試して覚えるPacemaker入門
『 リソース設定編 』

～ Pacemakerでノードやサービスを手玉に取ろう！ ～

2016年 11月 19日
OSC2016 Fukuoka

Linux-HA Japan
松浦　健太

HighAvailability Pacemaker

目次

- Pacemakerってなに？
- Pacemakerの設定とは？
- Pacemakerのリソース設定
  - リソース定義・パラメータ設定
  - リソース種類選択
  - リソース制約
  - クラスタ設定
- さいごに
  - Linux-HA Japanの紹介

Pacemakerってなに？

Pacemakerはオープンソースの
HAクラスタソフトです

Pacemakerってなに？

High Availability = 高可用性
サービス継続性

つまり

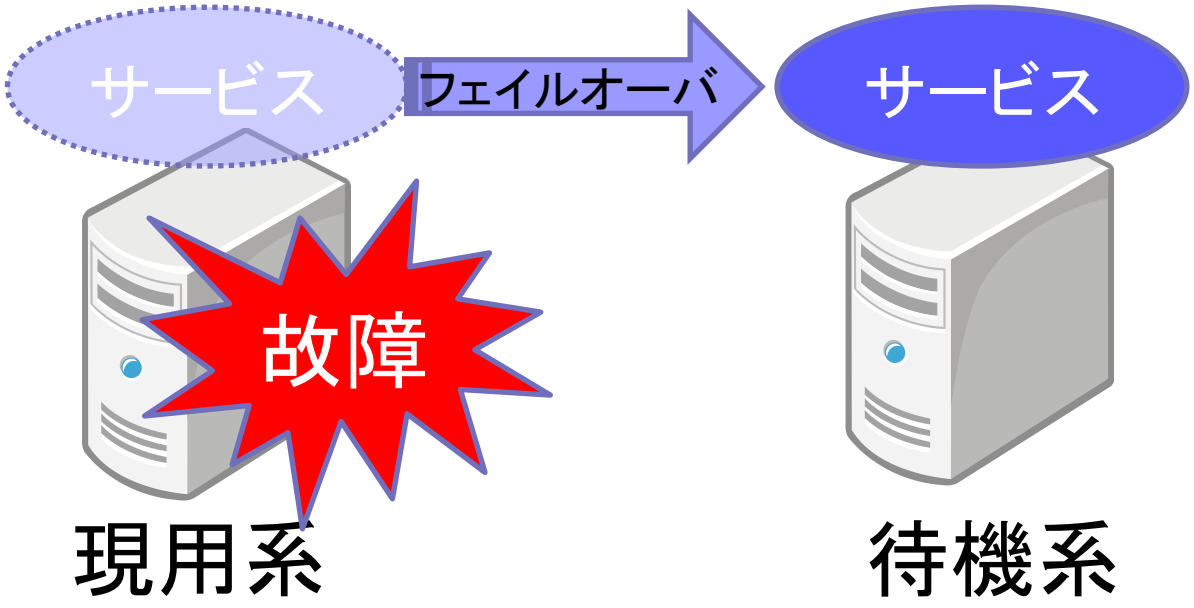
一台のコンピュータでは得られない高い信頼性を得るために、
複数のコンピュータを結合(クラスタ化)し、
ひとまとまりとする…

ためのソフトウェアです

Pacemakerってなに？

HAクラスタソフトを導入すると、
故障で現用系でサービスが運用できなくなったときに、
自動的に待機系でサービスを起動させます

→このことを**「フェイルオーバ」**と言います

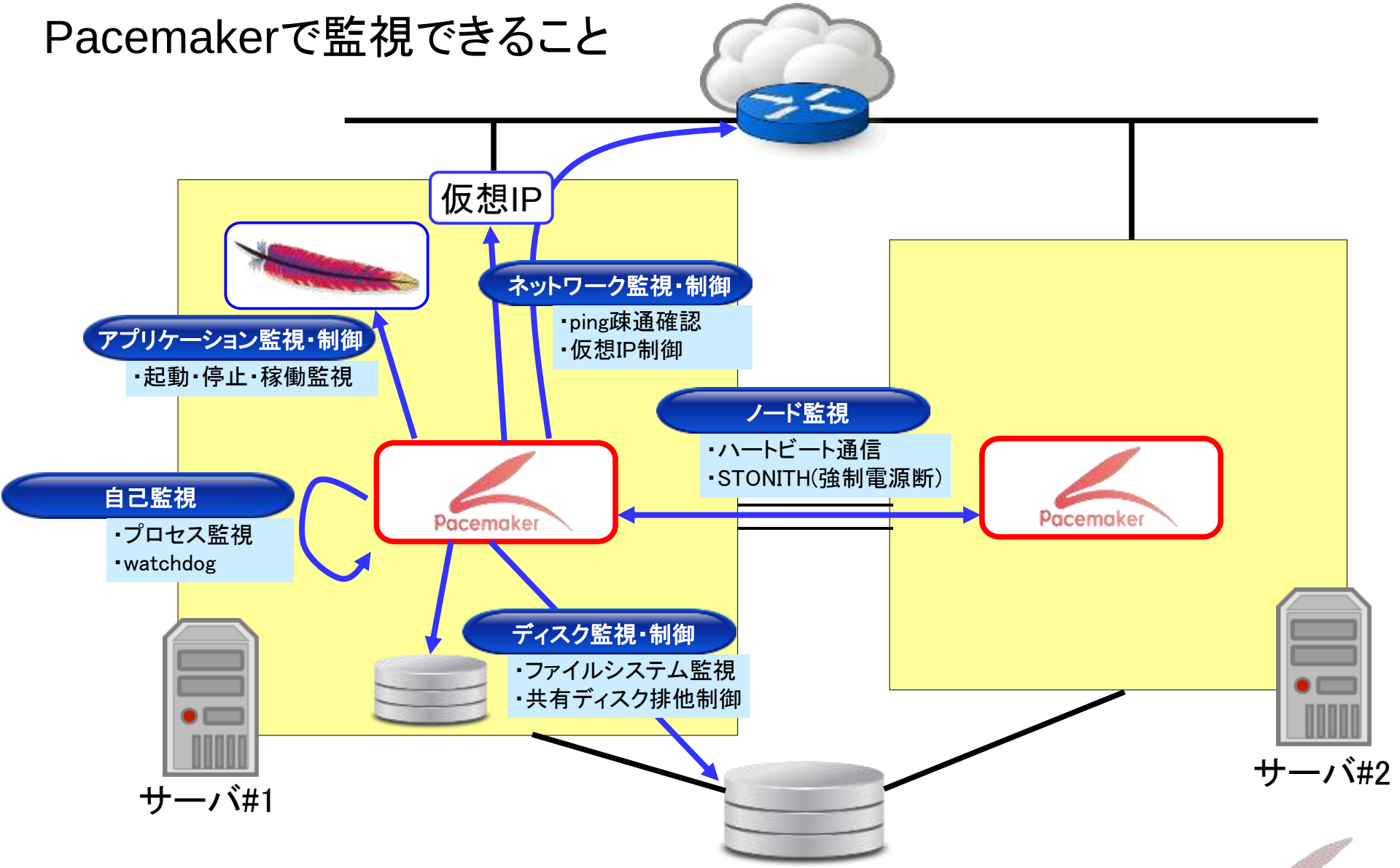


Pacemakerってなに？

はこのHAクラスタソフトとして
実績のある「Heartbeat」と
呼ばれていたソフトの後継です

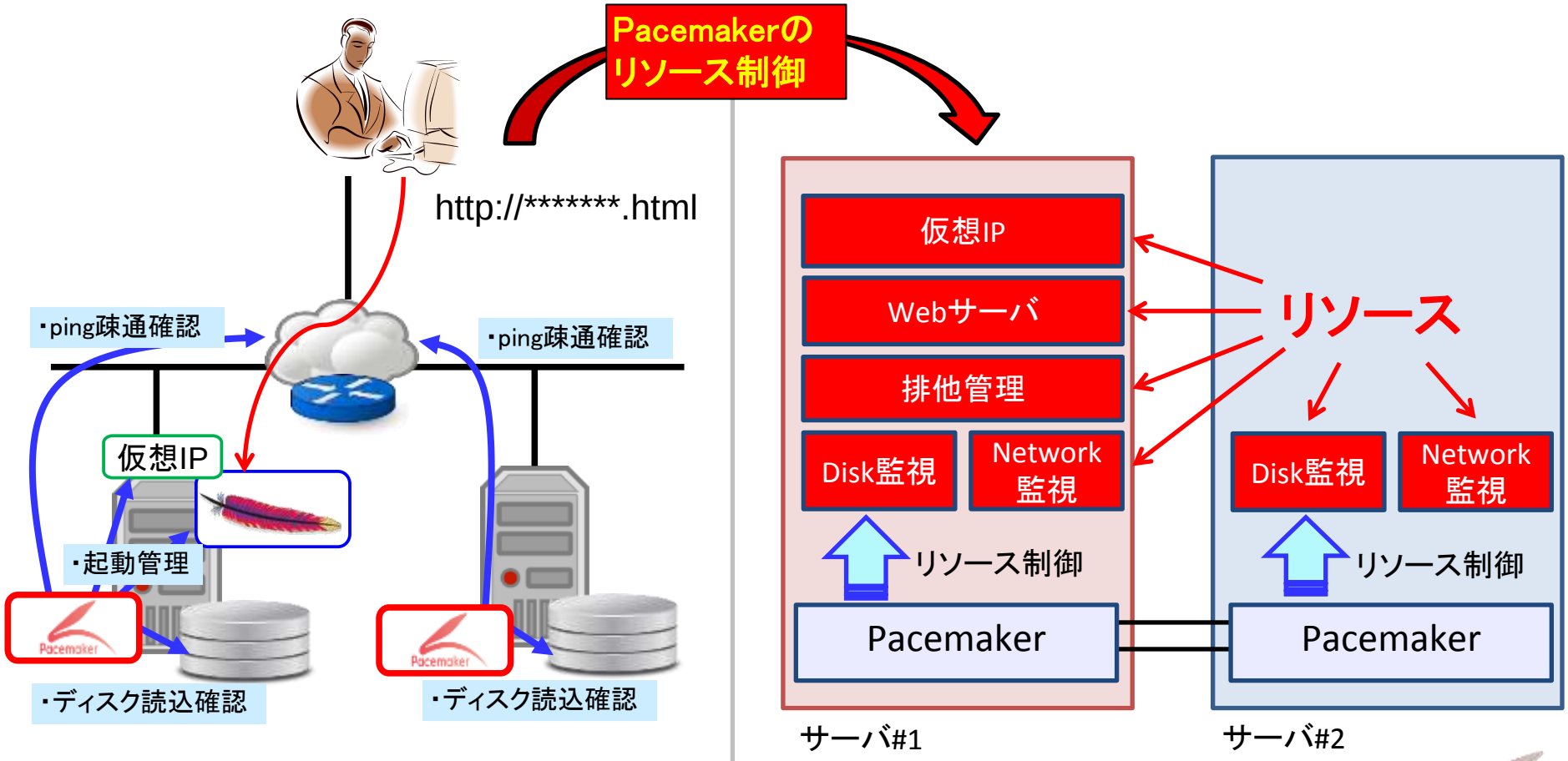
Pacemakerってなに？

Pacemakerで監視できること



Pacemakerってなに？

- Pacemakerはアプリケーションやディスク、ネットワーク等をリソースと呼び、このリソースを起動/停止/監視することで管理します。
  - リソースの例：Apache、PostgreSQL、共有ディスク、仮想IPアドレス…

