモード　※Linux版のみ

シナリオ4

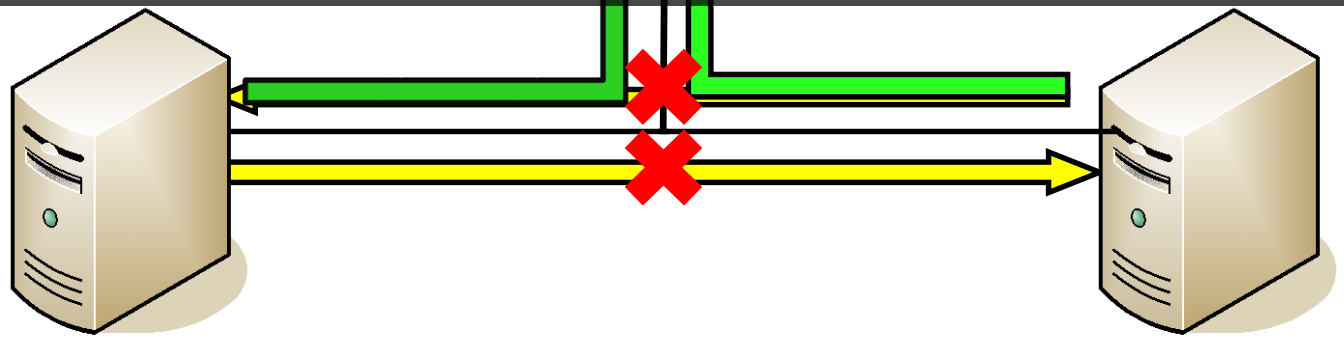
Node1とNode2 のコミュニケーションパスのみが断線した場合

Quorum Host



Witness Checkへつなげることも可能ですが、それだとWitnessサーバーが必要になるので、tcp_remoteモードを使う意味がなくなってしまいます。

IPMI/STONITHでスプリットブレイン回避（後述）



Node1(Active)

Node2(Standby)

■処理の流れ

1. Node1とNode2間のコミュニケーションパスのみが断線しました。
2. Node1、Node2のそれぞれでQuorum Checkが行われます。
3. Node1、Node2はともに、Quorum Hostに到達可能なのでどちらも多数派となってしまいます。
4. この場合、後に紹介するIPMI/STONITHでスプリットブレインを回避します。

Quorum Check / storageモード

storageモードで利用可能な領域

指定可能な設定（QWK_STORAGE_TYPE）	対象
block ※Linux版のみ	物理ストレージ、RDM(物理互換) iSCSI(VM内イニシエータ)、VMDK
file	Linux版：NFS（またはEFS） Windows版：SMB
aws_s3	Amazon S3

