Version: **2020.01**

Dernière mise-à-jour : 2020/01/30 03:28

SO210 - Gestion du système de fichiers ZFS

Présentation de ZFS

Solaris 10 comprend le système de fichiers Solaris ZFS, nouveau système de fichiers 128 bits. Solaris ZFS offre une administration simple, une sémantique transactionnelle, une intégrité complète des données et une capacité d'évolution extraordinaire. Solaris ZFS ne constitue pas l'amélioration d'une technologie existante. Il s'agit d'une approche totalement nouvelle de gestion de données.

Solaris ZFS utilise un modèle de stockage en pools qui supprime purement et simplement le concept de volumes. Ainsi, ZFS supprime les problèmes liés à la gestion des partitions, à l'approvisionnement et à la croissance des systèmes de fichiers. Des centaines de systèmes de fichiers peuvent faire partie d'un seul et même pool de stockage. Chaque système n'utilise que l'espace qui lui est strictement nécessaire. La bande passante d'E/S combinée de tous les périphériques du pool est disponible à tout moment pour tous les systèmes de fichiers.

Toutes les opérations sont des transactions copie-écriture. L'état sur disque est donc toujours valide. Chaque bloc comprend une somme de contrôle. Toute corruption des données silencieuses est donc impossible. Les données peuvent, en outre, être auto-rétablies dans des configurations répliquées. Cette fonctionnalité signifie que si une copie est endommagée, Solaris ZFS la détecte et utilise une autre copie pour réparer celle endommagée.

Le vocabulaire ZFS

L'introduction de ZFS a apporté un vocabulaire nouveau :

Terme	Description
pool	Un élément de stockage regroupant une ou plusieurs partitions de disques contenant un ou plusieurs file systems
file system	Un **dataset** contenant répertoires et fichiers
clone	Une copie d'un file system
snapshot	Une copie en lecture seule de l'état d'un file system

Terme	Description
compression	La réduction de l'occupation disque obtenue par la suppression de blocks de données identiques
checksum	Un chiffre long de 256 bits utilisés pour valider des données quand elles sont lues ou écrites
quota	La limite maximale d'occupation disque utilisée par un groupe ou un utilisateur
reservation	Une quantité d'espace disque réservée à un utilisateur ou à un file system
mirror	Une copie exacte d'un disque ou d'une partition
RAID-Z	L'implémentation ZFS de RAID-5
RAID-Z2	L'implémentation ZFS de RAID-6
RAID-Z3	L'implémentation ZFS de *Triple Parity RAID*

ZFS Commands

Les commandes ZFS sont :

Commande	Description
zpool	Utilisée pour gérer des pools
zfs	Utilisée pour gérer des file systems

La Commande zpool

La commande **zpool** utilise un jeu de sous-commandes :

Commande	Description
create	Crée un pool et configure son point de montage
destroy	Détruit un pool
list	Affiche la santé et l'utilisation d'un pool
get	Affiche la liste des caractéristiques d'un pool
set	Fixe une caractéristique d'un pool
status	Affiche la santé d'un pool
history	Affiche les commandes utilisées sur un pool depuis sa création
add	Ajoute un disque à un pool existant

Commande	Description
remove	Supprime un disque d'un pool existant
replace	Remplace un disque par un autre dans un pool existant
scrub	Vérifie les checksums d'un pool et répare des blocs de données endommagés

La Commande zfs

La commande **zfs** utilise également un jeu de sous-commandes :

Command	Description
create	Creates a ZFS file system, sets its properties and automatically mounts it
destroy	Destroys a ZFS file system or snapshot
list	Displays the properties and storage usage of a ZFS file system
get	Displays a list of ZFS file system properties
set	Sets a property for a ZFS file system
snapshot	Creates a read-only copy of the state of a ZFS file system
rollback	Returns the file system to the state of the **last** snapshot
send	Creates a file from a snapshot in order to migrate it to another pool
receive	Retrieves a file created by the subcommand **send**
clone	Creates a copy of a snapshot
promote	Transforms a clone into a ZFS file system
diff	Displays the file differences between two snapshots or a snapshot and its parent file system
mount	Mounts a ZFS file system at a specific mount point
unmount	Unmounts a ZFS file system

