

winDRBD 構築手順書

サイオステクノロジー株式会社

バージョン 0.9.1 2019/01/16

目次

1. 変更履歴	1
1.1. v0.9.1 2019/01/16	1
1.2. v0.9.0 2019/01/14	1
1.3. v0.1.2 2019/01/08	1
1.4. v0.1.0 2018/10/01	1
2. はじめに	2
2.1. 制限事項	2
3. インストール	3
3.1. ファイル構成	4
4. DRBD 設定	5
4.1. Kernel log ファイル	5
4.2. DRBD 構成ファイル	5
5. 構築例 (新規ボリューム)	6
5.1. Windows 側の設定	6
5.2. Linux 側の設定	8
6. 構築例 (既存のボリューム)	9
6.1. Windows 側の設定	9
6.2. Linux 側の設定	11
7. ソースファイルからビルド	12
7.1. cygwinをインストール	12
7.2. wdk をインストール	12
7.3. Linux 上の作業	12
7.4. Windows上の作業	13
7.4.1. drbd-utils	13
7.4.2. windrbd	13

1. 変更履歴

1.1. v0.9.1 2019/01/16

- 0.9.1 対応

1.2. v0.9.0 2019/01/14

- Beta5 対応

1.3. v0.1.2 2019/01/08

- サイオステクノロジーに変更

1.4. v0.1.0 2018/10/01

- 最初のバージョン

2. はじめに

winDRBD は Windows 版の DRBD であり、

<https://github.com/LINBIT/windrbd>

でOpenSourceとして開発が進んでいます。このたび、LinbitからBeta版としてバイナリパッケージがダウンロードできるようになりました。弊社ではwinDRBDを円滑に構築するための手順書を作成しましたので公開します。

なお、ここで公開する内容はBeta5時点のものであり将来のリリースで変更される可能性があることに注意してください。

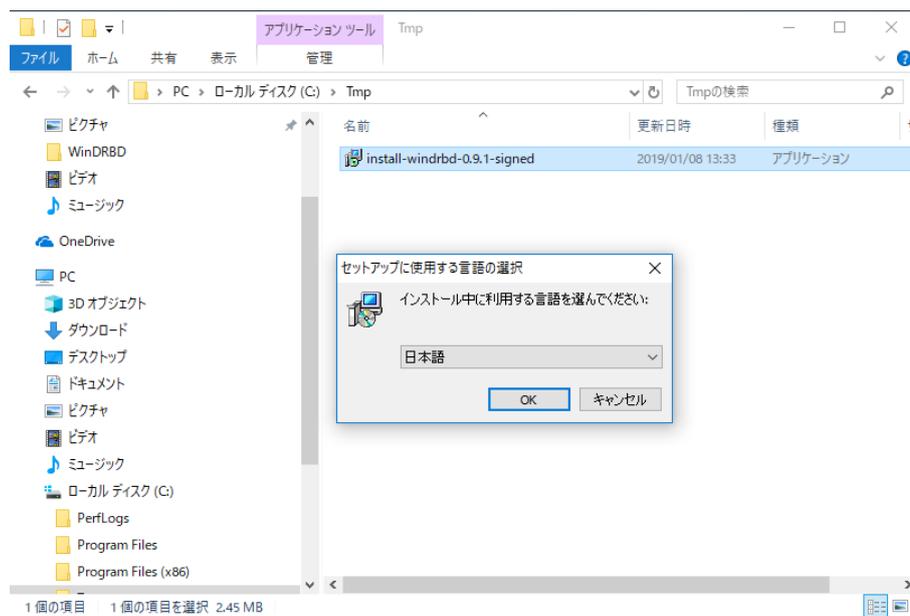
2.1. 制限事項

今のところいくつか制限があります。主要なものをリリースノートから抜粋します。

- 現在はNTFSのみサポートである。
- 3ノード以上は動作はするがあまりテストされてない。
- 自動プロモーションはサポートされない。
- C:ドライブはDRBDデバイスに使用できない。
- Windows Server 2016 (64 bit), Windows 10 (64 bit), Windows 7 (service pack 1) (64 bit) がサポート対象である。