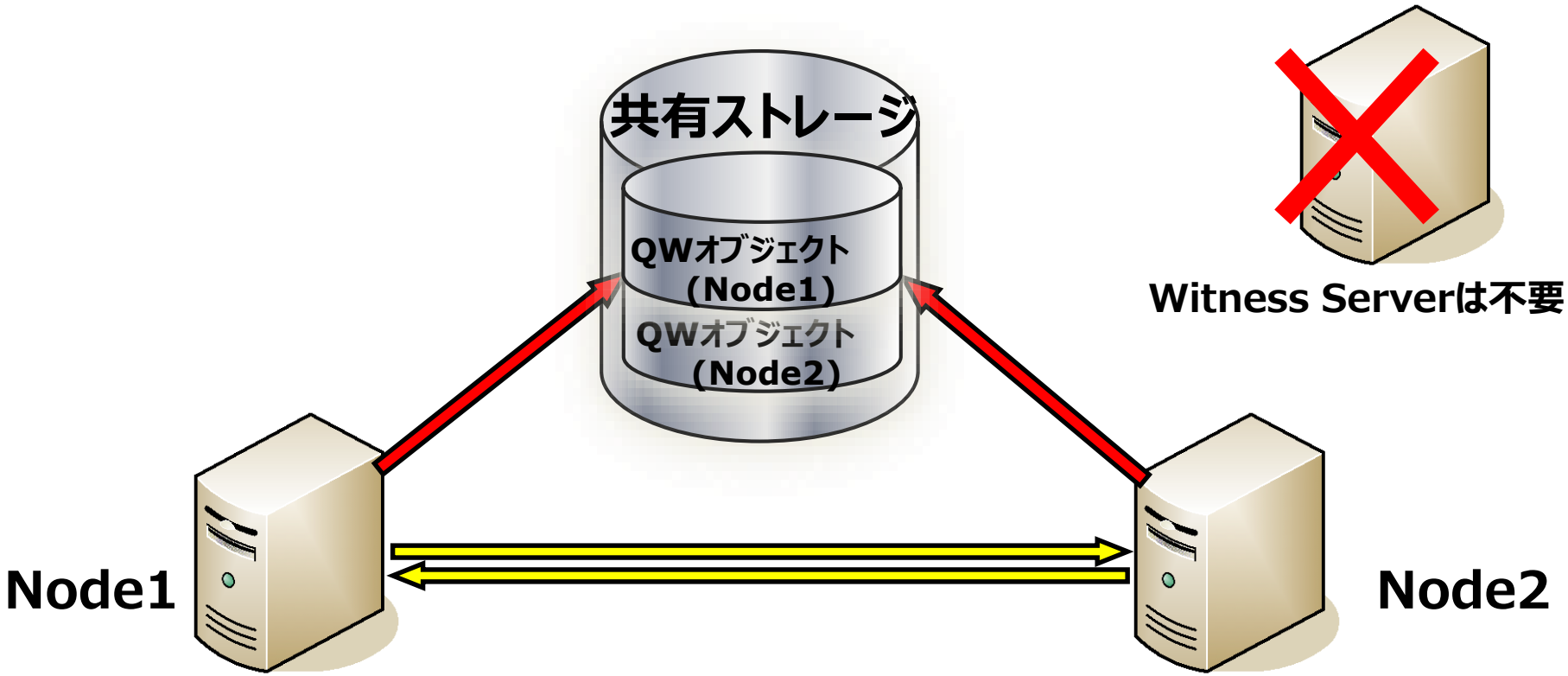
※注意事項

- 共有ストレージは各ノードからアクセス可能なストレージのことです。この共有ストレージは Quorum/Witness 機能専用であり、LifeKeeperがリソースとして保護することはできません。
- Storageモードを利用するにはQuorum Check/Witness Check共にstorageを選択する必要があります。

詳細はオンラインマニュアルをご参照ください。
・Linux版：https://docs.us.sios.com/spslinux/9.6.1/ja/topic/storage-mode
・Windows版：https://docs.us.sios.com/sps/8.9.0/ja/topic/lifekeeper-quorum

