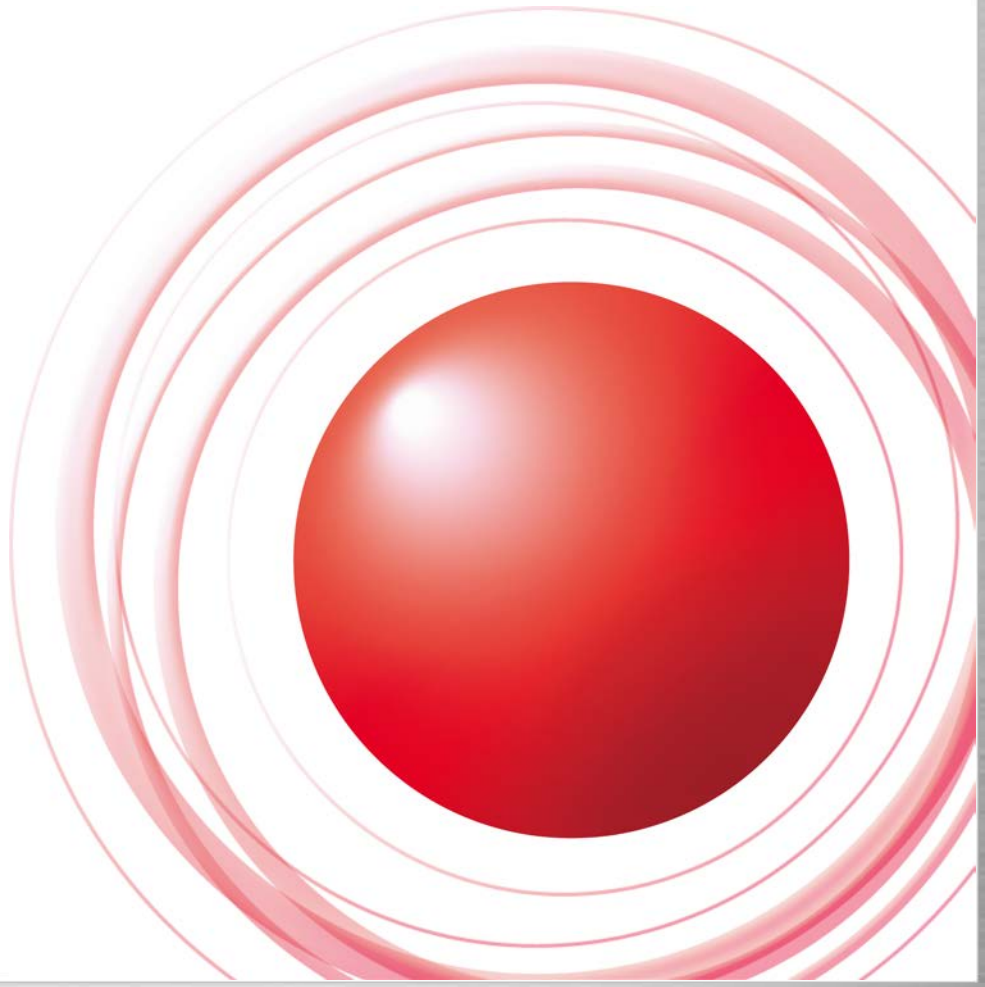
IIJ GIOインフラストラクチャP2 パブリックリソースにおけるストレージ機能強化に向けて

Internet Initiative Japan



株式会社インターネットイニシアティブ

Ongoing Innovation

本日の内容

- **ストレージ機能強化の内容に関して**

  クラウド本部
  クラウドサービス1部
  パブリックリソース２課
  安達　賢

- **カーネルモジュールの実装に関して**

  クラウド本部
  クラウドサービス1部
  パブリックリソース２課
  小野　航平

サービスのご紹介

IIJ GIOインフラストラクチャP2

パブリックリソース

仮想サーバ

仮想サーバ	ベストエフォートタイプ
	性能保証タイプ
	専有タイプ
システムストレージ	

追加ストレージ	ベストエフォートタイプ
	性能保証タイプ

ストレージアーカイブ

ネットワーク

FW+LB	ベストエフォートタイプ
	専有タイプ

プライベートリソース

仮想化プラットフォーム VWシリーズ

VWシリーズ	ベースセット
	vSphere ESXiサーバ
	データストア

物理サーバ

物理サーバ	シングルタイプ
	クラスタタイプ
FC接続ストレージ	

ネットワーク

コネクタ	サービスコネクタ
	DCコネクタ
	FCストレージコネクタ

ストレージリソース

NASサーバ	プライマリNFSサーバ
	プライマリCIFSサーバ

NASボリューム	NFSボリューム
	CIFSボリューム

パブリックリソース

仮想サーバ

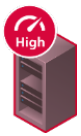
CPUリソースとメモリリソース、
ネットワーク帯域が割り当てられた
仮想サーバを提供

ベストエフォートタイプ

CPU性能を時間で割り当て、ベストエフォートで性能を発揮
•サーバ起動時間とネットワーク転送量の従量課金

性能保証タイプ

CPU、メモリを固定で割り当て、一定の安定した性能を発揮
•月額固定課金（日割り）



専有タイプ

物理サーバを1台の仮想サーバで専有し、SSD、ioMemoryのローカルストレージを利用可能
・月額固定課金（日割り）

×

システムストレージ

仮想サーバで利用可能なOSがインストールされたストレージを提供

CentOS
30GB

Ubuntu
30GB

RedHat Enterprise Linux
30GB

Windows Server
60GB

※ システムストレージの容量には、OSが含まれます。

＋

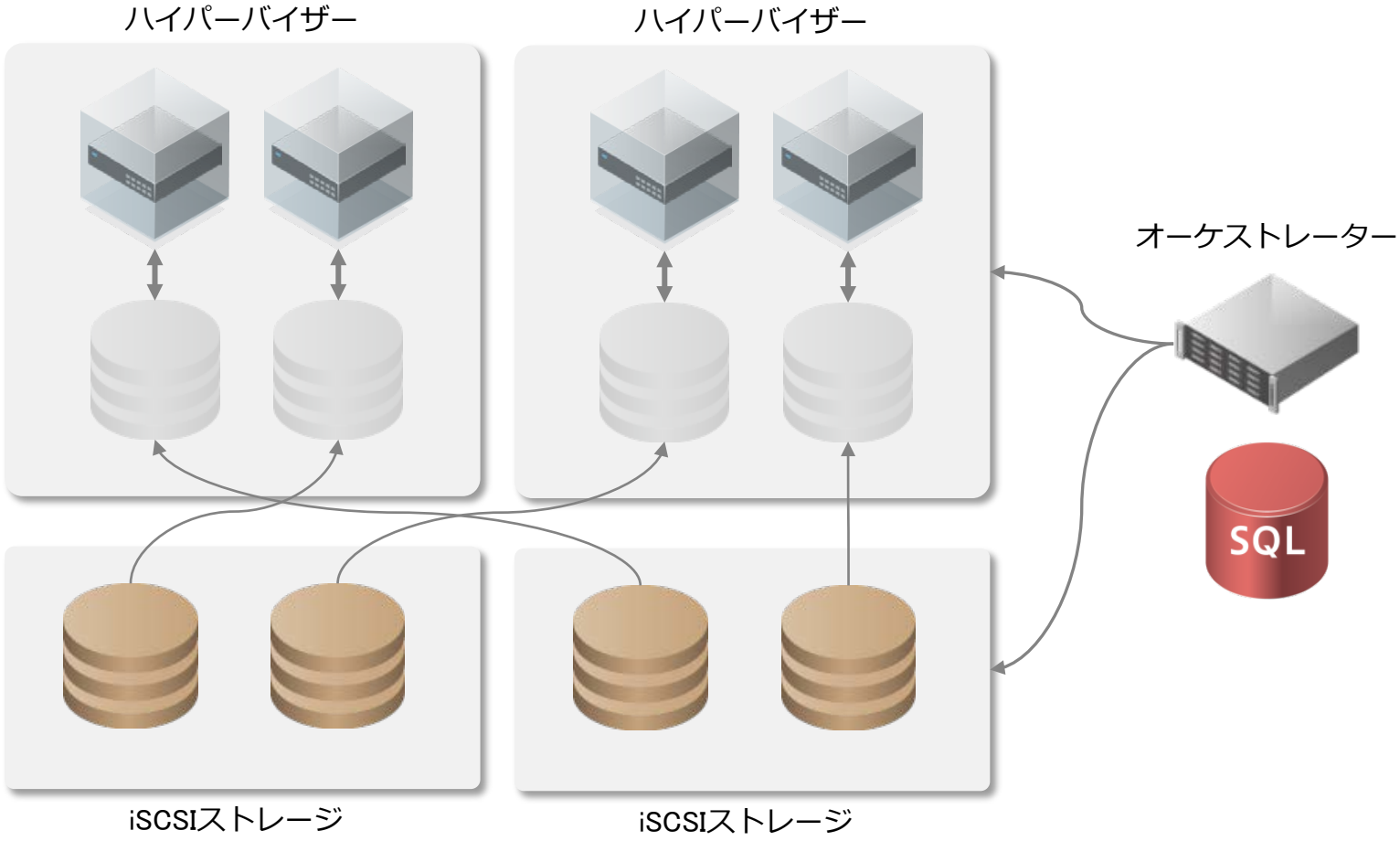
追加ストレージ

仮想サーバで利用可能な
追加用のストレージを提供

ベストエフォートタイプ
100GB～1TB
価格とパフォーマンスのバランスが優れている汎用的なI/O性能ストレージ

性能保証タイプ
100GB～1TB
容量に応じてIOPS上昇
DBなど安定的なI/O性能が必要なサーバに利用

パブリックリソースのストレージ構成



機能強化の概要

強化される機能

- **データの暗号化**
  - 透過的な暗号化機能の提供
- **オンラインでのストレージ接続・切断**
  - 稼働中仮想サーバへのストレージ接続・切断機能の提供
- **インスタントクローニング**
  - リードタイム極小のストレージ複製機能の提供
- **オンラインバックアップ**
  - 稼働中仮想サーバのストレージバックアップ機能の提供

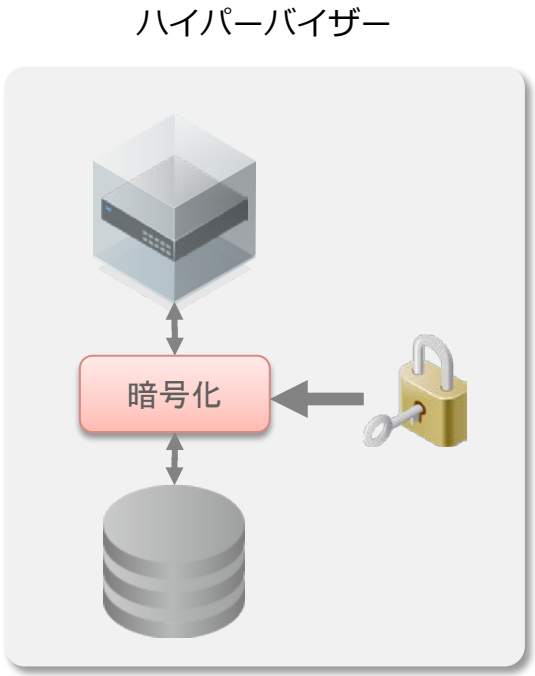
開発の前提条件

- **ストレージ製品に依存しない実装**
  - ストレージにかかるコストの圧縮を図る
  - より効率的なリソース活用を図る
  - 製品変更による追加開発のリスクを最小化する
- **ホストの機能による実装**
  - ホストサーバ側に備わっている機能を活用した実装を図る

機能強化の概要

データの暗号化

- **透過的なデータの暗号化**
  - セキュリティおよびコンプライアンス要件を満たすことができる
  - ハイパーバイザーにおいて仮想サーバとストレージ間に暗号化レイヤーを挟むことによりデータを透過的に暗号化・復号化する



ニーズに合わせたストレージの利用

- **ストレージの品目**
  - システムストレージ
    - Linux系は30GB
    - Windows系は60GB
  - 追加ストレージ
    - ベストエフォート
    - 性能保証
    - 100GB,300GB,500GB,1000GB
- **接続可能数**
  - 仮想サーバあたり最大８台の追加ストレージが追加可能（性能保証は2台まで）