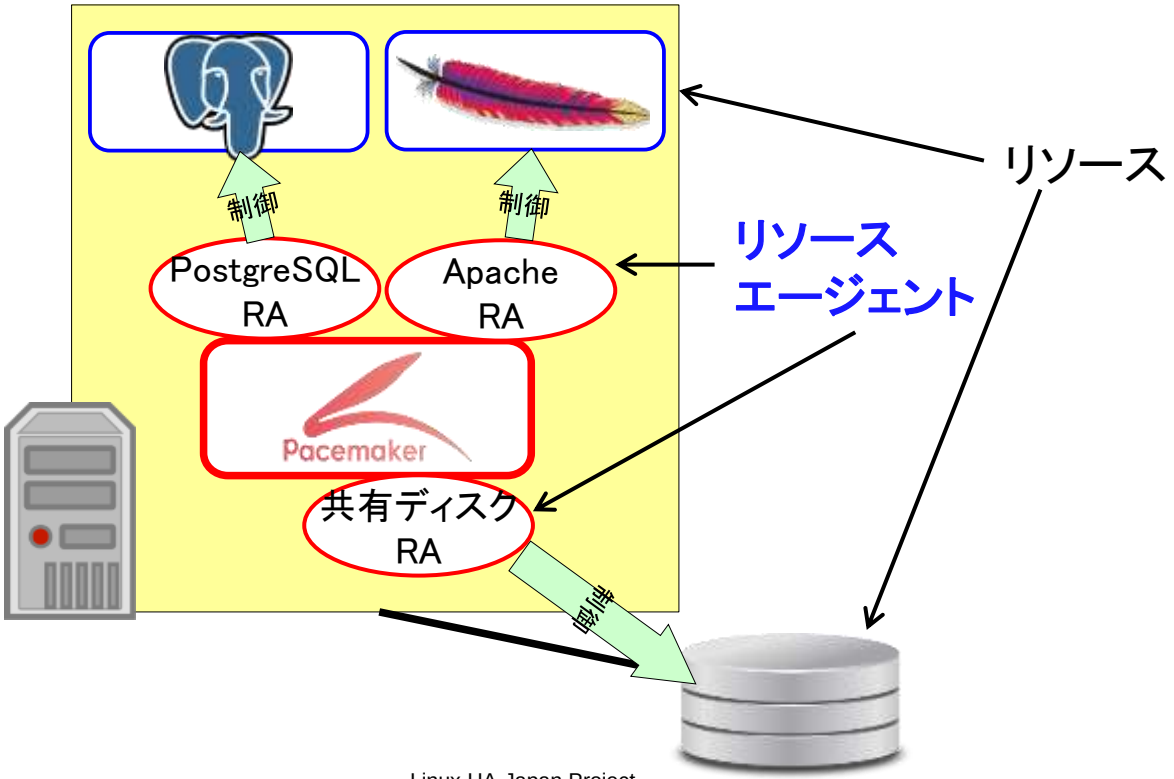


Pacemakerってなに？

Q. リソース毎に制御方法は違うのにPacemakerは正しく管理できるの？？？

A. リソースの制御は**リソースエージェント(RA)**を介して行います。

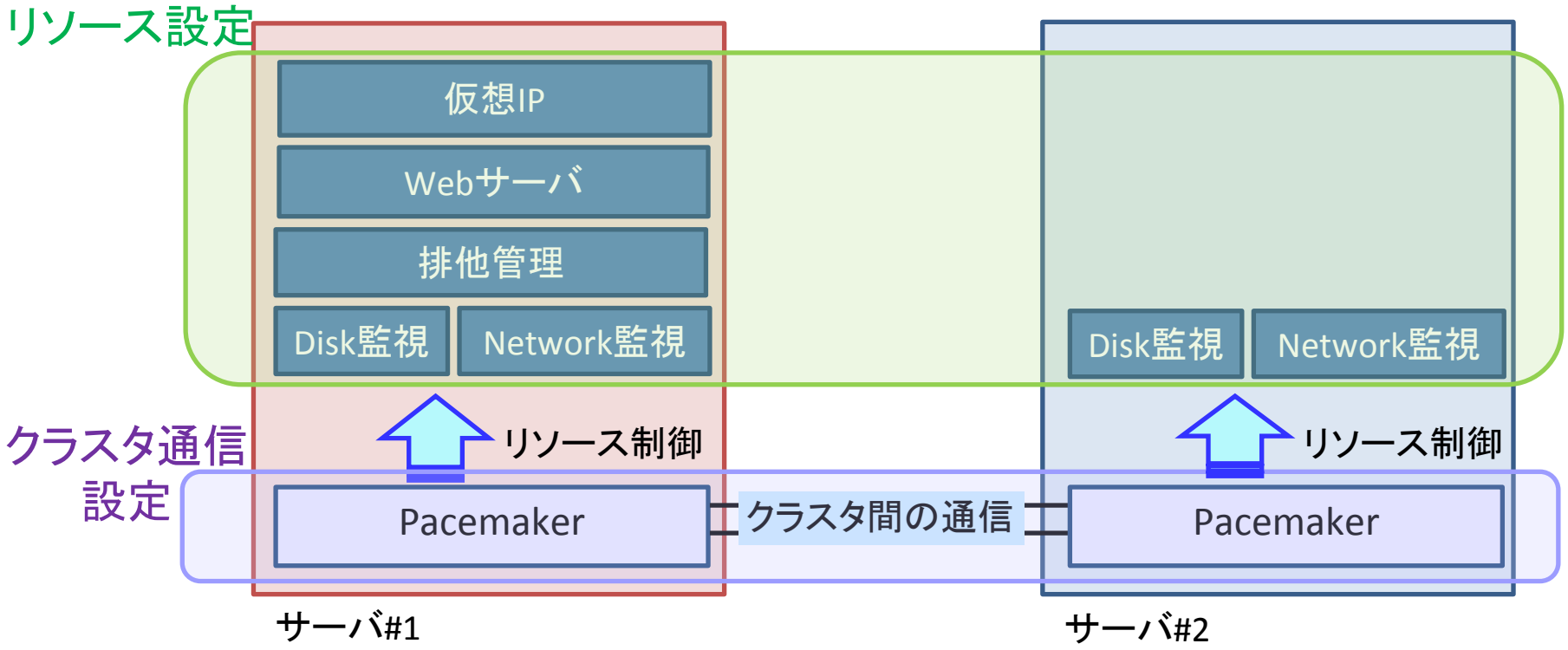
- RAが各リソースの操作方法の違いをラップし、Pacemakerで制御できるようにしている
- 多くは起動・停止・監視等を行う関数が実装されたシェルスクリプト



Pacemakerのリソース設定

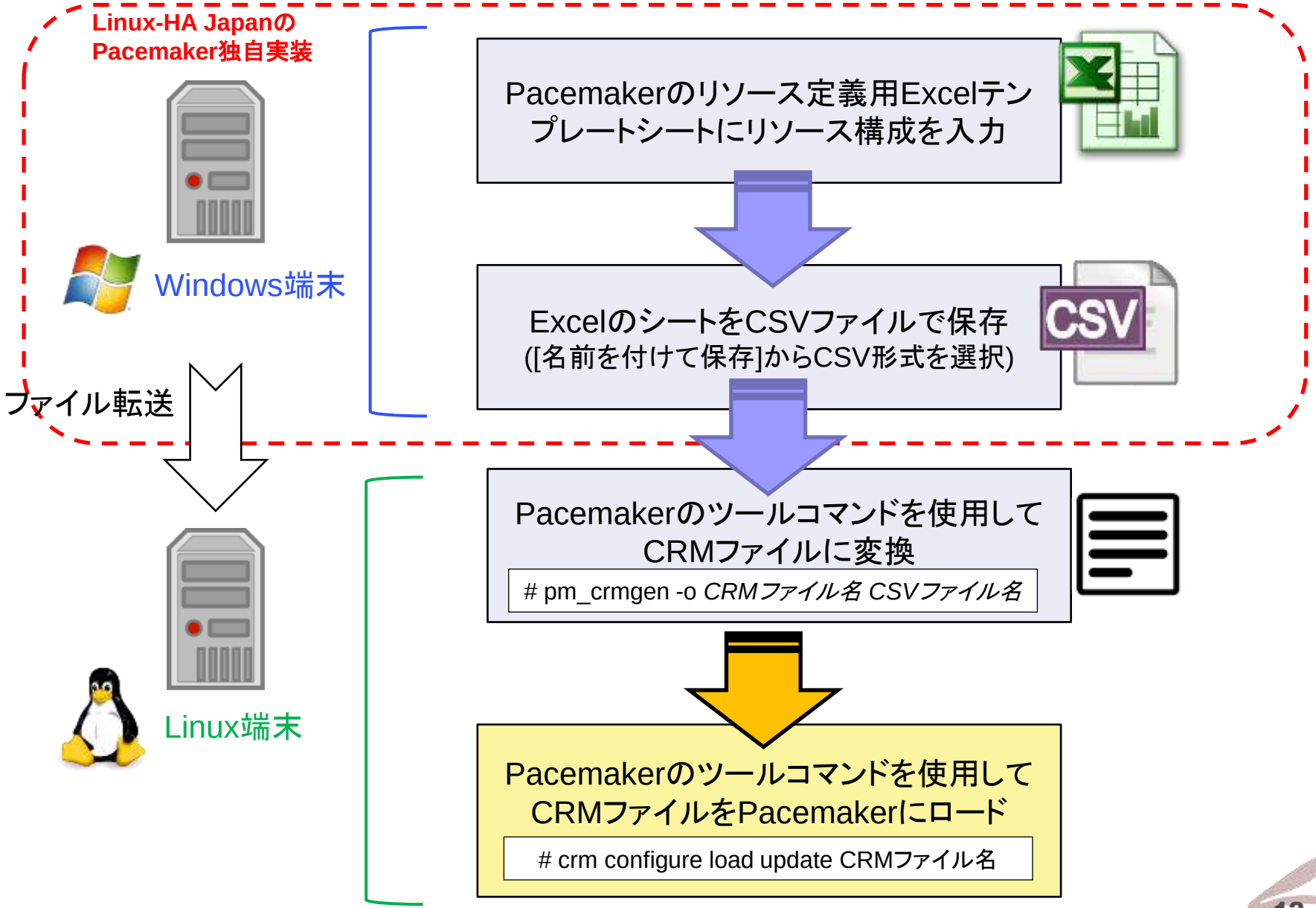
Pacemakerの設定とは？

- Pacemakerには主に2つの設定ファイルがあります。
  - クラスタ通信設定：クラスタを組むノードの指定やクラスタ間の通信方法の設定
  - リソース設定：クラスタ上でPacemakerが管理するリソースの管理方法を設定



本資料では『リソース設定』について説明します。

リソース設定の流れ


Linux-HA Japanの
Pacemaker独自実装
Windows端末
Pacemakerのリソース定義用Excelテンプレートシートにリソース構成を入力
ExcelのシートをCSVファイルで保存
([名前を付けて保存]からCSV形式を選択)
CSV
ファイル転送
Pacemakerのツールコマンドを使用してCRMファイルに変換
pm_crmgen -o CRMファイル名 CSVファイル名
Linux端末
Pacemakerのツールコマンドを使用してCRMファイルをPacemakerにロード
crm configure load update CRMファイル名

なんでこんな設定に手順必要なの？

- Pacemakerのリソース設定ファイルはCRMファイルのみであり、CRMファイルを記述できればExcelで設定する必要がありません。

（CRMファイル記入例）

```
### Cluster Option ###
property no-quorum-policy="ignore" ¥
                    stonith-enabled="true" ¥
                    startup-fencing="false“
### Resource Defaults ###
rsc_defaults resource-stickiness="200" ¥
                    migration-threshold="1“
### Group Configuration ###
group grpPostgreSQL ¥
                    prmFilesystem ¥
                    prmVip ¥
                    prmPostgreSQL
### Clone Configuration ###
clone clnPing ¥
                    prmPing
clone clnDiskd1 ¥
                    prmDiskd1
clone clnDiskd2 ¥
                    prmDiskd2
### Group Configuration ###
group grpStonith1 ¥
                    prmStonith1-1 ¥
                    prmStonith1-2
group grpStonith2 ¥
                    prmStonith2-1 ¥
                    prmStonith2-2
……
```

え、これ自分で入力するの？？？（汗）

そんな人のために、Excelで簡単に入力できるテンプレートシートと、そこからCRMファイルに変換するツールが作られたの。

でかした～！！

リソース設定用テンプレートシート

- Excelテンプレートシートは Pacemakerインストール時に以下に配置されます。
  - /usr/share/pacemaker/pm_crmgen/pm_crmgen_env.xls
  - 基本的にはこのテンプレートシートをWindowsマシンに転送して設定します。

pm_crmgen.xls

 # pm_crmgen 環境定義書 ファイル形式バージョン: 2.1

#表 1-1 クラスタ設定 … クラスタ・ノード属性

NODE

	uname	ntype	ptype	name	value
#	ノード名	ノード種別	パラメータ種別	項目	設定内容

#表 2-1 クラスタ設定 … クラスタ・プロパティ

PROPERTY

	name	value	
#	項目	設定内容	概要
	no-quorum-policy		ノード数によるリソース割当て
	stonith-enabled		障害ノード対処(STONITH制御)
	startup-fencing		起動時に状態不明ノードへSTONITH

#表 3-1 クラスタ設定 … リソース・デフォルト

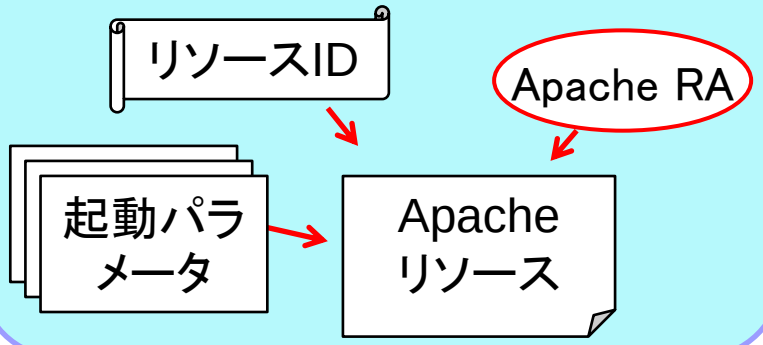
RSC_DEFAULTS

	name	value

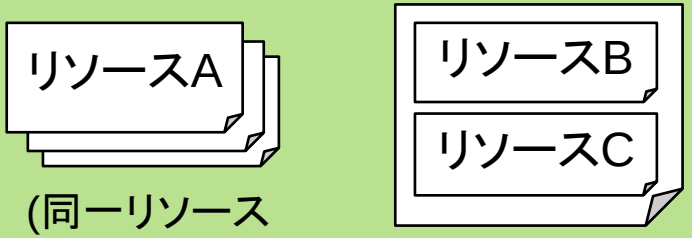
何を設定するの？

リソース設定では主に以下の4つを設定します。

リソース定義・パラメータ設定



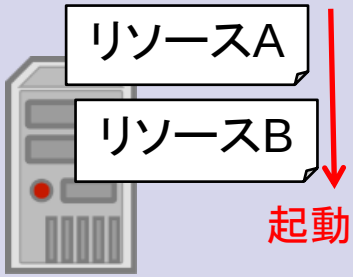
リソース種類選択



(同一リソースをクラスタ内に複数起動)