3. インストール

インストーラをクリックすると、管理者権限で起動します。インストーラのみ日本語対応です。



インストール後、画面の指示に従います。アップグレード以外はリブート不要です。

インストーラは

<https://www.linbit.com/en/drbd-community/drbd-download/>

からダウンロードできます。

ソースファイルからビルドして使用する場合は、「ソースファイルからビルド」の項を参照ください。

3.1. ファイル構成

以下に主要なファイルのパスを示します。

- winDRBD driver

```
C:\Windows\System32\drivers\windrbd.sys
```

- winDRBD 固有の設定ツール

```
C:\windrbd\usr\sbin\windrbd.exe
```

- ユーザランドのバイナリ(cygwin で動作)

```
C:\windrbd\usr\sbin\drbdadm.exe  
C:\windrbd\usr\sbin\drbdmeta.exe  
C:\windrbd\usr\sbin\drbdsetup.exe
```

- 構成ファイル

```
C:\windrbd\etc\drbd.conf  
C:\windrbd\etc\drbd.d\global_common.conf  
C:\windrbd\etc\drbd.d\windrbd-sample.res
```

- その他ファイル

```
C:\Program Files\WinDRBD\cygwin1.dll  
:
```

4. DRBD 設定

では実際に winDRBD を構成してみます。

4.1. Kernel log ファイル

以前のバージョンでは windrbd.exe のサブコマンド log-server を使用してログを確認していましたが、現バージョンでは C:\windrbd\windrbd-kernel.log にログが出力していますので、これを確認します。

4.2. DRBD 構成ファイル

DRBD構成ファイルはC:\windrbd\etc\drbd.d*.resを編集します。disk, device keyにはWindows固有のドライブ指定を使います。この例では下位デバイスがGUID 53ffdbac-0000-0000-0000-100000000000, DRBD デバイスがH:になり、ユーザはH:ドライブを介してDRBDデバイスを使用します。minor番号はノードで一意なものを指定します。GUIDはmountvolコマンドで確認することができます。

```
disk      "53ffdbac-0000-0000-0000-100000000000";  
device    "H:" minor 1;
```

diskにはドライブ文字"E:"等を指定することも可能ですが、推奨は下位デバイスにはドライブ文字を割り当てず、GUIDで指定するのがよいようです。このようにすることでエクスプローラからも見えなくなり誤って使われるのを防ぐことができます。

5. 構築例 (新規ボリューム)