ストレージ運用上の課題

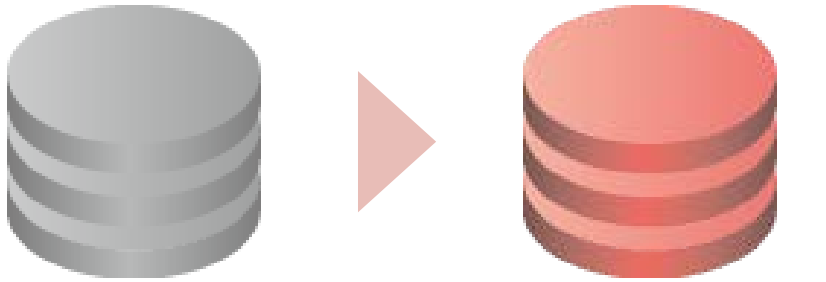
- **仮想サーバの停止を伴う**
  - ストレージの接続・切断時には仮想サーバを停止する必要がある
  - ミッションクリティカルなシステムでは必要に応じたタイムリーな拡張が困難
- **稼働中の仮想サーバへの接続・切断を可能に**
  - 必要なタイミングでストレージを増設可能になる
  - 予めLVM上にファイルシステムを構成しておけば運用中に容量が不足しても仮想サーバを止めることなく動的に容量の増加可能になる

ストレージ複製の現状と課題

- **２段階の操作が必要**
  - ストレージの複製という機能そのものはない
  - ストレージアーカイブの機能を使い一旦イメージを作成した上でリストアする必要がある
  - ストレージ製品の機能が使えなかったためDRDBによる実装など検討されたが構成が複雑になり採用されなかった
- **処理に時間がかかる**
  - ディスクイメージの作成、イメージからのリストアはいずれもデータコピーを伴う
  - ストレージの容量、実使用量に応じて処理時間がかかる
  - システムの停止、スタートアップ時間が長くなる

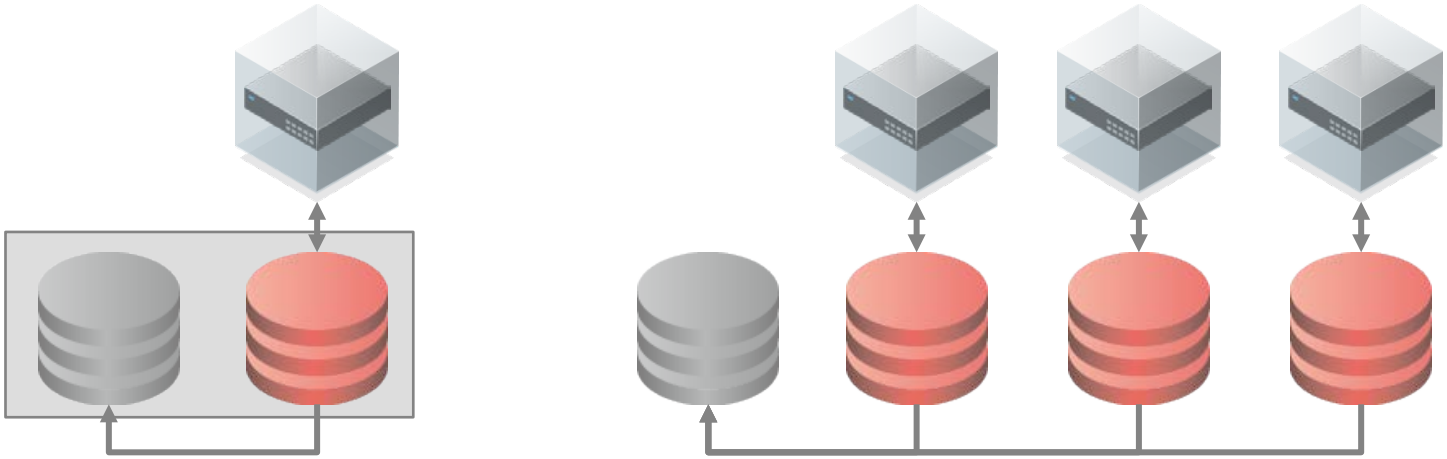
スナップショットを活用したストレージの複製（1）

- **複製元のスナップショットを作成**
  - Linuxカーネルに実装されたスナップショットの機能を使用し複製元のスナップショットを作成
  - スナップショットの作成はデータコピーを伴わないので瞬間的に完了する



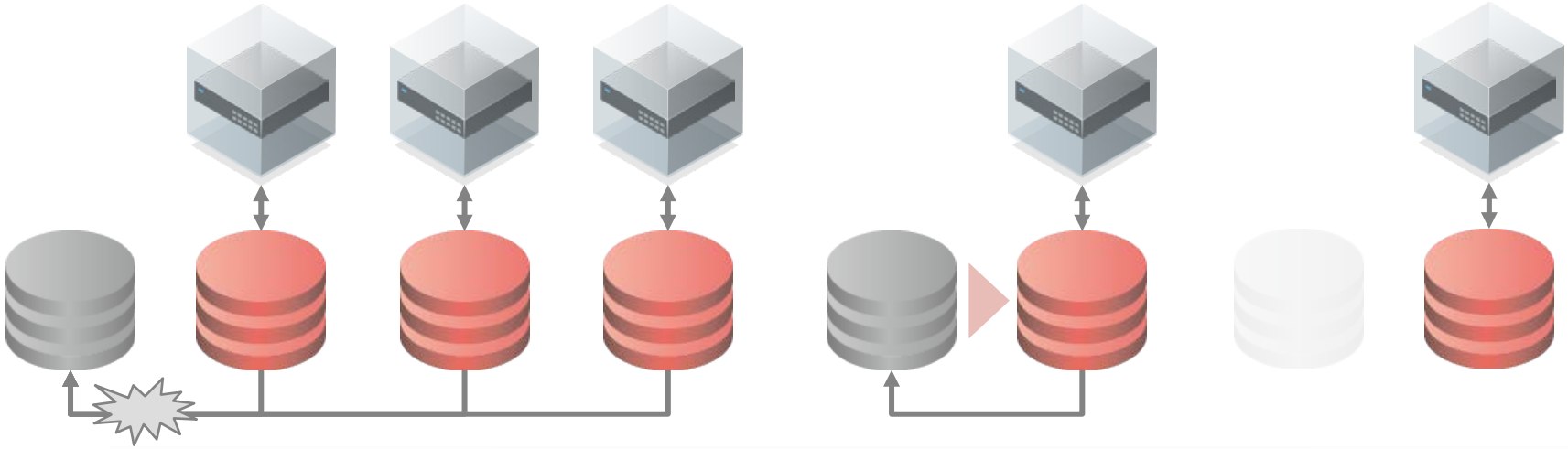
スナップショットを活用したストレージの複製（3）

- **スナップショットを仮想サーバに接続**
  - 書き込みはスナップショットに対して行われるため複製元は変更されない
  - スナップショットにまだ書き込まれていないデータは複製元より読み込まれる
  - 単一の複製元から複数のスナップショットを作成できるので同時に複数の複製を作成することが可能



スナップショット方式の課題と取り組み

- **読み込み負荷の集中**
  - 複数の複製を作成した場合に複製元に対する読み込み負荷が集中する可能性がある
- **バックグラウンドでデータをマージ**
  - バックグラウンドで未更新の領域を複製元よりスナップショットにコピーする
  - スナップショットのすべての領域が更新されたら複製元を切り離して単体で動作するようにする



バックアップ機能の現状

- **ストレージのディスクイメージを生成**
  - ストレージアーカイブの機能によりストレージの状態をディスクイメージとして保存可能
  - 保存されたイメージより同品目のストレージへのリストアが可能
  - 静止点を作成できないためディスクイメージの生成が完了するまで仮想サーバを停止する必要がある
  - 容量が応じて処理にかかる時間が長くなるため対象はシステムストレージのみ

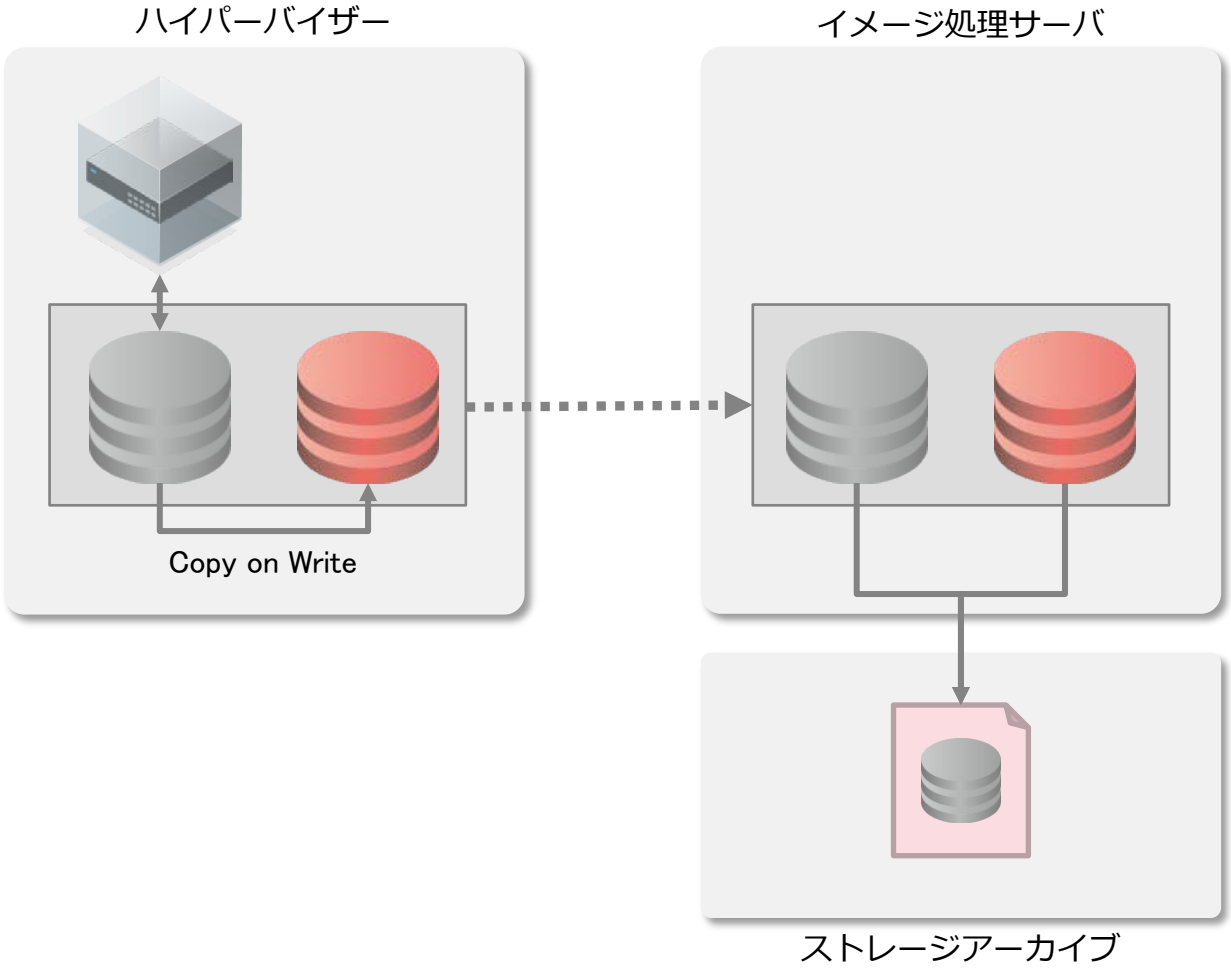
止めないバックアップの実装（1）

- **スナップショットによる静止点の作成**
  - Linuxカーネルに実装されたスナップショットの機能を使用しバックアップ元のスナップショットを作成
  - Copy on Writeにより更新前のデータがスナップショットに保存される
  - バックアップ元のデータとスナップショット上のデータを組み合わせることでスナップショット作成時点のストレージの状態を保持できる
  - 仮想サーバを止めることなくバックアップを作成することができる

止めないバックアップの実装（2）

- **バックアップ処理の流れ**
  - スナップショット用の物理ボリュームを確保する
  - Linuxカーネルの機能により確保した物理ボリュームをバックアップ元のスナップショットとしてセットアップする
  - バックアップ元物理ボリュームとスナップショット用物理ボリュームを組み合わせ論理ボリュームを構成する
  - 論理ボリュームをイメージ処理サーバにも接続する
  - イメージ処理サーバで論理ボリュームのイメージを作成しストレージアーカイブに保存する
  - 論理ボリュームからスナップショット用物理ボリュームを切り離し解放する

止めないバックアップの実装（3）



止めないバックアップの実装（4）

- **イメージ処理サーバの効果**
  - ハイパーバイザーでバックアップ処理を実行すると色々と不都合が生じる
  - バックアップ処理中に仮想サーバを止める場合はバックアップを中止する必要がある
  - 仮想サーバのライブマイグレーション時にバックアップ処理も移行する必要があり制御が煩雑になる
  - 複数のバックアップ処理が単一のハイパーバイザーで実行されるとハイパーバイザーのリソースが不足する恐れがある