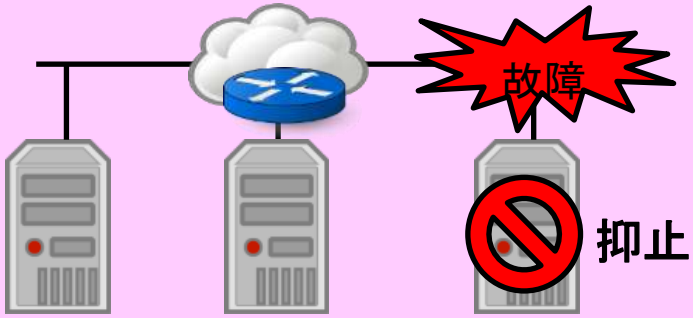
(複数のリソースを一括管理)

リソース制約



- リソースが起動するノードを制御
- リソースの起動順序を制御

クラスタ設定



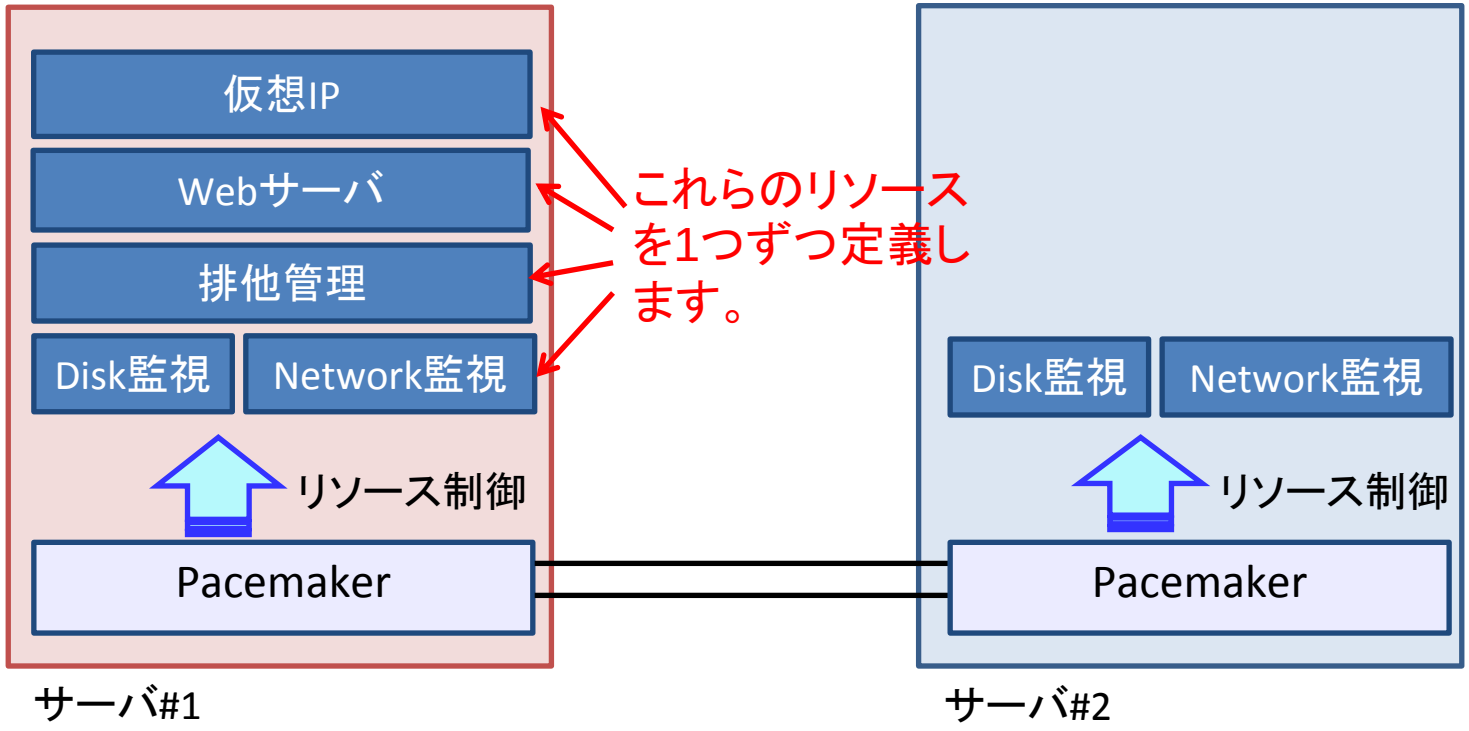
(スプリットブレイン時の排他制御)

Pacemakerのリソース設定

1. リソース定義・パラメータ設定
2. リソース種類選択
3. リソース制約
4. クラスタ設定

リソース定義・パラメータ設定(1/3)

- ここではリソースを1つずつ定義し、また起動に必要なパラメータの設定等も行います。
- 設定する内容は以下の通りです。
  - リソースID (リソース毎にユニークなIDを設定します)
  - リソースが使用するRA
  - パラメータ(リソースの起動に必要な設定。パラメータ項目はRAによって異なります)
  - 動作設定(起動・監視・停止処理のタイムアウトや失敗時の自動対処を設定します)



リソース定義・パラメータ設定(2/3)

- リソース定義はExcelテンプレートシート 表7-1に設定します。
  - 複数リソース設定する場合は表7-1をコピーして表を作成し、設定します。

リソースが使用するRA
設定例：/usr/lib/ocf/resource.d/heartbeat/apache

#表 7-1 クラスタ設定 … Primitiveリソース

	PRIMITIVE						
	P	id	class	provider	type		
#	リソースID	class	provider	type	概要		
		prmApache	ocf	heartbeat	apache		
	A	type	name	value			
#		パラメータ種別	項目	設定内容		概要	
		params	configfile	/etc/httpd/conf/httpd.conf			
	O	type	timeout	interval	on-fail	start-delay	
#		オペレーション	タイムアウト値	監視間隔	障害時の動作	起動前待機時間	備考
		start	60s	0s	restart		
		monitor	60s	10s	restart		
		stop	60s	0s	block		

リソースID

動作設定

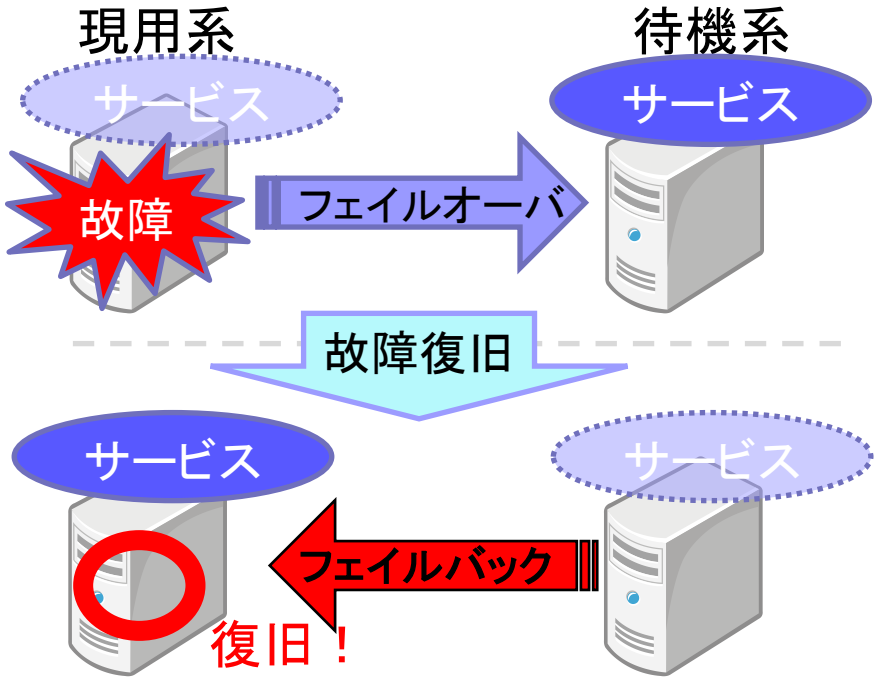
パラメータ

それぞれの処理の実行が失敗した際の挙動を設定します。
・restart : リソースを再起動
・block : 保守者の対応待ち

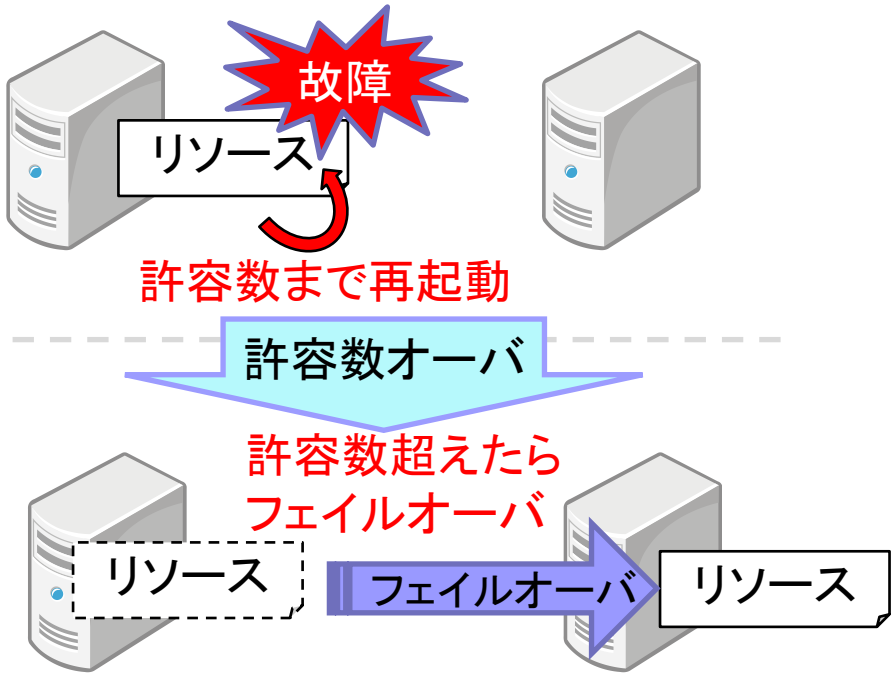
リソース定義・パラメータ設定(3/3)

- リソースの共通デフォルト設定はExcelテンプレートシート 表3-1に設定します。
  - 主に設定する共通設定項目は以下の2点です。

resource-stickiness
(フェイルバックの有無)



migration-threshold
(同一ノード上でのリソース故障許容数)



#表 3-1 クラスタ設定 … リソース・デフォルト

RSC_DEFAULTS

name / value

自動フェイルバックを有効にする場合は0、
無効の場合はINFINITYを設定

#	項目	設定内容	概要	備考
	resource-stickiness	INFINITY	リソース割当て	基本的には「INFINITY」推奨です。
	migration-threshold	1	リソース故障可能回数	何回監視に失敗したらフェイルオーバするか設定します。

Pacemakerのリソース設定

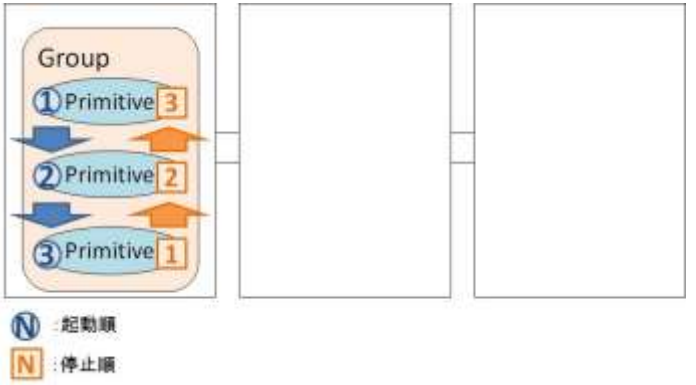
1. リソース定義・パラメータ設定
2. リソース種類選択
3. リソース制約
4. クラスタ設定

リソース種類選択(1/3)

