TeamDrive

Sync your data fast & securely

TeamDrive Host Server Administration

Release 3.0.013.13

Lenz Grimmer, Eckhard Pruehs

2015

CHAPTER

ONE

COPYRIGHT NOTICE

TeamDrive Systems GmbH
https://www.teamdrive.com
Max-Brauer-Allee 50
22765 Hamburg, Germany
Email: info@teamdrive.com

CHAPTER

TWO

TRADEMARK NOTICE

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Apache HTTP Server, Apache, and the Apache feather logo are trademarks of The Apache Software Foundation.

MySQL is a registered trademark of Oracle and/or its affiliates.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation.

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices.

VMware is a trademark or registered trademark of VMware, Inc. in the United States and/or other jurisdictions.

“Amazon Web Services”, “Amazon S3” are trademarks of Amazon.com, Inc. or its affiliates in the United States and/or other countries.

“Red Hat Linux” and “CentOS” are trademarks of Red Hat, Inc. in the U.S. and other countries.

All other names and trademarks used herein are the property of their respective owners.

CHAPTER

THREE

INTRODUCTION

This document will guide you through the administration and advanced configuration of a TeamDrive Host Server.

When managing the TeamDrive Hosting Service, we assume that you have basic knowledge of:

- **Linux system administration:**
  - Adding/configuring software packages
  - Editing configurations files
  - Creating user accounts
  - Assigning file ownerships and privileges
  - Creating and mounting file systems
  - Setting up environment variables
- Apache web server: installation and configuration, adding and enabling modules, modifying configuration files
- MySQL Database: installation and configuration, administration/maintenance, using the MySQL command line client, basic SQL
- Basic knowledge of application server technology

CHAPTER

FOUR

TEAMDRIVE HOSTING SERVICE ADMINISTRATION

4.1 Configuring the Storage Upgrade URL

Storage upgrade: The server informs the TeamDrive Clients how much storage space and traffic is used per Space or account. The Space owner can reserve storage space via the TeamDrive Client and the TeamDrive Clients will generate an URL that opens in the browser. The URL always points to the Hosting Service. This request can be forwarded as required via a rewrite statement.

Open the file `/etc/httpd/conf.d/td-hostserver.httpd.conf` in an editor and ensure the following configuration option matches your environment.

Please replace in the Rewrite-Rule "bestellung.hostserver.com" with the URL pointing to your own server that provides information about how to upgrade storage. If a user clicks on the "More Storage" button in the TeamDrive Client, the client will open the URL specified.

Using the Rewrite-Rule allows you to redirect these requests to a custom web page where you can offer storage upgrade options:

```
# This Rewrite is required for the storage-upgrade-buttons
# in the TD-Client (see storage-upgrade-note in the documentation)
RewriteRule ^/upgrade/([a-z][a-z])/order.html(.*) \
https://bestellung.hostserver.com/$1/order.php$2 [R,NE]
```

The URL called by the client is structured as follows:

```
http://<domain-name>/upgrade/<2-character-language-code>/order.html
```

Examples of language codes are: `en` (English), `de` (German) and `fr` (French).

Additionally, the following values are provided by the TeamDrive Client as URL parameters:

- **spaceid**: The Space ID of the Space
- **host**: The host name (host name and Space ID together are always unique)
- **user**: The TeamDrive user name (BASE64-encoded)
- **check**: Checksum used to verify whether the request is valid

This allows you to create an order page according to your requirements and adapt it to your own needs (payment link). However, this page must always be present so that the user does not see an error message or an empty page.

Information about Spaces and Accounts can be retrieved from the Hosting Service via the Hosting Service API (an HTTP based interface which uses XML-formatted requests and replies). Please consult the TeamDrive Hosting Service Reference Guide for details.

Functions to delete Spaces and increase Storage limits after payment, for example, are also available. Please contact support@teamdrive.net if you need assistance in using this API.

4.2 Disabling the Apache Access Log

In view of the amount of requests issued by the TeamDrive Clients, there is no point in keeping the normal access log activated. We therefore suggest to deactivate it in a production environment. Only the error log should be left enabled. To facilitate this, comment out the following line in the default `httpd.conf`:

```
# CustomLog logs/access_log combined
```

If problems occur in a Space, logging can be activated for a specific Space (see http://httpd.apache.org/docs/2.2/mod/mod_log_config.html). e.g. all access to Space ID 3204 will be logged (the required Apache logging module needs to be enabled again):

```
SetEnvIf Request_URI 3204 spaceid-3204
CustomLog logs/spaceid-3204-requests.log common env=spaceid-3204
```

Restart the Apache instance and check the log files for errors.

4.3 Changing the Admin Console Password

The Host Server Admin Console can only be accessed by the "HostAdmin" User, by entering the correct password that was defined during the initial installation.

If you want to change the password, you first need to log out from the Administration Console to return to the login page (click on **Logout** in the left navigation bar).

On the page, click on **Change Password...** to enable two input fields that allow you to enter the new password twice (to ensure you did not mistype it by accident). You also need to enter the current password in the **Password:** field above and click **Login and Change Password** to apply the new password.

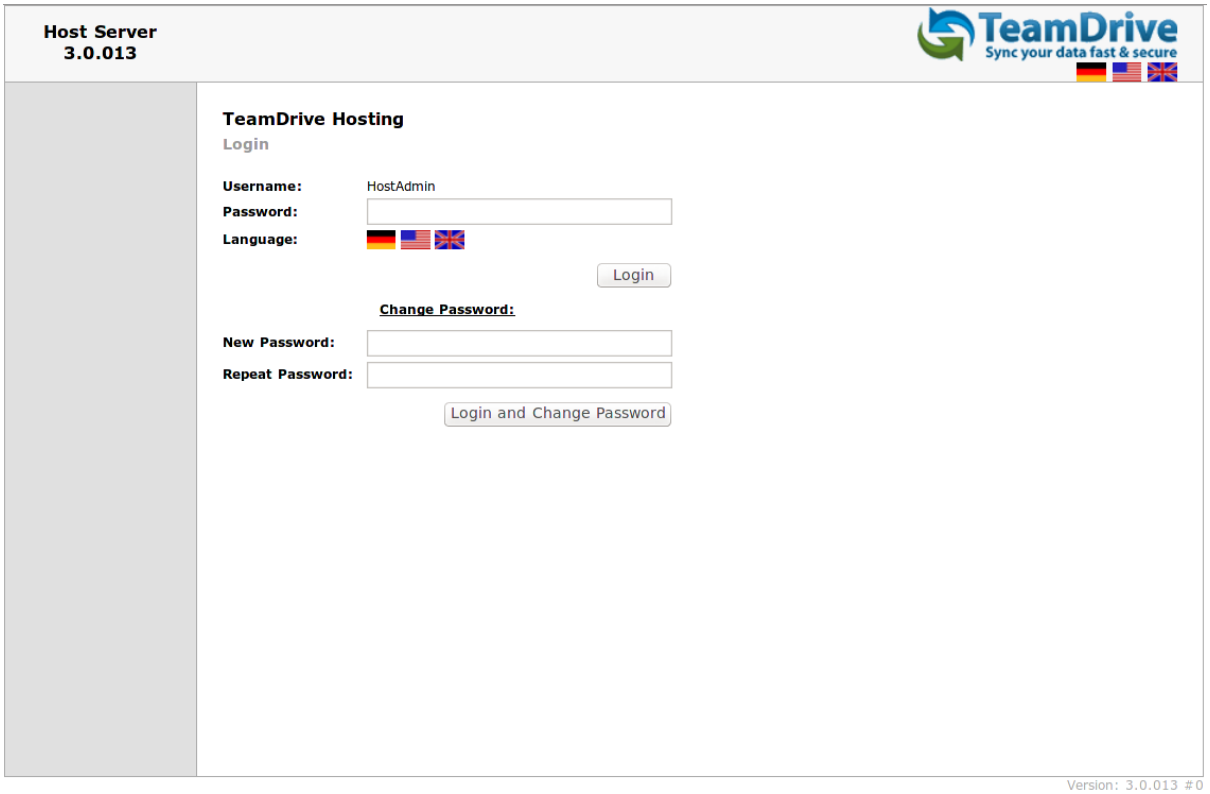

Fig. 4.1: Host Server Administration Console: Change Password

In case you lost or forgot the current password for the Administration Console, you need to remove

the current hashed password stored in the MySQL Database (Setting `AdminPassword`, located in Table `pspace.Setting`). This can be performed using the following SQL query.

Log into the MySQL database using the `teamdrive` user and the corresponding database password:

```
[root@hostserver ~]# mysql -u teamdrive -p
Enter password:

[...]

mysql> use pspace;
Database changed

mysql> SELECT * FROM Setting WHERE Name='AdminPassword'\G
*************************** 1. row ***************************
          ID: 6
        Name: AdminPassword
       Value: $2y$10$wGWlkeMt6K9qToIy.SxH3ufrP8Zrivv26HnICtKYDucQbtVmzme3u
    ReadOnly: 1
     Visible: 0
        Type: VARCHAR
CreationTime: 2014-05-20 15:25:09
  ModifyTime: 2014-05-26 11:09:28
1 row in set (0.00 sec)

mysql> UPDATE Setting SET Value='' WHERE Name='AdminPassword';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> quit
Bye
```

Now you can enter a new password on the login page as outlined above, by clicking the **Change Password** link, but leaving the **Password** field empty and only entering the new password twice, followed by clicking the **Login and Change Password** button.

4.4 Changing the MySQL Database Connection Information

The Host Server Apache modules `mod_pspace` and `mod_yvva` as well as the `yvvad` daemon that performs the `td-hostserver` background tasks need to be able to communicate with the MySQL management database of the Host Server.

If you want to change the password of the `teamdrive` user or move the MySQL database to a different host, the following changes need to be performed.

To change the MySQL login credentials, edit the file `/etc/td-hostserver.my.cnf`. The password for the `teamdrive` MySQL user in the `[p1db]` option group must match the one you defined earlier:

```
[p1db]
database=pspace
user=teamdrive
password=<password>
host=127.0.0.1
```

If the MySQL database is located on a different host, make sure to modify the `host` variable as well, providing the host name or IP address of the host that provides the MySQL service. If required, the TCP port can be changed from the default port (3306) to any other value by adding a `port=<port>` option.

4.5 Manually creating a Depot

The default Depot is always linked to a single user. Using the Host Server Admin Console, it is possible to create Depots that are not linked to a particular user. Each TeamDrive Client that has a Depot file can create Spaces within it. A Depot must always be assigned to a TeamDrive user when it is created via the Web Interface . This is the Depot owner, and only they can later expand the storage space from their TeamDrive Client by using the upgrade button.

The Depot files are encrypted using an external tool the `tshs` binary. The name and path of the executable is stored in the setting `TSHSExecutable`. This should be configured correctly during the TeamDrive software installation process, and need not be altered.

To set up a new Depot, click in the navigation on **Space Depots -> Add New Depot**.

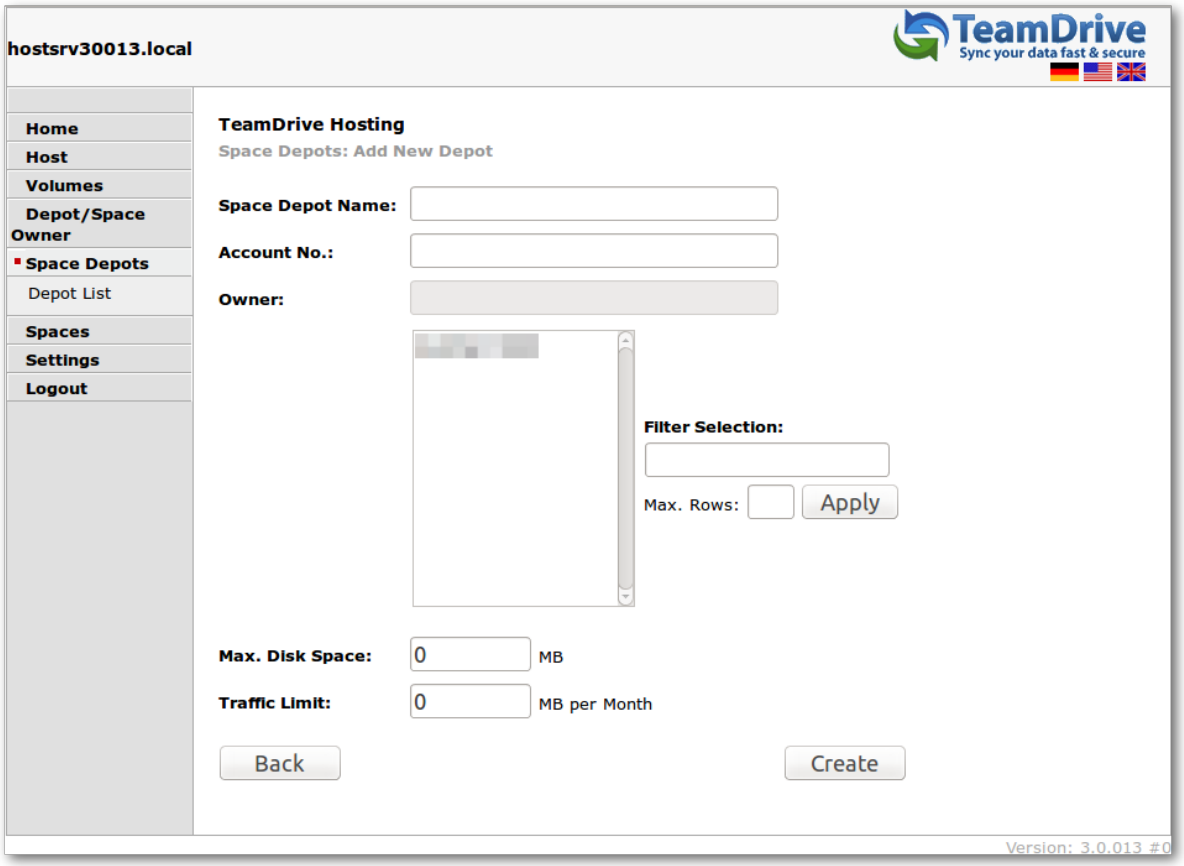

Fig. 4.2: Host Server Admin Console: Add New Depot

First select a user from the selection list to which the Depot will be assigned. Use the **Filter Selection** input field to search for a specific username after clicking the **Apply** button.