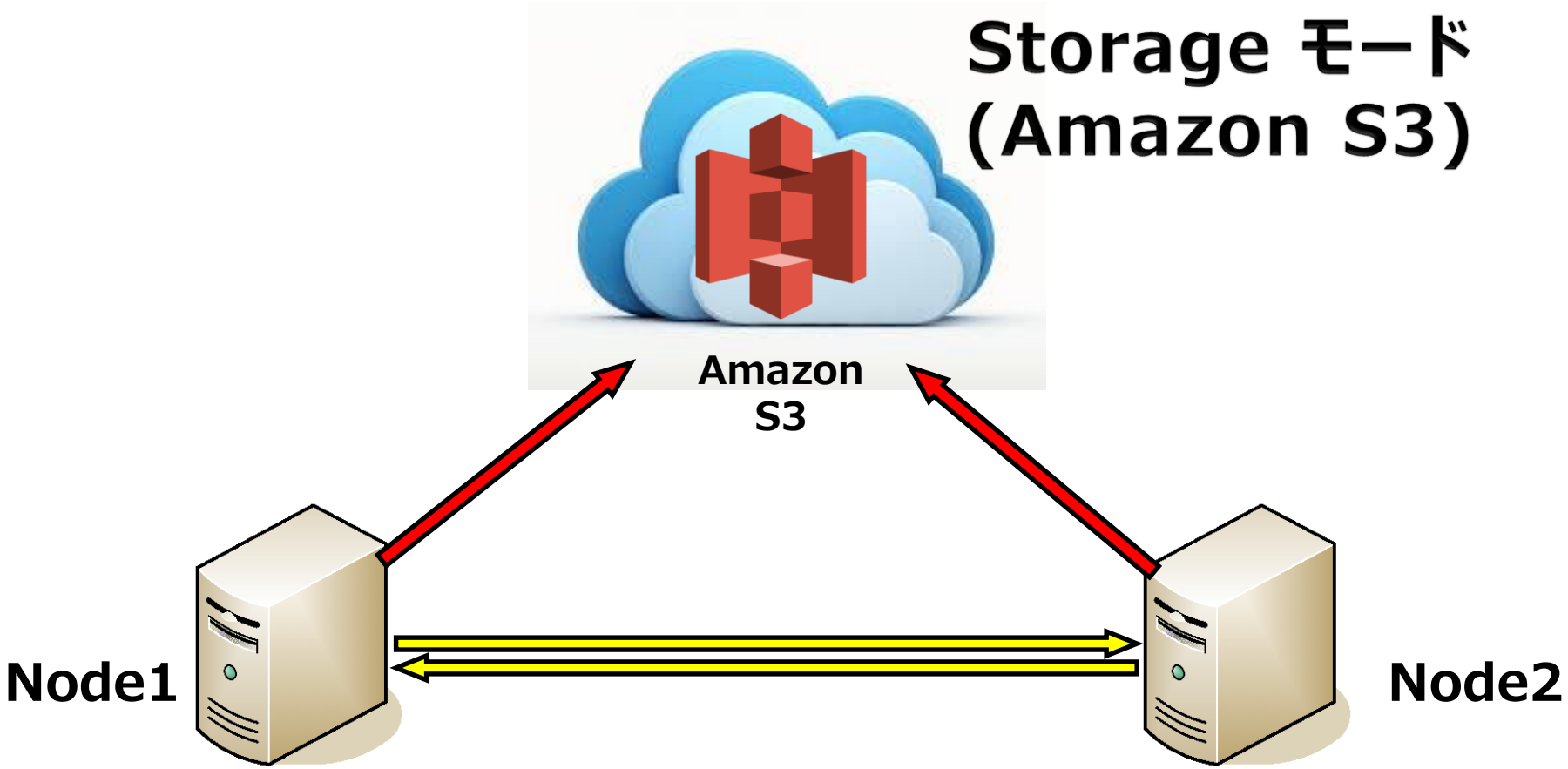
Quorum Check / storageモード

storage モード



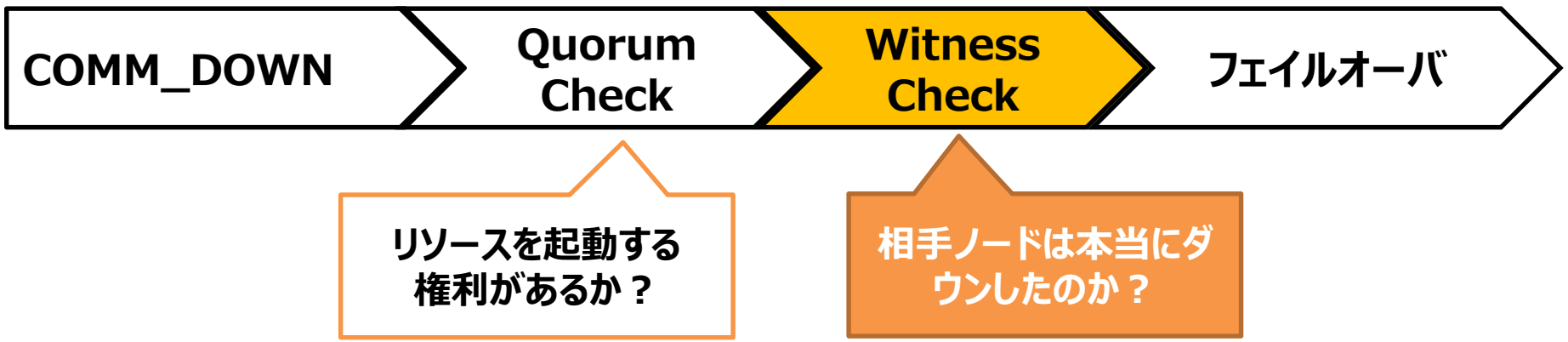
共有ストレージにアクセスできることでQuorumチェック成功と判断します。

Quorum Check / storageモード



Quorum Check用にAmazon S3も指定可能です。

Witness Check

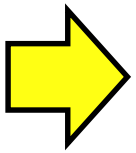


Witness Check

第三者(サーバーまたはディスク)で相手ノードの状態を確認して**比較**

比較結果が同じ

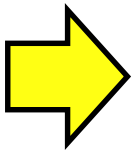
自ノードから見てDEADのノードは、
他ノードからも同様にDEADである