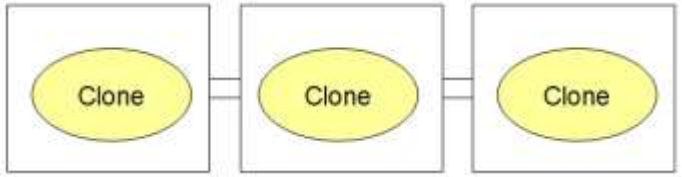
リソースは以下の4種類から選択できます。

- プリミティブリソース
  - クラスタ内で単独起動する。
  - 基本となるリソース種類。

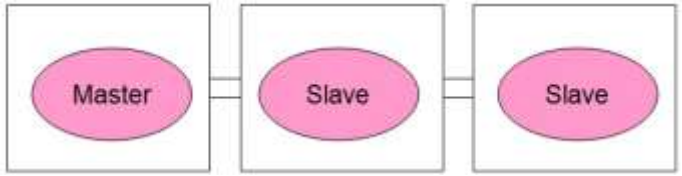
- リソースグループ
  - 複数のプリミティブリソースを一括管理し、1つのリソースの様に動作する。
  - 同一ノード内での起動、および起動順序を保証する。



- クローンリソース
  - 同一リソースを複数のノード上で起動する。



- マスタ/スレーブリソース
  - 複数ノード上にマスタとなるリソースとスレーブとなるリソースを起動する。



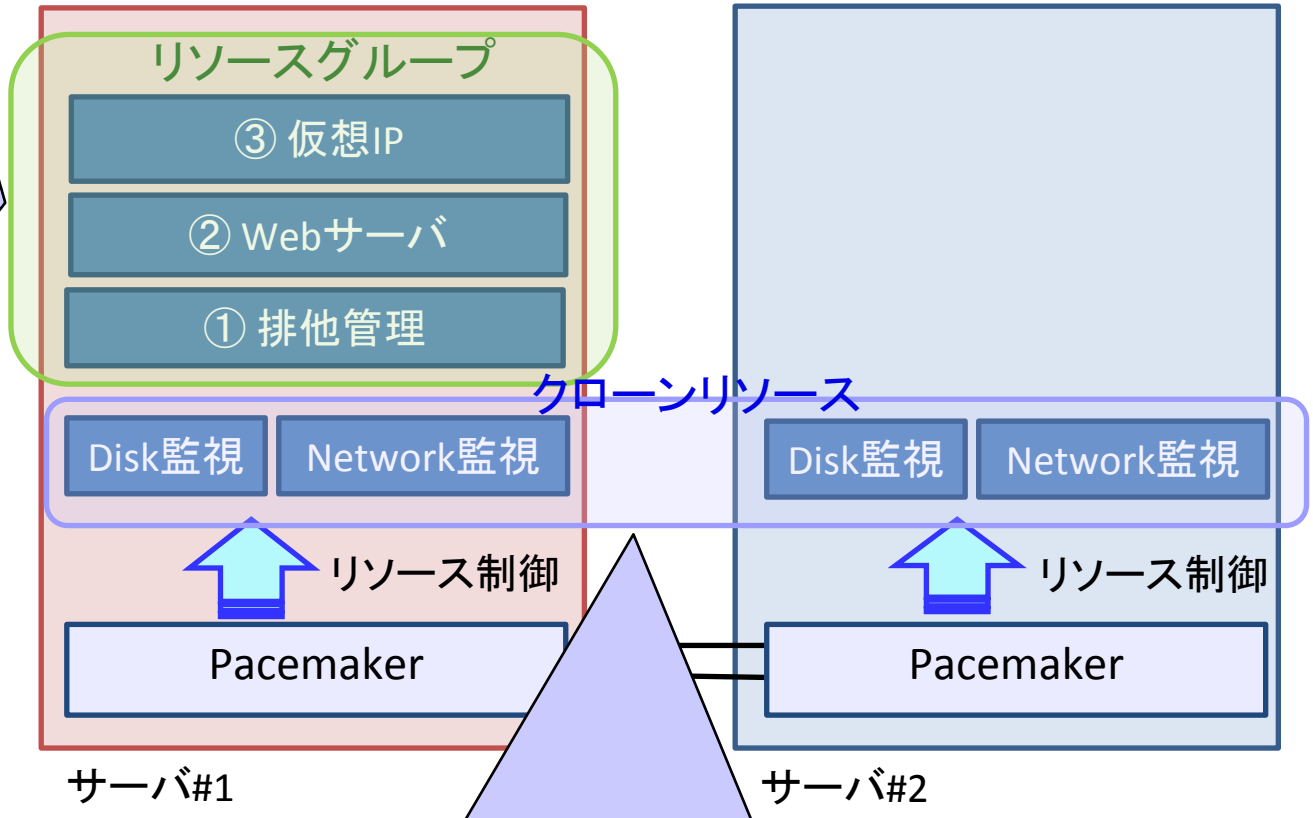
リソース種類選択(2/3)

■ リソース種類の利用例。

サービス提供には以下を現用機で実行する必要があります。

① 他ノードでサービスが起動していないか確認。
② Webサーバ起動
③ ユーザが現用機にアクセスするためのIP割当

上記を同一ノードで一括管理するため、リソースグループとして定義します。



ディスク監視、ネットワーク監視は現用機だけでなく、予備機でも実施し、フェイルオーバ先として問題ないか確認する必要があります。

そのため監視リソースをクローンリソースとして定義し、複数ノード上で起動する様にします。

リソース種類選択(3/3)

- リソースの種類はExcelテンプレートシート 表4-1に設定します。
  - リソース構成要素には以下を設定できます。
    - Primitive：プリミティブリソースを定義、もしくは他リソースに紐付ける際に指定。
    - Group：リソースグループを定義する際に指定。
      - 紐付けるプリミティブリソースの定義した順がリソースの起動順序となる。
    - Clone：クローンリソースを定義する際に指定。
    - Master：マスタ/スレーブリソースを定義する際に指定。
  - リソースIDには以下を設定します。
    - Primitive行：表7で定義したリソースのリソースID
    - Group、Clone、Master行：任意のユニークなリソースID

#表 4-1 クラスタ設定 … リソース構成

RESOURCES

#	resourceItem	resourceItem	resourceItem	id	概要	備考
	リソース構成要素			リソースID		
	Group			grpApache		
		Primitive		prmVipCheck		
		Primitive		prmApache		
		Primitive		prmVIP		
	Clone			clnPing		
		Primitive		prmPing		
	Clone			clnDiskd		
		Primitive		prmDiskd		

Group、Clone、Masterの場合はユニークなリソースID名を定義

リソース種類を指定

起動順序

Group、Clone、Masterを定義する際は先に表7のリソースをPrimitiveとして定義し、それを紐づけている

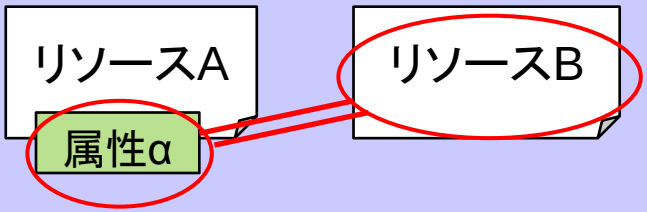
Pacemakerのリソース設定

1. リソース定義・パラメータ設定
2. リソース種類選択
3. リソース制約
4. クラスタ設定

リソース制約とは

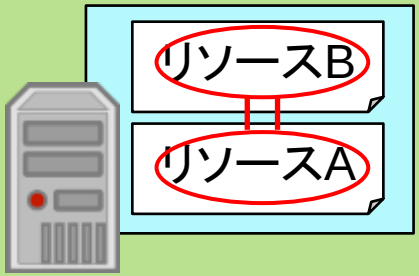
- リソース制約では各リソースが起動するノードや順序のルールを設定します。
  - リソース種類設定だけでは実現できない細かい動作をここで設定できます。
- 制約は主に以下の3種類あります。

リソース配置制約
(LOCATION)



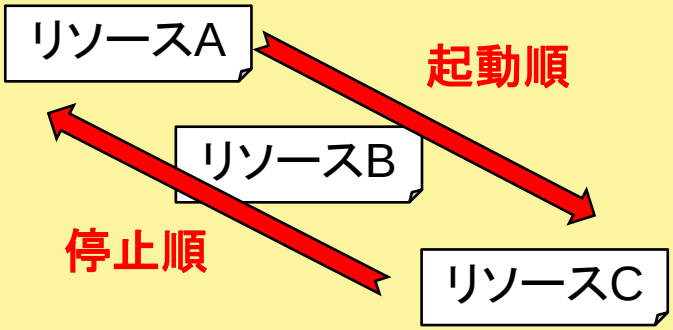
if(属性α == hoge)リソースBを起動

リソース同居制約
(COLOCATION)



リソースAとリソースBを同居

リソース起動順序制約
(ORDER)



リソースが起動するノードを決める設定

リソースが起動する順番を決める設定

リソース配置制約(1/2)

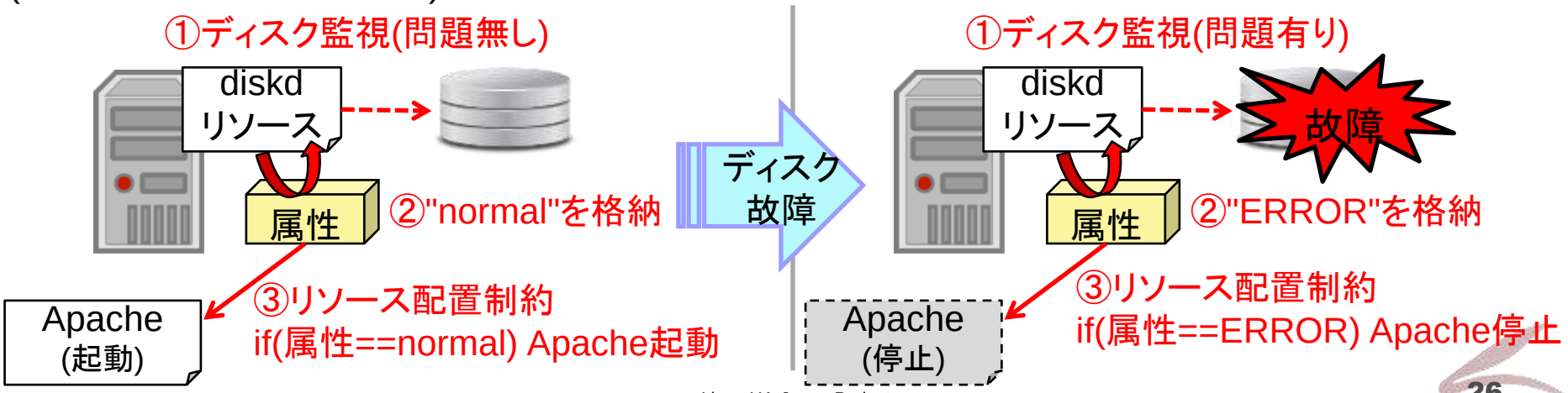
属性等の情報を元にリソースが起動するノードを決めるルールを設定できます。

属性とは
　属性とは、Pacemaker内部で扱われているkey-value型の変数です。
　主にRAがリソース管理に必要な情報を各ノード毎にリアルタイムで属性に格納・更新します。

★リソース配置制約を書く上でよく使われる2つの属性

- diskd RAとping RAのリソースが管理する属性
  - diskd RAはディスク、ping RAはネットワークを監視するためのRAです。
  - 監視結果はそれぞれ属性に格納され、他リソースはその値を元にリソース配置制約で起動・停止します。

(diskd RAのリソースの場合)



リソース配置制約(2/2)

- リソース配置制約はExcelテンプレートシート 表8-1 に設定します。
  - リソース配置制約を含むリソース制約は、各ノード毎にリソース配置の優先度であるスコアを設定します。
  - スコアは数値が入り、値が大きいほど優先度が高くなります。
  - また、「inf」は最優先の配置先、「-inf」は配置の禁止を意味します。

#表 8-1 クラスタ設定 … リソース配置制約

LOCATION_EXPERT

#	rsc リソースID	score スコア	bool_o and/or	attribute 条件属性名	op 条件	value 条件値	role 役割	id_spec ルールID	備考
	grpApache	200		#uname	eq	srv01			ホストsrv01を優先的にActiveにする
		100		#uname	eq	srv02			
		-inf	or	default_ping_set	not_defined				ネットワーク故障が発生したノードでは起動しない
				default_ping_set	lt	100			
		-inf	or	diskcheck_status	not_defined				共有ディスク故障が発生したノードでは起動しない
				diskcheck_status	eq	ERROR			

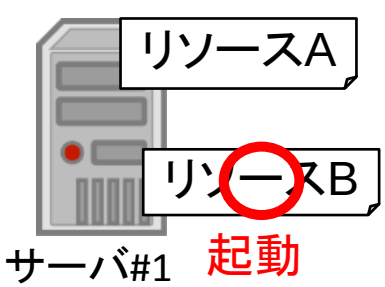
diskdリソースの属性"diskcheck_status"が存在しない (**not_defined**)場合、もしくは(**or**)、値が"**ERROR**"と等しい(**eq**)場合、そのノード上でのgrpApacheリソースの起動を禁止する(スコア = **-inf**)。

(※ diskdリソース、pingリソースの属性名は各リソース定義時にパラメータで指定します。)

リソース同居制約(1/2)