Space Depot Name: **TeamDrive Depot**
Account No.: **TeamDrive account no.**
Owner: **Depot Owner**
Max. Disk Space: **1024**
Traffic Limit: **10240**

Description:

Any name can be selected for the Space Depot name. The name appears in the TeamDrive Client in the list of available Depots.

Any account number can likewise be selected and can be used as a reference for other systems.

Disk space and traffic limit should be set up at a ratio of at least 1:10 because users invite each other to the Spaces and the traffic thus may always be higher than the storage space used.

Click on **Create** to create the Depot.

By clicking on the **Depot Access File** link in the Depot Details screen, you can download the respective Depot file, which can be imported into the TeamDrive Client.

4.6 Increasing Volume Storage Space

The first scaling strategy is to add additional volumes to increase the available storage capacity. You should consider adding more volumes (or increase the size of a volume), if any existing volume reaches 60% of utilization.

For additional scaling, we recommend to add an object store which will scale unlimited. We offer additional tools for moving local data to the object store. An extended apache module will redirect client requests to the data that was moved to the object store. The clients could read the data directly from the object store, if it supports the HTTP protocol.

Please contact sales@teamdrive.com for supported object stores.

4.7 Optional Configuration Settings

In the settings you can set up configurations, such as the IP address of the external server needed for the XML invocations referred to into access the account on the Hosting Service and automatically upgrade the storage space, if needed (after payment is received). Only requests from this IP are then accepted. The value in "UpgradeServerIP" is delivered empty so that no safety check is done.

4.7.1 Use HTTPS for publish files

The TeamDrive Professional client license enables TeamDrive Clients to publish files so that they can be accessed without using a TeamDrive Client. The default protocol for downloading the data is HTTP. You could add a boolean setting with the name `UseHTTPsForPublishFiles` and set the value to true to use HTTPS instead (additional changes in the Apache config and adding a valid SSL certificate are necessary by the admin of the Hosting Service). Please notice, that this value will only be used for new accounts. Existing accounts will not be changed to use HTTPS.

4.7.2 Enabling storing Space Names

Each created Space will be stored on the Hosting Service. This record has different information such as: user, account, status and usage information and also the original Space title. For security reasons, the storing the Space names on the server is disabled by default . To enable storing the Space names: look for a boolean setting named `StoreSpaceNames` and set the value to `true`.

4.7.3 API return Space Names

By default, Space names will not be returned in the API for security reasons. To enable returning Space names: add a boolean setting with the name `API_ReturnSpaceNames` and set the value to `true` (this setting will have no effect, if you disable `StoreSpaceNames` as described above).

4.7.4 Force HTTPS usage for Web Interface

By creating a boolean value `ForceHTTPSUsage` and set it to `true`, you could force using HTTPS in all host admin web pages to avoid transferring secure data without SSL encryption. To use HTTPS you have to configure the Apache http Server to support SSL and install a certificate for your server.

4.7.5 Reporting Usage Statistics

It's possible to generate a monthly report that contains detailed statistics about all existing Depots and Spaces within these depots, including the monthly traffic and disk usage. The report is prepared in the form of an XML file `statistic_from_MM_DD_YYYY_to_MM_DD_YYYY.xml` by `p1_autotask` at the beginning of each month. To enable the generation of these statistics, you need to change the Host Server setting `SpaceStatisticEnabled` from `False` to `True`. By default, the resulting report files will be written to `/tmp/`. If you prefer a different location, you can provide an alternative directory name in the configuration setting `SpaceStatisticExportPath`.

4.7.6 External Traffic

The Hosting Service can store data externally (e.g., Amazon S3 storage). Custom modules are needed for the connection, which you can request from sales@teamdrive.net. Outsourcing and directly accessing the client on external storage also generates external traffic. This is recorded separately and added to the direct traffic for the Depot. For transparency, this is displayed separately in the Web Interface . However, the value is only visible if the required module is used. The summarized value of the external traffic and the traffic directly to the host is provided to the TeamDrive Client.

CHAPTER

FIVE

BACKUPS AND MONITORING

5.1 Host Server Backup Considerations

The two most important assets of a TeamDrive Host Server are the storage volumes that host the actual TeamDrive Spaces as well as the MySQL database that stores the related meta data.

The backup schedule depends on the amount of users, their activity and your recovery point objective. We recommend to run a backup at least once a day. The backups should be safely stored on another system.

Ideally, the time and frequency of the Host Server backup should be synchronized with the backup schedule used on the associated Registration Server — this ensures that the information about Users and their Space Depots is consistent across these systems.

In a virtualized environment, the usage of VM snapshots is highly recommended, as these provide atomic and instant full-system copies across multiple instances that can be backed up offline.

The backup of the Host Server's Space Volume(s) can be performed by any given file system backup tool.

When planning a backup of the volume containing the TeamDrive Spaces, keep in mind that the `last.log` files, located in each Space directory in the directory `protolog` are frequently updated by the TeamDrive Clients. New space data and events are constantly appended to the file. When the log files reach a certain size (currently set to 8MB, but this value is not fixed and could change in later versions of TeamDrive), they get renamed and new `last.log` files will be created. This operation is initiated by the Clients. The naming scheme is to rename `last.log` to `<number>.log`, where `<number>` is the next free number, starting from 0. Previously renamed log files are not modified anymore, but must remain available to the clients since these logs must be read when a Space is joined.

To create a consistent backup, the best approach is to perform a snapshot of the entire Space Volume file system, preferably after shutting down the Apache http Server beforehand. If you are using an incremental backup method like `rsync`, keep in mind that some Spaces may have been changed while the rsync job is still running. For consistency, we suggest to perform a full rsync run while the service is running (to sync the bulk of the changes), then briefly change the volume's status to **Standby** or shut down the Apache http Server and run rsync once more, to transfer the remaining changes that have occurred in the meanwhile. Once the rsync job has finished, the Apache http Server can be started again.

The MySQL databases must also be backed up periodically, ideally at the same time the Space Volume(s) are being backed up. This ensures a consistent snapshot of the file system and the related meta data included in the MySQL database.

The Host Server's MySQL databases that need to be backed up are named `pspace` and (optionally) `hostapilog`. They use MySQL's InnoDB storage engine to provide transaction support, fast recovery and consistency. Any of the ususal MySQL backup methods may be used, e.g. `mysqldump`. The size of the Host Server's MySQL Databases is usually quite small, if API logging is not enabled.

The MySQL backup can be performed using any established MySQL backup method, e.g. running a `mysqldump` via a cron job, or using more sophisticated tools like Percona XtraBackup or Oracle's MySQL Enterprise Backup. Other commercial backup solutions usually offer MySQL-specific plugins or extensions as well.

An example MySQL backup job using `mysqldump` could look like as follows. The SQL dump is piped through `gzip` for compression before it is written to a directory `/backup`, using a time stamp for the file name:

```
[root@regserver ~]# mysqldump -u root -p --single-transaction \
--databases pspace hostapilog \
| gzip > /backup/td-hostserver-mysql-$(date +%Y-%m-%d_%H.%M).sql.gz
```

See the MySQL documentation at https://dev.mysql.com/doc/refman/5.1/en/backup-and-recovery.html for more details and hints on how to define a MySQL backup strategy.

If the I/O overhead introduced by running the backup job on the production database is a concern, we recommend setting up a MySQL replication slave on another host and use this one to perform the backup. This second MySQL instance can also function as a hot standby server for high-availability purposes.

More details about MySQL replication and high availability can be found in the MySQL reference manual at https://dev.mysql.com/doc/refman/5.1/en/replication.html and https://dev.mysql.com/doc/refman/5.1/en/ha-overview.html.

In addition to the Space Volumes and MySQL databases, we recommend to create backup copies of the Server's configuration files. Please refer to the *TeamDrive Host Server Installation Guide* for details on the relevant configuration files.

These files should be backed up at least every time you changed them. These backups can be performed using any file-based backup method, e.g. using `tar`, `rsync` or more sophisticated backup tools, e.g. Amanda or Bacula.

5.2 Setting up Server Monitoring

It's highly recommended to set up some kind of system monitoring, to receive notifications in case of any critical conditions or failures.

Since the TeamDrive Host Server is based on standard Linux components like the Apache http Server and the MySQL database, almost any system monitoring solution can be used to monitor the health of these services.

We recommend using Nagios or a derivative like Icinga or Centreon. Other well-established monitoring systems like Zabbix or Munin will also work. Most of these offer standard checks to monitor CPU usage, memory utilization, disk space (especially the file systems providing the TeamDrive Space Volumes) and other critical server parameters.

In addition to these basic system parameters, the existence and operational status of the following services/processes should be monitored:

- The MySQL Server (system process `mysqld`) is up and running and answering to SQL queries
- The Apache http Server (`httpd`) is up and running and answering to http requests (this can be verified by accessing the files http://hostserver.yourdomain.com/ping.xml and http://hostserver.yourdomain.com/admin/ping.xml)
- The `td-hostserver` service is up and running (process name `yvvad`)
- For Host Servers using an Amazon S3 compatible object store (see *Setting up an Amazon S3-Compatible Object Store* (page 27) for details): the `s3d` process is up and running
- For Host Servers using TeamDrive Scalable Hosting Storage (TSHS, see *TeamDrive Scalable Hosting Storage* (page 31) for details): the `tshs` process is up and running (and all related MySQL nodes are up and running, too)

5.3 Restoring individual Spaces or Volumes

In case of corrupted or lost data of a single Space or complete Volume, is is possible to restore the Space or Volume data from a previously created backup.

An example scenario would be a Space that was entirely deleted by a user by accident, or the recovery of a file that was moved to the Space's Trash Folder and the Trash was then emptied.

Note: Note that it is not possible to restore an individual file from a particular Space on the Host Server — due to the client-side encryption it's impossible to determine the correct file on the server side. However it is possible to restore the entire **state** of a Space and all of its files to a previous version, which will allow the user to extract the missing file(s) on the client side.

An additional challenge is identifying the Space(s) you want to restore; by default, Space names are not stored on the Host Server and are only referenced by their ID. Take extra caution and double check you're working on the correct Space.

The process of restoring a Space or Volume involves the following steps:

1. Identify the ID of the Space or Volume you want to restore.
2. Deactivate the Space or all Spaces of a Volume by setting the "Deactivated for restore" status, to prevent TeamDrive Clients from accessing the affected TeamDrive Spaces.
3. Restore the Space(s) by restoring the necessary Space directory or entire volume directory from your backup to the corresponding location.
4. Reactivate the Space(s) to make it/them available to the TeamDrive Clients again.
5. The Clients will be notified that a Space recovery is required, which should be performed according to the procedure outlined in the TeamDrive Client documentation.

If a restore of a single Space is required, the task of (de-)activating it can be performed via the Hosting Service Administration Web Interface. Open the Space Details page, check the **Deactivated for restore** checkbox and click **Save** to change the state. After the restore has finished, click **Complete Restore** to re-enable the Space.

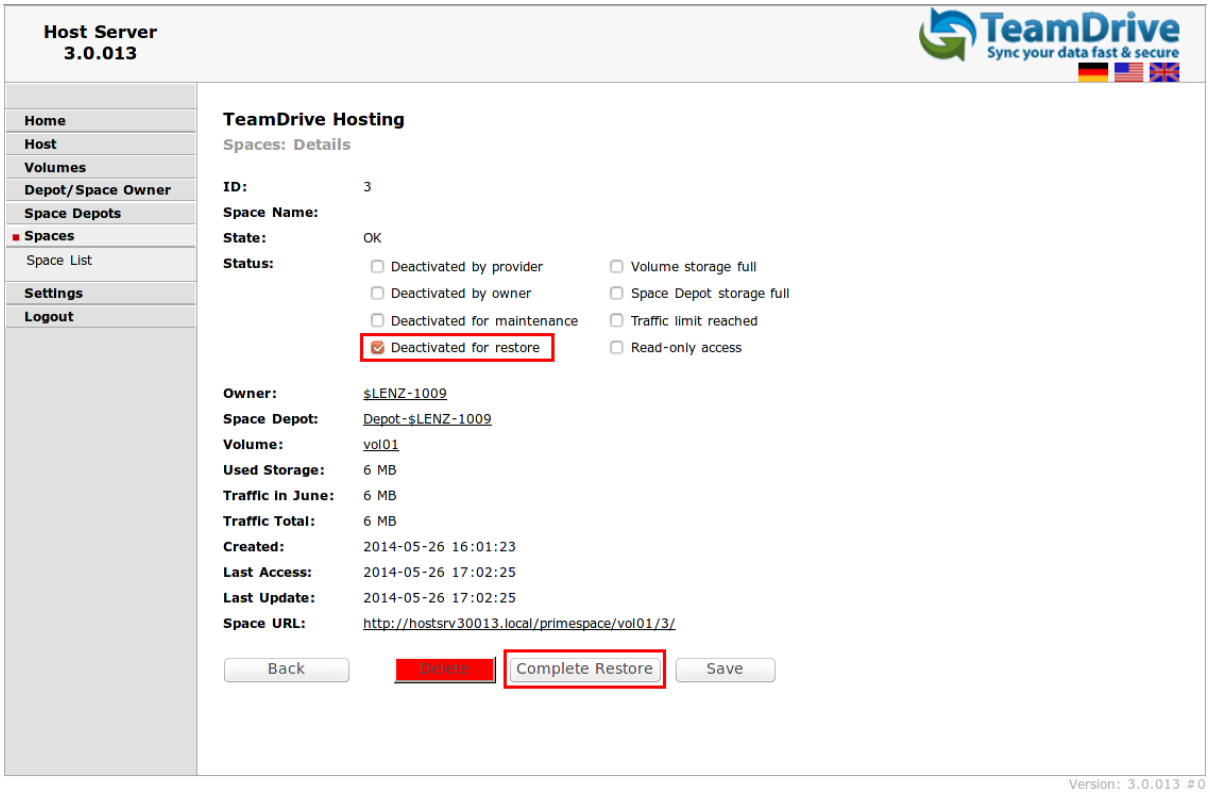

Fig. 5.1: Host Server Administration Console: Restore Space

Alternatively, the Space deactivation and reactivation can be performed on the command line, as outlined below.

If you need to restore an entire volume, all Spaces contained in this volume need to be marked for restore. You can use a script included in the Host Server installation, which performs the tasks of deactivating and reactivation of all Spaces on the affected volume.