実際にノード障害が発生
していると判断できる
フェイルオーバ処理を継続

比較結果が異なる

自ノードから見るとDEADであるが、
他ノードからはALIVEである



実際にはノード障害では
ないと判断できる
フェイルオーバ処理を中止

Witness Checkで設定可能なモード

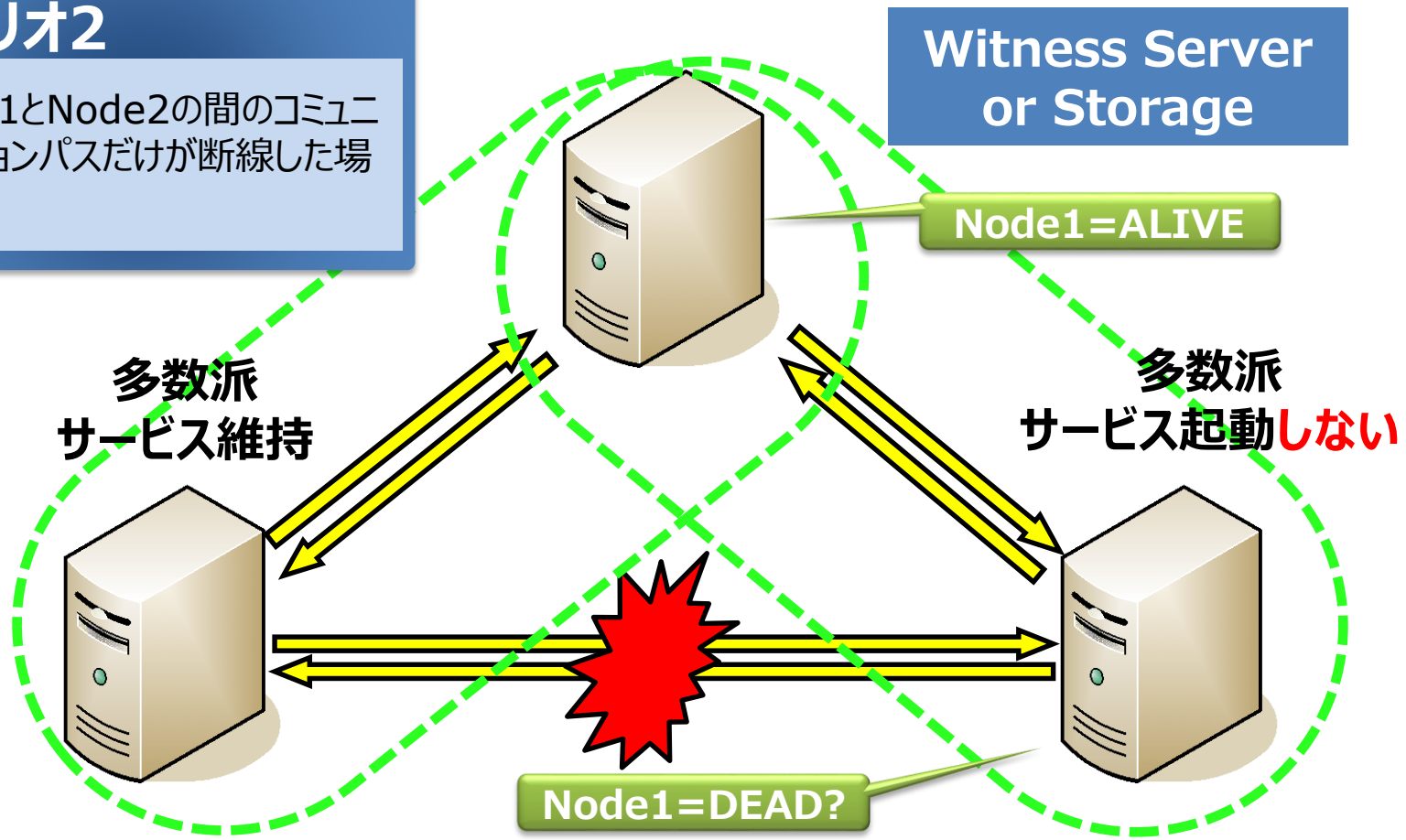
Witness Checkモード	特徴
remote_verify （デフォルト）	• クラスター内の他のすべてのノードに対して障害が疑われるノードのステータスに関する意見を求めます。1つのノードでも障害なしと判断した場合は、Witness チェックの結果は障害なしと判断します。すべてのノードが障害と判断した場合は、Witness チェックの結果は障害が発生していると判断します。 • Quorum Checkにmajorityを選択した場合は、Witness Checkはremote_verifyを選択してください。 • Linux環境でQuorum Checkにtcp_remoteを選択した場合でもWitness Checkにremote_verifyは選択できますが、それだと結局Witnessサーバーが必要になります。tcp_remoteを選択するのはWitnessサーバーを立てないケースが一般的なので、Witnessサーバーを立てない場合はnone/offを選択して後述のIPMI STONITHとの組み合わせをご検討ください。
storage	• 共有ストレージに書き込まれた情報を定期的に確認し、更新されていれば障害なし、更新が止まれば障害が発生していると判断します。 • WitnessモードにstorageをえらんだばあいQuorumモードもstorageをせんたくしなければなりません。
none/off	• Witness チェックが無効になっています。この設定では、常に障害が発生していると判断されます。