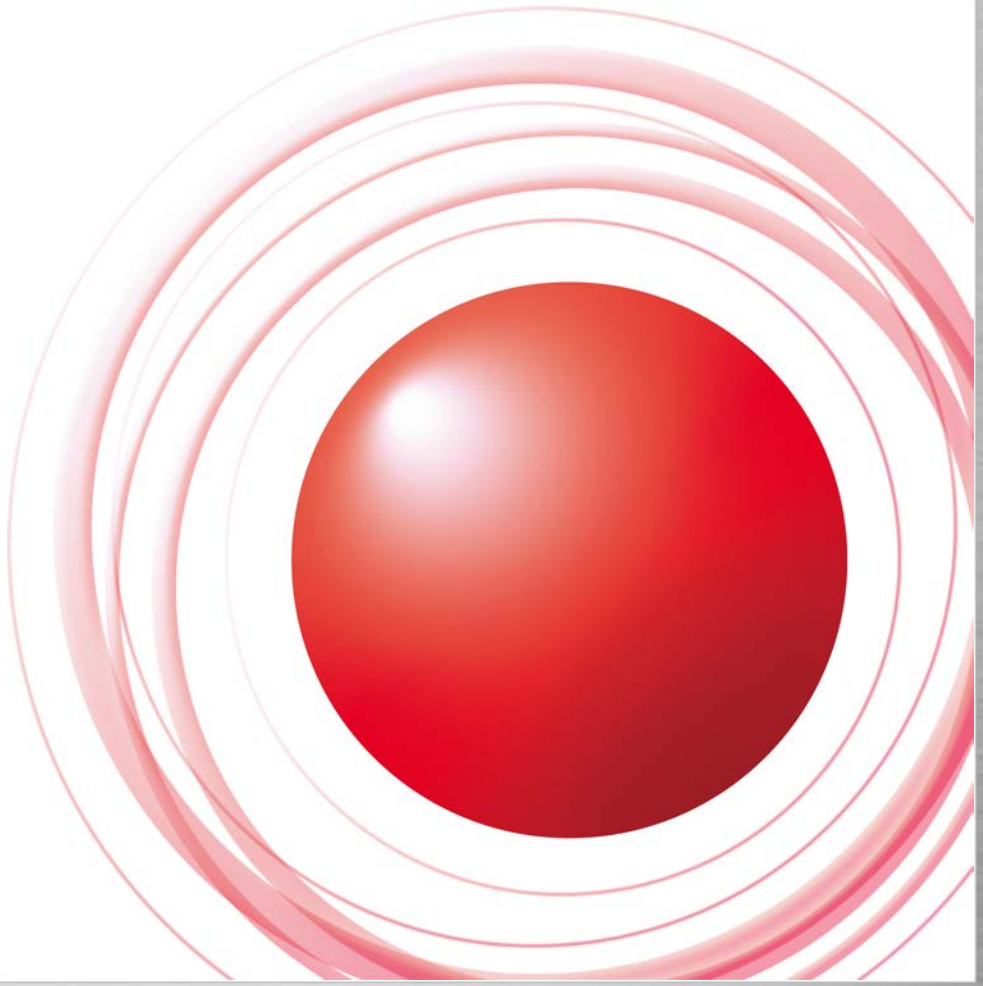
カーネルモジュール

カーネルモジュールの開発

- **LVMによる実装の試み**
  - 複数ホストから同時にアクセスされることを想定していない
  - イメージ処理サーバによるバックアップが実現できない
  - 仮想サーバのライブマイグレーションができない
- **QCOWによる実装の試み**
  - オンラインバックアップを実現することができない
- **カーネルモジュールの開発**
  - LVM複数ホストから同時にアクセスできない問題を解決できる
  - インスタントクローニングにおける複製元データのマージ機能を追加できる

IIJ GIOインフラストラクチャーP2
ストレージ機能を実現するカーネルモジュール実装

IIJ
Internet Initiative Japan



株式会社インターネットイニシアティブ

Ongoing Innovation

ここからの対象機能

ここからの対象機能

- オンラインバックアップ
- インスタントクローニング

技術的な検討の経緯

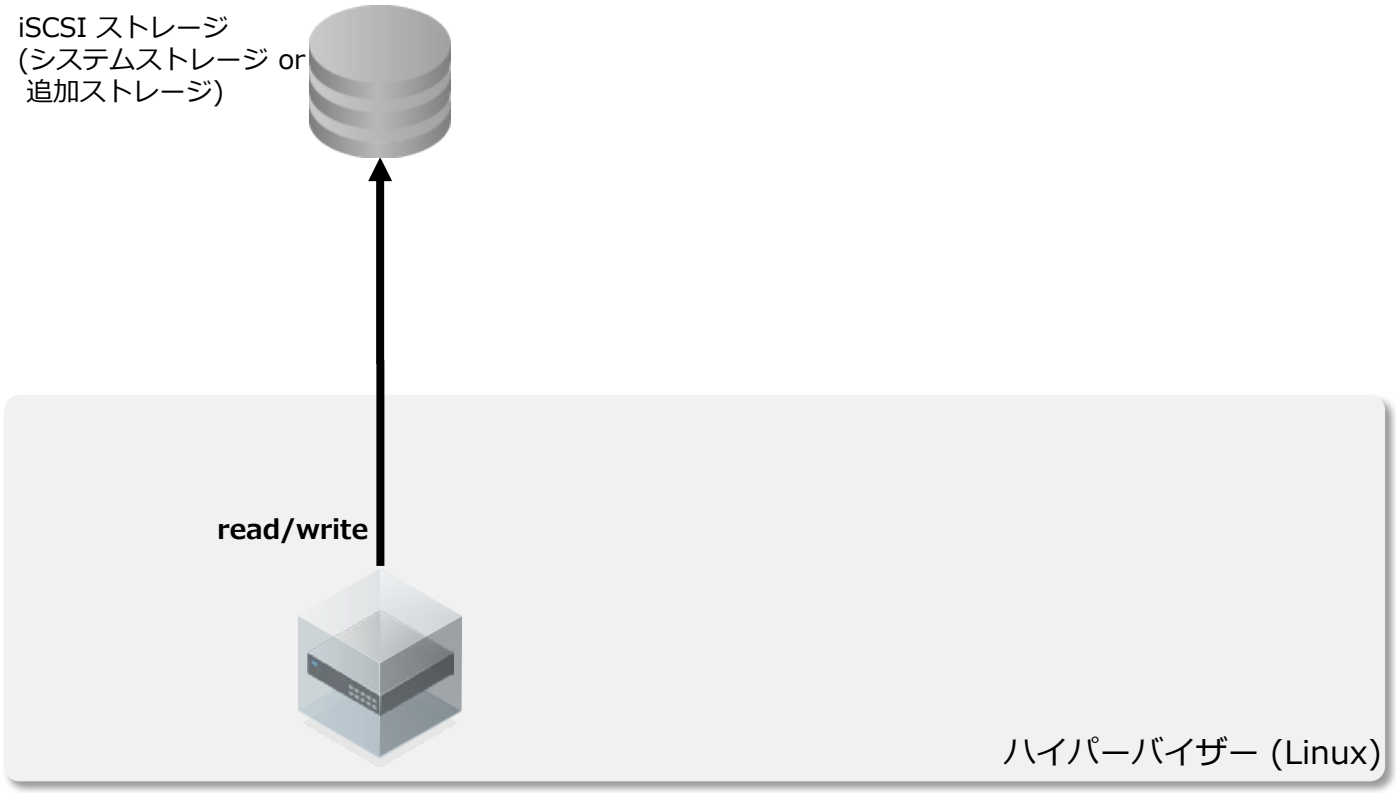
当初検討していた実現方法

- 当初、オンラインバックアップ機能・インスタントクローニング機能は、LVM のスナップショット機能を用いて実現することを検討していた。
- しかし、LVM スナップショット方式には問題があることが分かり、別方式を検討せざるを得なくなった (後述) 。