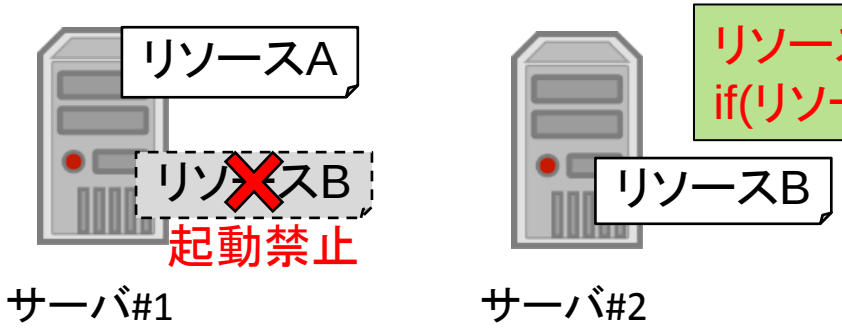
他のリソースの状態を元にリソースが起動するノードを決めるルールを設定できます。

- スコアには「inf」と「-inf」が設定でき、それぞれ以下の挙動になります。
  - inf : 他のリソースが起動しているノードで対象リソースを起動する。




```
リソース同居制約
if(リソースA == start) リソースB.score = inf
```

  - -inf : 他のリソースが起動しているノードで対象リソースを起動しない。



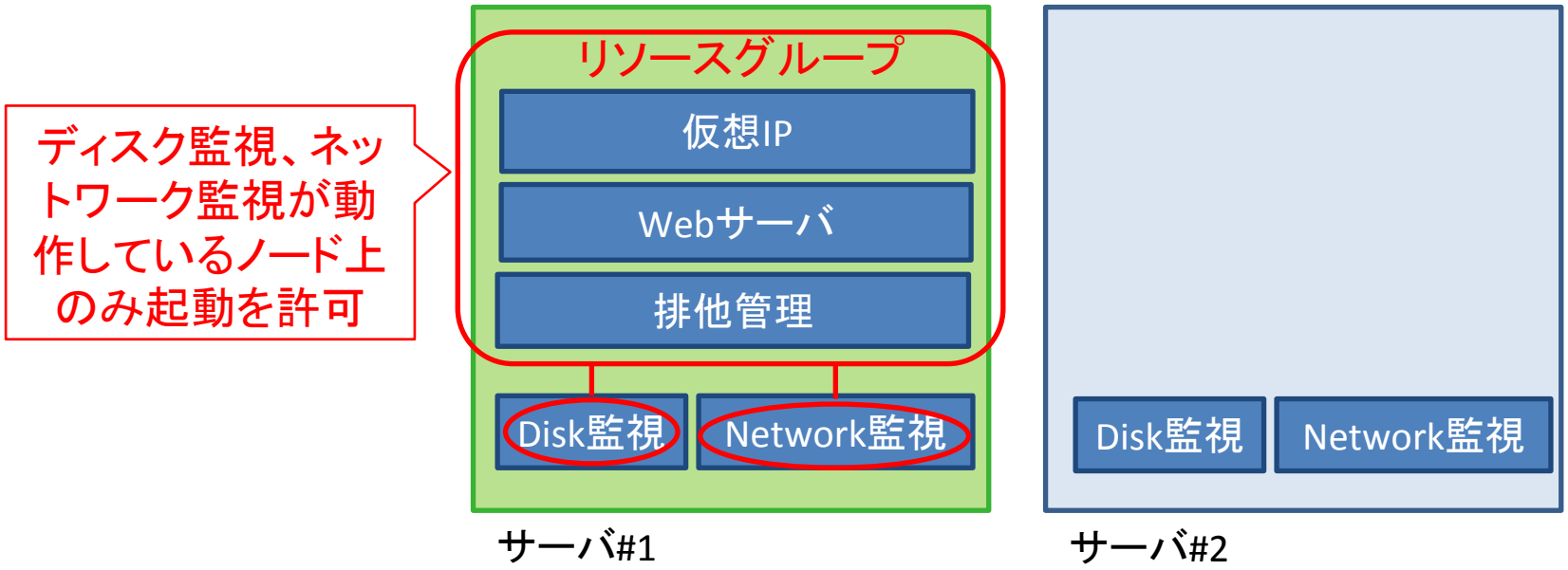

```
リソース同居制約
if(リソースA == start) リソースB.score = -inf
```

リソース同居制約(2/2)

- リソース同居制約はExcelテンプレートシート 表9-1 に設定します。
  - 以下ではclnPing、clnDiskdをgrpApacheと同居する様、設定しています。

#表 9-1 クラスタ設定 … リソース同居制約						
	COLOCATION					
	rsc	with-rsc	score	rsc-role	with-rsc-role	
#	制約関連リソースID	制約対象リソースID	スコア（重み付け）	制約関連リソースの役割	制約対象リソースの役割	備考
	grpApache	clnPing	inf			ネットワーク監視・ディスク監視が行われているノードで起動する
	grpApache	clnDiskd	inf			

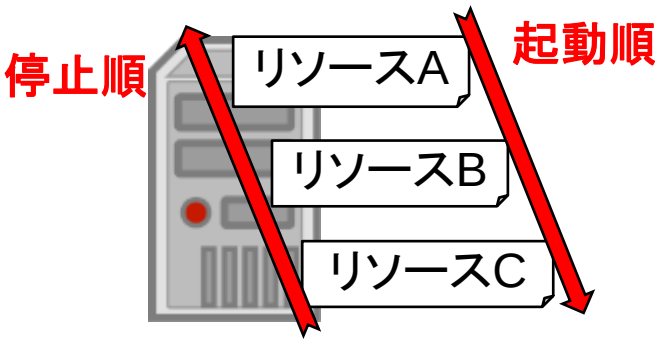
(動作イメージ)



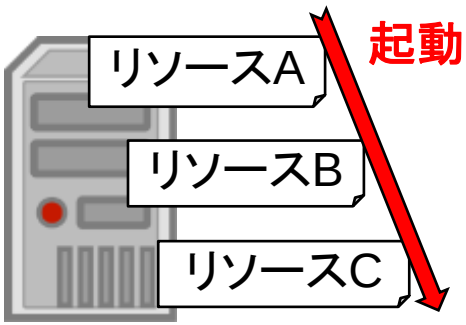
リソース起動順序制約(1/2)

リソースの起動順序を決めるルールを設定できます。
また、リソースグループの様に停止時に逆順で処理する様に設定することも可能です。

逆順の処理を有効にした場合

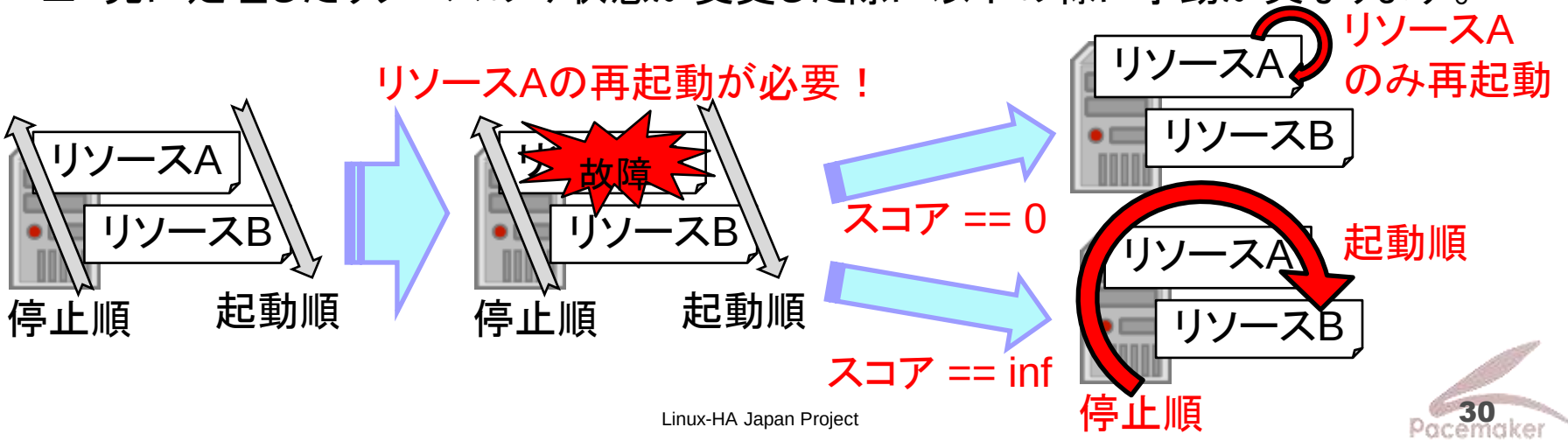


逆順の処理を無効にした場合



この場合、停止順序は別途制約で指定することも可能です。

- スコアにはinfと0が設定できます。
  - infの場合も0の場合も起動順は変わりません。
  - 先に処理したリソースのみ状態が変更した際に以下の様に挙動が異なります。



リソース起動順序制約(2/2)

- リソース起動順序制約はExcelテンプレートシート 表10-1 に設定します。
  - 以下ではclnPing、clnDiskd起動後、grpApacheを起動する様、設定しています。

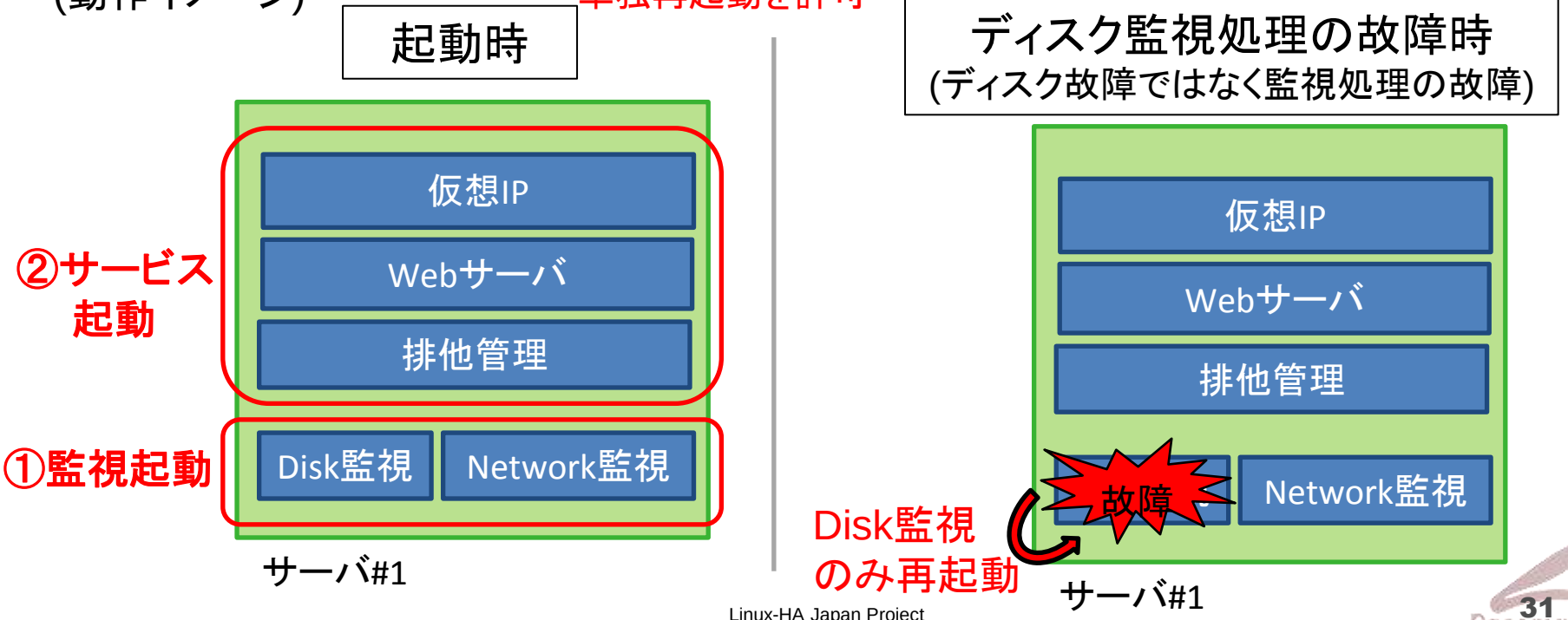
#表 10-1 クラスタ設定 … リソース起動順序制約

ORDER	first-rsc	then-rsc	score	first-action	then-action	symmetrical	
#	先に起動するリソースID	後に起動するリソースID	スコア（重み付け）	先起動リソースのアクション	後起動リソースのアクション	起動と逆順に停止（y/n）	備考
	clnPing	grpApache	0			n	ネットワーク監視・ディスク監視起動後に起動する
	clnDiskd	grpApache	0			n	

先に起動するリソースの単独再起動を許可

逆順の処理を無効

(動作イメージ)