In any case, the actual task of restoring the Space(s) from backup has to be perfomed manually by the administrator and the TeamDrive clients will have to perform a local Space recovery to get the local Spaces back into a consistent state.

The examples describe the usage of the "Restore Script" used to restore an individual Space as well as an entire Space Volume.

5.3.1 Load and start the Restore Script

The Restore Script has to be executed using the Yvva Runtime Environment's commandline shell `yvva`:

```
[root@hostserver ~] yvva
Welcome to yvva shell (version 1.0.0).
Enter "go" or end the line with ';;' to execute submitted code.
For a list of commands enter "help".

> execute file 'RestoreSpace.pbt' location 'setup/scripts/Restore';;

Usage:
-------
[] List volumes available for restore:
        RestoreSpace:volumes();
[] List spaces availbale for restore:
        RestoreSpace:spaces();
[] Deactivate volume before restoring:
        RestoreSpace:deactivateVolume(<ID>);
        i.e. 'RestoreSpace:deactivateVolume(5)'; to deactivate volume with ID 5.
[] Deactivate space before restoring:
        RestoreSpace:deactivateSpace(<ID>);
        i.e. 'RestoreSpace:deactivatSpace(7);' to deactivate space with ID 6.
[] Reactivate volume after restoring:
        RestoreSpace:reactivateVolume(<ID>);
        i.e. 'RestoreSpace:reactivateVolume(5);' to reactivate volume with ID 5.
[] Reactivate space after restoring:
        RestoreSpace:reactivateSpace(<ID>);
        i.e. 'RestoreSpace:reactivateSpace(7);' to reactivate space with ID 7.
```

Loading the Restore Script does not execute any task yet, it only displays a short usage information as shown above. You can execute selected tasks by entering their name and providing the respective Volume or Space ID in brackets. Finish the command with two consecutive semicolons to immediately execute the submitted code.

5.3.2 Identify the Space you want to Restore

Usually, the ID of the Space to restore should be obtained from the TeamDrive Client. The **Space Information** displayed in the Client window for each Space contains a field **Space ID** that contains the ID used on the Host Server. The actual Space data will is stored in a subdirectory below the Space volume, using the Space ID as the directory name.

To list all available Spaces on the Hosting Service you can execute the command `RestoreSpace:spaces()`:

```
> RestoreSpace:spaces();;
.-------------------------------------------------------------------------.
| Spaces                                                                  |
|-------------------------------------------------------------------------|
| Volume | ID     | Title                          | ResID | Status           |
|--------|--------|--------------------------------|-------|------------------|
| vol01  | 3      |                                | 2     | 0                |
| vol01  | 4      |                                | 3     | 0                |
'-------------------------------------------------------------------------'
```

A list of all active Spaces will be displayed which should look similar to the list shown above. Identify the ID of the Space you want to restore.

5.3.3 Deactivate the Space to Restore

After identifying the Space you want to restore, you have to deactivate it by providing the ID to the command `RestoreSpace:deactivateSpace(<ID>)`:

```
> RestoreSpace:deactivateSpace(3);;
140610 13:49:56 [Note] Deactivate space [3]
140610 13:49:56 [Note] Space with ID '3' deactivated.
```

In the example above the Space with the ID '3' has been deactivated. After a short while, the Client will notice this change and mark the Space accordingly on its side.

5.3.4 Restore Backup

After deactivating a Space, you can now restore its data by copying the backup of that Space into the corresponding location on the Space volume, e.g. `/spacedata/vol01/3` in our case.

5.3.5 Reactivate Space

After copying the backup to the deactivated Space, you have to reactivate the Space which makes it available to the TeamDrive Clients again. To reactivate a certain Space you have to execute the command `RestoreSpace:reactivateSpace(<ID>)`:

```
> RestoreSpace:reactivateSpace(3);;
140610 13:54:07 [Note] Reactivate space [3]
140610 13:54:07 [Note] /spacedata/vol01/3
140610 13:54:07 [Note] Space '3' reactivated successful. Restore ID: 3.
Restore Log No: 0, Restore Log Offset: 855
```

In the example above the Space with the ID '3' has been reactivated.

The Client will now notify that the Space has been reactivated and a local Space recovery operation has to be performed.

5.3.6 Identify the Volume you want to Restore

To identify all available volumes on the Hosting Service you have to execute the command `RestoreSpace:volumes()`:

```
> RestoreSpace:volumes();;
.-------------------------------------------------------.
| Volumes                                               |
|-------------------------------------------------------|
| ID    | Name   | Status                               |
|-------|--------|--------------------------------------|
| 1     | vol01  | Operational                          |
| 2     | vol02  | Operational                          |
'-------------------------------------------------------'
```

A list with the volumes will be displayed which should look similar to the list shown above. Identify the ID of the volume you want to restore.

5.3.7 Deactivate the Volume to Restore

After identifying the volume you want to restore, you have to deactivate the volume with all its Spaces. To deactivate a volume with a certain ID you have to execute the command `RestoreSpace:deactivateVolume(<ID>)`:

```
> RestoreSpace:deactivateVolume(1);;
140603 17:13:21 [Note] Deactivate volume [1]
140603 17:13:21 [Note] 2 Spaces of Volume 'vol01' left to deactivate.
140603 17:13:21 [Note] Space with ID '3' deactivated.
140603 17:13:21 [Note] Space with ID '4' deactivated.
```

In the example above the volume with the ID '1' and all the spaces within that volume will be deactivated.

5.3.8 Restore Backup

After deactivating a volume, you can now restore its data by copying the backup of that volume into the corresponding location.

5.3.9 Reactivate Volume

After copying the backup to the deactivated Volume, you have to reactivate the Volume which makes the Spaces available to the TeamDrive Clients again. To reactivate a certain Volume you have to execute the command `RestoreSpace:reactivateVolume(<ID>)`:

```
> RestoreSpace:reactivateVolume(1);;
140603 17:15:37 [Note] Reactivate volume [1]
140603 17:15:37 [Note] Reactivate space [3]
140603 17:15:37 [Note] /spacedata/vol01/3
140603 17:15:37 [Note] Space '3' reactivated successful. Restore ID: 2.
Restore Log No: 0, Restore Log Offset: 855
140603 17:15:37 [Note] Reactivate space [4]
140603 17:15:37 [Note] /spacedata/vol01/4
140603 17:15:37 [Note] Space '4' reactivated successful. Restore ID: 2.
Restore Log No: 0, Restore Log Offset: 384
```

In the example above all Spaces on the Volume with the ID '1' have been reactivated.

5.3.10 Exit the `yvva` session

You can close the `yvva` session by typing `quit` or pressing Ctrl+D on the > prompt.

CHAPTER

SIX

HOST SERVER FAILOVER CONSIDERATIONS AND SCENARIOS

6.1 Scaling a TeamDrive Host Server Setup

A first step in increasing a single Host Server's performance would be to monitor and review the system's CPU, RAM and Disk I/O utilization, and to adjust the server configuration by adding more RAM/CPUs or increasing the storage bandwidth, if necessary (also called "scale-up strategy").

Adding more CPUs typically increases the maximum number of possible concurrent connections to the service and reduces the latency. However, the ability to handle more connections also requires more memory, as the system needs to spawn more concurrent Apache instances. So usually both parameters need to be adjusted.

Adding more RAM can also help to improve database and file system throughput and latency, as it allows the operating system and database to keep more of its working set and caches in memory, which enables it to return data quicker.

If your setup has reached the physical limits of a single server instance, you can further improve the scalability as well as the redundancy of a TeamDrive Host Server by implementing a "scale out" strategy.

In this setup, you distribute the load across several independent systems, by deploying multiple virtual or physical Apache server instances of the TeamDrive Host Server behind one or more load balancers.

This configuration also mitigates the risk of a service outage, e.g. if an instance fails or needs to be taken offline for maintenance purposes.

Note: The Host Server's Space Volumes must be placed on a shared storage medium like an NFSv4 server or shared disk file systems like OCFS2 or GFS2, as each Host Server instance requires concurrent access to the same Space Volume(s). As an alternative to shared storage, using an S3-compatible object store (e.g. Amazon S3 or OpenStack Swift) or the TeamDrive Scalable Hosting Storage (TSHS) can be utilized.

A migration from a single instance setup to such a scaled-out configuration can usually be performed with very little downtime, so you can start small and grow your setup as the need arises.

However, you must ensure that in case of a node failover/outage, the remaining nodes can handle the load that is usually distributed across all server instances.

Note: In a scale-out scenario, the Host Server's MySQL database server must be set up as a separate instance, so each Host Server node has access to the same data set.

To avoid the MySQL database to become a single point of failure, we recommend to set up MySQL in a redundant configuration, too (e.g. by using MySQL replication or other clustering technologies like Galera/Percona Cluster).

Note: The TeamDrive Host Server configuration does not support accessing more than one MySQL Server; you need to use a floating/virtual IP address that gets assigned to the currently active MySQL instance.

If you intend to run multiple independent Host Server instances (e.g. to serve a globally distributed user base), you can assign users to different Host Servers, e.g. by registering more than one Host Server to a given Provider, or using multiple Provider Codes on different Registration Servers.

These independent TeamDrive Host Server instances can then be scaled using the strategies above.

In a single instance configuration, a re-appearing server can suffer from a "thundering herd problem", as a large number of TeamDrive Clients will try to synchronize their accumulated pending changes simultaneously. This can lead to a peak in the number of concurrent connections to this server, its MySQL database and storage subsystem, resulting in a noticable increase in network and disk I/O.

This effect can be mitigated by temporarily extending the poll interval used by the Clients, increasing the number of Apache instances, or by temporarily assigning more resources like vCPUs or vRAM to a virtual machine.

The MySQL server's configuration might also need to be reviewed in order to support more concurrent database connections.

6.2 Host Server Failure Scenarios

This chapter discusses most likely outages that can occur on a TeamDrive Host Server, if no additional redundancy is provided.

Chapter *Host Server Failover Test Plan* (page 22) outlines some possible tests you should perform, and what results to expect.

6.2.1 Entire Host Server Outage

An outage of the entire TeamDrive Host Server can be triggered by any of the following events:

- Failure of the entire Host Server host system (e.g. a hardware or OS crash/failure)
- Network failure that renders the Host Server unavailable
- Failure of the Host Server's Apache http Server
- Failure of the Host Server's Space Volume storage system
- Failure of the Host Server's MySQL Database
- Failure of the S3 compatible object store
- Failure of the TeamDrive Scalable Hosting Storage (TSHS)

In case of an outage of the entire TeamDrive Host Server, the TeamDrive Clients will mark any existing Spaces on that Host Server as "Offline".

However, it is still possible to work with these Spaces locally; the Clients will record any local changes (e.g. adding or removing files, making modifications) and queue these events for later submission once the Host Server is available again.

The following Client operations can not be performed while the Host Server is unavailable:

- Creating new Spaces
- Publishing files
- Inviting users to existing Spaces

In addition to that, the following administrative operations via the Host Server's API or Administration Console will not be possible:

- Creating new Space Depots via the API (e.g. using the Registration Server's Administration Console)
- Changing the limits of existing Space Depots
- Assigning a default depot to a newly registered user upon first login

Except for the MySQL Server outage, this failure scenario can be avoided by setting up multiple instances of the Host Server behind a load balancer with failover capabilities and using a shared/scalable and reduntant storage system for all nodes.

6.2.2 Space Volume Outage

The storage volume hosting the Space Volumes might become unavailable, e.g. because the mount point `/spacedata/vol01` is missing or empty due to a failed mount after a reboot, an outage of the NFS server or a network connection failure between the Host Server and the storage subsystem. The Host Server will notice the missing volume or Space data and log error messages to the Apache error log `/var/log/httd/error_log`, e.g.:

```
[error] [client x.x.x.x] Unable to create space path: Possible Mount Error:
Volume Global ID required: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx", but Space not
found: xx, path: /spacedata/vol01
```

or:

```
[error] [client 10.0.3.1] Space data root path missing: /spacedata/vol01
```

The Clients will receive a notification for Spaces hosted on that volume, indicating that the Space has been disabled for maintenance.

The Server will return to normal operation automatically, as soon as the missing volume is available again, re-enabling the affected Spaces on the Clients.

Increasing the availablility of the storage subsystem can be performed in numerous ways and is highly dependent on the technology or vendor used. Consult the documentation of your storage technology for details/options.

6.2.3 MySQL Database Outage

A failure of the Host Server's MySQL Database could be triggered by one of the following events:

- Failure of the entire MySQL Server host system (e.g. a hardware or OS crash/failure)
- Network failure that renders the MySQL Server unavailable for the Host Server
- Failure of the MySQL Server's `mysqld` process

The failure will be indicated by error messages in the following Host Server log files.

`/var/log/mod_yvva.log`:

```
[Error] -12036 (2002): Can't connect to local MySQL server through socket
'/var/lib/mysql/mysql.sock' (2)
[Error] "startup.yv" (80)
```

`/var/log/mod_pspace.log`:

```
[Error] db_connect(pspace_mdb.c:318) Failed to connect to default group:
[p1db]
[Error] db_connect(pspace_mdb.c:318) MySQL Config file:
/etc/td-hostserver.my.cnf
[Error] p1_send_xml_response(mod_pspace.c:986) Space x: [HTTP 503] Status:
0 Error code: 0
```

To mitigate the risk of a MySQL Server outage, consider setting up a cluster of MySQL Servers, using MySQL replication, DRBD or other replication and HA technologies to provide synchronization and redundancy.

6.2.4 Outage of the `td-hostserver` Background Service

If the `td-hostserver` background service (process name `yvvad`) has failed or was not started at bootup time, regular Client operations are not affected immediately.

However, the following background tasks are no longer performed, which may lead to unwanted consequences over time:

- Spaces marked for deletion are no longer physically removed from the Space Volume's file system, which could lead to the file system filling up until it runs full.
- The disk usage of Volumes and Spaces is no longer calculated. Clients will not be notified if they have reached their storage limits.
- Monthly traffic statistics are not being reset at the end of the month. Clients that have exceeded their traffic in the meanwhile might be blocked from synchronizing Space data once the `td-hostserver` service has been re-enabled.

See chapter *Background Tasks Performed by ''td-hostserver''* in the *TeamDrive Host Server Installation Guide* for further details on the individual tasks performed by this service.

Restarting the `td-hostserver` background service will pick up where the previous process has stopped.

For increased redundancy, it is possible to run this service on each TeamDrive Host Server instance in a multi-server installation.