技術的な検討の経緯 2/11

LVM によるオンラインバックアップの実現 1/4

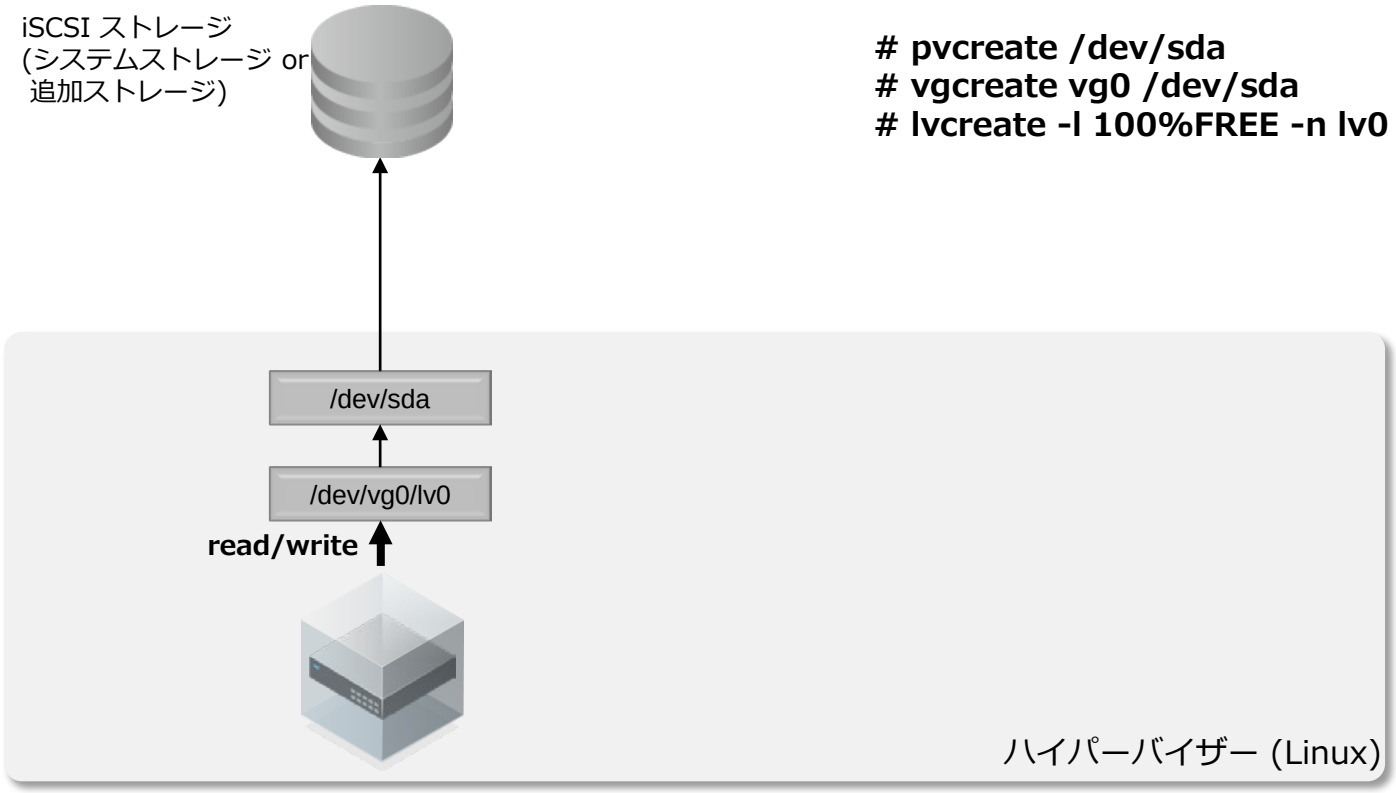
- **現状のストレージ接続方式**



技術的な検討の経緯 3/11

LVM によるオンラインバックアップの実現 2/4

- **オンラインバックアップ機能実現のために、iSCSI ストレージを LVM 化**



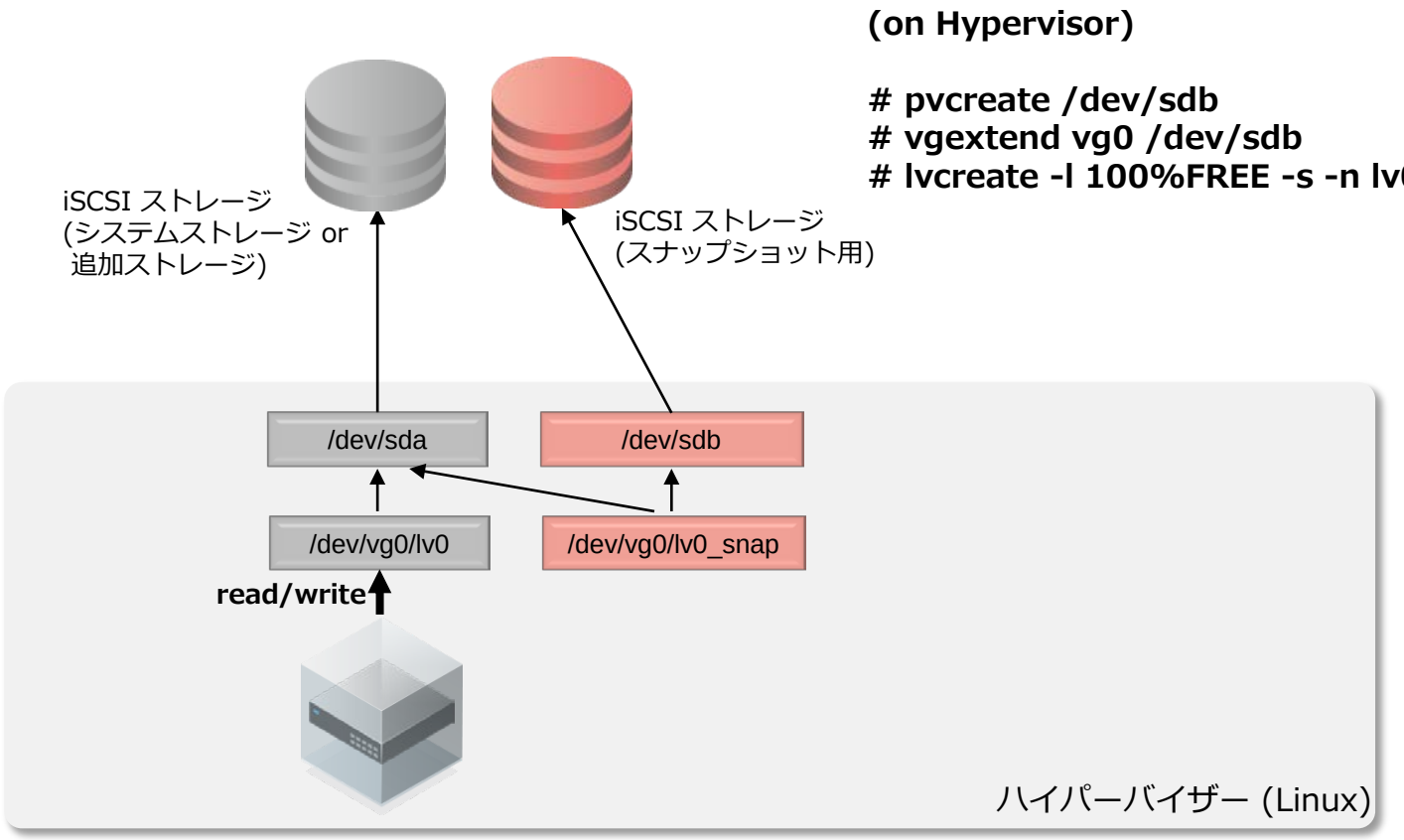
(on Hypervisor)

```
# pvcreate /dev/sda
# vgcreate vg0 /dev/sda
# lvcreate -l 100%FREE -n lv0 vg0
```

技術的な検討の経緯 4/11

LVM によるオンラインバックアップの実現 3/4

- **オンラインバックアップ開始時、LVM スナップショットを作成し、オンラインバックアップのための静止点を作成**



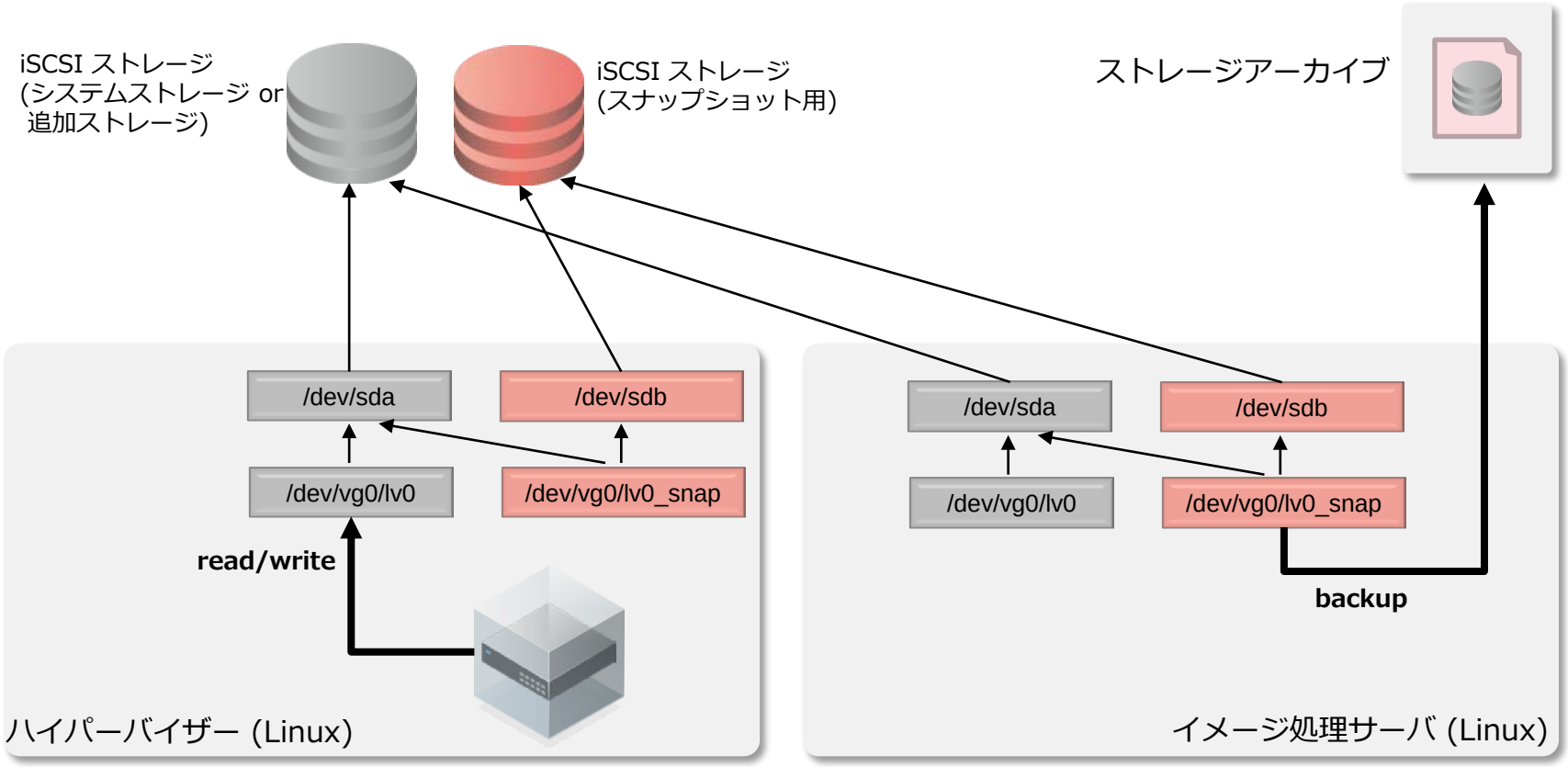
(on Hypervisor)

```
# pvcreate /dev/sdb
# vgextend vg0 /dev/sdb
# lvcreate -l 100%FREE -s -n lv0_snap vg0/lv0
```

技術的な検討の経緯 5/11

LVM によるオンラインバックアップの実現 4/4

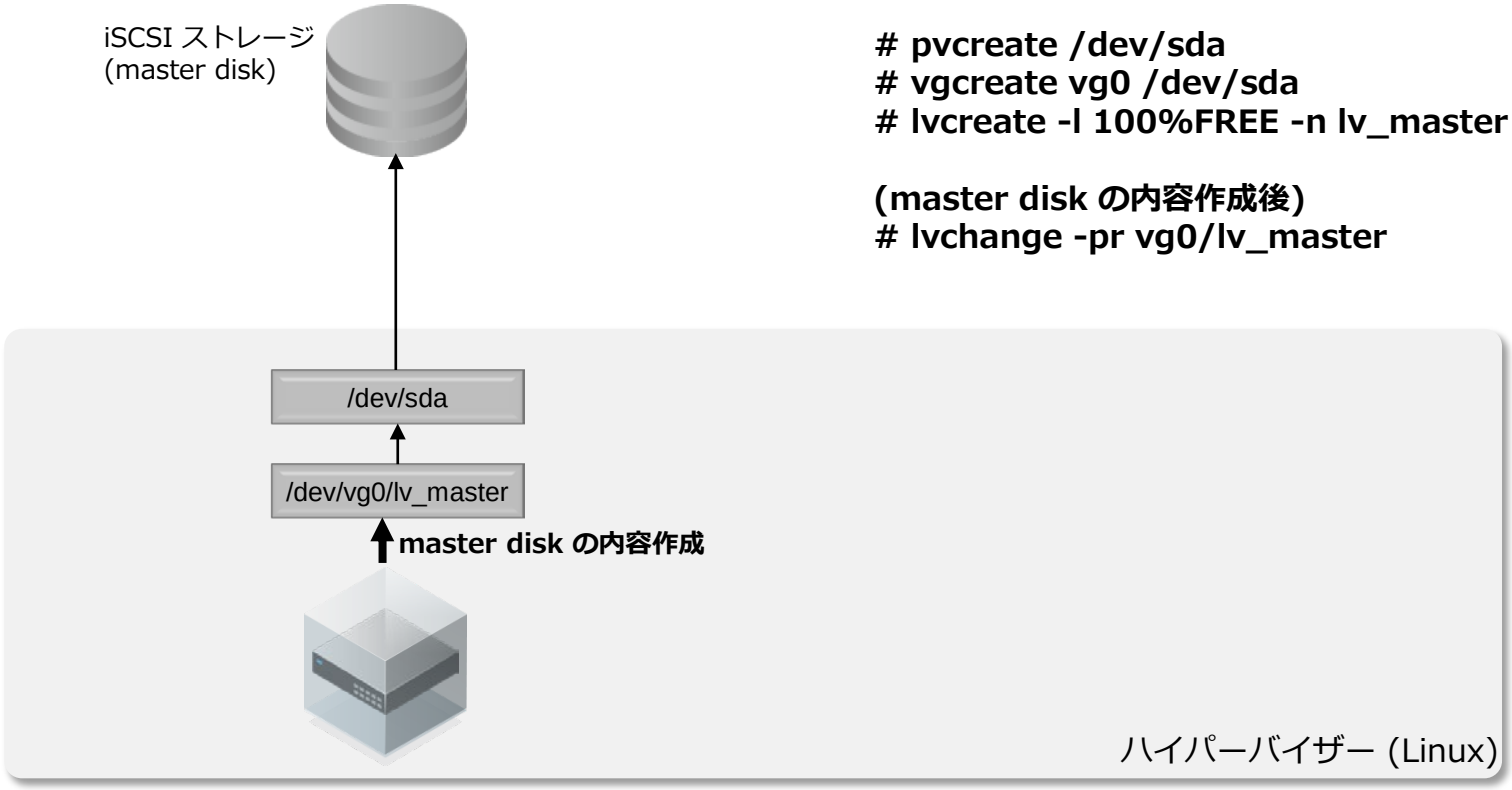
- **作成したスナップショット（静止点）からストレージアーカイブへのコピーをイメージ処理サーバ上で行い、バックアップを実行**



技術的な検討の経緯 6/11

LVM によるインスタントクローニングの実現 1/2

- ひな形として使用する master disk (/dev/vg0/lv_master) を、LVM の論理ボリュームとして用意する
  - ひな形の内容は、お客様の仮想サーバから作成可能
  - master disk は read only とする



(on Hypervisor)

```
# pvcreate /dev/sda
# vgcreate vg0 /dev/sda
# lvcreate -l 100%FREE -n lv_master vg0

(master disk の内容作成後)
# lvchange -pr vg0/lv_master
```