Préparation de la VM Solaris 10

Éteignez la VM Solaris 10. Dans la section **Stockage** de l'**Oracle VM VirtualBox Manager**, ajoutez les disques suivants au contrôleur **existant** SATA :

Type	Taille	Nom
vmdk	256 Mb	Disk1
vmdk	256 Mb	Disk2
vmdk	256 Mb	Disk3
vmdk	256 Mb	Disk4
vmdk	256 Mb	Disk5
vmdk	20 Gb	Mirror

Démarrez la VM Solaris 10, créez ensuite le fichier **reconfigure** à la racine du système et re-démarrez le :

```
# touch /reconfigure
# shutdown -i0 -g1 -y
```

Connectez-vous à la VM et vérifiez que les disques ont été détectés :

```
# format
Searching for disks...done


AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <ATA    -VBOX HARDDISK  -1.0 cyl 2085 alt 2 hd 255 sec 63>  My Disk
          /pci@0,0/pci8086,2829@d/disk@0,0
       1. c0t2d0 <ATA    -VBOX HARDDISK  -1.0  cyl 253 alt 2 hd 64 sec 32>
          /pci@0,0/pci8086,2829@d/disk@2,0
       2. c0t3d0 <ATA    -VBOX HARDDISK  -1.0  cyl 253 alt 2 hd 64 sec 32>
          /pci@0,0/pci8086,2829@d/disk@3,0
       3. c0t4d0 <ATA    -VBOX HARDDISK  -1.0  cyl 253 alt 2 hd 64 sec 32>
          /pci@0,0/pci8086,2829@d/disk@4,0
       4. c0t5d0 <ATA    -VBOX HARDDISK  -1.0  cyl 253 alt 2 hd 64 sec 32>
          /pci@0,0/pci8086,2829@d/disk@5,0
       5. c0t6d0 <ATA    -VBOX HARDDISK  -1.0  cyl 253 alt 2 hd 64 sec 32>
          /pci@0,0/pci8086,2829@d/disk@6,0
       6. c0t7d0 <ATA    -VBOX HARDDISK  -1.0  cyl 2608 alt 2 hd 255 sec 63>
```

```
          /pci@0,0/pci8086,2829@d/disk@7,0
Specify disk (enter its number): ^C
```

LAB #1 - La Gestion du Stockage ZFS

La Création d'un Pool en Miroir

Créez maintenant un pool en miroir appelé **mypool** en utilisant les deux premiers des cinq disques supplémentaires attachés à votre VM :

```
# zpool create mypool mirror c0t2d0 c0t3d0
```

Vérifiez que votre pool a bien été créé :

```
# zpool list
NAME    SIZE  ALLOC  FREE  CAP  HEALTH  ALTROOT
mypool  242M    77K  242M   0%  ONLINE  -
```

Utilisez maintenant la sous-commande **status** :

```
# zpool status
  pool: mypool
 state: ONLINE
 scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        mypool      ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
```

```
errors: No known data errors
```

Utilisez maintenant les commande **zfs** et la sous-commande **list** afin de visualiser les file systems :

```
# zfs list
NAME     USED  AVAIL  REFER  MOUNTPOINT
mypool  83.5K   210M    31K  /mypool
```

Notez que la commande **zpool** crée automatiquement un file system sur **mypool** puis le monte au point de montage **/mypool**. Si vous ne souhaitez pas que le file system soit monté automatiquement, il convient d'utiliser la commande **zfs set mountpoint=legacy mypool**.

L'Addition de File Systems à un Pool Existant

Créez maintenant deux file systems dans mypool appelés respectivement **/home** et **/home/user1** puis affichez le résultat :

```
# zfs create mypool/home
# zfs create mypool/home/user1
# zfs list
NAME               USED  AVAIL  REFER  MOUNTPOINT
mypool             163K   210M    32K  /mypool
mypool/home         62K   210M    31K  /mypool/home
mypool/home/user1   31K   210M    31K  /mypool/home/user1
```

Notez que les deux file systems partagent le même espace disque avec le pool parent.