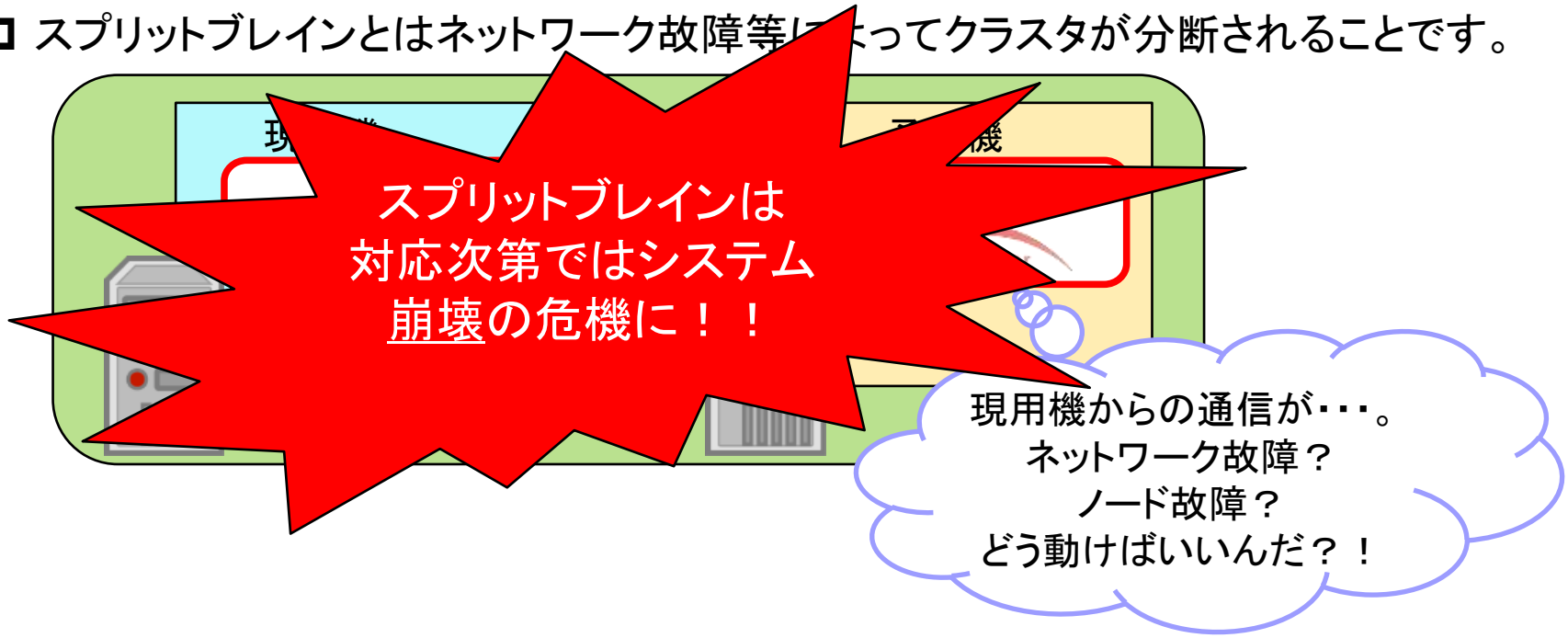
Pacemakerのリソース設定

1. リソース定義・パラメータ設定
2. リソース種類選択
3. リソース制約
4. クラスタ設定

クラスタ設定(1/3)

クラスタ設定ではスプリットブレイン時の動作を設定します。

- スプリットブレインとは？
  - スプリットブレインとはネットワーク故障等によってクラスタが分断されることです。

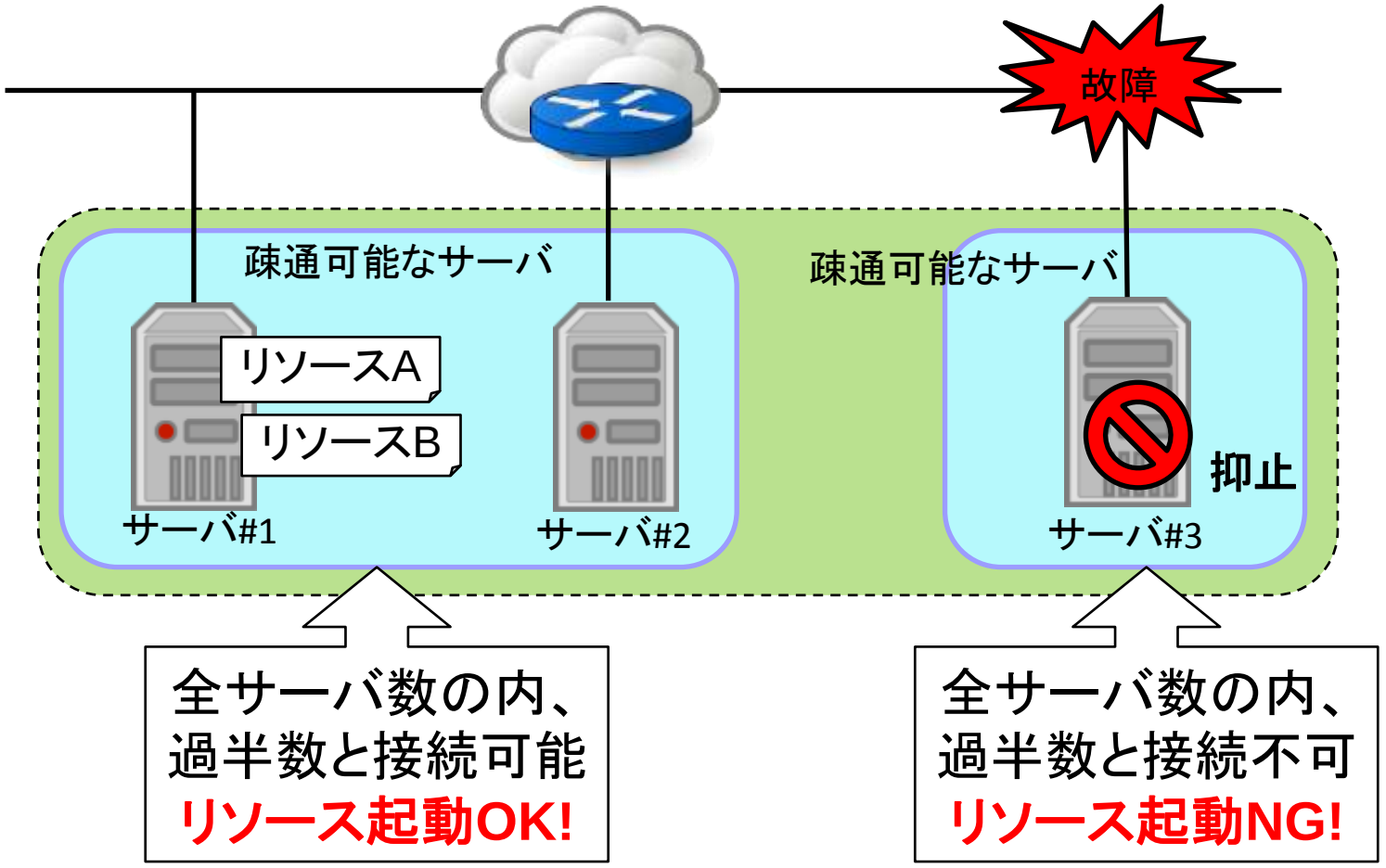


Pacemakerにはスプリットブレインを想定した様々な排他制御が実装されています！詳細は以下をご参照ください。(とっても大切な機能です！)

参照：http://linux-ha.osdn.jp/wp/wp-content/uploads/076783ca53a363270d253bbb98b59e83.pdf

クラスタ設定(2/3)

クラスタ設定ではスプリットブレイン対策の排他制御の1つである「接続可能なサーバ数を参照したリソース制御機能」を設定できます。



クラスタ設定(3/3)

- クラスタ設定はExcelテンプレートシート 表2-1 に設定します。
  - 以下では2ノード構成なので先ほどのスプリットブレイン対策は無効(ignore)にしてます。

#表 2-1 クラスタ設定　…　クラスタ・プロパティ

PROPERTY

#	name	value		
	項目	設定内容	概要	備考
	no-quorum-policy	ignore	ノード数によるリソース割当て	1+1構成の場合は「ignore」を、3台以上の場合は「freeze」を指定します。
	stonith-enabled	false	障害ノード対処（STONITH制御）	STONITHを使用する場合は「ture」、使用しない場合は「false」を指定します。

他のスプリットブレイン対策の1つであるSTONITH機能の有効・無効を設定してます。

STONITH機能については以下を参照ください。
http://linux-ha.osdn.jp/wp/wp-content/uploads/076783ca53a363270d253bbb98b59e83.pdf

さいごに

さいごに

Linux-HA Japan URL

http://linux-ha.osdn.jp/
https://ja.osdn.net/projects/linux-ha/

Pacemaker関連の最新情報を日本語で発信

Pacemakerのダウンロードもこちらからどうぞ
(インストールが楽なリポジトリパッケージを公開しています)

さいごに

日本におけるHAクラスタについての活発な意見交換の場として
「Linux-HA Japan日本語メーリングリスト」も開設しています。

Linux-HA-Japan MLでは、Pacemaker、Heartbeat3、Corosync
DRBDなど、HAクラスタに関連する話題は歓迎！

- ML登録用URL

  http://linux-ha.osdn.jp/
  の「メーリングリスト」をクリック

- MLアドレス

  linux-ha-japan@lists.osdn.me

※スパム防止のために、登録者以外の投稿は許可制です

さいごに

本資料の他にも以下の資料を公開しております。
合わせてご参照ください。

- Pacemakerインストール方法とCorosync設定
  http://linux-ha.osdn.jp/wp/wp-content/uploads/OSC2015_Nagoya.pdf

- リソース排他制御
  http://linux-ha.osdn.jp/wp/wp-content/uploads/076783ca53a363270d253bbb98b59e83.pdf

http://linux-ha.osdn.jp/の『イベント情報』
にて上記を含む活動内容を確認できます。

ご清聴ありがとうございました。

Linux-HA Japan

検索

付録A リソース設定詳細説明

付録A-1. クラスタ･プロパティ設定

- クラスタの設定をします。
  - Excelテンプレートシート 表2-1クラスタ･プロパティ

- 設定内容
  - no-quorum-policy :
    - スプリットブレインが発生した際に、孤立したノードの動作を指定します。
    - 基本的には1+1構成なら"ignore"(リソース起動)、3台以上なら"freeze"(何もしない)を指定します。
  - stonith-enabled :
    - スプリットブレインが発生した際に、STONITHによる対向ノード強制停止(排他制御)を行うか指定します。
    - STONITHを使用できる環境であれば"true"(使用する)、使用できない環境であれば"false"(使用しない)を指定します。
    - "true"を指定するには、表7 にてSTONITHリソースを定義してある必要があります。
  - startup-fencing :
    - 起動時に状態不明ノードが存在する場合に、STONITHを実行するか指定します。
    - STONITHを使用する場合には"true"、使用しない場合は"fase"を指定します。
    - 起動時の状態異常はSTONITHではなく、運用者の保守対応が望ましいため、基本的には"false"を指定します。

#表 2-1 クラスタ設定 … クラスタ･プロパティ

PROPERTY

	name	value		
#	項目	設定内容	概要	備考
	no-quorum-policy	ignore	ノード数によるリソース割当て	1+1構成の場合は「ignore」を、N+1構成の場合は「freeze」を指定します。
	stonith-enabled	false	障害ノード対処（STONITH制御）	STONITHを使用する場合は「ture」、使用しない場合は「false」を指定します。
	startup-fencing	false	起動時に状態不明ノードへSTONITH	基本的には「false」推奨です。

付録A-2. リソース・デフォルト

- リソースのデフォルト設定をします。
  - Excelテンプレートシート 表3-1リソース・デフォルト

- 設定内容
  - resource-stickiness :
    - 故障復旧時などに自動的にフェイルバックするかを指定します。
    - 基本的には以下のいずれかを設定します。
      - 0 : リソースを最適なノードにフェイルバックする。
      - INFINITY : 現状起動しているノードで起動状態を継続し、フェイルバックしない。

    ※ "INFINITY"を指定した場合でも、故障時のフェイルオーバは行います。
  - migration-threshold :
    - 何回monitor(監視)処理に失敗した場合にフェイルオーバを実施するかを指定します。
    - この設定はリソースのオペレーションにmonitorを設定し、且つmonitorの"on-fail"値にrestartを指定している場合のみ動作に反映されます。
    - monitor処理に失敗した際に、失敗回数がこの設定値に満たない場合は、同一ノードでリソースを再起動し、失敗回数がこの設定値以上であればリソースを別ノードへフェイルオーバします。

#表 3-1 クラスタ設定 … リソース・デフォルト

RSC_DEFAULTS

	name	value		
#	項目	設定内容	概要	備考
	resource-stickiness	INFINITY	リソース割当て	基本的には「INFINITY」推奨です。
	migration-threshold	1	リソース故障可能回数	何回監視に失敗したらフェイルオーバするか設定します。

付録A-3. リソース構成

- リソースの構成を設定します。
  - Excelテンプレートシート 表4-1リソース構成

- 設定内容
  - リソースの種類を選択します。
  - 選択可能な種類は以下の4種類です。
    - Primitive : プリミティブリソースを定義する。
    - Group : リソースグループを定義する。
    - Clone : クローンリソースを定義する。
    - Master: マスタ/スレーブリソースを定義する。
  - Primitiveには表7で定義したリソースIDを指定します。
  - Group、Clone、Masterには任意のユニークなリソースIDを設定し、内包するPrimitiveに表7で定義したリソースIDを指定します。
  - Groupは表4-1で定義した際の内包するPrimitiveの順番が、リソースの起動順序となります(上から順に起動する)。