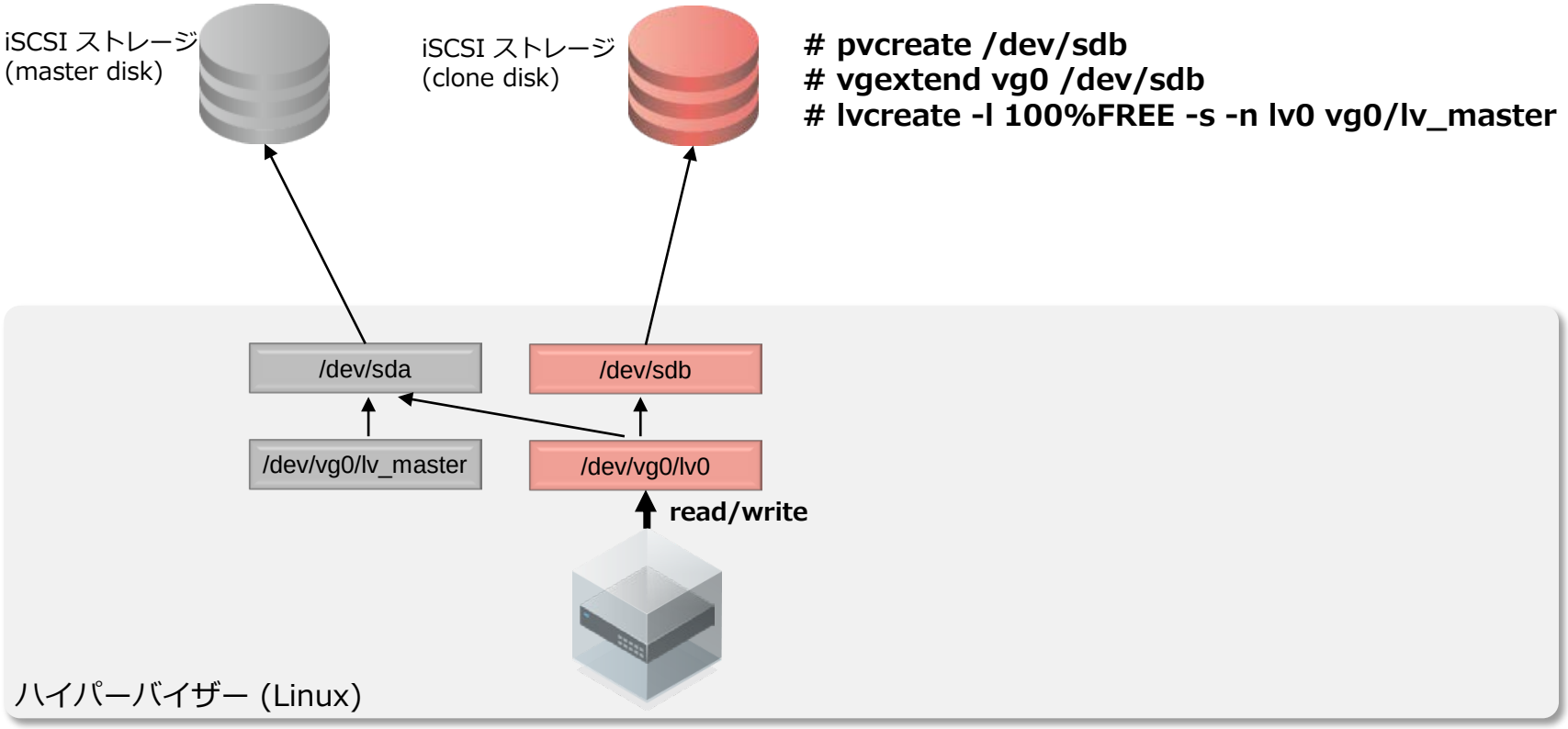
技術的な検討の経緯 7/11

LVM によるインスタントクローニングの実現 2/2

- **master disk の内容をクローンするストレージを短時間で作成**
- **そのクローンしたストレージを用いて、仮想サーバを起動**

(on Hypervisor)

```
# pvcreate /dev/sdb
# vgextend vg0 /dev/sdb
# lvcreate -l 100%FREE -s -n lv0 vg0/lv_master
```



技術的な検討の経緯 8/11

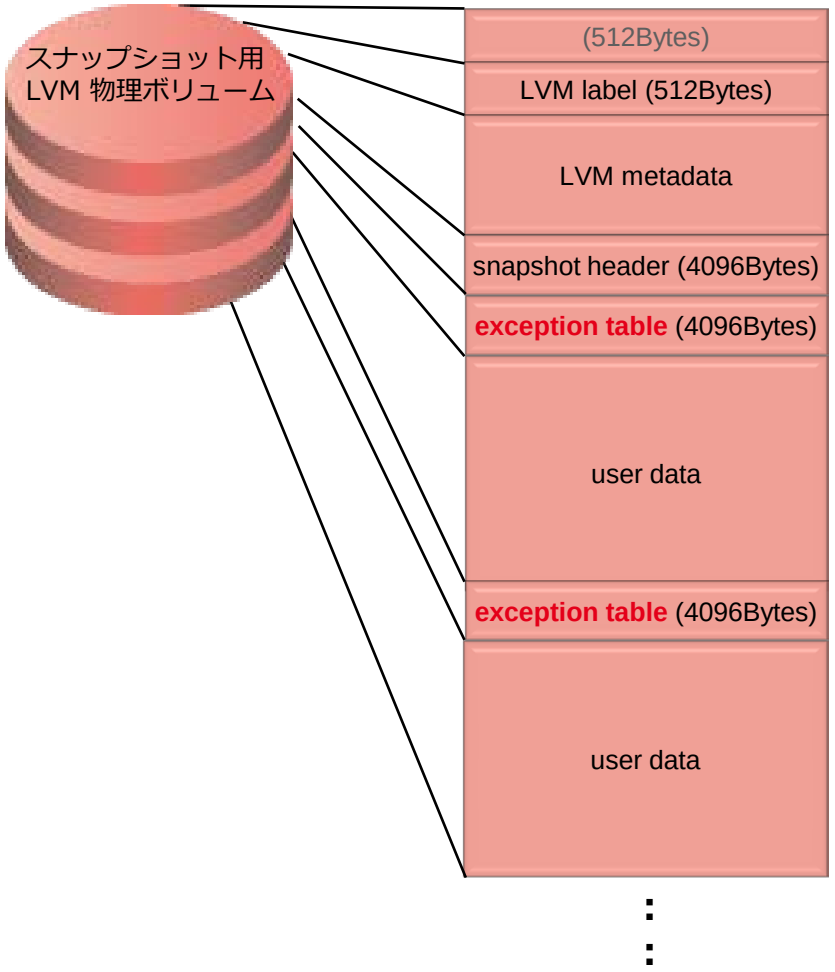
LVM スナップショット使用時の問題点

- オンラインバックアップを行っても、静止点でのディスクイメージが正しく取得できない場合がある
- インスタントクローニングで作成したストレージを参照している仮想サーバをライブマイグレーションすると、クローンで作成したストレージの内容が正しく認識されなくなる可能性がある
- なぜか？

技術的な検討の経緯 9/11

LVM スナップショット用ストレージのフォーマット

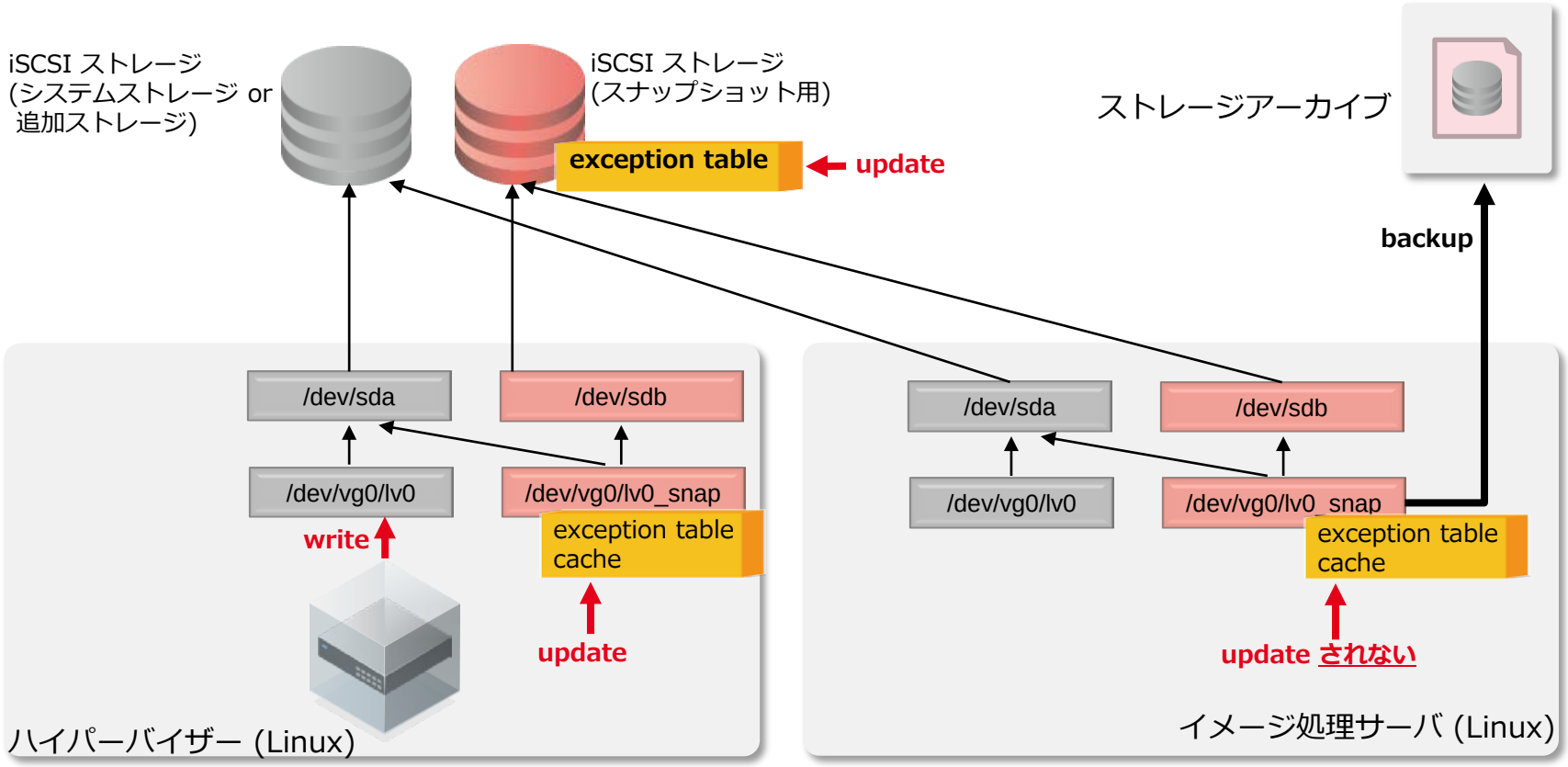
- LVM label
  - 物理デバイスの ID 等を保持
- LVM metadata
  - ボリュームグループの詳細情報を保持
- snapshot header
  - スナップショット領域用ヘッダ
- **exception table**
  - 差分データを保持する user data 領域のディスク上の位置情報を保持
  - **Linux kernel のメモリ空間上に cache される**
- user data
  - 差分データを保持するためのデータ領域



技術的な検討の経緯 10/11

exception table cache 問題 (オンラインバックアップ)

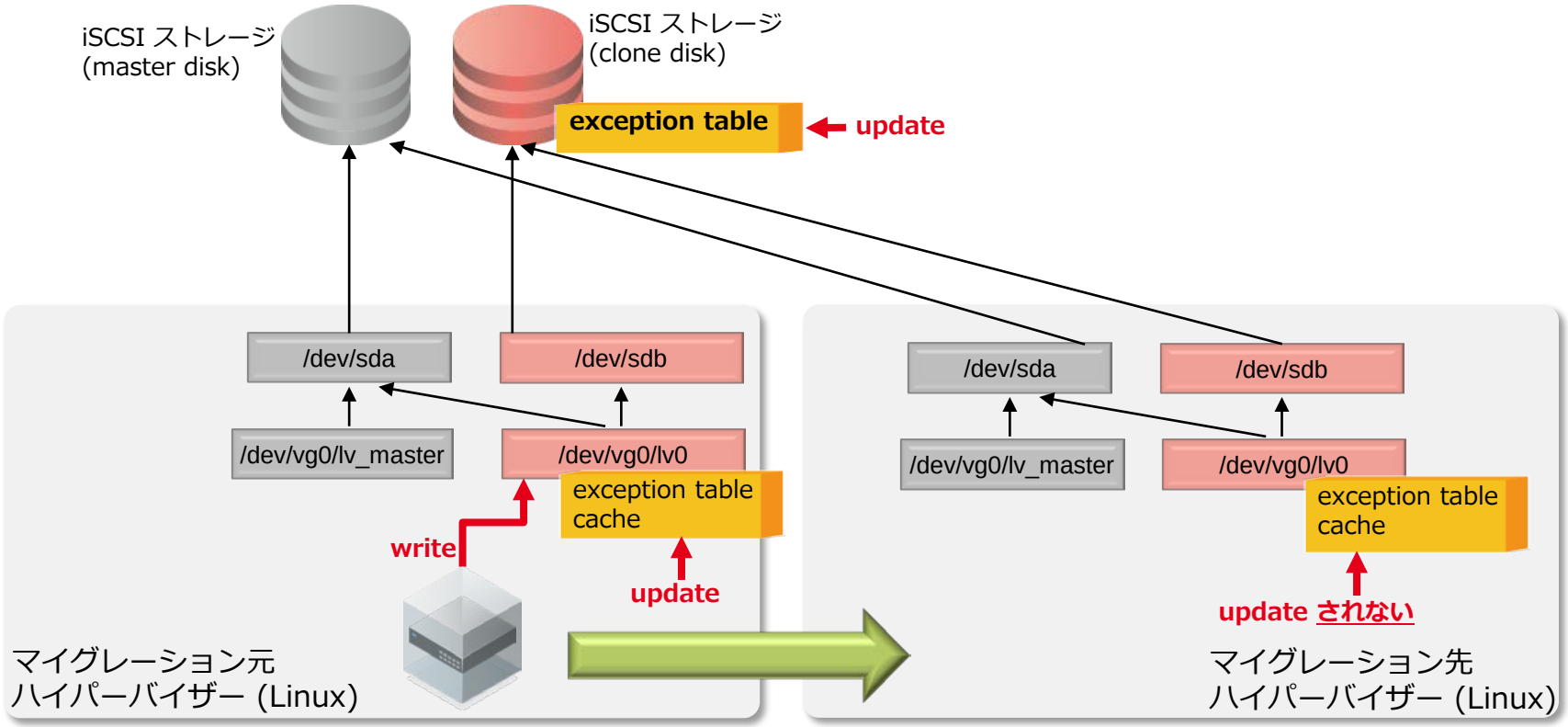
- ハイパーバイザー, イメージ処理サーバの両方に exception table cache が構築される
- 仮想サーバから write が発生すると、exception table の内容が更新される (場合がある)
- イメージ処理サーバ側は write があったことを検知できない
- そのため、イメージ処理サーバ側の exception table cache が更新されず、正しくバックアップを行えなくなる