La Modification du Point de Montage d'un Pool

Imaginons que vous souhaitiez que le file system **/home** soit monté ailleurs que sous /mypool. Avec ZFS, cet objectif est atteint simplement :

```
# mkdir /users
# zfs set mountpoint=/users mypool/home
# zfs list
NAME                USED  AVAIL  REFER  MOUNTPOINT
mypool              173K   210M    32K  /mypool
mypool/home          63K   210M    32K  /users
mypool/home/user1    31K   210M    31K  /users/user1
```

Notez que ZFS démonte le file system **/mypool/home** et le re-monte à **/users** d'une manière automatique et transparente.

L'Ajout d'un Hot Spare

Pour visualiser tous le caractéristiques associées à **mypool**, utilisez la commande **zpool** avec la sous-commande **get** :

```
# zpool get all mypool
NAME    PROPERTY       VALUE                  SOURCE
mypool  size           242M                   -
mypool  capacity       0%                     -
mypool  altroot        -                      default
mypool  health         ONLINE                 -
mypool  guid           10980242944530705327   -
mypool  version        32                     default
mypool  bootfs         -                      default
mypool  delegation     on                     default
```

```
mypool  autoreplace    off                    default
mypool  cachefile      -                      default
mypool  failmode       wait                   default
mypool  listsnapshots  on                     default
mypool  autoexpand     off                    default
mypool  free           242M                   -
mypool  allocated      257K                   -
mypool  readonly       off                    -
```

Notez que la valeur d'**autoreplace** est **off**. Afin d'utiliser un hot spare, il est nécessaire de changer cette valeur à **on** :

```
# zpool set autoreplace=on mypool
# zpool get autoreplace mypool
NAME    PROPERTY     VALUE    SOURCE
mypool  autoreplace  on       local
```

Ajoutez maintenant en tant que spare le quatrième disque que vous avez précédemment créé :

```
# zpool add mypool spare c0t5d0
# zpool status mypool
  pool: mypool
 state: ONLINE
 scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        mypool      ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
        spares
          c0t5d0    AVAIL
```

```
errors: No known data errors
```

L'Observation de l'Activité du Pool

Créez un fichier de données aléatoires dans **/users/user1** :

```
# cat /dev/urandom > /users/user1/randomfile &
909
```

Notez le PID. Vous en aurez besoin pour tuer le processus ultérieurement.

Affichez maintenant l'activité du pool en utilisant la sous-commande **iostat** de la commande **zpool** :

```
# zpool iostat -v 3
               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
mypool      48.6M   193M      0      7    172   150K
  mirror    48.6M   193M      0      7    172   150K
    c0t2d0      -      -      0      5    230   155K
    c0t3d0      -      -      0      5     58   155K
----------  -----  -----  -----  -----  -----  -----

               capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
mypool      67.0M   175M      0     77      0  1.66M
  mirror    67.0M   175M      0     77      0  1.66M
    c0t2d0      -      -      0     54      0  1.66M
    c0t3d0      -      -      0     54      0  1.66M
```

```
----------  -----  -----  -----  -----  -----  -----

              capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
mypool      78.6M   163M      0    157      0  3.65M
  mirror    78.6M   163M      0    157      0  3.65M
    c0t2d0      -      -      0     89      0  3.66M
    c0t3d0      -      -      0     90      0  3.74M
----------  -----  -----  -----  -----  -----  -----

              capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
mypool      98.8M   143M      0    118      0  1.77M
  mirror    98.8M   143M      0    118      0  1.77M
    c0t2d0      -      -      0     64      0  1.77M
    c0t3d0      -      -      0     64      0  1.75M
----------  -----  -----  -----  -----  -----  -----

              capacity     operations    bandwidth
pool        alloc   free   read  write   read  write
----------  -----  -----  -----  -----  -----  -----
mypool       111M   131M      0    290      0  3.87M
  mirror     111M   131M      0    290      0  3.87M
    c0t2d0      -      -      0    139      0  3.88M
    c0t3d0      -      -      0    139      0  3.88M
----------  -----  -----  -----  -----  -----  -----

^C#
```

Vérifiez que votre miroir fonctionne.