詳細はオンラインマニュアルをご参照ください。
・Linux版：https://docs.us.sios.com/spslinux/9.6.1/ja/topic/storage-mode
・Windows版：https://docs.us.sios.com/sps/8.9.0/ja/topic/lifekeeper-quorum

Witness Check（remote_verifyの場合）（Quorum Check=majorityのケース）

シナリオ2

Node1とNode2の間のコミュニケーションパスだけが断線した場合



■処理の流れ

Quorum Check＝majoriyを選択した際に、Quorum Checkだけでは「引き分け」になり、スプリットブレインの発生を回避できなかったケースでも、Witness Checkは下記の処理でフェイルオーバーの信頼性を高めます。

1. Node1とNode2間のコミュニケーションパスが全断
2. Quorum Checkの結果、両ノードが多数派となります。
3. その後、Witness Checkを行い、Node2はWitness Serverに対してNode1の状態を問合せます。
4. 今回の場合は、Witness ServerとNode1の間のコミュニケーションパスは健在です。
5. Node2は、Witness Serverから受け取ったNode1の状態と自ノードから見たNode1の状態を比較し、異なるのでフェイルオーバ処理をそこで終了します。
6. そのため、サービスの起動は行われません。
7. 同様に、Node1からWitness Serverに対してもWitness Checkが行われます。ここでも同じように、Node1の想定と異なる結果が返ってきますので、フェイルオーバ処理を終了します。つまり、稼働中のサービスは維持されます。

Witness Check /storageモード

- 常時、自ノードの情報を共有ストレージに定期的に書き込む
  （この書き込まれた情報をQWKオブジェクトと呼ぶ）
- 常時、他ノードのQWKオブジェクトを定期的に読み込む

Quorum Checkでは、全ノードのQWKオブジェクトにアクセスできることを確認する

QWKオブジェクトにアクセス可能→Quorumを持っている

QWKオブジェクトにアクセス不可→Quorumを持っていない
⇒ QUORUM_LOSS_ACTIONで指定された処理*1を実行

Witness Checkでは、死活状態を確認したい対向ノードのQWKオブジェクト*2を一定期間監視して、
更新がなければノードはDEAD、更新されていればALIVEと判断する

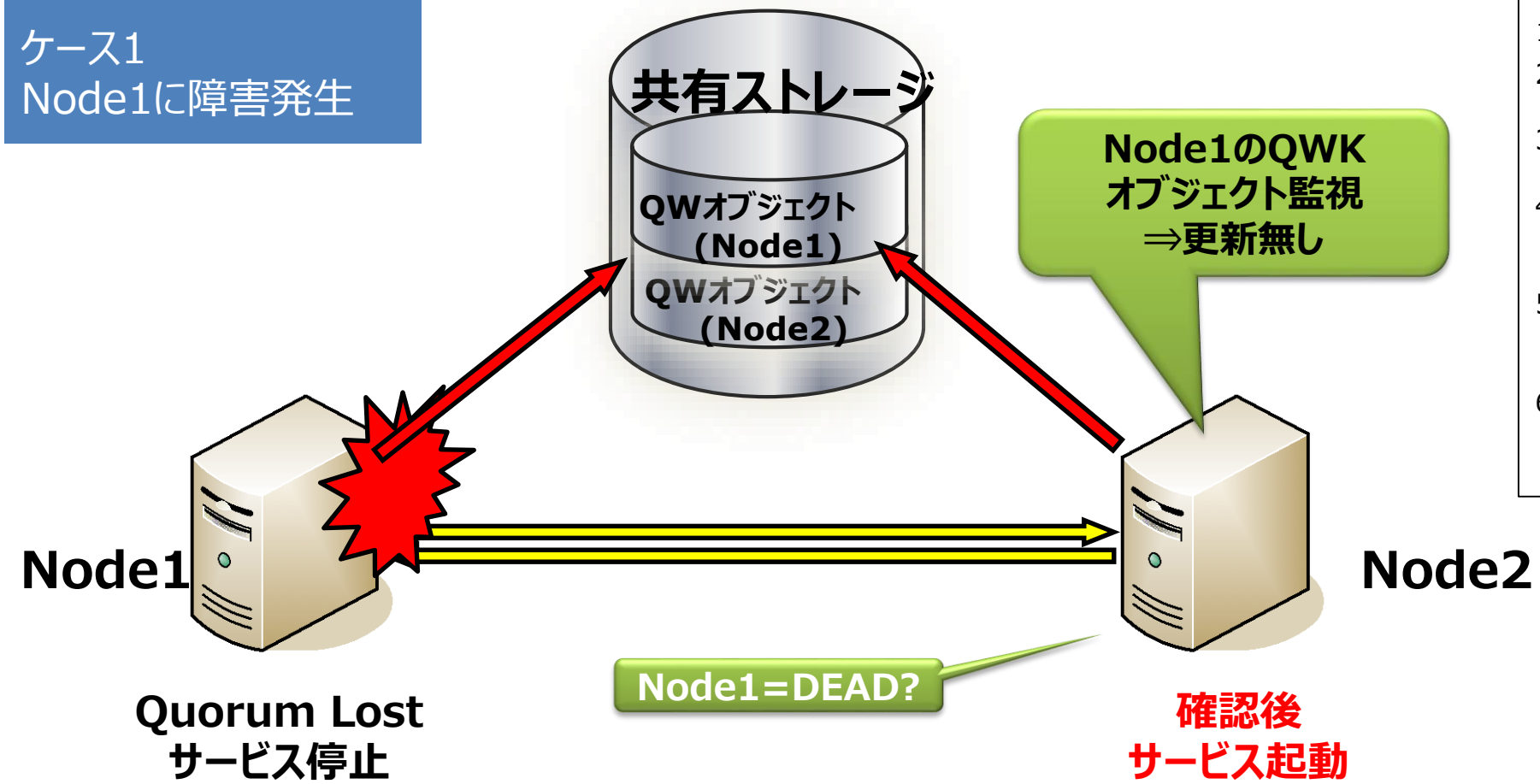
*1： QUORUM_LOSS_ACTIONパラメータについては下記のマニュアルをご参照ください。
・Linux版：https://docs.us.sios.com/spslinux/9.6.1/ja/topic/quorum-parameters-list
・Windows版：https://docs.us.sios.com/sps/8.9.0/ja/topic/quorum-parameters-list