技術的な検討の経緯 11/11

exception table cache 問題 (インスタントクローニング)

- 仮想サーバをライブマイグレーションする場合を考える
- マイグレーション先のハイパーバイザーでは、予め仮想サーバが使用するストレージの用意 (LVM 論理ボリュームも) が必要
- マイグレーション先のストレージ用意後、マイグレーション元で write 発生するとマイグレーション先の exception table cache が更新されないまま残ってしまい、マイグレーション後の仮想サーバからのストレージ操作が正しく行われなくなる



対策は？

検討した対策

- **インスタントクローニングで作成したストレージを参照している仮想サーバをライブマイグレーションする際には、仮想サーバをサスペンドしてしまう**
  - ライブマイグレーションができなくなり、サービス品質が低下
  - オンラインバックアップの問題を別途解決する必要がある
- **LVM スナップショットではなく、qcow2 の backing file 機能を使用**
  - L2 table のディスクへの write が遅延されるため、ハイパーバイザー突然死の場合等データ損失の恐れあり
  - オンラインバックアップを実現することができない
- **スナップショット機能を提供するカーネルモジュールを新規に実装し問題を回避**
  - 前述の問題を回避できそう

新 device mapper 用カーネルモジュールの開発

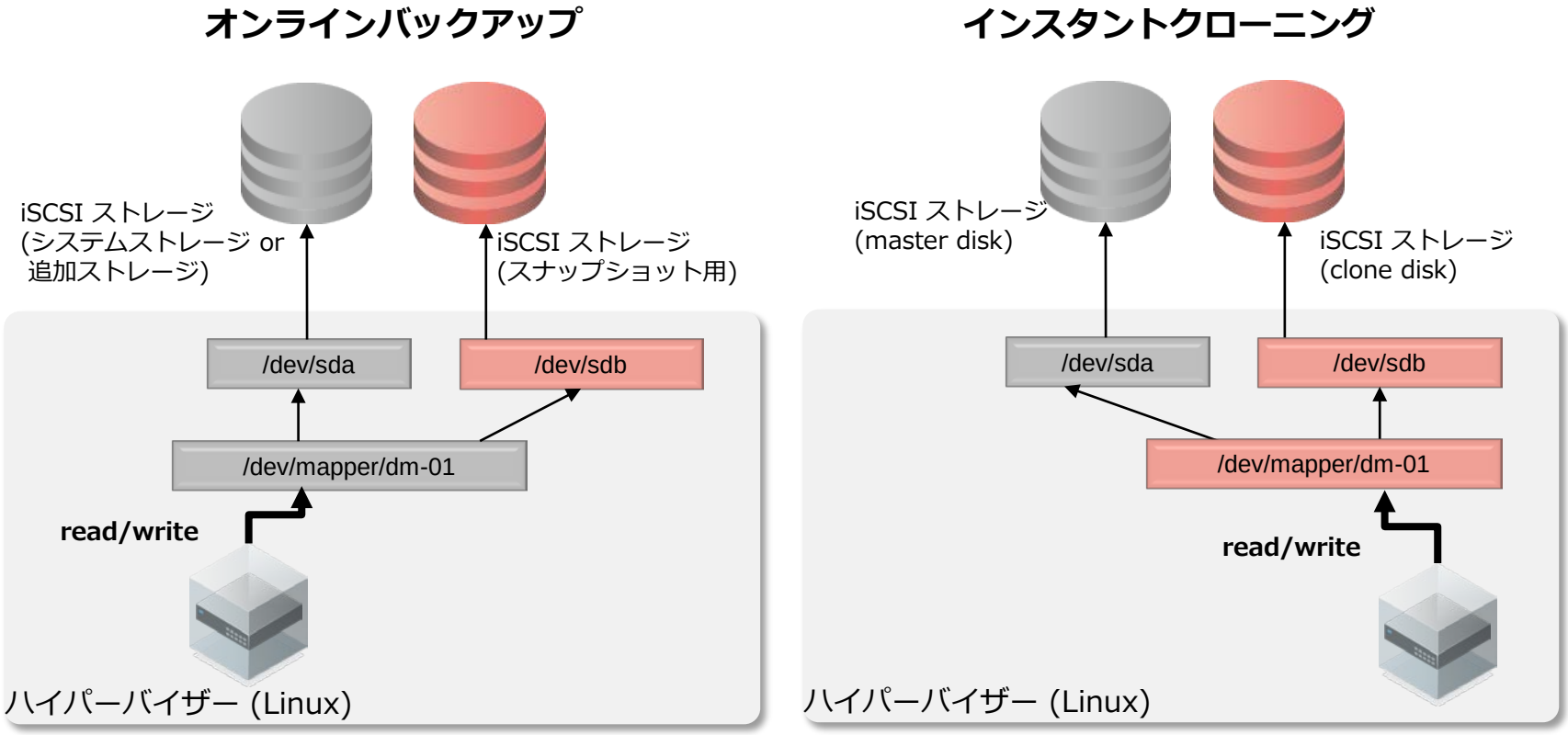
新 device mapper 用カーネルモジュールの開発 1/10

開発した device mapper

- LVM のスナップショットのような機能を提供する device mapper
- LVM との違いは、メモリ上の exception table cache の on/off 切り替えを行えるようになっていること (LVM の場合常に on)

新 device mapper 用カーネルモジュールの開発 2/10

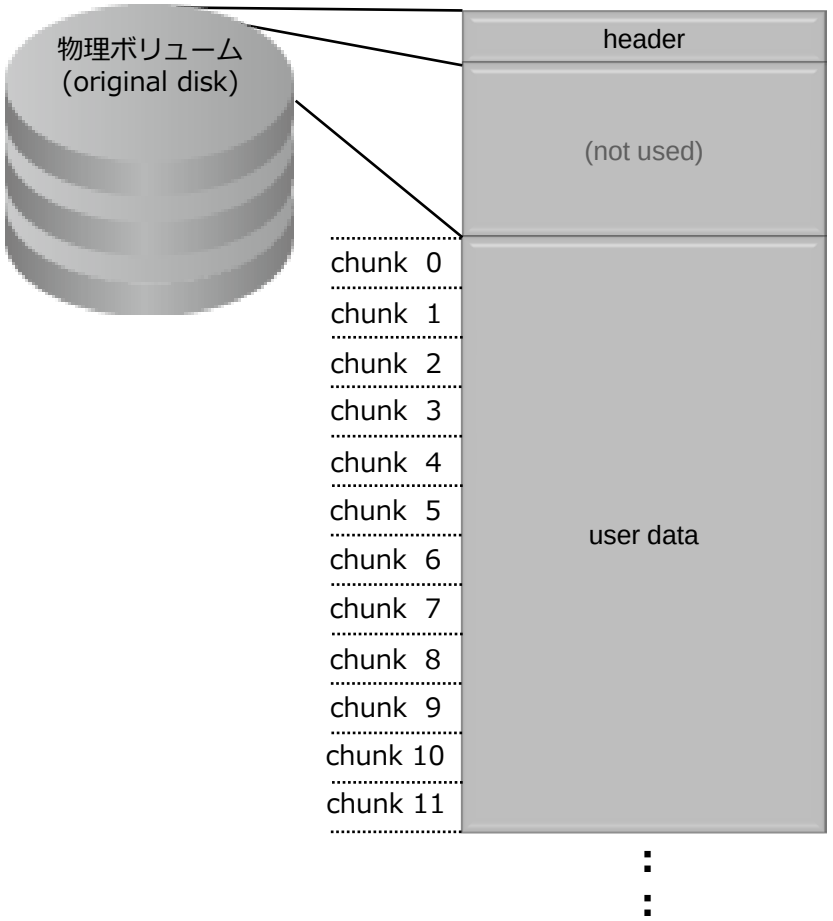
構成イメージ



新 device mapper 用カーネルモジュールの開発 3/10

物理ディスクフォーマット(original disk 側)

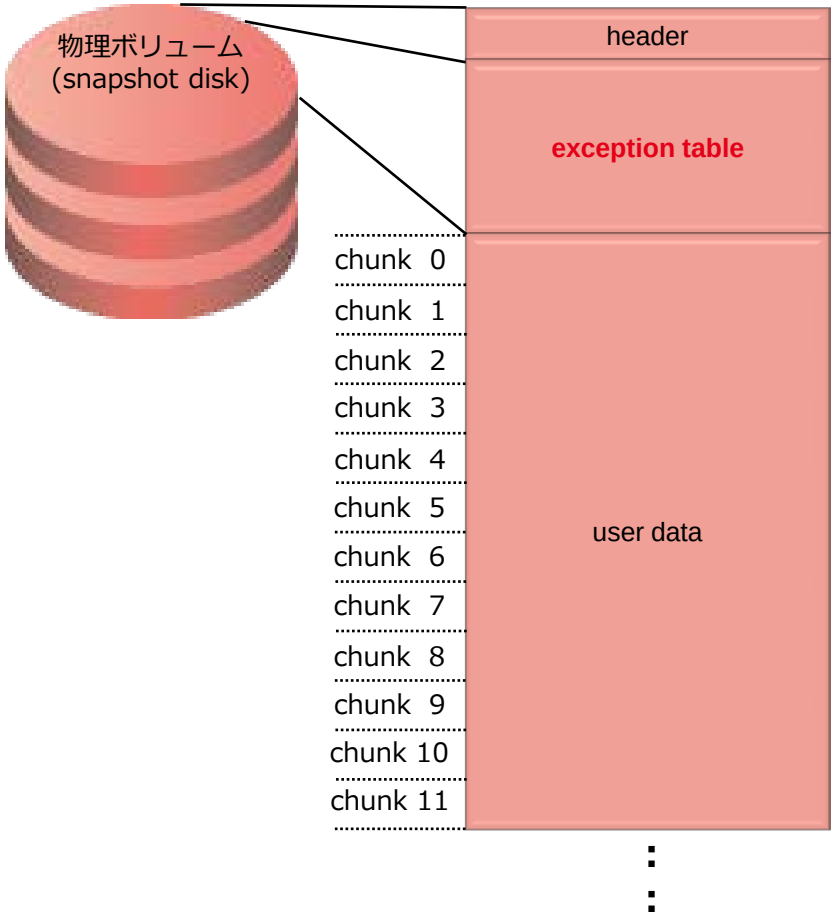
- header
  - 新 device mapper で使用される、device ID 等を保持
- user data
  - ユーザが read/write するデータそのものを保存
  - chunk という固定長の領域毎に管理



新 device mapper 用カーネルモジュールの開発 4/10

物理ディスクフォーマット (snapshot disk 側)