6.3 Host Server Failover Test Plan

Based on the failover scenarios described in chapter *Host Server Failover Considerations and Scenarios* (page 19), the following tests should be performed to verify the correct behaviour and recovery from failures of individual TeamDrive Host Server components.

This test plan assumes an environment consisting of two virtualized TeamDrive Host Server instances (hostsrv01 and hostsrv02), located behind a load balancer, using a shared NFSv4 share for storing the Space data and using a dedicated MySQL Server instance (td-mysql) for storing Space management information. Other setups/configurations may require additional tests, depending on the environment.

Note: Note that the configuration described above contains several components for which no redundancy is provided, therefore these components are considered single points of failure (SPOF). In particular, the following components can become a SPOF:

- The Host Server's Space Volume. In this scenario, the Space data coming from the TeamDrive Clients is stored on an NFS server. If the NFS share becomes corrupted or unavailable, the TeamDrive Clients will be unable to synchronize Space data with their peers until the file system has been restored or made available again.
- The MySQL database instance (td-mysql). If this instance becomes unavailable, the entire TeamDrive service will be affected and rendered unavailable until the service is restored.
- The load balancer/firewall. If the public-facing load balancer/firewall fails, the TeamDrive service will be unavailable.

6.3.1 Single Host Server Instance Failure

An outage of one of the TeamDrive Host Server instances (hostsrv01 or hostsrv02) should be simulated/triggered in the following ways:

- Shutting down the Apache httpd Server running `service httpd stop`.
- Shutting down the network connection, e.g. by running `service network stop`, `ifconfig eth0 down` or by disconnecting the virtual network interface via the virtual machine management console.
- Shutting down the entire virtual machine e.g. via the virtual machine management console or by running `poweroff`.

Expected results:

- The load balancer should detect that the Host Server instance is no longer available and redirect any incoming traffic to the remaining instance instead. If configured, a notification about the outage should be sent out to the monitoring software.

- The monitoring software should raise an alert about the Host Server instance being unavailable, specifying the nature of the outage (e.g. `httpd` process missing, network unavailable, etc.).
- The remaining Host Server instance should handle all incoming Client requests. The TeamDrive Service should not be impacted/affected in any way.

Once the outage has been resolved and the instance has recovered, the following is expected to happen:

- The load balancer should detect that the Host Server instance is available again. Incoming traffic should be spread across both instances again.
- The monitoring software should detect the service recovery and perform the respective actions (e.g. resetting the alert, sending an update notification).
- The TeamDrive Service should continue unaffected throughout this process

6.3.2 Multiple Host Server Failures

An outage of **both** of the TeamDrive Host Server instances (hostsrv01 and hostserv02) should be simulated/triggered in the following ways:

- Shutting down the Apache httpd Servers running `service httpd stop` on both instances.
- Shutting down the network connections, e.g. by running `service network stop`, `ifconfig eth0 down` on both instances, or by disconnecting the virtual network interfaces via the virtual machine management console.
- Shutting down the entire virtual machines e.g. via the virtual machine management console or by running `poweroff`.

Expected results:

- The load balancer should detect that the Host Server instances are no longer available and stop redirecting any incoming traffic to the instances. Incoming requests should be answered with an appropriate error code (HTTP error code 503 - Service Unavailable). If configured, a notification about the outage should be sent out to the monitoring software.
- The monitoring software should raise an alert about the Host Server instances being unavailable, specifying the nature of the outage (e.g. httpd process missing, network unavailable, etc.).
- The TeamDrive Service will be impacted/affected as outlined in chapter *Entire Host Server Outage* (page 20).

Once the outage has been resolved and at least one of the Host Server instances has been recovered, the following is expected to happen:

- The load balancer should detect that the Host Server instance is available again. Incoming traffic should be redirected to the instance and incoming requests should no longer result in HTTP errors.
- The monitoring software should detect the service recovery and perform the respective actions (e.g. resetting the alert, sending an update notification).
- Once the TeamDrive Clients have noticed the service being available again, operations should proceed as before.

6.3.3 Testing Space Volume Outage

An outage of the TeamDrive Host Server instance's Space Volume (NFS share) should be simulated/triggered in the following ways:

- Detaching/unmounting the NFS share from the Space Volume mount point, for example by temporarily shutting down the Apache http Server and the Host Server background tasks and unmounting the volume, e.g. by running the following commands:

```
# service httpd stop
# service td-hostserver stop
# umount /spacedata/vol01
# service td-hostserver start
# service httpd start
```

- If technically possible, disconnecting NFS share from the virtual machine at run time, e.g. by detaching the network connection (by running `ifconfig <device> down` for the respective network interface, detaching the virtual network card from the virtual machine) or shutting down the NFS server. **Note that this operation may lead to data inconsistencies or file system corruption and should only be performed on non-criticial test data**.

Expected results:

- The TeamDrive Host Server should detect the missing volume and react as outlined in chapter *Space Volume Outage* (page 21).
- The monitoring software should raise an alert about the missing volume.
- Optionally, the load balancer could be instructed to return an error (e.g. HTTP error code 503 - Service Unavailable), to fend off incoming Client requests until the outage has been resolved.

Once the outage has been resolved and the Space Volume has been mounted again, the following is expected to happen:

- The Clients should continue the Space synchronization at the point where they were interrupted by the outage. Incomplete Spaces will be restarted again. In case of a severe corruption of the Space Volume (e.g. file system errors), a restore from backup and a Space/Volume recovery might be required, as documented in the *Team Drive Host Server Administration Guide*.
- The monitoring software should detect the service recovery and perform the respective actions (e.g. resetting the alert, sending an update notification).
- If configured, the load balancer should be instructed to stop returning 503 Errors to Client requests (e.g. by the administrator).

6.3.4 Testing MySQL Server Failures

An outage of one of the MySQL Server instance (td-mysql) should be simulated/triggered in the following ways:

- Shutting down the MySQL Server by running `service mysqld stop`.
- Shutting down the network connection, e.g. by running `service network stop`, `ifconfig eth0 down` or by disconnecting the virtual network interface via the virtual machine management console.
- Shutting down the entire virtual machine e.g. via the virtual machine management console or by running `poweroff`.

Expected results:

- The TeamDrive Host Server instances will no longer be able to handle incoming Client requests as outlined in chapter *MySQL Database Outage* (page 21).
- The monitoring software should raise an alert about the MySQL Server instance being unavailable, specifying the nature of the outage (e.g. `mysqld` process missing, network unavailable, etc.).

Once the outage has been resolved and the MySQL Server is available again, the following is expected to happen:

- The TeamDrive Host Server instances will continue to operate where they were interrupted by the MySQL Server outage. The TeamDrive Clients will pick up where they left, synchronizing all accumulated/pending changes.
- The monitoring software should detect the service recovery and perform the respective actions (e.g. resetting the alert, sending an update notification).

6.3.5 Testing Load Balancer Failure

Since all TeamDrive instances are accessed through a load-balancer, an outage of this component should be tested as well:

- Shutting down the load balancer
- Removing the network connections to the TeamDrive Server components

Expected results:

- The TeamDrive Host Server instances will no longer be able to handle incoming Client or API requests as outlined in chapter *Entire Host Server Outage* (page 20).
- The monitoring software should raise an alert about the load balancer instance being unavailable, specifying the nature of the outage.

Once the outage has been resolved and the load balancer is available again, the following is expected to happen:

- The TeamDrive Host Server instances will continue to operate as soon as they receive incoming Client requests again. The TeamDrive Clients will pick up where they left, synchronizing all pending changes that have accumulated in the meanwhile.
- The monitoring software should detect the service recovery and perform the respective actions (e.g. resetting the alert, sending an update notification).

CHAPTER

SEVEN

SETTING UP AN AMAZON S3-COMPATIBLE OBJECT STORE

Installation and configuration of the TeamDrive S3 Daemon `s3d` is only required if TSHS (see *TeamDrive Scalable Hosting Storage* (page 31) for details) is not enabled and you have an Amazon S3 compatible object store you wish to use as a secondary storage tier.

Currently, Amazon S3, OpenStack Swift and the Ceph Object Gateway (version 0.8 or higher, using the S3-compatible API) are supported.

`s3d` is a process that runs in the background and provides secondary storage by transferring files to the object store. It does this by monitoring the hosted data directory structure and transferring files to the object store when a file reaches an age as specified in the service's configuration. The configuration settings also allows you to specify what files are eligible to be transferred via the use of pattern matching.

Note: Because all object store requests will be signed using the current timestamp, it's essential that the system time is accurate when running `s3d`. Make sure that the NTP service is installed and running. See the chapters about NTP configuration in the installation guides for details.

7.1 Configuring `s3d`

The configuration of `s3d` is performed by changing the relevant configuration settings using the Host Server Administration Console.

Log into your Host Servers Administration Console at https://hostserver.yourdomain.com/admin/ and click on **Settings**. The S3 daemon Settings all begin with `S3`.

The following information is needed by `s3d` to connect to the object store.

`S3Brand` This setting specifies the type of S3 storage. Valid options are: **Amazon** or **OpenStack**.

`S3Server` Your object store's domain name, e.g. `s3.amazonaws.com`.

`S3DataBucketName` The name of the Bucket in the object store that will contain the Space data. The bucket must already exist.

Warning: If you are setting up multiple TeamDrive Hosting servers it is important that they do not use the same Bucket. Doing so can result in data loss.

`S3AccessKey` Your S3 access (public) key, used to access the specified bucket.

`S3SecretKey` Your S3 secret (private) key used to access the specified bucket.

`S3SyncActive` Set this to `True` to enable the synchronization of data stored by the Host Server (Space data) to the specified bucket on an S3 compatible object store. Note that the synchronization won't start until the `s3d` service has been started (see below).

`S3Options` These options control the way S3 is accessed, for example the number of parallel threads during upload, whether to use multipart upload, etc. The options may also contain `S3Brand` specific settings.

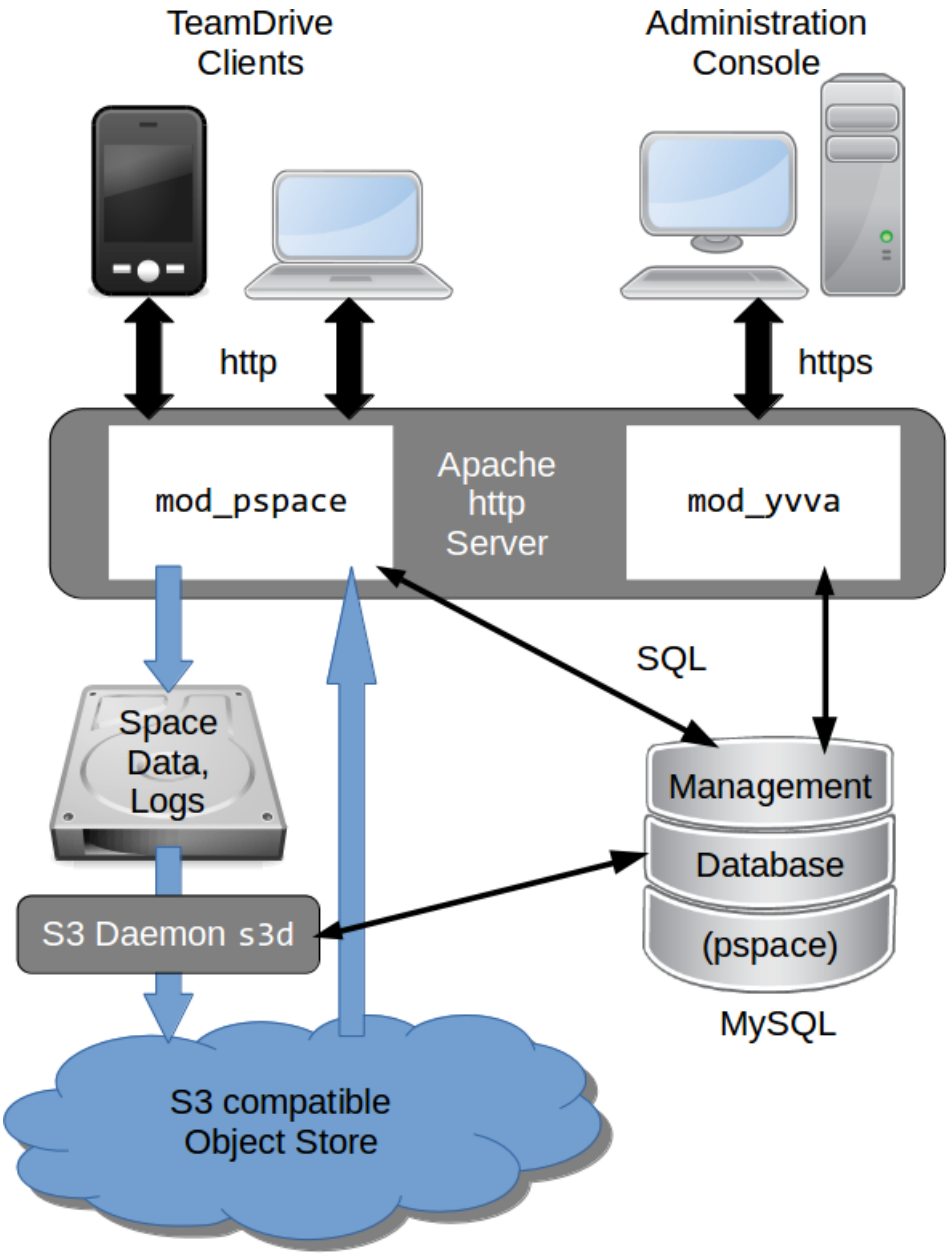

Fig. 7.1: TeamDrive Hosting Service using an S3-compatible object store

S3EnableRedirect When S3 redirect is enabled, the Host Server will redirect the Client to a download directly from the object store, when appropriate. This helps to offload traffic from the Host Server to the object store. If set to `False`, the Host Server fetches the requested object from the object store and serves it to the Client directly.

7.2 Starting and Stopping the `s3d` service

You can use the `/etc/init.d/s3d` init script to start and stop s3d. The configuration setting `S3SyncActive` needs to be set to `True`, otherwise `s3d` will abort with a corresponding error message.

> **Warning:** Enabling the S3 Daemon means that any new data in existing Spaces and new Spaces will be transferred to the object store. Currently, there is no automatic way to return back to a pure file-based Host Server setup — data that was moved to S3 stays in S3. Disabling the S3 secondary storage tier would result in Clients no longer being able to access their Space data.