#表 4-1 クラスタ設定 … リソース構成

RESOURCES

#	resourceItem	resourceItem	resourceItem	id	概要	備考
	リソース構成要素			リソースID		
	Primitive			vip-slave	プリミティブ定義	
	Group			master-group	グループ定義	
		Primitive		vip-master	リソース定義	
		Primitive		vip-rep	リソース定義	
	Master			msPostgresql	Master/Slave定義	
		Primitive		pgsql	リソース定義	
	Clone			clnPing	クローン定義	
		Primitive		prmPing	リソース定義	

付録A-4. リソース・パラメータ

- リソースのパラメータを設定します。
  - Excelテンプレートシート 表5-1リソース・パラメータ

- 設定内容
  - 表7と同様にリソースのパラメータを設定可能です。
  - 表7と異なる点として、プリミティブリソースだけでなくリソースグループやクローンリソース、マスタ/スレーブリソースに対しての設定が可能です。
  - 基本的にはマスタ/スレーブリソースの以下のパラメータを設定するのに使用します。
    - master-max :
      - クラスタ内にマスタリソースを起動する数を指定します。
    - master-node-max :
      - 1ノード内にマスタリソースを起動する数を指定します。
    - clone-max :
      - クラスタ内にマスタ or スレーブリソースを起動する数を指定します。
    - clone-node-max :
      - 1ノード内にマスタ or スレーブリソースを起動する数を指定します。
    - notify :
      - マスタ/スレーブリソースの起動/停止処理の結果を他ノードのリソースへ通知するか否かを設定します(詳細の動きに関してはRAによって異なります)。
      - 通知する場合は"true"、通史しない場合は"false"を指定します。

#表 5-1 クラスタ設定 … リソース・パラメータ

RSC_ATTRIBUTES

	id	type	name	value	
#	リソースID	ﾊﾟﾗﾒｰﾀ種別	項目	設定内容	備考
	msPostgresql	meta	master-max	1	
			master-node-max	1	
			clone-max	2	
			clone-node-max	1	
			notify	true	

付録A-5. STONITHの実行順序

- STONITHの実行順序を設定します。
  - Excelテンプレートシート 表6-1STONITHの実行順序

- 設定内容
  - STONITH対象のノード名、表7に定義したSTONITHリソースのリソースID、実行順序を設定します。
  - 基本的には各ノード毎にstonith-helperリソースを起動順序"1"として、STONITH実行リソースを起動順序"2"として定義します。

#表 6-1 クラスタ設定 … STONITHの実行順序

FENCING_TOPOLOGY

#	node STONITHの対象ノード	rsc 実行するSTONITHリソースID	index 実行順序	備考
	srv01	prmStonith1-1	1	rsc列には、「STONITH(Primitive)リソースのID」を設定すること
		prmStonith1-2	2	
	srv02	prmStonith2-1	1	
		prmStonith2-2	2	

付録A-6-1. Primitiveリソース

- リソースを定義します。
  - Excelテンプレートシート 表7Primitiveリソース

- 設定内容
  - 任意のユニークなリソースID、使用するRA、各パラメータ、動作を設定します。
  - RAは以下を設定して指定します。
    - class : RAの仕様規格を指定します。
    - provider : /usr/lib/ocf/resource.d/配下にあるRA格納ディレクトリを指定します。
    - type : RAのファイル名を指定します。
  - パラメータの設定項目はRAによって異なります。
  - オペレーションは以下が定義でき、それぞれタイムアウトや監視間隔(monitorのみ)、故障時の動作を設定できます(故障時の動作については次ページで詳細説明)。
    - start : リソースを起動する際の動作を設定します。
    - stop : リソースを停止する際の動作を設定します。
    - monitor :
      - リソースを監視する際の動作を設定します。
      - マスタ/スレーブリソースの場合は、マスタ用とスレーブ用のmonitorを分けて定義する必要があります。この場合、新規に"role"の設定列を追加し、マスタ用のmonitorには"Master"を、スレーブ用のmonitorには"Slave"を設定します。

    (以下はマスタ/スレーブリソース限定のオペレーション)
    - promote : リソースをマスタに昇格する際の動作を設定します。
    - demote : リソースをスレーブに降格する際の動作を設定します。
    - notify :
      - start、stopの実行結果を他ノードに通知する処理の動作を設定します。
      - このオペレーションは別途表5-1にてnotifyに"ture"を設定する必要があります。

付録A-6-2. Primitiveリソース

- オペレーション故障時の動作設定
  - オペレーション故障時の動作は各オペレーションのon-failにて設定します。
  - 設定可能な値は以下の通りです。
    - block : Pacemakerによるリソース管理を停止し、保守者の対応待ちとなります。
    - fence : 故障発生したノードをクラスタから隔離し、フェイルオーバします(STONITH使用時限定)。
    - ignore : 故障対応は特に行わず、次の処理へ遷移します。
    - stop : リソースを停止し、他ノードへのフェイルオーバは行われません。
    - restart : リソースを再起動します(故障回数によっては他ノードへフェイルオーバされる)。
    - nothing : restartと同様の処理となります。
  - 基本的にはstart、monitor、promoteのon-failに"restart"を設定します。また、stop、demoteのon-failには、故障してもサービスに直接影響のないリソースの場合は"ignore"、サービスに影響がありSTONITHを使用している場合は"fence"、サービスに影響がありSTONITHを使用していない場合は"block"を設定します。

#表 7-1 クラスタ設定 … Primitiveリソース

	PRIMITIVE							
	P	id	class	provider	type			
#		リソースID	class	provider	type	概要		
		prmDiskd	ocf	pacemaker	diskd			
	A	type	name	value				
#		パラメータ種別	項目	設定内容		概要		
		params	name	diskcheck_status				
			device	/dev/sda				
			options	-e				
			interval	10				
			dampen	2				
	O	type	timeout	interval	on-fail	start-delay		
#		オペレーション	タイムアウト値	監視間隔	障害時の動作	起動前待機時間	備考	
		start	60s	0s	restart			
		monitor	60s	10s	restart			
		stop	60s	0s	ignore			

付録A-7-1. リソース配置制約

- リソース配置制約を設定します。
  - Excelテンプレートシート 表8-1リソース配置制約
  - Pacemakerはリソース起動可能なノードが複数ある場合、より優先度の高いノードでリソースを起動する。
  - リソース配置制約では属性(Pacemaker内部変数)の値やノード名を元に、対象リソースが起動するノードの優先度を決定づけるルールを設定できます。
  - 優先度はリソース毎にスコアと条件を指定することで設定でき、指定した条件を満たす場合にスコアの値を優先度として反映されます。

- 設定内容
  - rsc：
    - 制約の対象となるリソースのリソースIDを設定します。
    - 表4で定義したリソースIDを指定してください。
      - プリミティブリソースとして使用するリソースのみ、表7で定義したリソースIDを指定します。
  - score：
    - "-INFINITY"～"INFINITY"の数字を指定でき、値が大きいほど優先度が高くなります。また、数字ではなく"-INFINITY"、"INFINITY"を指定した場合は、以下の意味として反映されます。
      - -INFINITY : 条件が満たされているノードでは対象リソースの起動(もしくは昇格)を禁止する。
      - INFINITY : 条件が満たされているノードで対象リソースを起動(もしくは昇格) する。
  - bool_op :
    - 複数の条件のいずれか、もしくは両方が満たされた際に優先度を変更したい場合に"and"もしくは"or"を設定します。
      - and : 複数の条件を全て満たした場合に優先度を変更する。
      - or : 複数の条件のいずれかを満たした場合に優先度を変更する。
  - attribute :
    - ルールにて参照する情報を設定します。ノード名を参照する場合は"#uname"を、属性の値を参照する場合は属性名を設定します。

付録A-7-2. リソース配置制約

- 設定内容
  - op :
    - attributeとvalueの比較式、もしくはattributeが属性名の場合はattributeの有無を設定します。
      - eq : attribute = value の場合に適合する。
      - ne : attribute ≠ value の場合に適合する。
      - lt : attribute ＜ value の場合に適合する。
      - gt : attribute ＞ value の場合に適合する。
      - lte : attribute ≦ value の場合に適合する。
      - gte : attribute ≧ value の場合に適合する。
      - defined : attributeが存在する場合に適合する。
      - not_defined : attributeが存在しない場合に適合する。
  - value :
    - opにeq、ne、lt、gt、lte、gteを設定した場合に設定します。
  - role :
    - 対象リソースのどの処理に対して優先度を設定するかを指定できます。
    - 設定しない場合はリソースの起動に対しての優先度となります。
    - "master"を設定した場合はリソースの昇格に対して優先度を設定できます(マスタ/スレーブリソースのみ可能)。

付録A-7-3. リソース配置制約

- よく使用される設定

#表 8-1 クラスタ設定 … リソース配置制約

LOCATION_EXPERT

	rsc	score	bool_op	attribute	op	value	role	id_spec	
#	リソースID	スコア	and/or	条件属性名	条件	条件値	役割	ルールID	備考
①	grpPostgreSQL	200		#uname	eq	srv01			
		100		#uname	eq	srv02			
②		-inf	or	default_ping_set	not_defined				
				default_ping_set	lt	100			
		-inf	or	diskcheck_status	not_defined				
				diskcheck_status	eq	ERROR			
③	grpStonith1	-inf		#uname	eq	srv01			
	grpStonith2	-inf		#uname	eq	srv02			

①: リソースに対して、サーバ名に応じた優先度を設定しています。
上記の例では、grpPostgreSQLリソースグループが起動するノードとして、
サーバ名がsrv01のノードの方がsrv02のノードよりも優先度を高く設定
しています。

②: リソースに対して、属性の値に応じた優先度を設定しています。
上記の例では、grpPostgreSQLリソースグループに対して、default_ping_set
(pingリソースの属性名)が存在しない場合、もしくは値が100未満の場合に、
起動禁止の制約を設定しています。
またdiskcheck_status(diskdリソースの属性名)が存在しない場合、値が
ERRORの場合も同様の制約を設定しています。

③: リソースに対して、特定のノード上では起動しない様に設定しています。
上記の例では、STONITHリソース(grpStonith1、grpStonith2)をSTONITH対象
と同一ノード上で起動しない様に設定しています。

付録A-8-1. リソース同居制約

- リソース同居制約を設定します。
  - Excelテンプレートシート 表9-1リソース同居制約
  - Pacemakerはリソース起動可能なノードが複数ある場合、より優先度の高いノードでリソースを起動する。
  - リソース同居制約では他リソースの起動有無を元に、対象リソースが起動するノードの優先度を決定づけるルールを設定できます。
  - 優先度はリソース毎にスコアと条件を指定することで設定でき、指定した条件を満たす場合にスコアの値を優先度として反映されます。

- 設定内容
  - rsc：
    - 優先度を設定する対象リソースのリソースIDを設定します。
    - 表4で定義したリソースIDを指定してください。
      - プリミティブリソースとして使用するリソースのみ、表7で定義したリソースIDを指定します。
  - with-rsc：
    - 起動条件で参照するリソースIDを指定します。
  - score：
    - "-INFINITY"もしくは"INFINITY"を指定できます。
      - -INFINITY：条件が満たされているノードでは対象リソースの起動(もしくは昇格)を禁止する。
      - INFINITY：条件が満たされているノードで対象リソースを起動(もしくは昇格)する 。
  - rsc-role：
    - rscで指定した対象リソースのどの処理に対して優先度を設定するかを指定できます。
    - 設定しない場合はリソースの起動に対しての優先度となります。
    - マスタ/スレーブリソースの場合は"master"を設定することで、リソースの昇格に対して優先度を設定できます。

付録A-8-2. リソース同居制約

- 設定内容
  - with-rsc-role :
    - 条件としてwith-rscで指定したリソースがどの状態の場合に適合するかを指定できます。
    - 設定しない場合はリソースが起動している際に適合します。
    - "master"を設定した場合、リソースがマスタの場合に適合します(マスタ/スレーブリソースのみ可能)。

- よく使用される設定

#表 9-1 クラスタ設定 … リソース同居制約

COLOCATION						
	rsc	with-rsc	score	rsc-role	with-rsc-role	
#	制約関連リソースID	制約対象リソー	スコア（重み付け）	制約関連リソースの役割	制約対象リソース	備考
①	grpPostgres	clnPing	inf			
	grpPostgres	clnDiskd	inf			

①: リソースに対して、他リソースの起動状態に応じた優先度を設定しています。
上記の例では、clnPing(pingリソース)およびclnDiskd(diskdリソース)が起動しているノードで
grpPostgresリソースグループが起動を許可しています。