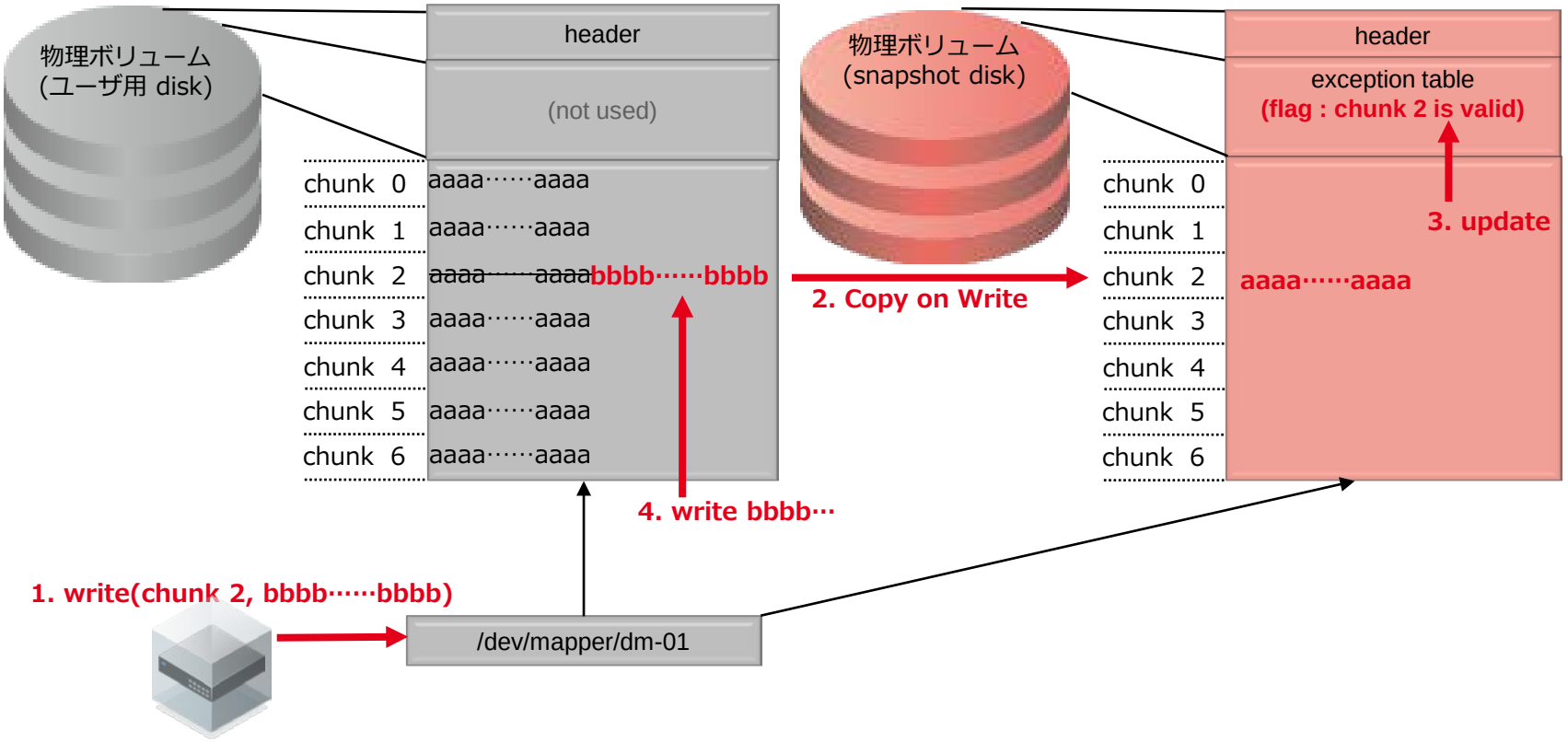
- header
  - 新 device mapper で使用される、device ID 等を保持
- **exception table**
  - user data 領域の各 chunk に、差分データが保存されているかどうかの flag を保持
  - **Linux kernel メモリ空間上の cache を on/off できる**
- user data
  - 差分データを保持
  - chunk という固定長の領域毎に管理



新 device mapper 用カーネルモジュールの開発 5/10

disk write 時の動作 (オンラインバックアップ)

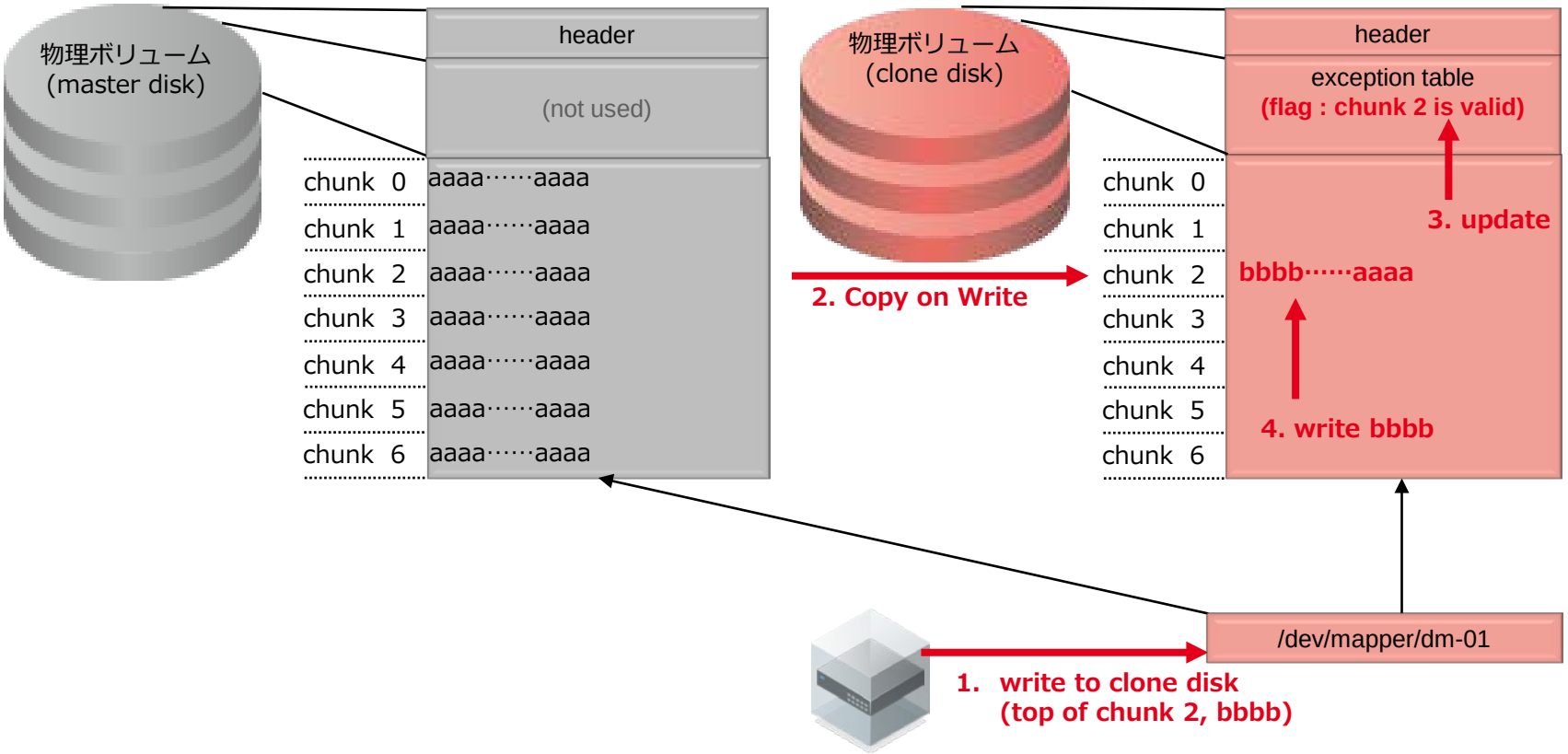
- スナップショット作成前は、ユーザ用 disk 側の user data 領域は “a” で埋め尽くされていた
- お客様の仮想サーバから、chunk 2 の領域に対して “bbbb……bbbb” を write
- 下図のように write 前の data が Copy on Write されるため、スナップショットをとった時点でのデータが保持され、オンラインバックアップが可能になる



新 device mapper 用カーネルモジュールの開発 6/10

disk write 時の動作 (インスタントクローニング)

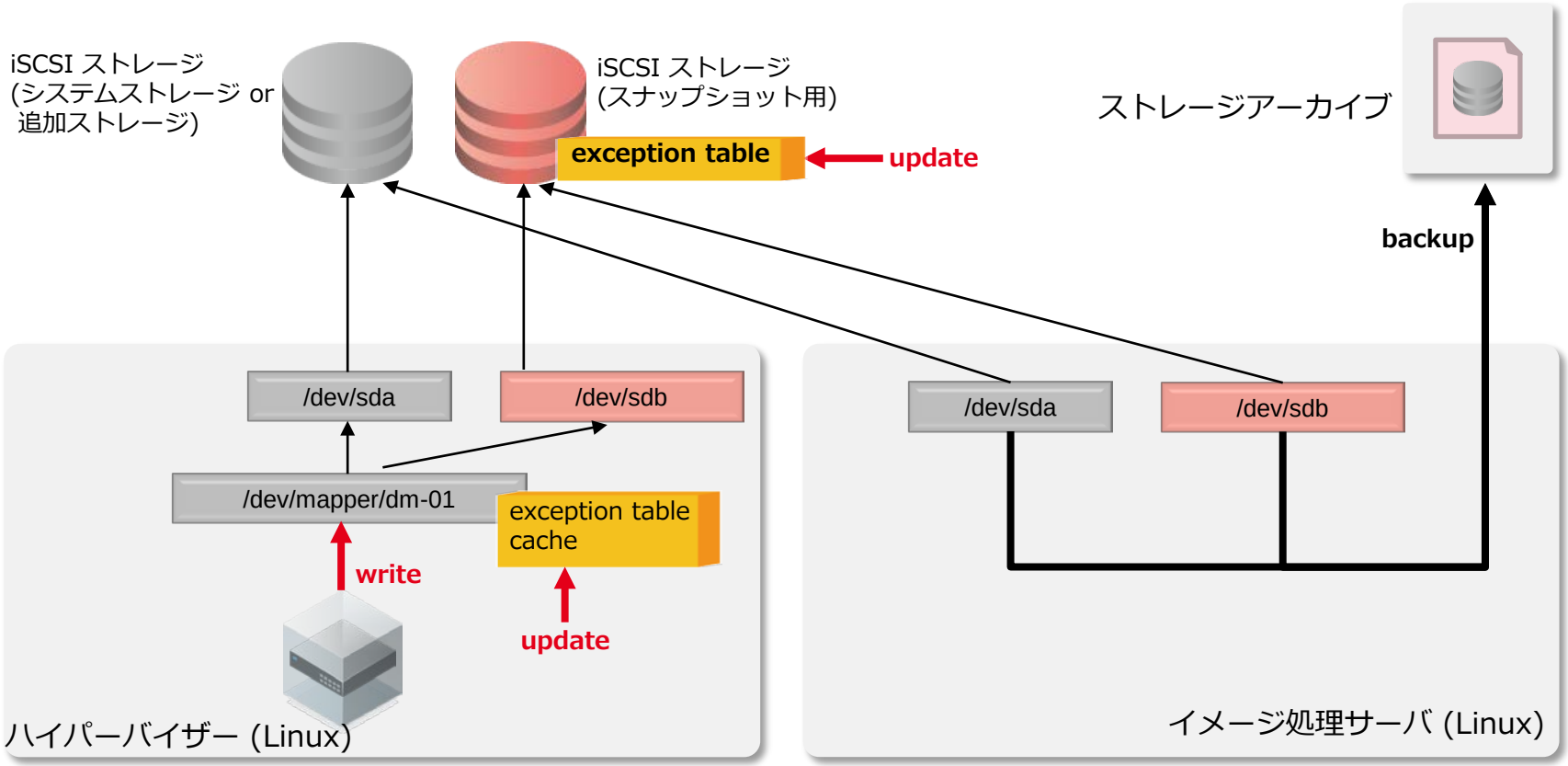
- user data 領域が ”a” で埋め尽くされている master disk から、clone disk を作成
- お客様の仮想サーバから、clone disk の chunk 2 の一部領域に “bbbb” を write
- 物理 disk 上は、write したデータは clone disk 上に差分データとして保存されていく