After starting `s3d` with the command `service s3d start`, check the log file `/var/log/s3d.log` for startup messages. You can use `service s3d status` to check if `s3d` is up and running.

`s3d` should be added to the processes to be started at boot time. To do this execute the following commands as root:

```
[root@hostserver ~]# chkconfig s3d on
```

7.3 Optional configuration parameters

The following optional configuration options can be modified in the `S3Options` configuration setting:

MinBlockSize The minimum block size that can be used for multi-part upload. The valid range of this parameter will be determined by the implementation of the object store being used.

MaxUploadParts The maximum number of parts a file can be divided into for upload. The valid range of this parameter will be determined by the implementation of the object store being used.

MaxFileUploadSize Files larger than this will not be transferred to S3. The default, 0, means all files are uploaded.

MaxUploadThreads The thread pool size to use for multi-part uploads. The default, 0, means single threaded.

BucketAsSubdomain Amazon S3 uses the bucket name as part of the domain name (`BucketAsSubdomain=1`), while other S3-compatible object stores (e.g. OpenStack) include the bucket name as part of the path name after the domain (`BucketAsSubdomain=0`). This option usually does not have to be set explicitly, as the `S3Brand` setting determines this value automatically.

7.4 OpenStack configuration parameters

If the `S3Brand` is set to `OpenStack` then 2 additional parameters are required in `S3Options`, to enable the generation of temporary URLs. Temporary URLs give clients temporary direct access to objects in the object store, helping to reduce network traffic on the Host Server. This requires the setting `S3EnableRedirect` to be set to `True`.

OpenStackAuthPath This is the path component of the OpenStack Authorization URL. For example if the OpenStack Authorization URL is https://swift-cluster.example.com/v1/AUTH_a422b2-91f3-2f46-74b7-d7c9e8958f5d30 then the `OpenStackAuthPath` would be /v1/AUTH_a422b2-91f3-2f46-74b7-d7c9e8958f5d30.

OpenStackAuthKey Your OpenStack temp URL key used to generate temporary URLs.

Your OpenStack administrator should be able to provide you with the OpenStack Authorization URL and the corresponding temp URL key.

Warning: The generation of new temp URL keys will invalidate older keys. It is important that once the temp URL key has been set for your TeamDrive Hosting server that no new keys are being generated.

7.5 Enabling Object Store Traffic Usage Processing

When S3 storage has been enabled, TeamDrive Clients access the data in the object store directly. The traffic required by these operations is recorded by reading the S3 access log files.

This means that setting `S3SyncActive` to `True` changes the way Traffic usage is calculated. When `S3SyncActive` to `False`, the Hosting Service is able to record all traffic usage in the `pspace` database. When `S3SyncActive` to `True` the S3 access logs myst be downloaded and parsed to get the required information.

Note: The traffic usage processing expects the Amazon S3 access log format.

The calculation of traffic usage requires an external Perl-based tool named `aws` to be installed. Install the aws tool as described on its home page at http://timkay.com/aws/. The tools `s3get`, `s3ls` and `s3delete` need to be in the `PATH` of the `apache` user.

In addition, the following configuration settings have to be defined:

S3LogBucketName: Name of the bucket that contains the S3 access logs. Note that if this setting is empty, the object store traffic will not be calculated. You must configure your object store to save the access logs to this bucket. Please refer to your object store documentation to determine how this is done.

S3ToProcessPath: Local path in the filesystem to download above access logs for further processing. By default, this path is set to `/var/opt/teamdrive/td-hostserver/s3-logs-incoming/`

S3ArchiveLogs: If set to `True`, the processed access logs will be moved to the folder defined in the next setting `S3ProcessedPath`.

S3ProcessedPath: If logs need to be kept for own additional analysing, then the S3 log files will be moved to this directory once traffic usage Processing is complete.

By default, the task that calculates the S3 traffic runs every 10 minutes.

CHAPTER

EIGHT

TEAMDRIVE SCALABLE HOSTING STORAGE

If you require a scalable hosting system that grows with the number of users, then you have 2 alternatives:

- Use a scalable file system that allows multiple access points, or,
- Use TSHS, which stores the data in a cluster of MySQL databases.

The TeamDrive Scalable Hosting Storage, TSHS, is a scalable storage system for the TeamDrive Hosting Service based on the MySQL database. It is an alternative to the standard file system based TeamDrive Hosting Service.

If you are not sure of your requirements, you can begin with file system based storage. If your requirements outgrow a file system-based configuration, you can upgrade to TSHS at any time. How to do this is described in the section *Upgrading to TSHS* (page 36).

By default, TeamDrive Hosting Service stores files in the local file system. When TSHS is enabled, the Hosting Service stores files in a cluster of MySQL Servers (not to be confused with MySQL Cluster NDB). Each server in the cluster stores a partition (known as a "shard") of the data. TSHS refers to such a database server as a "Storage Node".

One of the databases in the cluster contains the administration data for the cluster, and this is known as the "Admin Node". The Admin Node contains a list of Storage Nodes in the cluster, and is the starting point for connecting to the cluster.

Scale-out can be achieved by placing each database (Storage Node) in the cluster on a different machine. When a Storage Node exceeds its capacity (either in the terms of storage or computing power), the shard that it contains can be split, which creates a new Storage Node.

Whenever a Storage Node (or the Admin Node) is created, the system administrator must first create an empty MySQL database that will be used by the Node. In this way the administrator can control where the data for the Node will be stored, and how scale-out is achieved.

8.1 TSHS and S3 Compatible Object Storage

In general, the TeamDrive Hosting Service is capable of using an S3 compatible object store as secondary storage. This is also the case when using TSHS. An object store is used as secondary storage to provide unlimited capacity if this is not provided by the primary storage (the file system or TSHS database cluster).

When using the file system for storage, `s3d` (the TeamDrive S3 Daemon) is responsible for transferring files to the object store. This job is done by the `tshs` command line tool when TSHS is enabled (see below).

The object store also serves as a backup for the data in the primary storage. If the primary storage is lost for some reason the data can be restored from the object store. The actuality of the restored data will depend on the frequency with which data is moved to the object store.

The frequency of this operation can be set by the administrator in the configuration of the `s3d` or the `tshs` tool. Nevertheless, it is likely that data is lost in the most active spaces upon restore. This eventuality is handled by the TeamDrive Client if the correct restore procedure is followed, as described before.

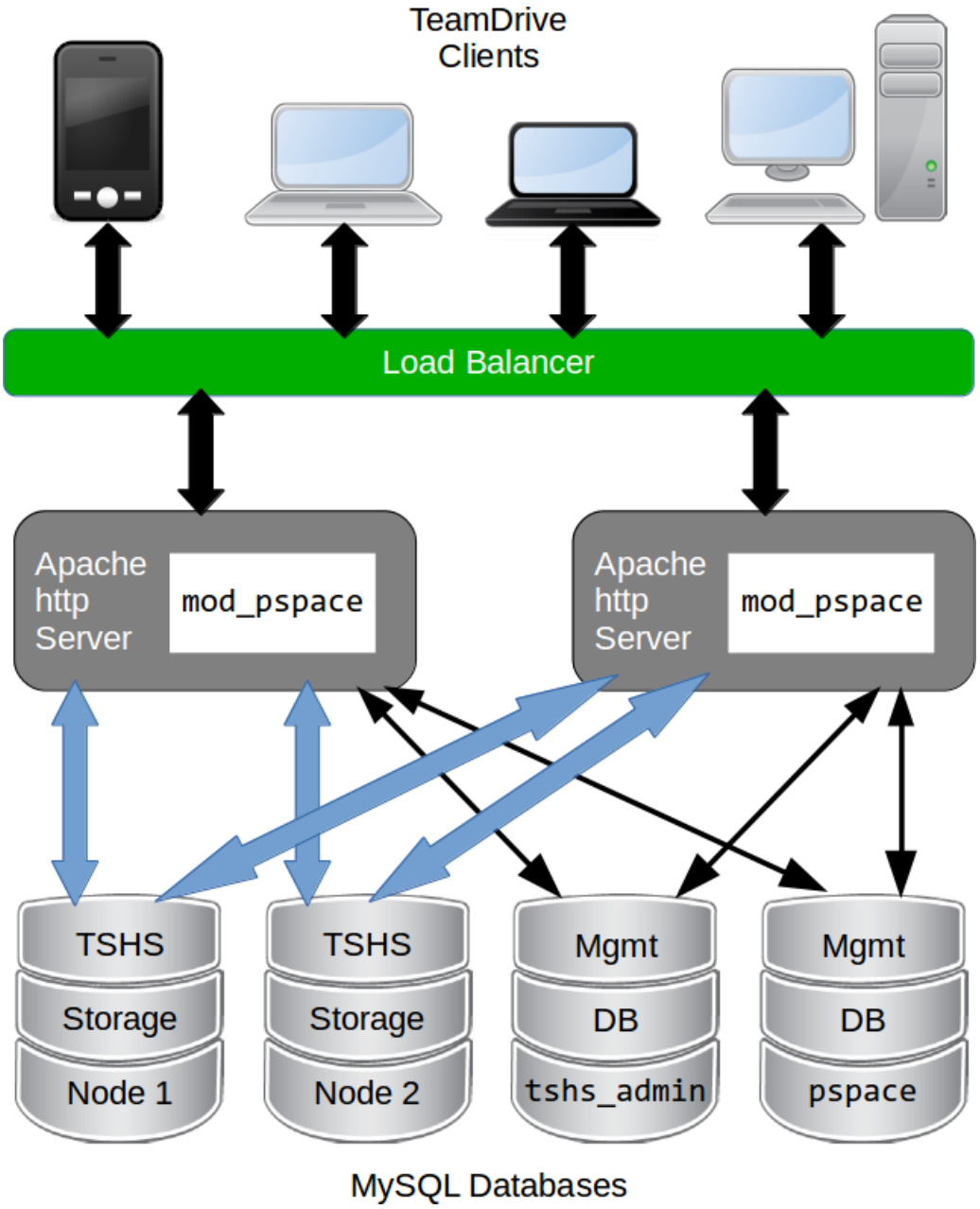

Fig. 8.1: TeamDrive Scalable Hosting Storage (TSHS)

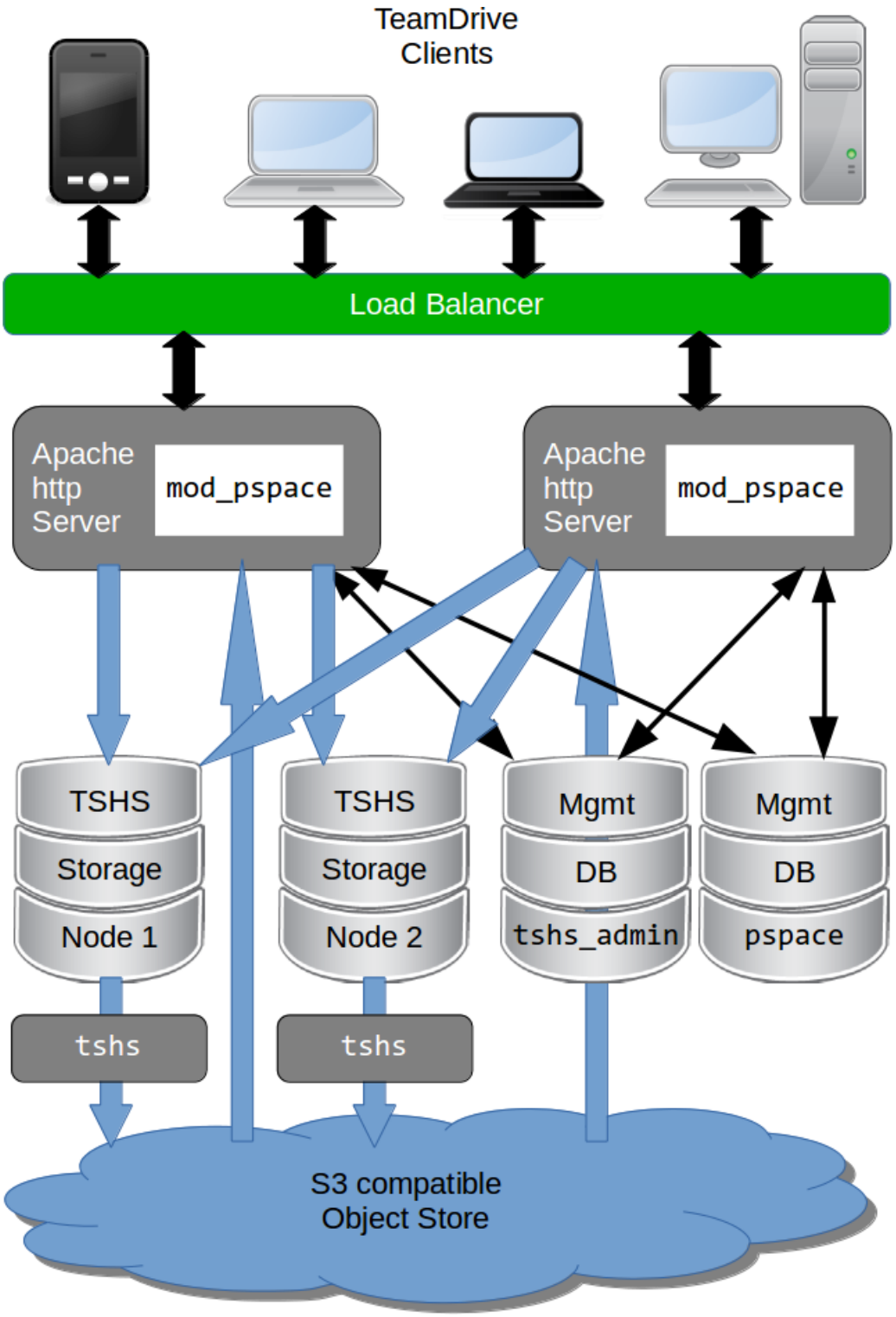

Fig. 8.2: TeamDrive Scalable Hosting Storage (TSHS) with S3 object store

8.2 The tshs Command Line Tool

The TSHS cluster is managed using the tshs command line tool. The command line tool is all you need to create, maintain and expand the TSHS cluster. Alternatively, you can use the Host Server Administration Console, which calls tshs in the background to perform the requested task.