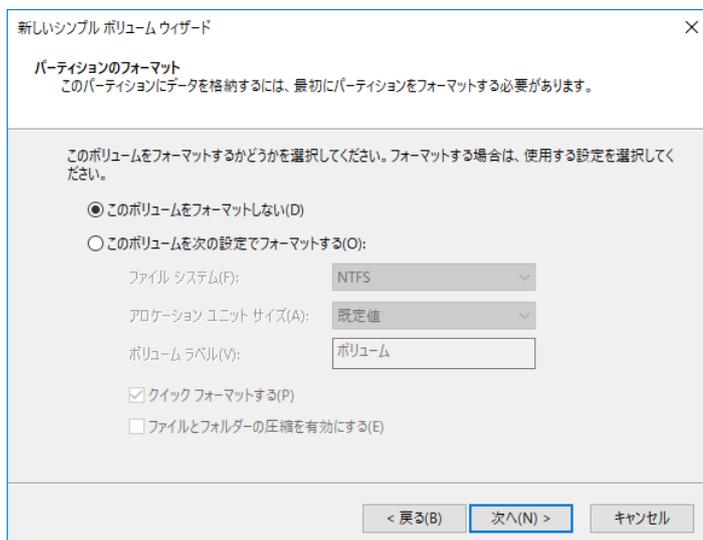
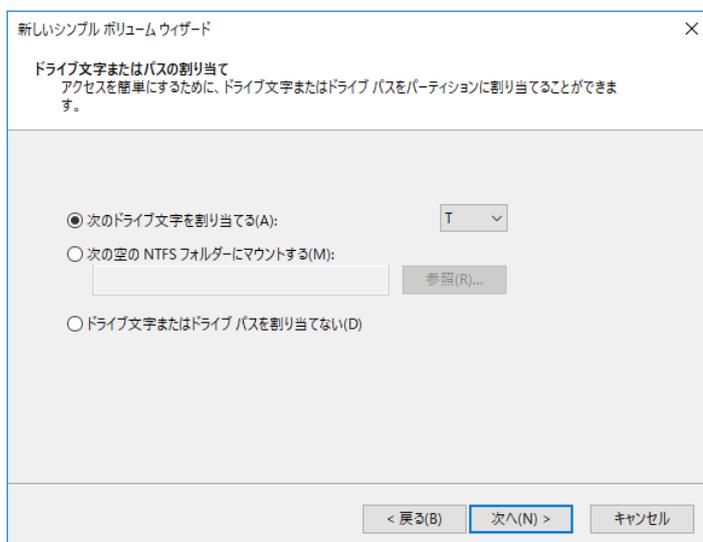
ここでは、Windows 10とCentOS7.5を用いた2node構成とし、Windows上のボリュームをLinux側に複製します。Windows上のボリュームは新規に作成します。既存のボリュームをDRBDで使用する場合は、次節の「構築例 (既存のボリューム)」を参照ください。

	Windows	Linux
ホスト	tate-z2, 10.30.103.2	tate-z4, 10.30.103.4
ボリューム	2048M, meta-disk internal	2048M, meta-disk internal

リソース名はw0とする。

5.1. Windows 側の設定

- firewallを適切に設定するか(port 7600)、一時的に無効にします。
- 管理ツールを用いて、下位デバイス用の新規ボリュームの準備します。この時フォーマットはしません。GUIDを使用する場合(推奨)、ドライブ文字割り当ては不要なのですが、mountvolコマンドで該当GUIDを調べる場合、ここで割り当てておいたほうがわかりやすいので適当にT:とかを割り当てておきます。



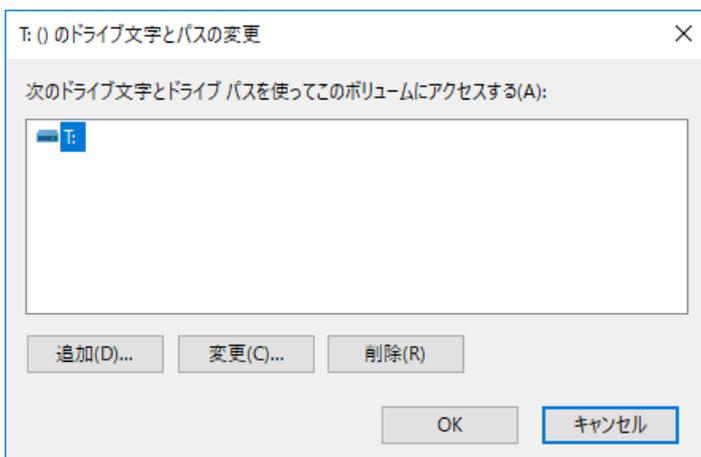
- GUIDを使用する場合、mountvolコマンドで下位デバイス用のGUIDを調べます。

```
\\?\Volume{53ffdbac-0000-0000-0000-100000000000}\  
T:\
```

- コマンドプロンプトでC:\windrbd\etc\w0.resを作成します。サンプルwindrbd-sample.resがインストールされるのでそれをベースに環境に合わせて変更します。

```
include "global_common.conf";  
  
resource "w0" {  
    protocol C;  
  
    net {  
        use-rle no;  
    }  
  
    on tate-z4 {  
        address 10.30.103.4:7600;  
        node-id 1;  
        volume 1 {  
            disk /dev/sdb1;  
            device /dev/drbd1;  
            flexible-meta-disk internal;  
        }  
    }  
    on tate-z2 {  
        address 10.30.103.2:7600;  
        node-id 2;  
        volume 1 {  
            disk "53ffdbac-0000-0000-0000-100000000000";  
            device "H:" minor 1;  
            meta-disk internal;  
        }  
    }  
}
```