新 device mapper 用カーネルモジュールの開発 7/10

オンラインバックアップ

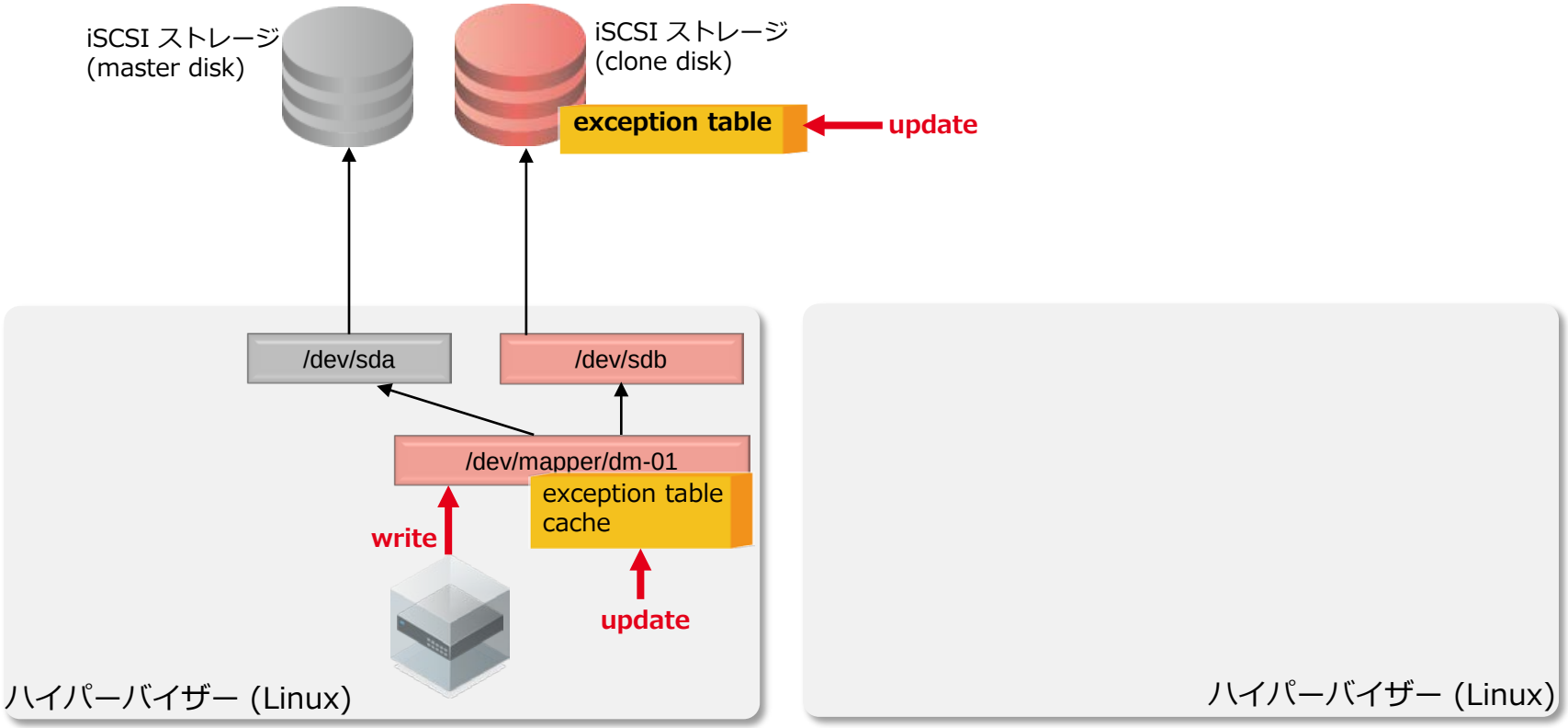
- ハイパーバイザー側に、オンラインバックアップ用に構築した新 device mapper の device を dmsetup create
- dmsetup create した device mapper を仮想サーバに attach して利用
- イメージ処理サーバ側では、物理ストレージ上の exception table と user data を直接 scan してバックアップを取得
- イメージ処理サーバ側に exception table cache が存在しないので、exception table の食い違いが発生しない



新 device mapper 用カーネルモジュールの開発 8/10

インスタントクローニング

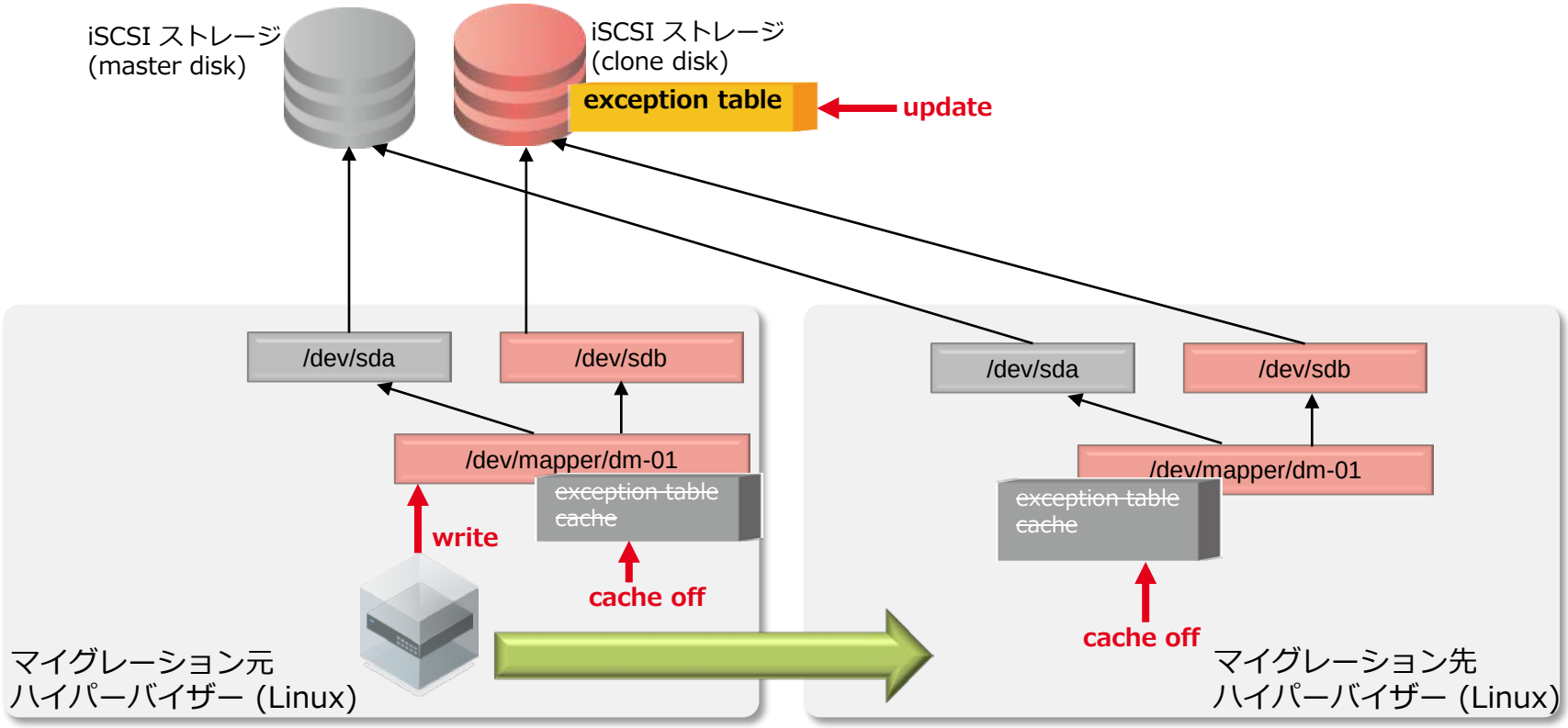
- ハイパーバイザー上で、インスタントクローニング用に構築した新 device mapper の device を dmsetup create
- dmsetup create した device mapper を仮想サーバに attach して利用



新 device mapper 用カーネルモジュールの開発 9/10

インスタントクローニング利用時のライブマイグレーション

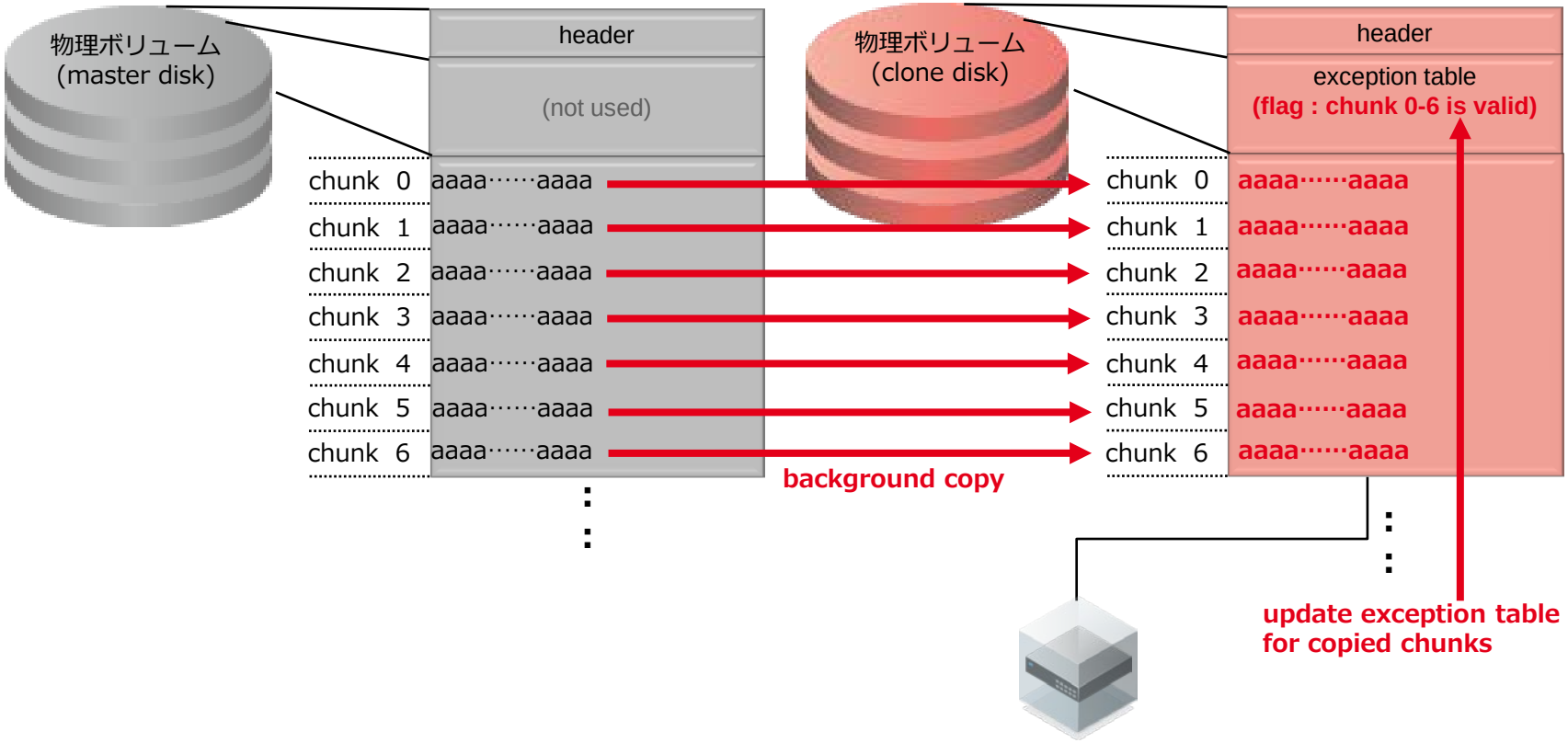
- ライブマイグレーション前に、マイグレーション元ハイパーバイザーで exception table cache を off にする
- マイグレーション先ハイパーバイザーでも、ストレージに iSCSI 接続後 exception table cache なしで dmsetup create
- マイグレーション中 exception table cache が存在しないので、write しても exception table の食い違いが発生しない
- マイグレーションが完了したら、ライブマイグレーション先のハイパーバイザーで exception table cache を on にする



新 device mapper 用カーネルモジュールの開発 10/10

clone disk へのバックグラウンドコピー

- インスタントクローニングで作成した clone disk へは、disk write がない未更新の領域でも、master disk からのバックグラウンドコピーが逐次的に行われる
- user data 領域全体のコピーが完了した段階で、clone disk を master disk から切り離す
- 切り離しを行うことで、read が master disk に集中しないようにできるため、disk 性能が向上する



まとめ

まとめ

- オンラインバックアップ・インスタントクローニングの実装に LVM のスナップショット機能を利用すると、不具合が発生する場合がある
- その不具合は、差分データを管理するための exception table と呼ばれる領域の cache が原因
- exception table の cache を on/off できるスナップショット機能を新規に device mapper として実装し、問題の不具合発生を防ぐことに成功した

IIJ Internet Initiative Japan

ご清聴ありがとうございました

お問い合わせ先　IIJインフォメーションセンター
TEL：03-5205-4466 （9：30～17：30 土/日/祝日除く）
info@iij.ad.jp

Ongoing Innovation