付録A-9-1. リソース起動順序制約

- リソース起動順序制約を設定します。
  - Excelテンプレートシート 表10-1リソース起動順序制約
  - リソースの起動順序を制約で定義します。

- 設定内容
  - first-rsc：
    - 先に処理を実行するリソースのリソースIDを設定します。
    - 表4で定義したリソースIDを指定してください。
      - プリミティブリソースとして使用するリソースのみ、表7で定義したリソースIDを指定します。
    - 処理内容はfirst-actionで設定します。
  - then-rsc：
    - 後に処理を実行するリソースのリソースIDを指定します。
  - score：
    - "INFINITY"もしくは"0"を指定します。どちらを設定した場合も処理順は変わりません。
      先に処理したリソースの状態が変更した場合に以下の様に挙動が変わります。
      - INFINITY : 先に処理したリソースのみ状態変化(リソース停止等)が発生し条件を満たせなくなった場合に、それに合わせて後から処理したリソースにも停止処理を実行する。
      - 0 : 先に処理したリソースに状態変化が発生し条件を満たせなくなった場合でも、後から処理したリソースは状態を継続する。
  - first-action :
    - first-rscで指定した対象リソースの処理を指定できます。
    - 以下の設定が可能であり、設定をしていない場合は"start"が適用されます。
      - start : リソースを起動する。
      - stop : リソースを停止する。
      - promote : リソースを昇格する(マスタ/スレーブリソースのみ)。
      - demote : リソースを降格する(マスタ/スレーブリソースのみ)。

付録A-9-2. リソース起動順序制約

- then-action :
  - first-action 確認後に行う、then-rscで指定した対象リソースの処理を指定できます。
  - 以下の設定が可能であり、設定をしていない場合はfirst-actionと同様の値が適用されます。
    - start : リソースを起動する。
    - stop : リソースを停止する。
    - promote : リソースを昇格する(マスタ/スレーブリソースのみ)。
    - demote : リソースを降格する(マスタ/スレーブリソースのみ)。
- symmetrical :
  - 制約と逆順の制約を設けることができます。
    - true : 逆順の制約を定義する。
      - (例) リソースA起動 ⇒ リソースB起動 の制約でsymmetrical=trueの場合：
        リソースB停止 ⇒ リソースA停止 の制約を定義する。
    - false : 逆順の制約を定義しない。
- よく使用される設定

#表 10-1 クラスタ設定 … リソース起動順序制約

ORDER

#	first-rsc 先に起動するリソースID *A	then-rsc 後に起動するリソースID *A	score スコア（重み付け） *A	first-action 先起動ﾘｿｰｽのアクション *A	then-action 後起動ﾘｿｰｽのアクション *A	symmetrical 起動と逆順に停止（y/n） *A	備考
①	clnPing	grpPostgreSQL	0			n	
	clnDiskd	grpPostgreSQL	0			n	

①: リソースの起動順序を設定しています。
上記の例では、clnPing(pingリソース)およびclnDiskd(diskdリソース)が起動してから
grpPostgreSQLリソースグループを起動する様にしています。
尚、grpPostgreSQLリソースグループ内の各プリミティブリソースの起動順序は既に
リソースグループ定義時に決定しているため、ここでの制約は不要です。

付録B リソース設定反映手順

付録B-1. リソース設定反映手順

- 設定反映手順
  - CSVファイル作成
    - Excelテンプレートシートを編集し、CSVファイルで保存
    - [ファイル] > [名前を付けて保存] > ファイルの種類[CSV (カンマ区切り)(*.csv)]
  - CRMファイル作成
    - CSVファイルをサーバへ転送し、以下のコマンドを実行

```
# pm_crmgen -o <CRMファイル名> <CSVファイル名>
```

  - CRMファイルをPacemakerにロード
    - Pacemakerが起動中且つ、他の設定が反映されていない状態で以下のコマンドを実行
      (以下のコマンドはどちらか片系のみで実行)

```
# crm configure load update <CRMファイル名>
```

ここに注意！！！
(詳細は次ページ参照)

- リソース起動状態を確認
  - 以下のコマンドを実行することで、Pacemakerのリソース状態を確認できます。

```
# crm_mon -fA
```

付録B-2. リソース設定反映手順

- 既にCRMファイルを読み込ませているPacemakerに追加で読み込ませた場合、設定が重複して読み込まれ、正常動作しない恐れがあります。
- 以下のいずれかの手順で以前の設定を削除してから読み込ませてください。
  - オフライン

Pacemakerが停止している状態で以下を実行
1. 以下のファイルを削除【両系でコマンド実行】
   # rm /var/lib/pacemaker/cib/*
2. Pacemaker起動【両系でコマンド実行】
   (RHEL 6系) # initctl start pacemaker.combined
   (RHEL 7系) # systemctl start pacemaker
3. 新しいCRMファイルを反映【片系でコマンド実行】
   # crm configure load update *<CRMファイル名>*

  - オンライン

Pacemakerが起動している状態で以下を実行
1. 起動中にリソースを全て停止【片系でコマンド実行】
   # crm resource stop *<リソースID>*
2. 入力済みの設定を削除【片系でコマンド実行】
   # crm configure erase
3. 新しいCRMファイルを反映【片系でコマンド実行】
   # crm configure load update *<CRMファイル名>*

付録C 講演で使用したリソース設定紹介

付録C-1.講演のリソース設定

 # pm_crmgen 環境定義書 ファイル形式バージョン: 2.1

#表 1-1 クラスタ設定　…　クラスタ・ノード属性

NODE

	uname	ntype	ptype	name	value	
#	ノード名	ノード種	パラメータ種別	項目	設定内容	備考

#表 2-1 クラスタ設定　…　クラスタ・プロパティ

PROPERTY

	name	value		
#	項目	設定内容	概要	備考
	no-quorum-policy	ignore	ノード数によるリソース割当て	
	stonith-enabled	false	障害ノード対処（STONITH制御）	

#表 3-1 クラスタ設定　…　リソース・デフォルト

RSC_DEFAULTS

	name	value		
#	項目	設定内容	概要	備考
	resource-stickiness	INFINITY	リソース割当て	
	migration-threshold	1	リソース故障可能回数	

#表 3-2 クラスタ設定　…　オペレーション・デフォルト

OP_DEFAULTS

	name	value		
#	項目	設定内容	概要	備考

付録C-2.講演のリソース設定

#表 4-1 クラスタ設定　…　リソース構成

RESOURCES

#	resourceItem	resourceIt	resourc	id	概要	備考
	リソース構成要素			リソースID		
	Group			grpApache	グループ定義	
		Primitive		prmVIPcheck	仮想IP(VIP)排他制御	
		Primitive		prmVIP	仮想IP割当	
		Primitive		prmApache	Apache制御	
	Clone			clnPing	クローン定義	
		Primitive		prmPing	ネットワーク監視	
	Clone			clnDiskd	クローン定義	
		Primitive		prmDiskd	内蔵ディスク監視	

#表 5-1 クラスタ設定　…　リソース・パラメータ

RSC_ATTRIBUTES

#	id	type	name	value	備考
	リソースID	ﾊﾟﾗﾒｰﾀ	項目	設定内容	

#表 6-1 クラスタ設定　…　STONITHの実行順序

FENCING_TOPOLOGY

#	node	rsc	index	備考
	STONITHの対象ノード	実行するSTONITHリソ	実行順序	

付録C-3.講演のリソース設定

#表 7-1-1 クラスタ設定 … Primitiveリソース (id=prmVIPcheck)

PRIMITIVE

	P	id	class	provider	type	
#		リソースID	class	provider	type	概要
		prmVIPcheck	ocf	heartbeat	VIPcheck	仮想IP(VIP)排他制御

	A	type	name	value	
#		パラメータ種別	項目	設定内容	概要
		params	target_ip	**192.168.146.200**	確認するIPアドレス (接続用VIP以外の同セグメント内の範囲で適当なVIP)
			count	1	実行回数
			wait	10	wait値

	O	type	timeout	interval	on-fail	start-delay	
#		オペレーション	タイムアウト値	監視間隔	on_fail(障害時の動作	起動前待機時間	備考
		start	90s	0s	restart	6s	

#表 7-1-2 クラスタ設定 … Primitiveリソース (id=prmVIP)

PRIMITIVE

	P	id	class	provider	type	
#		リソースID	class	provider	type	概要
		prmVIP	ocf	heartbeat	IPaddr2	仮想IP割当

	A	type	name	value	
#		パラメータ種別	項目	設定内容	概要
		params	ip	**192.168.146.200**	Apache接続用仮想IPアドレス
			nic	**eno33554984**	同デバイス名
			cidr_netmask	**28**	同ネットマスク

	O	type	timeout	interval	on-fail	start-delay	
#		オペレーション	タイムアウト値	監視間隔	on_fail(障害時の動作	起動前待機時間	備考
		start	60s	0s	restart		
		monitor	60s	10s	restart		
		stop	60s	0s	block		

付録C-4.講演のリソース設定

#表 7-1-3 クラスタ設定 … Primitiveリソース (id=prmApache)

	PRIMITIVE						
	P	id	class	provider	type		
#		リソースID	class	provider	type		概要
		prmApache	ocf	heartbeat	apache		ldirectord監視
	A	type	name	value			
#		パラメータ種別	項目	設定内容			概要
		params	configfile	/etc/httpd/conf/httpd.conf			Apache設定ファイル
	O	type	timeout	interval	on-fail	start-delay	
#		オペレーション	タイムアウト値	監視間隔	on_fail(障害時の動作	起動前待機時間	備考
		start	60s	0s	restart		
		monitor	60s	10s	restart		
		stop	60s	0s	block		

#表 7-2-1 クラスタ設定 … Primitiveリソース (id=prmPing)

	PRIMITIVE						
	P	id	class	provider	type		
#		リソースID	class	provider	type		概要
		prmPing	ocf	pacemaker	ping		ネットワーク監視
	A	type	name	value			
#		パラメータ種別	項目	設定内容			概要
		params	name	default_ping_set			ネットワーク監視用の属性名
			host_list	**192.168.146.1**			監視するネットワークのIPアドレス
			multiplier	100			属性値
			attempts	2			リトライ回数
			timeout	2			タイムアウト
			debug	true			ping先故障検知時のログ出力
	O	type	timeout	interval	on-fail	start-delay	
#		オペレーション	タイムアウト値	監視間隔	on_fail(障害時の動作	起動前待機時間	備考
		start	60s	0s	restart		
		monitor	60s	10s	restart		
		stop	60s	0s	ignore		

付録C-5.講演のリソース設定

#表 7-3-1 クラスタ設定　…　Primitiveリソース (id=prmDiskd)

PRIMITIVE

	P	id	class	provider	type	
#		リソースID	class	provider	type	概要
		prmDiskd	ocf	pacemaker	diskd	内蔵ディスク監視

	A	type	name	value	
#		パラメータ種別	項目	設定内容	概要
		params	name	diskcheck_status_internal	内蔵ディスク用の属性名
			device	**/dev/sda**	【内蔵ディスクパーティション名】 監視する内蔵ディスクのパーティション番号を除いたディスクパーティション名
			options	-e	監視オプション(スレッドタイマー→有効)
			interval	10	監視間隔
			dampen	2	属性更新待ち時間

	O	type	timeout	interval	on-fail	start-delay	
#		オペレーション	タイムアウト値	監視間隔	on_fail(障害時の動作	起動前待機時間	備考
		start	60s	0s	restart		
		monitor	60s	10s	restart		
		stop	60s	0s	ignore		

#表 8-1 クラスタ設定　…　リソース配置制約

LOCATION_EXPERT

	rsc	score	bool_op	attribute	op	value	role	id_spec	
#	リソースID	スコア	and/or	条件属性名	条件	条件値	役割	ルールID	備考
	grpApache	200		#uname	eq	**srv01**			ホストsrv01を優先的にActiveにする
		100		#uname	eq	**srv02**			
		-inf	or	default_ping_set	not_defined				ネットワーク故障が発生したノードでは起動しない
				default_ping_set	lt	100			
		-inf	or	diskcheck_status_	not_defined				内蔵ディスク故障が発生したノードでは起動しない
				diskcheck_status_	eq	ERROR			

付録C-6.講演のリソース設定

#表 9-1 クラスタ設定 … リソース同居制約

COLOCATION

#	rsc	with-rsc	score	rsc-role	with-rsc-role	備考
	制約関連リソースID	制約対象リソー	スコア（重み付け）	制約関連ﾘｿｰｽの役割	制約対象リソース	
	grpApache	clnPing	inf			
	grpApache	clnDiskd	inf			

#表 10-1 クラスタ設定 … リソース起動順序制約

ORDER

#	first-rsc	then-rsc	score	first-action	then-action	symmetrical	備考
	先に起動するリソースID	後に起動するリ	スコア（重み付け）	先起動ﾘｿｰｽのアクシ	後起動ﾘｿｰｽのア	起動と逆順に	
	clnPing	grpApache	0			n	
	clnDiskd	grpApache	0			n	

#表 11-1 クラスタ設定 … リソースチケット制約

RSC_TICKET

#	ticket	rsc	role	loss-policy	備考
	チケットID	チケット付与時に起動す	役割	ﾁｹｯﾄ剥奪時のア	備考

#表 12-1 クラスタ設定 … 追加設定

ADDITIONAL_CONFIG

#	config	備考
	追加設定 (記述内容がそのまま出力されます)	備考