Tuez maintenant le processus **randomfile** :

```
^C# kill -9 909
#
```

Supprimez le fichier **/users/user1/randomfile**:

```
# rm -rf /users/user1/randomfile
909 Killed
```

Le Mise en Place d'un Quota pour un Utilisateur

Afin de mettre en place un quota pour un utilisateur, vous devez utiliser la sous-commande **set** de la commande **zpool** :

```
# zfs set quota=50M mypool/home/user1
# zfs list
NAME               USED  AVAIL  REFER  MOUNTPOINT
mypool             390K   210M    31K  /mypool
mypool/home         63K   210M    32K  /users
mypool/home/user1   31K  50.0M    31K  /users/user1
```

Créez maintenant un fichier de données aléatoires :

```
# cat /dev/urandom > /users/user1/testfile
cat: output error (0/1040 characters written)
Disc quota exceeded
```

Constatez que l'espace disque disponible dans /users/user1 est de 0 :

```
# zfs list mypool/home/user1
NAME                USED  AVAIL  REFER  MOUNTPOINT
mypool/home/user1  50.1M      0  50.1M  /users/user1
```

Supprimez le fichier **testfile** :

```
# rm -f /users/user1/testfile
```

La Mise en Place d'une Réservation pour un Utilisateur

Une réservation se met en place aussi simplement qu'un quota :

```
# zfs set reservation=25M mypool/home/user1
# zfs get reservation mypool/home/user1
NAME                PROPERTY     VALUE   SOURCE
mypool/home/user1  reservation  25M     local
```

L'Utilisation de Snapshots

Créez un fichier dans **/users/user1**:

```
# echo "This is a test file for the first snapshot" > /users/user1/snapshot1
# ls /users/user1
snapshot1
```

Pour créer un snapshot, il convient d'utiliser la sous-commande **snapshot** de la commande **zfs** :

```
# zfs snapshot mypool/home/user1@snapshot1
```

Le snapshot est stocké dans le répertoire caché **/users/user1/.zfs/snapshot** :

```
# ls -l /users/user1/.zfs/snapshot
total 3
drwxr-xr-x   2 root     root           3 Dec  2 17:30 snapshot1
```

Comme vous pouvez constaté, le snapshot contient le fichier **snapshot1** :

```
# ls -l /users/user1/.zfs/snapshot/snapshot1/
total 2
-rw-r--r--   1 root     root          43 Dec  2 17:30 snapshot1
```

Il est important à noter que le répertoire .zfs ne peut pas être listé par la commande ls, même en utilisant l'option **-a** :

```
# ls -laR /users/user1
/users/user1:
total 8
drwxr-xr-x   2 root     root           3 Dec  2 17:30 .
drwxr-xr-x   3 root     root           3 Dec  2 15:26 ..
-rw-r--r--   1 root     root          43 Dec  2 17:30 snapshot1
```

Créez maintenant un snapshot récursif de tous les file systems dans votre pool :

```
# zfs snapshot -r mypool@snapshot2
```

Les snapshots sont stockés dans leur répertoires **.zfs** réspectifs :

```
# ls /users/.zfs/snapshot
snapshot2
# ls /users/user1/.zfs/snapshot
snapshot1  snapshot2
```

Listez maintenant tous les snapshots :

```
# zfs list -t snapshot -r mypool
NAME                         USED  AVAIL  REFER  MOUNTPOINT
mypool@snapshot2                0      -    31K  -
mypool/home@snapshot2           0      -    32K  -
mypool/home/user1@snapshot1     0      -  31.5K  -
```

```
mypool/home/user1@snapshot2     0      -  31.5K  -
```