- GUIDを使用した場合で、ドライブ文字を割り当てた場合はもう不要ですので、ディスク管理の「ドライブ文字とパスの変更」で削除しておきます。



- あとはLinux版同様に構成します。

```
$ drbdadm.exe create-md w0
$ drbdadm.exe up w0
$ drbdadm.exe --force primary w0
```

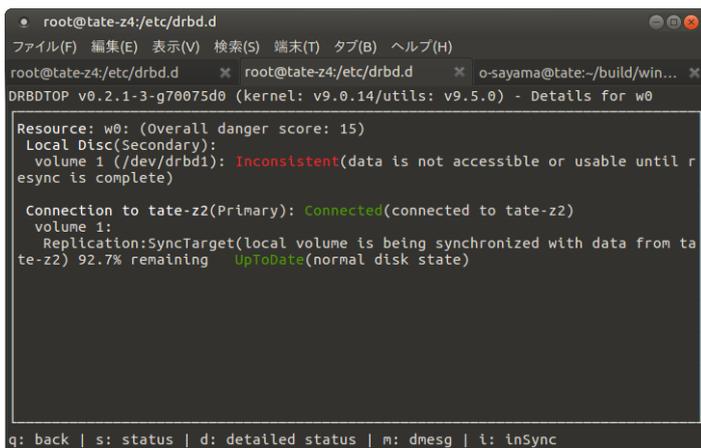
- ここでドライブH:にアクセスできるようになりますので、NTFSでフォーマットすれば使用できるようになります。

5.2. Linux 側の設定

- firewall(port 7600),selinuxを適切に設定するか、一時的に無効にします。
- DRBD9をインストールし、/dev/sdb1にWindows側と同じサイズ2048Mのパーティションを準備します。
- 端末から/etc/drbd.d/w0.resを構成します。基本的にWindowsのものといっしょです。

```
# drbdam create-md w0
# drbdam up w0
```

ここで同期が始まりますので、別端末でdrbdtop等で監視します。



```
root@tate-z4:/etc/drbd.d
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) タブ(B) ヘルプ(H)
root@tate-z4:/etc/drbd.d * root@tate-z4:/etc/drbd.d * o-sayama@tate:~/build/win... *
DRBDTOP v0.2.1-3-g70075d0 (kernel: v9.0.14/utils: v9.5.0) - Details for w0

Resource: w0: (Overall danger score: 15)
Local Disc(Secondary):
  volume 1 (/dev/drbd1): Inconsistent(data is not accessible or usable until r
esync is complete)

Connection to tate-z2(Primary): Connected(connected to tate-z2)
volume 1:
  Replication:SyncTarget(local volume is being synchronized with data from ta
te-z2) 92.7% remaining UpToDate(normal disk state)

q: back | s: status | d: detailed status | m: dmesg | i: inSync
```

6. 構築例 (既存のボリューム)

ここでは、Windowsの既存のボリュームをDRBDで使用する構築例を示します。E:ドライブには既存のNTFSボリュームがあり、外部のメタデバイスを使用してDRBDドライブH:を構成します。

	Windows	Linux
ホスト	tate-z2, 10.30.103.2	tate-z4, 10.30.103.4
ボリューム	E: ドライブ 2048M, meta-disk external 10M	2048M, meta-disk internal

リソース名はw0とする。

6.1. Windows 側の設定

- firewallを適切に設定するか(port 7600)、一時的に無効にします。
- 管理ツールを用いて、下位デバイス用のmeta-diskボリュームを準備します。この時フォーマットはしません。GUIDを使用する場合(推奨)、ドライブ文字割り当ては不要なのですが、mountvolコマンドで該当GUIDを調べる場合、ここで割り当てておいたほうがわかりやすいので適当にT:とかを割り当てておきます。

新しいシンプル ボリューム ウィザード

ドライブ文字またはパスの割り当て
アクセスを簡単にするために、ドライブ文字またはドライブ パスをパーティションに割り当てることができます。

次のドライブ文字を割り当てる(A): T

次の空の NTFS フォルダにマウントする(M): 参照(R)...