The tshs command line tool can be used in 3 different modes:

1. **One off commands**: Run tshs with command line arguments to perform various once-off tasks, such as create a new Storage Node, split a Storage Node or add a reference to an S3 compatible object store.
2. **Interactive "shell" mode**: To introspect and perform a number of commands on a cluster the easiest is to start tshs in interactive mode, also know as shell mode. In this mode you can run all command line operations, in addition to a few shell only commands.
3. **Background "daemon" mode**: A number of tasks need to be started regularly. These are basically TSHS commands that are run repeatedly by tshs running in daemon mode. For example, if a Storage Node is split, then data must be copied from one node to the other. The copy-shards maintenance command does this.

All maintenance tasks can be run in parallel, and from any machine. This way, it is possible to scale-out the maintenance operations. Additional tshs commands can be run to increase the throughput of any task that needs to be performed.

For example, when a node split occurs, a lot of data may need to be transferred from the old to the new Storage Node. Each copy-shards command only transfers one file at a time. So, starting (for example) 10 tshs instances running the copy-shard command will transfer 10 files simultaneously.

Enter tshs help to get a full list of commands provided by the tshs command line tool.

8.3 Creating a TSHS-based TeamDrive Hosting Service

The first step is to create a tshs_admin database, setup the [tshs] MySQL group and enable TSHS in the Hosting Administration Console as described in the section *Initializing a TSHS Cluster* (page 35) below.

Following this, to the menu item **Host -> TSHS** and create a Storage Node as described in the section *Creating a Storage Node* (page 35).

After you have successfully created a storage node, you need to restart all Apache instances. TSHS cluster usage should then begin immediately. Check the Apache error log (e.g. /var/log/httpd/error_log) to ensure that the module has loaded correctly, and that no errors occurred. You should see something like this:

```
[notice] mod_pspace 1.5.04 Loaded; Build Jun 18 2014 18:15:23;
Crash-Reporting-Disabled
[notice] Admin API booted: TSHS Enabled; Importing Volumes; S3 n/a; Path:
/spacedata
```

You can confirm that the TeamDrive Apache module has connected to the TSHS cluster by running the tshs list-access-points command:

```
$ tshs list-access-points
ID       Ack Access time         Description
-------- --- ------------------- -----------
205      1   2014-06-20 15:57:41 pspace module;host=www.teamdrive.com;pid=75744
206      1   2014-06-20 15:57:41 pspace module;host=www.teamdrive.com;pid=75742
207      1   2014-06-20 15:57:41 pspace module;host=www.teamdrive.com;pid=75741
208      1   2014-06-20 15:57:41 pspace module;host=www.teamdrive.com;pid=75743
209      1   2014-06-20 15:57:41 pspace module;host=www.teamdrive.com;pid=75745
210      1   2013-03-26 21:24:42 pspace module;host=www.teamdrive.com;pid=75746
211      1   2014-06-20 15:57:41 tshs shell;host=www.teamdrive.com;pid=75765
```

This command lists all processes connected to the TSHS cluster. In the example above, the "pspace module" is listed a number of times, once for each process started by Apache.

8.4 Initializing a TSHS Cluster

A TSHS Cluster is initialized by creating an "Admin Node", which is simply a MySQL database within the cluster with the name (by default) `tshs_admin`.

As mentioned above, a TSHS Cluster consists of a group of MySQL databases. The topology (i.e. the configuration of databases, MySQL servers and hardware) of the database cluster is determined by the administrator. However, it is obvious that maximum scalability is achieved by placing each database on its own hardware.

The databases used by TSHS must always be created by the administrator. The administrator determines the name of the user that will be used by TSHS to access the database, and grants the TSHS user the required rights. In order to create a node (Admin or Storage), TSHS requires the right to create its own tables in the database. After that point it only needs complete access to the tables it has created.

Observe the following steps to initialize a TSHS cluster:

1. **Create the tshs_admin database**: Start by creating a database called `tshs_admin`. Add a user to the database (for the purpose of this documentation we will call the user `teamdrive`) and grant the user the right to create tables.
2. **Configure the MySQL connection**: Create an options group called `[tshs]` in your MySQL options file (`/etc/td-hostserver.my.cnf` file). The `[tshs]` group contains the MySQL connection parameters used to connect to the tshs_admin database:

```
[tshs]
database=tshs_admin
user=<tshs-username>
password=<thsh-password>
host=localhost
port=3306
socket=/var/lib/mysql/mysql.sock
```

   How to setup a MySQL connection is explained in the MySQL documentation: http://dev.mysql.com/doc/refman/5.6/en/option-files.html . Use the `--mysql-cfg-file` option to specify the location of the MySQL options file on the `tshs` command line, if you are not using the default location (`/etc/td-hostserver.my.cnf`).

3. **Create the Admin tables**: This is done by using the Hosting Administration Console (menu item **Settings -> TSHSENabled**) to set the configuration setting `TSHSENabled` to `True`.

   When TSHS is enabled, the Console will run the `tshs create-admin-node` command which creates and initializes the tables used by TSHS in the `tshs_admin` database. The command will succeed if TSHS is able to access the `tshs_admin` database you have already created. If the tables already exist, the command has no effect.

The `[tshs]` group described above is the entry point for accessing the entire TSHS cluster. Connection information to other nodes is stored in the `tshs_admin` database itself.

After TSHS has been enabled, the Administration Console will display an additional entry **TSHS** located below the **Host** section of the left navigation bar. Use this page to manage TSHS.

8.5 Creating a Storage Node

Before data can be stored in a TSHS cluster you need to create one or more Storage Nodes.

To create a Storage Node, you must first create an empty MySQL database. The location of the database depends in the topology of your cluster but, as mentioned before, the database must run on its own machine for maximum scalability.

The database may have any name you choose and must contain a user that has the right to create tables. We will call this user `teamdrive` for the purpose of this documentation.

Under the menu item **Host -> TSHS**, choose the `create-storage-node` command, and enter the connection options for accessing the database you have just created in the **Parameters** text field. The `<connection-options>` is a list of MySQL connection options separated by a semicolon (';' character).

For example:

```
user=teamdrive;password=<password>;host=127.0.0.1;database=tshs_storage_1
```

You can use any options that you would otherwise use in a MySQL options file. A complete list of options is provided in the table on this page:

http://dev.mysql.com/doc/refman/5.6/en/mysql-options.html

Now press the **Execute** button to run this command using `tshs`.

This will execute the command `tshs create-storage-node <connection-options>` to create TSHS Storage Node.

This command may only be executed when all Storage Nodes are empty in the cluster. As soon as the cluster contains data, you need to use the `split-shard` command to create new Storage Nodes.

Under **Host -> TSHS**, the Hosting Administration Console displays a list of Storage nodes you have created. The list all the Storage Nodes displayed by the Console is obtained by using the `tshs list-storage-nodes` command:

```
$ tshs list-storage-nodes
ID  Phase        Boundary (size)         Connection options (totals)
--- ------------ ----------------------- ---------------------------
1   ACTIVE_SHARD 0 (2147483648)          user= td;port=3306;database=tshs_storage_1
(InDB=4558 OnS3=2677 R/O=3013 Bytes=4651260)
2   MIRROR_SHARD 2147483648 (2147483648) user=td;port=3306;database=tshs_storage_2
(InDB=5812 OnS3=3908 R/O=3102 Bytes=4523606)
```

Data can be stored in the cluster as soon as you have one Storage Node. However, it is possible to start with several Storage Nodes if you are expecting the cluster to grow large. This will save splitting shards later, which is an expensive operation. Create additional Storage Nodes by calling `create-storage-node` several, times. Each time you must provide a new empty MySQL database. As long as the cluster is empty you can create and delete Storage Nodes until you have the desired start configuration.

Ideally, you should not need to change the topology of your TSHS cluster (other than adding new Storage Nodes) once it is in use. Changing the topology effectively requires changing the connection options stored for a particular Storage Node, as a result of moving one of the storage databases to a different machine or MySQL server. Although this is possible, it currently requires the entire cluster to be shutdown before changing the connection options (shutting down the cluster is done by stopping all the Hosting Service Apache instances).

If the database needs to be copied, then the cluster must remain shut down during this time. You can shorten the downtime, by using MySQL replication to create a copy of a Storage Node database before changing the topology of the cluster.

8.6 Upgrading to TSHS

An existing file system based TeamDrive Hosting Service can be easily upgraded to use TSHS. This process is automatic, and runs while the system is online.

Note: If your existing Hosting Service uses an S3 object store you must **stop and disable the TeamDrive S3 Daemon background task** (`s3d`) before you enable TSHS:

```
[root@hostserver ~] service s3d stop
[root@hostserver ~] chkconfig s3d off
```

However, the setting `S3SyncActive` must remain set to `True`.

When you enable TSHS, the Hosting Administration Console will check to see if file system storage is active. If so, it will automatically set the configuration setting `TSHSImportVolumes` to True.

When `TSHSImportVolumes` is enabled, the data in the File System volumes will be transfered in the background to the TSHS cluster. This transfer occurs automatically, while the system is online.

The main work of importing the data into TSHS is done by the `tshs perform-import` command. TSHS is configured to perform this command by default.

The command checks to if an import job is active, and if so, executes the import. You retrieve the status of an import jobs by listing all "jobs" in the system using `tshs list-jobs`:

```
$ tshs list-jobs
ID     Job    Parameters
------ ------ ----------
236887 IMPORT path=/spacedata;s3-host-id=6
```

The example shows that an import job is in progress. If an S3 object store is in use, then the job contains a reference to an S3 host (see `tshs list-s3-hosts` above).

Although it can be run while the Hosting Storage is online, the `perform-import` command places quite a load on the system. If necessary the command can be run at non-peak times, for example, for a few hours every night until the import is complete. Using the `--time-limit` option you can limit the time a maintenance task will run.

As soon as the `perform-import` task has completed successfully, it is recommended to disable `TSHSImportVolumes` (set the setting to False). The name of the job is changed from IMPORT to IMPORTED when all data has been transfered from the File System volumes to TSHS.

Disabling `TSHSImportVolumes` is an important optimisation because TSHS no longer to search the File System (or S3 Storage if enabled) in order to find files requested by the TeamDrive Client. When import is complete, all the required information on the location of files is stored in the TSHS cluster.

8.7 Scaling Out the Cluster

When a Storage Node reaches its capacity, either in terms of storage or computing power, then the data shard on the node needs to be split.

The `tshs split-storage-node <storage-id> <connection-options>` operation splits the shard on a given node, and creates a new Storage Node at the same time. For example:

```
tshs split-storage-node 2
user=teamdrive;password=<password>;database=tshs_storage_3
```

The `<storage-id>` parameter is a reference an existing Storage Node which you want to split. Approximately half the files that reside on the this Storage Node will be moved to the new Storage Node (using copy and delete operations).

As described above, before you can create a new Storage Node, you must create an empty MySQL database. The `<connection-options>` operation parameters specify the MySQL options required to connect to the new empty MySQL database.

After you have split a Storage Node shard you need to run the `tshs copy-shards` maintenance command. This command copies all files as required from the old Storage Node to the new Storage Node. This operation can be sped up by starting a number of `tshs copy-shards` commands in parallel. Of course, the more copy operations running, the greater the load on the MySQL databases involved. As a result it is recommended that you monitor the database load during this time and start or stop copy operations as appropriate. The new Storage Node remains in MIRROR_SHARD state until the copy operation is complete.

Note that by default the tshs command is setup to run in daemon mode, and perform all outstanding tasks. This incudes the `copy-shards` command.

A Storage Node shard can be split while the cluster is online. Neither the `split-storage-node` or `copy-shards` command interrupt normal operation of the cluster, other than causing increased load due to the copy operation.

The copy operation must be complete (i.e. the Storage Node must be in the ACTIVE_SHARD phase) before it can be split again.

8.8 Connecting TSHS to an S3 Compatible Object Store

When TSHS is connected to an S3 compatible Object Store, it periodically transfers files from the Storage Node database(s) to the Object Store. This is done by the `tshs move-to-s3` maintenance command.

In order to enable the transfer of data to the object store, TSHS needs to be configured for accessing it. This can done by taking the S3-specific setting from the Host Server settings and adding the necessary access details to the `thsh_admin` database using the command `tshs add-s3-host`.

Ensure that the settings `S3Brand`, `S3Server`, `S3DataBucketName`, `S3AccessKey` and `S3SecretKey` contain the correct information to access the S3 bucket that will be used to store the Hosting Service data. See chapter *Configuring s3d* (page 27) for details.

`add-s3-host` will ping the S3 service before actually adding the host details.

The Hosting Administration Console will then take these settings to create the TSHS-specific S3 host entry when you enable S3 by setting the configuration setting `S3SyncActive` to `True`.

You can verify this by running `tshs list-s3-hosts` on the command line or via the Hosting Administration Console.

If any of these setting are incorrect, enabling S3 will fail with an appropriate error message.

Note: You need to restart the Apache http Server using `service httpd restart` after enabling `S3SyncActive`. Check the log file `/var/log/httpd/error_log` for the following notice:

```
[notice] Admin API booted: TSHS Enabled; S3 Enabled (s3d n/a)
```

The Hosting Administration Console sets up the S3 host in the cluster by executing the following command:

```
add-s3-host <brand> <server> <bucket> <access-key> <secret-key> [<options>]
```

For example:

```
tshs add-s3-host AMAZON s3.amazonaws.com my-bucket 022QF06E7MXBSAMPLE
kWcrlUX5JEDGM/SAMPLE/aVmYvHNif5zB+d9+ct
MaxAge=1;MinSize=1;MinBlockSize=5M;MaxUploadParts=1000;MaxUploadThreads=10
```

Currently, `<brand>` must be set to AMAZON or OPENSTACK. `<server>` is the host name of the server (for amazon this is `s3.amazonaws.com`). `<bucket>` is the name of the S3 bucket. `<access-key>` and `<secret-key>` are the keys required to access the bucket.

The Hosting Administration Console takes these values from the associated configuration settings.

The `<options>` string has the form: `<name>=<value>;<name>=<value>;...`

Options are:

`MinSize` The minimum size of a file that will be uploaded to S3. Use K=Kilobytes, M=Megabytes, etc.

`UrlTimeout` The time in seconds that a redirect for reading is valid. The default is 10.

`MinBlockSize` The minimum block size that can be used for multi-part upload. Default is 5MB. The valid range of this parameter will be determined by the implementation of the object store being used.