Créez un autre fichier dans **/users/user1** :

```
# echo "This is a test file for the second snapshot" > /users/user1/snapshot2
# ls -l /users/user1
total 3
-rw-r--r--   1 root     root          43 Dec  2 17:30 snapshot1
-rw-r--r--   1 root     root          44 Dec  2 17:36 snapshot2
```

Créez maintenant un autre snapshot récursif de **mypool**:

```
# zfs snapshot -r mypool@snapshot3
# zfs list -t snapshot -r mypool
NAME                          USED  AVAIL  REFER  MOUNTPOINT
mypool@snapshot2                 0      -    31K  -
mypool@snapshot3                 0      -    31K  -
mypool/home@snapshot2            0      -    32K  -
mypool/home@snapshot3            0      -    32K  -
mypool/home/user1@snapshot1      0      -  31.5K  -
mypool/home/user1@snapshot2      0      -  31.5K  -
mypool/home/user1@snapshot3      0      -    32K  -
```

La sous-commande **diff** de la commande **zfs** permet de voir les différences entre les deux snapshots :

```
# zfs diff mypool/home/user1@snapshot2 mypool/home/user1@snapshot3
M       /users/user1/
+       /users/user1/snapshot2
```

La sortie ci-dessus démontre que le fichier **/users/user1/snapshot2** a été ajouté au deuxième snapshot sur la ligne de commande saisie.

Notez que vous pouvez retrouver d'autres caractères dans la sortie de **zfs diff** selon les circonstances :

Charactère	Description
M	**M**odification
R	**R**enommer
+	Ajouté
-	Supprimé

Notez que vous ne pouvez pas comparer les snapshots dans l'ordre inverse :

```
# zfs diff mypool/home/user1@snapshot2 mypool/home/user1@snapshot1
Unable to obtain diffs: mypool/home/user1@snapshot1 is not a descendant dataset of mypool/home/user1@snapshot2
```

Faire un RollBack vers un Snapshot

Notez que vous ne pouvez faire un RollBack que vers le dernier snapshot dans la sortie de la commande **zfs list** :

```
# zfs list -t snapshot -r mypool
NAME                          USED  AVAIL  REFER  MOUNTPOINT
mypool@snapshot2                 0      -    31K  -
mypool@snapshot3                 0      -    31K  -
mypool/home@snapshot2            0      -    32K  -
mypool/home@snapshot3            0      -    32K  -
mypool/home/user1@snapshot1      0      -  31.5K  -
mypool/home/user1@snapshot2      0      -  31.5K  -
mypool/home/user1@snapshot3      0      -    32K  -
# zfs rollback mypool/home/user1@snapshot2
cannot rollback to 'mypool/home/user1@snapshot2': more recent snapshots exist
use '-r' to force deletion of the following snapshots:
mypool/home/user1@snapshot3
```

Supprimez donc le snapshot **snapshot3** :

```
# zfs destroy mypool/home/user1@snapshot3
# zfs list -t snapshot -r mypool
NAME                          USED  AVAIL  REFER  MOUNTPOINT
mypool@snapshot2                 0      -    31K  -
mypool@snapshot3                 0      -    31K  -
mypool/home@snapshot2            0      -    32K  -
mypool/home@snapshot3            0      -    32K  -
mypool/home/user1@snapshot1      0      -  31.5K  -
mypool/home/user1@snapshot2      0      -  31.5K  -
```

Maintenant fair un RollBack vers le snapshot **snapshot2** :

```
# zfs rollback mypool/home/user1@snapshot2
# ls -l /users/user1
total 2
-rw-r--r--   1 root     root          43 Dec  2 17:30 snapshot1
```

Notez l'absence du fichier **snapshot2**.