ドライブ文字またはドライブ パスを割り当てない(D)

< 戻る(B) 次へ(N) > キャンセル

新しいシンプル ボリューム ウィザード

パーティションのフォーマット
このパーティションにデータを格納するには、最初にパーティションをフォーマットする必要があります。

このボリュームをフォーマットするかどうかを選択してください。フォーマットする場合は、使用する設定を選択してください。

このボリュームをフォーマットしない(D)

このボリュームを次の設定でフォーマットする(O):

ファイル システム(F): NTFS

アロケーション ユニット サイズ(A): 既定値

ボリューム ラベル(V): ボリューム

クイック フォーマットする(P)

ファイルとフォルダの圧縮を有効にする(E)

< 戻る(B) 次へ(N) > キャンセル

meta-diskボリュームの概算は、

Cmb / 32768 + 1 = 2048/32768 + 1 = 2M

なので、ここでは余裕を持って10Mのボリュームを準備します。

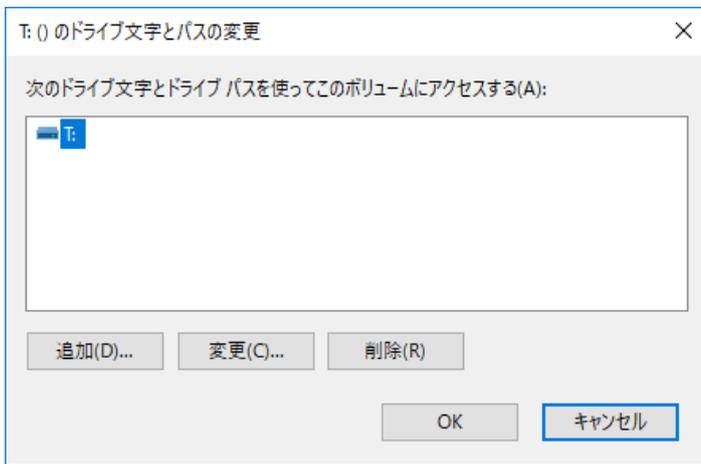
- GUIDを使用する場合、mountvolコマンドで下位デバイス、meta-disk用のGUIDを調べます。

```
\\?\Volume{53ffdbac-0000-0000-0000-200000000000}\  
E:\  
\\?\Volume{53ffdbac-0000-0000-0000-300000000000}\  
T:\
```

- コマンドプロンプトでC:\windrbd\etc\w0.resを作成します。サンプルwindrbd-sample.resがインストールされるのでそれをベースに環境に合わせて変更します。

```
include "global_common.conf";  
  
resource "w0" {  
    protocol C;  
  
    net {  
        use-rle no;  
    }  
  
    on tate-z4 {  
        address 10.30.103.4:7600;  
        node-id 1;  
        volume 1 {  
            disk /dev/sdb1;  
            device /dev/drbd1;  
            flexible-meta-disk internal;  
        }  
    }  
  
    on tate-z2 {  
        address 10.30.103.2:7600;  
        node-id 2;  
        volume 1 {  
            disk "53ffdbac-0000-0000-0000-200000000000";  
            device "H:" minor 1;  
            meta-disk "53ffdbac-0000-0000-0000-300000000000";  
        }  
    }  
}
```

- GUIDを使用した場合で、ドライブ文字を割り当てた場合はもう不要ですので、ディスク管理の「ドライブ文字とパスの変更」で削除しておきます。



- あとはLinux版同様に構成します。

```
$ drbdadm.exe create-md w0  
$ drbdadm.exe up w0  
$ drbdadm.exe --force primary w0
```