`MaxUploadParts` The maximum number of parts a file can be divided into for upload. Default is 1000. The valid range of this parameter will be determined by the implementation of the object store being used.

MaxFileUploadSize Files larger than this will not be transferred to S3. The default, 0, means all files are uploaded.

MaxUploadThreads The thread pool size to use for multi-part uploads. The default, 0, means single threaded.

The options value used by Hosting Administration Console is stored in the `S3Options` configuration settings.

When an S3 host is added it, becomes the current S3 host. All files uploaded to S3 storage automatically go to the current S3 host. You can set the current S3 host using the command `tshs set-current-s3-host <host-id>`.

You can list all S3 hosts with the `list-s3-hosts` command:

```
$ tshs list-s3-hosts
ID  Cur Server           Bucket               Public Key           Private Key
Options
--- --- ---------------- -------------------- --------------------
---------------------------------------- -------
6   yes s3.amazonaws.com mybucket  022QF06E7MXBSAMPLE
kWcrlUX5JEDGM/SAMPLE/aVmYvHNif5zB+d9+ct
MinSize=1;UrlTimeout=10;MinBlockSize=5K;MaxUploadParts=1000;MaxUploadThreads=2
```

To check the file transfer progress, you can use the command `tshs transfer-to-s3-info`, which will display how many files and their total size are ready to be transferred.

These commands can be executed from the command line, or by using the **Host -> TSHS** Hosting Administration Console page.

Note: If you wish to change S3 parameters, then first set `S3SyncActive` to `False`, then change the parameters and set `S3SyncActive` back to `True`. Note that if the S3 parameters are changed, data is already stored in S3 will continue to use the old parameters! If you wish to change the S3 parameters of existing S3 data, then you must use the `tshs update-s3-host` command.

8.9 Running Maintenance Tasks

The following maintenance tasks must be run periodically:

copy-shards If a Storage Node has been split, this task copies files from the old storage node to the new node, which contains the new shard.

remove-deleted This task removes files from the cluster that have been marked for deletion. This includes removing files from S3 storage.

In general, the TeamDrive Scalable Hosting Storage does not delete files in the cluster immediately, but rather marks them for deletion. This can also take the form of a DELETE job, if an entire space is deleted. DELETE jobs that have been scheduled are displayed using `tshs list-jobs`. If a file is marked for deletion then this can been seen when you use the `tshs list-files` command.

perform-import This task imports existing file system based Hosting data into the TSHS cluster.

move-to-s3 This task moves files to S3 storage that have been scheduled for transfer. The TeamDrive Hosting Apache module schedules files to be moved to S3 storage, as soon as the writing of the file is concluded. Files marked to be transferred, and copied to S3 by the `move-to-s3` and then removed from the cluster.

copy-to-s3 This task performs a backup of files that need to remain in the TSHS cluster. Currently this is all files called `last.log`. The `last.log` files are constantly updated and may therefore never be moved to S3 permanently. Whenever a `last.log` file is modified it is marked to be copied to S3. In order to revent `last.log` files from being constantly copied to S3, the `copy-to-s3` task should run less frequently than the other task (the default is once every 4 hours).

You can run each maintenance task as a separate command, or you can use the `tshs do-tasks` command:

```
tshs do-tasks [all] [copy-shards] [remove-deleted] [perform-import]
[move-to-s3] [copy-to-s3] [all-not-s3-copy]
```

Generally it is recommended to use the `tshs do-tasks all` or `tshs do-tasks all-not-s3-copy` commands which will start all maintenance tasks at once. Use `all-not-s3-copy` , which omits the `copy-to-s3` task, to avoid too frequent backups. The advantage of using the `do-tasks all` is that new tasks that may be added to future versions of TSHS will automatically be executed.

In order to run maintenance tasks regularly you can start `tshs` in the daemon mode with the –start option, and use the `repeat` command to execute a command or list of commands separated by `and`.

By default, `tshs` is setup as a service to run as follows:

```
tshs repeat 60 do-tasks all-not-s3-copy repeat 14400 tshs do-tasks all
```

Which means that maintenance tasks except copy will be run once a minute (60 seconds), and the copy task which does a backup of the `last.log` files will be run every 4 hours (14400 seconds).

A number of options allow you to control how maintenance tasks are run:

--max-threads=<number> Specifies the maximum number of threads to be started by a command. By default, the maximum number of threads is 10. This is the maximum number of threads that will run, no matter how many tasks are started. Note that this does not include the threads used to upload files to S3 as specified using the `MaxUploadThreads` parameter.

--pause-on-error=<seconds> Specifies the time in seconds to pause after an error occurs during a maintenance task. The default is 2 minutes. When running in daemon mode, `tshs` ignores this option.

--retries=<number> The number of times to retry if an error occurs during a maintenance task. Set to 'unlimited' if you want tshs to retry indefinitely. The default is 0, which means that the task quits immediately if an error occurs. When running in daemon mode, `tshs` ignores this option.

--time-limit=<seconds> Specifies the time limit in seconds for running maintenances tasks. tshs will stop running the tasks after the specified time, whether the tasks are finished or not. The default is 0, which means unlimited. When a task is restarted it will continue from where it left off so you do not loose anything by stopping a task and restarting it again later. Use this parameter to prevent tasks from running during high load times when it could disturb normal operation.

--node-affinity=<node-id> This options tells `tshs` to only transfer data from a specific storage node. This optiona may be used if you have a `tshs` service running on each storage node. Such a configuration reduces the amount of network traffic when data is transfered between nodes, or to and from S3 storage.

These options may be specified on the command line, or placed in the `tshs` options file: `/etc/tshs.conf`, which is consulted by the `tshs` background service. When running in daemon mode, errors and other messages are written to the `tshs` log file, which is `/var/log/tshs.log` by default.

CHAPTER

NINE

UPGRADING THE TEAMDRIVE HOST SERVER

9.1 General Upgrade Notes

There are two basic approaches to updating a TeamDrive Host Server: **in-place**, by replacing the software with a newer version on the live system, or starting a **new instance and migrating the configuration** and data (MySQL Database and Space Volumes) to the new instance.

For older installations, performing a migration to a freshly installed instance might be the better approach, to get rid of accumulated "cruft" and to start from a clean slate. In case the current system is still running a 32-bit installation, moving to a 64-bit system is required, as newer versions of the Host Server **no longer support 32-bit environments**.

Updating requires a service interruption, as the Host Server components (e.g. the Apache http Server) need to be stopped while the update is in progress. Short downtimes usually pass unnoticed by the TeamDrive Clients, they will simply try again after a short waiting period. Local Client operations can continue.

The Host Server-specific MySQL Databases and Space Volumes are the two crucial pieces of data that need to be preserved during updates. Take backups prior to performing an update and *verify they worked correctly*. In case of an in-place upgrade, both the databases and Space Volumes can be taken over "as is". When performing a migration to a new instance, the databases and volumes need to be copied or moved to the new host.

Updates between different Host Server versions (e.g. from 3.0.011 to 3.0.013) usually require changes to the MySQL table structures. Starting with version 3.0.013, these changes are applied automatically when starting the service after updating. Reversing these changes (e.g. reverting to the previous version) requires going back to the previous backup, there is **no automatic roll-back of changes to the database/table structures**.

Starting with version 3.0.013, updates to a new build (e.g. from 3.0.013.0 to 3.0.013.1) can be performed using yum/RPM. Updating from older versions requires manual intervention, as the installations were performed without automatic package management.

9.2 In-place Upgrading Version 3.0.013 to a Newer Build

The use of RPM packages makes updating from one build to another (e.g. from 3.0.013.0 to 3.0.013.1) a fairly straightforward and automatic process.

Usually, you can simply replace the existing packages while the service is running. The update performs an immediate restart of the services (`httpd` and `td-hostserver` automatically):

```
[root@hostserver ~]# yum update td-hostserver
```

Check the chapter releasenotes-3.0.013 for the changes introduced in each build.

9.3 In-place Upgrading from Older Versions to 3.0.013

These instructions assume a default installation of the TeamDrive Host Server (version 3.0.009, 3.0.010 or 3.0.011) on RHEL6 or a derivative distribution like CentOS 6 (64-bit) that was set up based on the Host Server installation

instructions or using the TeamDrive Host Server Virtual Appliance for VMware.

Note: The following approach does not work on 32-bit systems and it does not apply to custom installations on other Linux distributions. If you performed an installation to different locations/directories, the process might work, but has not been tested/verified.

The overall procedure is similar in all cases — we'll remove the old software components while maintaining the MySQL databases and Space Volumes, install the current versions of the Host Server packages and and migrate a few configuration settings by performing the following steps:

- Stop the Apache http Server and PrimeBase processes (PBAC, PBAS)
- Perform a backup of the Host Server's MySQL Databases
- Remove the PrimeBase Application Environment and related files
- Remove old Apache modules
- Remove the `teamdrive` user and its home directory `/home/teamdrive`
- Install the new Host Server RPM packages `yvva` and `td-hostserver`
- Update the configuration files, remove the old configuration files after merging the settings
- Update the privileges for the `teamdrive` MySQL user, if required
- Start the TeamDrive Hosting Service and Apache http Server, check the log files for any errors
- Test the new setup with a local test client before allowing all user Clients to connect to the new instance again

The following paragraphs explain these steps in more detail.

9.3.1 Stop the TeamDrive Services

As a first step, the currently running TeamDrive Hosting Services need to be shut down.

Start by stopping the Apache http Server:

```
[root@hostserver ~]# service httpd stop
```

Next, stop the API, Admin Console and background tasks:

```
[root@hostserver ~]# pbctl stop
```

Use `pbctl status` to check that the services have been stopped (their `Status` needs to be `Stopped`) and `ps` or `pstree` to double check that there are no stray `httpd`, `pbeas`, `ase`, `pbas`, `pbac` or `smm` processes running. Use `kill <pid>` or `pkill <name>` to terminate these, if they don't disappear shortly after you issued the stop commands.

9.3.2 Create a MySQL Backup

After all TeamDrive Services have been stopped, you should now create a backup of the MySQL databases, e.g. using `mysqldump`:

```
[root@hostserver ~]# mysqldump -u root -p --force \
--databases hostapilog pbpg pbur pspace td2apilog \
| gzip > td-hostserver-mysql-$(date +%Y-%m-%d_%H.%M).sql.gz
```

Note: Older versions of of the Host Server (3.0.011.3 and older) used `td2apilog` instead of `hostapilog` and an additional `pbpg` database. The `--force` option in the example above ensures that the dump continues, even if some of the databases don't exist in your setup.

9.3.3 Backup the old Installation and Configuration Files

Next, create a backup the old PrimeBase Application Environment, Apache Modules and config files, if you don't have a full system backup already (e.g. a VM snapshot) that you could revert to in case of issues.

Note that some of these files might not exist on your local installation. The following sample shell script will skip these and add all existing ones to a backup tar archive named `td-hostserver-backup-YYYY-MM-DD.tar.gz` in the current directory:

```
#!/bin/sh
BACKUP="td-hostserver-backup-$(date +%Y-%m-%d).tar"

FILES="
/etc/httpd/conf.d/ssl.conf
/etc/httpd/conf.d/pbas.conf
/etc/httpd/conf.d/teamdrive.conf
/etc/httpd/conf/httpd.conf
/etc/httpd/modules/mod_pbas*.so
/etc/httpd/modules/mod_pspace*.so
/etc/httpd/myssl
/etc/init.d/primebase.boot
/etc/logrotate.d/teamdrive
/etc/primebase
/etc/profile.d/custom.csh
/etc/profile.d/custom.sh
/etc/profile.d/primebase.sh
/etc/profile.d/teamdrive.sh
/home/teamdrive
/usr/local/lib
/usr/local/lib64
/usr/local/primebase"
for a in $FILES
do
  if [ -e $a ]
  then
      tar rvf $BACKUP $a
  fi
done
gzip $BACKUP
```

9.3.4 Remove Obsolete Files and Binaries

Now, remove the files and binaries that are no longer required (except for some config files). Again, note that not all of these files might exist on your local installation:

```
[root@hostserver ~]# rm -rf \
/etc/httpd/conf.d/pbas.conf \
/etc/httpd/modules/mod_pbas*.so \
/etc/httpd/modules/mod_pbt.so \
/etc/httpd/modules/mod_pspace*.so \
/etc/init.d/primebase.boot \
/etc/logrotate.d/teamdrive \
/etc/primebase \
/etc/profile.d/custom.csh \
/etc/profile.d/custom.sh \
/etc/profile.d/primebase.sh \
/etc/profile.d/teamdrive.sh \
/home/teamdrive \
/usr/local/primebase
```

Finally, remove the `teamdrive` user (Host Server version 3.0.013 runs under the user `httpd` instead):

```
[root@hostserver ~]# userdel teamdrive
```

The system has now been prepared for installing the new version of the Host Server Software.

9.3.5 Install the new Host Server Software

Install the new Host Server components (`td-hostserver` and `yvva`) from the dedicated TeamDrive Host Server yum repository:

```
[root@hostserver ~]# wget -O /etc/yum.repos.d/td-hostserver.repo \
http://repo.teamdrive.net/td-hostserver.repo
[root@hostserver ~]# yum install td-hostserver
```

The new Host Server software version 3.0.013 uses a different configuration file layout. The required values need to be migrated manually from the previous configuration files.

Update the following configuration files with a text editor before deleting the old ones:

- Migrate the MySQL login credentials (especially `username` and `password`) from the `[p1db]` configuration group in `my.cnf` to the new configuration file `/etc/td-hostserver.my.cnf` and delete the `[p1db]` group from `my.cnf` when done. If your Host Server version was installed before September 2013, your `my.cnf` file likely does not include a `[p1db]` group. In this case, fill out the correct values like `username` and `password` in `/etc/td-hostserver.my.cnf` based on your installation notes. Also ensure that the `socket` path used in `td-hostserver.my.cnf` matches the one defined in the `[mysqld]` group in‘‘my.cnf‘‘ and adjust the path name, if necessary.
- Migrate the `RewriteRule` for the Storage Upgrade button from `/etc/httpd/conf.d/teamdrive.conf` to `/etc/httpd/conf.d/td-hostserver.httpd.conf` and remove `teamdrive.conf` afterwards.
- Edit `/etc/httpd/conf.d/ssl.conf`, ensure the the following rewrite rules exist (usually at the bottom of the file within the `VirtualHost` section and look as follows:

```
    RewriteEngine on
    RewriteLogLevel 0
    RewriteLog "/var/log/httpd/rewrite.log"
    RewriteRule ^/admin$ /admin/ [R]
    RewriteRule ^/admin(.*) /yvva/p1a$1 [PT]
    RewriteRule ^/pbas/p1_as/api/(.*)$ /yvva/api/$1 [PT]
```

  If the following rewrite rules already exist in this file, make sure to replace them with the rules above:

```
    RewriteRule ^/admin(.*) /pbas/p1_as/p1a$1 [PT]
    RewriteRule ^/depot(.*) /pbas/p1_as/p1p$1 [PT]
```

9.3.6 Check and Update the privileges for the `teamdrive` MySQL User

Starting with version 3.0.013, the `teamdrive` MySQL user needs privileges on the following MySQL databases: `hostapilog` and `pspace`. Make sure that these exist and add the missing ones, if required. The following example assumes a MySQL database running on `localhost`. Amend the commands accordingly, if you are using a remote MySQL database.