*2：QWKオブジェクトはノードごとにあります。

Witness Checkについて(storageモード)

障害が疑われるノードのQWオブジェクトを確認し、一定期間内に更新が有れば障害無し、
更新が無ければ障害有りと判断します。

ケース1
Node1に障害発生

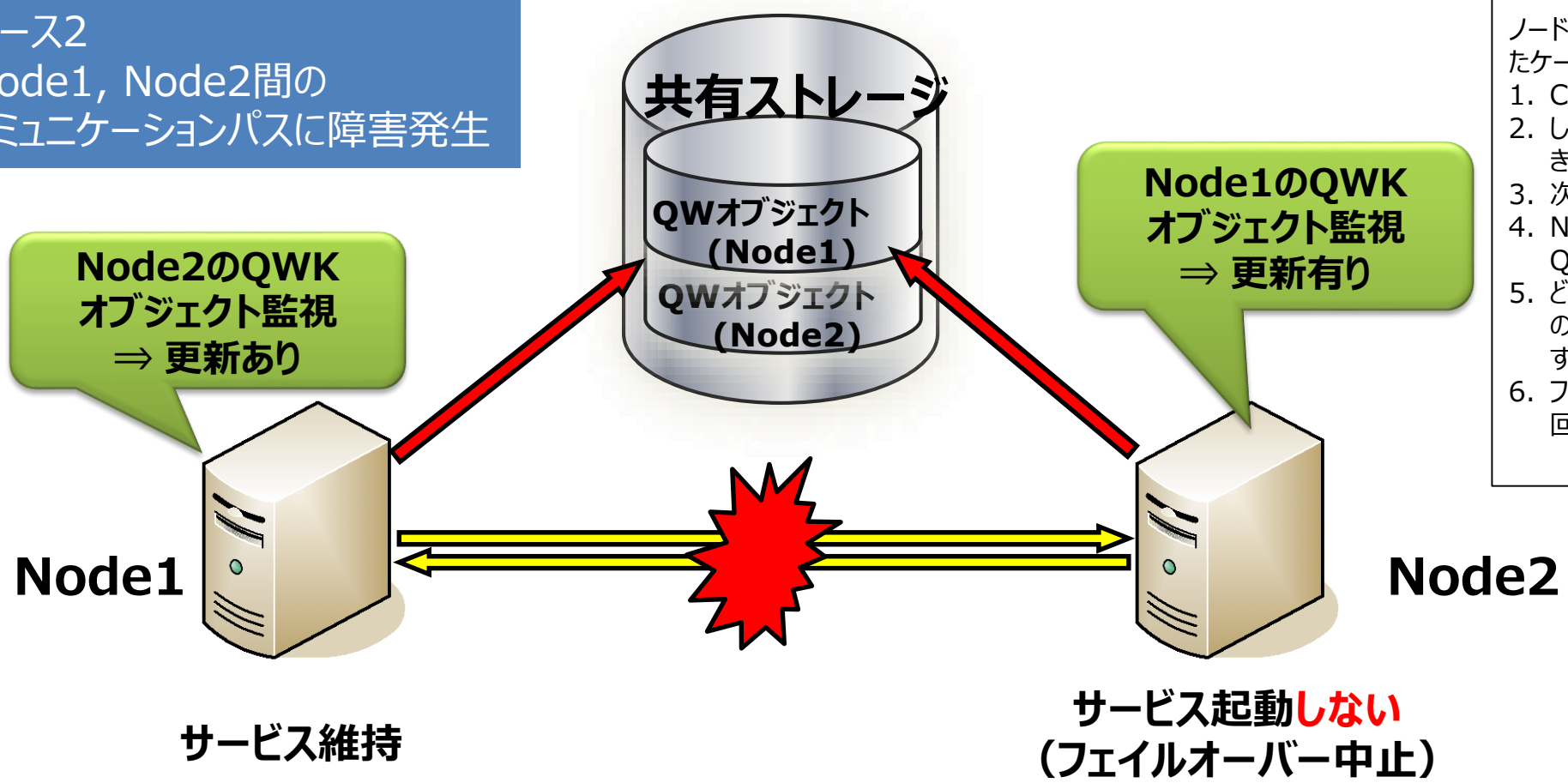


■処理の流れ
1. ノード1に障害が発生
2. 定期的に実行しているNode1のQWKオブジェクトの更新が停止
3. COMM_DOWNを契機にQuorum Check
4. Node1はQWKオブジェクトにアクセスできないので、QUORUM_LOSS_ACTIONを実行
5. Node2はNode1のQWKオブジェクトをチェック、更新が停止しているのでNode1はDEADであると判断する
6. その結果、フェイルオーバーが実行されNode2でリソースを起動

Witness Checkについて(storageモード)

障害が疑われるノードのQWオブジェクトを確認し、一定期間内に更新が有れば障害無し、
更新が無ければ障害有りと判断します。

ケース2
Node1, Node2間の
コミュニケーションパスに障害発生



■処理の流れ

ノード1とノード2の間の通信経路に障害が発生したケース

1. COMM_DOWNを契機にQuorum Check
2. しかし、両ノードとも共有ストレージにアクセスでき、Quorumを持っている
3. 次のWitness Check
4. Node1とNode2はそれぞれ、相手ノードのQWKオブジェクトを監視する
5. どちらのノードのQWKオブジェクトも更新されるので、その結果お互いに稼動状態であると判断する
6. フェイルオーバーを中止する（スプリットブレイン回避）

使用可能なQuorumモードとWitnessモードの組み合わせ

Linux版

		QUORUM_MODE			
		majority	*tcp_remote*	*storage*	*none/off*
WITNESS_MODE	*remote_verify*	利用可能 ３ノード以上	利用可能 ３ノード以上	利用不可	利用可能 ３ノード以上
	storage	利用不可	利用不可	利用可能 ２ノード以上 ４ノード以下	利用不可
	none/off	利用可能 ３ノード以上	利用可能 ２ノード以上	利用不可	利用可能

https://docs.us.sios.com/spslinux/9.6.1/ja/topic/quorum-witness

Windows版

		QUORUM_MODE		
		majority	*storage*	*none/off*
WITNESS_MODE	*remote_verify*	利用可能 3 ノード	利用不可	利用可能 3 ノード
	storage	利用不可	利用可能 2 ノード以上 4 ノード以下	利用不可
	none/off	利用可能 3 ノード	利用不可	利用可能

https://docs.us.sios.com/sps/8.9.0/ja/topic/lifekeeper-quorum

IPMI/STONITH
※Linux版のみ

IPMI/STONITH ※Linux版のみ

IPMI ＝ Intelligent Platform Management Interface

- OSやアプリケーションソフトなどを介さずに、ネットワークを通じて管理用端末のシステム監視ソフトなどと直接通信することができ、遠隔からCPUやデータ伝送路(バス)、ファンなど各部品の稼働状態や、筐体内部の温度や電圧などを監視したり、電源のオン/オフや再起動などを行うことができる