Cloner un Snapshot

Les snapshots sont des file system en lecture seule. Pour transformer un snapshot en un file system en lecture-écriture, utilisez la sous-commande **clone** de la commande **zfs** :

```
# zfs clone mypool/home/user1@snapshot2 mypool/home/user3
# zfs list
NAME                          USED  AVAIL  REFER  MOUNTPOINT
mypool                       25.7M   184M    31K  /mypool
mypool@snapshot2                 0      -    31K  -
```

```
mypool@snapshot3                 0      -    31K  -
mypool/home                  25.0M   184M    32K  /users
mypool/home@snapshot2            0      -    32K  -
mypool/home@snapshot3            0      -    32K  -
mypool/home/user1            32.5K  50.0M  31.5K  /users/user1
mypool/home/user1@snapshot1      0      -  31.5K  -
mypool/home/user1@snapshot2      0      -  31.5K  -
mypool/home/user3               1K   184M  31.5K  /users/user3
```

Listez le contenu de **/users/user3** :

```
# ls -l /users/user3
total 2
-rw-r--r--   1 root     root          43 Dec  2 17:30 snapshot1
```

L'Utilisation de la Compression

La compression peut être activée soit lors de la création du file system, soit après. La compression ne fonctionne que pour les données sauvegardée après l'activation. Activez la compression sur **/mypool/home/user1** :

```
# zfs set compression=on mypool/home/user1
# zfs get compression mypool/home/user1
NAME               PROPERTY     VALUE     SOURCE
mypool/home/user1  compression  on        local
```

Remplacement Manuel d'un Disque Défectif

Dans le cas d'un remplacement d'un disque sans spare, il convient d'utiliser la sous-commande **replace** de la commande **zpool** :

```
# zpool status mypool
  pool: mypool
```

```
 state: ONLINE
 scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        mypool      ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
        spares
          c0t5d0    AVAIL

errors: No known data errors

# zpool replace mypool c0t2d0 c0t4d0
```

Utilisez de nouveau la sous-commande **status** de la commande **zpool** afn d'observer le résultat de la commande précédente :

```
# zpool status mypool
  pool: mypool
 state: ONLINE
 scan: resilvered 646K in 0h0m with 0 errors on Mon Dec  2 17:51:36 2019
config:

        NAME        STATE     READ WRITE CKSUM
        mypool      ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
        spares
          c0t5d0    AVAIL

errors: No known data errors
```

Notez que le *Resilvering* ZFS est l'équivalent de la re-synchronization sous UFS.

La Destruction d'un Pool

La destruction d'un pool est obtenue en utilisant la sous-commande **destroy** de la commande **zpool** :

```
# zpool destroy mypool
# zfs list
no datasets available
```

Notez que cette opération détruit aussi les snapshots !!

La Création d'un Pool en RAID-5

Créez un pool RAID-5 en utilisant l'algorythme RAID-Z :

```
# zpool create mypool raidz c0t2d0 c0t3d0 c0t4d0 spare c0t5d0
# zpool status mypool
  pool: mypool
 state: ONLINE
 scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        mypool      ONLINE       0     0     0
```

```
          raidz1-0  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
        spares
          c0t5d0    AVAIL

errors: No known data errors
```

Détruisez **mypool** :

```
# zpool destroy mypool
```

La Création d'un Pool en RAID-6

Créez un pool RAID-6 en utilisant l'algorythme RAID-Z2 :

```
# zpool create mypool raidz2 c0t2d0 c0t3d0 c0t4d0 c0t5d0 spare c0t6d0
# zpool status mypool
  pool: mypool
 state: ONLINE
 scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        mypool      ONLINE       0     0     0
          raidz2-0  ONLINE       0     0     0
            c0t2d0  ONLINE       0     0     0
            c0t3d0  ONLINE       0     0     0
            c0t4d0  ONLINE       0     0     0
            c0t5d0  ONLINE       0     0     0
        spares
          c0t6d0    AVAIL
```

```
errors: No known data errors
```

Détruisez **mypool** :

```
# zpool destroy mypool
```

Consulter l'Historique Zpool

La sous-commande **history** de la command **zpool** permet de consulter l'historique des actions sur un pool :

```
# zpool history
no pools available
```

Notez que l'historique des pool supprimés a été aussi supprimée !

<html> Copyright © 2020 Hugh Norris. </html>