- ここでエクスプローラからドライブH:にアクセスすれば、既存のボリュームE:の内容を参照できます。

6.2. Linux 側の設定

- 構築例 (新規ボリューム)のLinux 側の設定と同じですのでそちらを参照ください。なお、/dev/sdb1にWindows側と同じサイズ2048Mとmeta-disk 10Mの合計サイズ、2058Mのパーティションを準備します。

7. ソースファイルからビルド

Windows7のマシンを用意し以下を実行します。テストモードに切り替えて使用しますので、専用マシンを準備して使用することをお勧めします。

7.1. cygwinをインストール

<http://www.cygwin.com/> からsetup-x86_64.exeをダウンロードし、インストールします。

Select PackagesでDevelを選んでおきます。

7.2. wdk をインストール

<https://docs.microsoft.com/en-us/windows-hardware/drivers/develop/installing-the-enterprise-wdk> からWDKをダウンロードします。その後、c:\ewdkに展開します。

```
$ mkdir c:\ewdk
$ unzip <where>/EnterpriseWDK_rs2_release_15063_20170317-1834.zip
```

7.3. Linux 上の作業

ubuntu 16.04 LTS を用意し、あらかじめ build-essential, coccinelleあたりを入れておきます。

```
# apt-get install build-essential
# apt-get install coccinelle
```

windrbdのソースを持ってきて、makeを実行しWindows対応のソースを作成します。spatchコマンドを使用しますので、coccinelleパッケージがインストールされていないと失敗します。

```
$ git clone --recursive git://github.com/LINBIT/windrbd.git
$ cd windrbd
$ make
```

作成したwindrbdソースをwindowsマシンのc:\tmpにコピーします。

drbd-utils を持ってきます。

```
$ git clone git://github.com/LINBIT/drbd-utils.git drbd-utils-windrbd
$ cd drbd-utils-windrbd
```

drbd-utils-windrbdをwindowsマシンのc:\tmpにコピーします。

7.4. Windows上の作業

Windows 7をテストモードに切り替えます。コマンドプロンプトの管理者権限で以下を実行しリブートします。

```
$ bcdedit /set TESTSIGNING ON
```

リブート後、コマンドプロンプトの管理者権限で以下を実行します。

7.4.1. drbd-utils

```
$ cd drbd-utils-windrbd
$ ./autogen.sh
$ ./configure --prefix=/cygdrive/c/windrbd/usr --localstatedir=/cygdrive/c/windrbd/var
--sysconfdir=/cygdrive/c/windrbd/etc --without-83support --without-84support --without
-drbdmon --with-windrbd
$ make
$ make install
```

c:\windrbd にインストールされますので、~/.profileに

```
PATH=$PATH:/cygdrive/c/windrbd/usr/sbin
```

を加えることで、source ~/.profileでdrbdad.exe等のコマンドが使用できるようになります。

7.4.2. windrbd

self-signのキーを作成します。Create Private Key Passwd画面はそのまま空でOKを押します。

```
$ cd windrbd/crypto
$ cat makekey.bat
cd "c:\ewdk\Program Files\Windows Kits\10\bin\x64"
makecert -r -sv C:\Tmp\windrbd\crypto\linbit-ha.pvk -n CN=LINBIT
C:\Tmp\windrbd\crypto\linbit-ha.cer
cert2spc C:\Tmp\windrbd\crypto\linbit-ha.cer C:\Tmp\windrbd\crypto\linbit-ha.spc
pvk2pfx -pvk C:\Tmp\windrbd\crypto\linbit-ha.pvk -pi a -spc C:\Tmp\windrbd\crypto\linbit-
ha.spc -pfx C:\Tmp\windrbd\crypto\linbit-ha.pfx -po ""
$ chmod 755 makekey.bat
$ ./makekey.bat
$ ln -s linbit-ha.cer user.crt
```

あとはtopからmakeすることで、windrbd.sysが作成されます。

```
$ make
$ make install
```

画面の指示に従いリブートします。