STONITH ＝ Shoot the Other Node in the Head

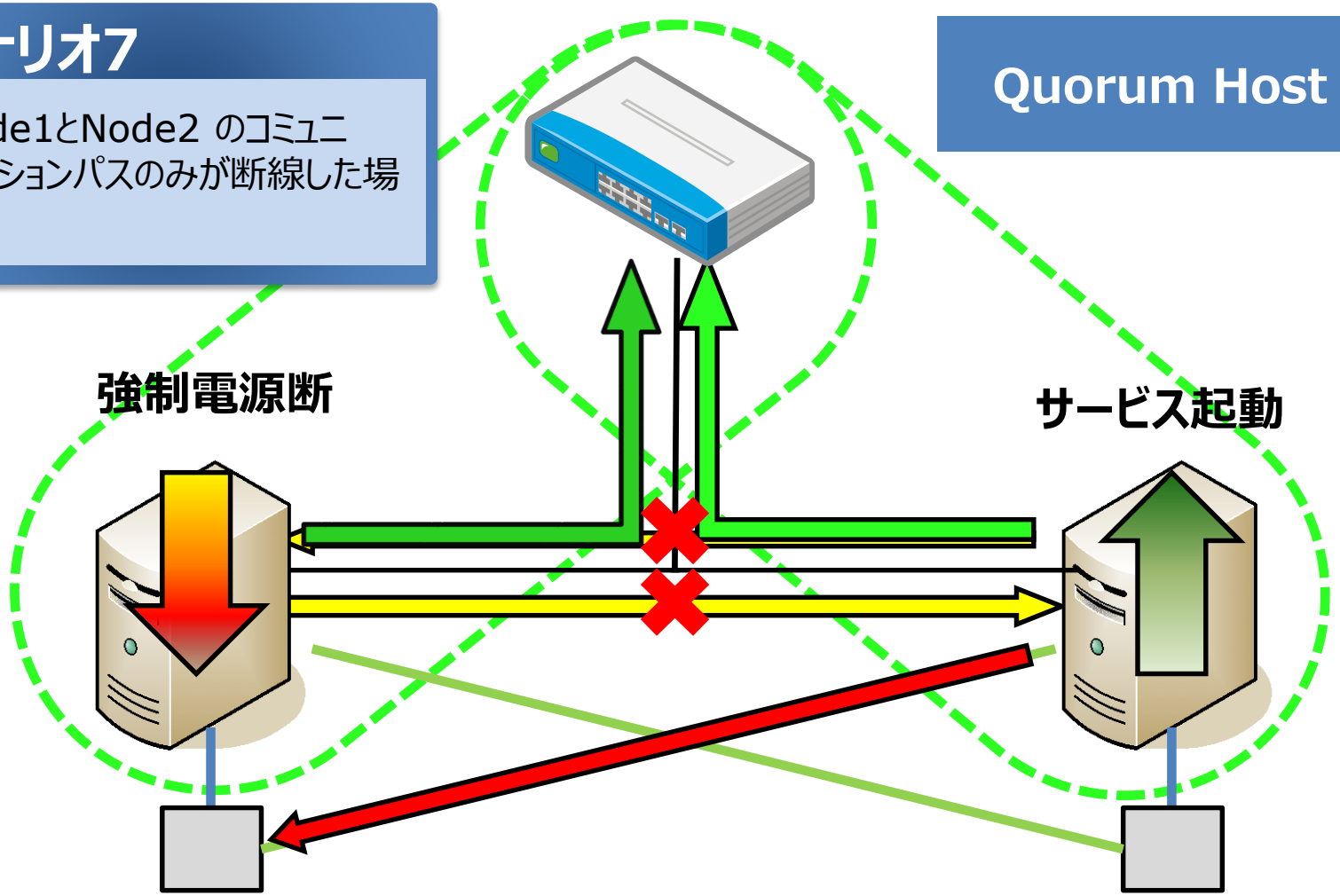
- クラスタノードの電源をリモートから切断するフェンシング方式

LifeKeeperでは、STONITHは物理・vSphere環境だけではなく、AWSとAzureにも対応しています。
詳しくはマニュアルをご参照ください。
https://docs.us.sios.com/spslinux/9.6.1/ja/topic/stonith

TCP_Remote + IPMI/STONITH ※Linux版のみ

シナリオ7

Node1とNode2 のコミュニケーションパスのみが断線した場合



■処理の流れ

1. Node1とNode2間のコミュニケーションパスのみが全断
2. Node1、Node2のそれぞれでQuorum Checkが行われます。
3. Node1、Node2はともに、Quorum Hostに到達可能なのでどちらも多数派となり引き分けになります。
4. IPMI/STONITH 発動で Node1 が強制電源断
5. Node2で「サービス起動」

（注）どちらのノードからSTONITHが発動するかはタイミングに依存します。

Quorum/Witnessの関連資料

- オンラインマニュアル
  - http://docs.us.sios.com/spslinux/9.6.1/ja/topic/quorum-witness
  - https://docs.us.sios.com/sps/8.9.0/ja/topic/lifekeeper-quorum
- ユーザーポータル
  - https://lkdkuserportal.sios.jp/hc/ja/search?utf8=%E2%9C%93&query=quorum
- ブログ
  - https://bcblog.sios.jp/?s=quorum


SIOS