Log into the MySQL database as the `root` user and use the `SHOW GRANTS` statement to check the user's current privileges:

```
mysql> SHOW GRANTS FOR 'teamdrive'@'localhost';
+-----------------------------------------------------------------------+
| Grants for teamdrive@localhost                                        |
+-----------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'teamdrive'@'localhost' IDENTIFIED BY PASSWORD |
| '*0132BD69C7F09A437AF424D3B94A6A56C3AC4AC1'                           |
```

```
| GRANT ALL PRIVILEGES ON `pspace`.* TO 'teamdrive'@'localhost'        |
| GRANT ALL PRIVILEGES ON `pbpg`.* TO 'teamdrive'@'localhost'          |
| GRANT ALL PRIVILEGES ON `pbur`.* TO 'teamdrive'@'localhost'          |
| GRANT ALL PRIVILEGES ON `td2apilog`.* TO 'teamdrive'@'localhost'     |
+----------------------------------------------------------------------+
5 rows in set (0.00 sec)
```

In this case, the `teamdrive` user does not have privileges on the new `hostapilog` database. You can add these using the following command:

```
mysql> GRANT ALL PRIVILEGES ON `hostapilog`.* TO 'teamdrive'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS FOR 'teamdrive'@'localhost';
+----------------------------------------------------------------------+
| Grants for teamdrive@localhost                                       |
+----------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'teamdrive'@'localhost' IDENTIFIED BY PASSWORD |
| '*0132BD69C7F09A437AF424D3B94A6A56C3AC4AC1'                          |
| GRANT ALL PRIVILEGES ON `pspace`.* TO 'teamdrive'@'localhost'        |
| GRANT ALL PRIVILEGES ON `pbpg`.* TO 'teamdrive'@'localhost'          |
| GRANT ALL PRIVILEGES ON `pbur`.* TO 'teamdrive'@'localhost'          |
| GRANT ALL PRIVILEGES ON `td2apilog`.* TO 'teamdrive'@'localhost'     |
| GRANT ALL PRIVILEGES ON `hostapilog`.* TO 'teamdrive'@'localhost'    |
+----------------------------------------------------------------------+
6 rows in set (0.00 sec)
```

As a final step, the `hostapilog` database needs to be created:

```
mysql> CREATE DATABASE hostapilog;
Query OK, 1 row affected (0.00 sec)
```

We'll keep the other privileges for now, until we have started the new version of the Host Server and the table conversion has concluded. Once everything works, the privileges on the `pbpg`, `pbur` and `td2apilog` databases can be revoked.

9.3.7 Start the Host Server Components

Now start the TeamDrive Hosting Service:

```
[root@hostserver ~]# service td-hostserver start
Starting TeamDrive Hosting Services:                       [  OK  ]
```

Check the log file for any errors:

```
[root@hostserver ~]# less /var/log/td-hostserver.log
```

Example:

```
140613 15:03:31 [Note] yvvad startup
140613 15:03:31 [Note] Using config file: /etc/td-hosting.conf
140613 15:03:31 [Note] yvvad running in repeat 60 (seconds) mode
140613 15:03:31 [Note] *** Version 3.0.013: Adding VolGlobalID column to
Volume table
140613 15:03:31 [Note] ALTER TABLE pspace.Volume ADD COLUMN VolGlobalID
VARCHAR(40) CHARACTER SET ascii NULL AFTER Identifier;
140613 15:03:32 [Note] Setting VolGlobalID of Volume: vol01
140613 15:03:32 [Note] *** Version 3.0.013: Upgrading Session table
140613 15:03:32 [Note] DELETE FROM Session;
140613 15:03:32 [Note] ALTER TABLE pspace.Session ADD COLUMN VolGlobalID
VARCHAR(40) CHARACTER SET ascii NULL AFTER VolumeName;
140613 15:03:32 [Note] *** Version 3.0.013: Creating table AdminSession
140613 15:03:32 [Note] CREATE TABLE pspace.AdminSession (
```

```
140613 15:03:32 [Note]   ID             INT UNSIGNED    NOT NULL
AUTO_INCREMENT PRIMARY KEY,
140613 15:03:32 [Note]   Cookie         VARCHAR(100)    CHARACTER SET ascii
UNIQUE NOT NULL,
140613 15:03:32 [Note]   CreationTime   TIMESTAMP NOT NULL DEFAULT
CURRENT_TIMESTAMP,
140613 15:03:32 [Note]   AccessTime     TIMESTAMP NULL,
140613 15:03:32 [Note]   SessionData    MEDIUMBLOB
140613 15:03:32 [Note] ) ENGINE=InnoDB;
140613 15:03:32 [Note] *** Version 3.0.013: Dropping Setting.IndNameSetting
index
140613 15:03:32 [Note] ALTER TABLE pspace.Setting DROP INDEX IndNameSetting;
140613 15:03:32 [Note] *** Version 3.0.013: Dropping database pbur
140613 15:03:32 [Note] DROP DATABASE pbur;
140613 15:03:32 [Note] *** Version 3.0.013: Setting PathToSAKConverter to
"/opt/teamdrive/sakh/"
```

Note the automatic conversion of the MySQL databases to the 3.0.013 format and the changes to some configuration settings.

The number of changes to the table structures depends on the release date of the previous version. We have successfully tested this process on Host Server versions back to 3.0.009 (August 2012).

The amount of data stored in these tables affects how long the conversion will take.

After the table transformation has finished, the now obsolete MySQL databases have also been removed:

```
[root@hostserver ~]# mysql -u teamdrive -p -e "SHOW DATABASES;"
Enter password:
+--------------------+
| Database           |
+--------------------+
| information_schema |
| hostapilog         |
| pspace             |
+--------------------+
```

You can now revoke the no longer needed privileges on these databases.

Log in as the MySQL `root` user to perform these changes:

```
mysql> SHOW GRANTS FOR teamdrive@localhost;
+-----------------------------------------------------------------------------+
| Grants for teamdrive@localhost                                              |
+-----------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'teamdrive'@'localhost' IDENTIFIED BY PASSWORD        |
| '*0132BD69C7F09A437AF424D3B94A6A56C3AC4AC1'                                 |
| GRANT ALL PRIVILEGES ON `pspace`.* TO 'teamdrive'@'localhost'               |
| GRANT ALL PRIVILEGES ON `pbpg`.* TO 'teamdrive'@'localhost'                 |
| GRANT ALL PRIVILEGES ON `pbur`.* TO 'teamdrive'@'localhost'                 |
| GRANT ALL PRIVILEGES ON `td2apilog`.* TO 'teamdrive'@'localhost'            |
| GRANT ALL PRIVILEGES ON `hostapilog`.* TO 'teamdrive'@'localhost'           |
+-----------------------------------------------------------------------------+
6 rows in set (0.00 sec)

mysql> REVOKE ALL PRIVILEGES ON pbpg.* FROM 'teamdrive'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> REVOKE ALL PRIVILEGES ON pbur.* FROM 'teamdrive'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> REVOKE ALL PRIVILEGES ON td2apilog.* FROM 'teamdrive'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> show grants for teamdrive@localhost;
```

```
+-------------------------------------------------------------------------------+
| Grants for teamdrive@localhost                                                |
+-------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'teamdrive'@'localhost' IDENTIFIED BY PASSWORD          |
| '*0132BD69C7F09A437AF424D3B94A6A56C3AC4AC1'                                   |
| GRANT ALL PRIVILEGES ON `pspace`.* TO 'teamdrive'@'localhost'                 |
| GRANT ALL PRIVILEGES ON `hostapilog`.* TO 'teamdrive'@'localhost'             |
+-------------------------------------------------------------------------------+
3 rows in set (0.00 sec)
```

Next, start the Apache http Server:

```
[root@hostserver ~]# service httpd start
Starting httpd:                                            [  OK  ]
```

Check the log files for any errors:

```
[root@hostserver ~]# less /var/log/httpd/error_log
[root@hostserver ~]# less /var/log/mod_pspace.log
[root@hostserver ~]# less /var/log/mod_yvva.log
```

In case of any errors, check the chapter troubleshooting for guidance.

9.3.8 Log into the Administration Console

After the services have been started, try logging into the Administration Console and verify the settings.

Note that logging into the Administration Console is only possible for the `HostAdmin` user account. This user's password has been migrated from the `pbur` database automatically. If you used a different user name or don't recall the password you used, see chapter *Changing the Admin Console Password* (page 8) for details on how to reset it.

9.3.9 Enable the TeamDrive Hosting Service at System Boot

If the update was successful and the service is up and running, make sure it gets started automatically when the system reboots:

```
[root@hostserver ~]# chkconfig | grep td-hostserver
td-hostserver          0:off   1:off   2:off   3:off   4:off   5:off   6:off
[root@hostserver ~]# chkconfig td-hostserver on
[root@hostserver ~]# chkconfig | grep td-hostserver
td-hostserver          0:off   1:off   2:on    3:on    4:on    5:on    6:off
[root@hostserver ~]# chkconfig | grep httpd
httpd                  0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

9.4 Migrating an Older Host Server Version to a 3.0.013 Instance

Perform the installation of the 3.0.013 Host Server instance from scratch as outlined in the Installation Guide or start off booting and setting up the Host Server Virtual Appliance.

For the setup and initial installation, the new instance can be started with a different IP address and host name. However, before starting the services and making the new Host Server available to the TeamDrive clients again, it **must** be accessible under the original hostname, e.g. "hostserver.yourdomain.com". Make sure to update your DNS records accordingly (a change of IP address is fine).

Once the Software has been installed and configured, don't perform the steps to register the new Host Server with the Registration Server. Instead, first stop the Apache Server and Services on the old Host Server and make sure they are not yet running on the new instance, either. Copy over the MySQL databases from the old system, e.g. by

creating a `mysqldump` as outlined in chapter *Create a MySQL Backup* (page 42) and importing these databases into the new instance's MySQL Server.

Note: Please ensure that the MySQL import is performed into a "clean" database and there are no remains of a previous installation (e.g. when using the Host Server Virtual Appliance, which ships with pre-populated databases).

If the Host Server Database is located on an external MySQL host, it's sufficient to simply point the new Host Server instance to this server by providing the appropriate login credentials as explained in chapter *Changing the MySQL Database Connection Information* (page 9). (You likely have to create a new MySQL user account and grant privileges to allow incoming connections to these databases from the new Host Server).

Next, the existing Space Volumes need to be migrated to the new Host Server instance. In case you're using NFS, simply update `/etc/fstab` on the new Host Server and mount the file system. Double check that the `apache` user has write permissions on this mount point.

If you are using other means of storage that can be migrated from one host to another (e.g. iSCSI targets, virtual hard disk devices), take the appropriate actions to make them available to the new Host Server instance. Alternatively, create a backup of the Space Volumes and restore them on the new instance or use a tool like `rsync` to transfer the Space data files.

Note: Ensure that the Space Volumes are mounted on the same paths as they were on the previous Host Server instance! It is essential to preserve the directory structure and mount points, as previous Host Server versions included the Space Volume name in the Space URLs that are used by the TeamDrive Clients.

At this point, proceed with the update process as outlined in chapter *Check and Update the privileges for the teamdrive MySQL User* (page 44) and following. For performing the table conversions, the `teamdrive` MySQL user needs to to have privileges on both the existing as well as the new MySQL databases required for the TeamDrive Hosting Service. As a final step, make sure to update your DNS so the new Host Server is reachable under the old host name.

CHAPTER

TEN

TROUBLESHOOTING

10.1 List of relevant configuration files

/etc/httpd/conf.d/td-hostserver.httpd.conf: The configuration file that loads and enables the TeamDrive Host Server-specific modules for the the Apache http Server:

- `mod_pspace.so`: this Apache module provides the actual Host Server functionality by accepting incoming data from the TeamDrive clients as well as delivering data to other clients upon request.
- `mod_yvva.so`: this Apache module is responsible for providing the web-based Host Server Administration Console as well as the Host Server API interface.

/etc/logrotate.d/td-hostserver: This file configures how the log files belonging to the TeamDrive Host Service are being rotated. See the `logrotate(8)` manual page for details.

/etc/td-hosting.conf: This file defines how the `td-hostserver` background service is started using the `yvvad` daemon.

/etc/td-hostserver.my.cnf: This configuration file defines the MySQL credentials used to access the `pspace` MySQL database. It is read by the Apache modules `mod_yvva` and `mod_pspace` as well as the `yvvad` daemon that runs the `td-hostserver` background tasks and the `yvva` command line client.

/etc/yvva.conf: This configuration file contains configuration settings specific to the Yvva Runtime Environment that are shared by all Yvva components, namely the `mod_yyva` Apache module, the `yvvad` daemon and the `yvva` command line shell.

/etc/tshs.conf: This configuration file defines a number of maintenance tasks performed by the `tshs` background service.

10.2 List of relevant log files

In order to debug and analyse problems with the Host Server configuration, there are several log files that you should consult:

/var/log/mod_yvva.log: The log file for the Yvva Application Server module which provides the web-based Host Server Administration Console and API. Consult this log file when you have issues with associating the Host Server with the Registration Server, errors when issuing API requests or problems with the Administration Console. You can increase the amount of logging by changing the Yvva setting `log-level` from `error` to `trace` or `debug` in `/etc/httpd/conf.d/td-hostserver.httpd.conf`:

```
<Location /yvva>
  SetHandler yvva-handler
  YvvaSet root-path=/opt/teamdrive/hostserver
  YvvaSet mysql-cnf-file=/etc/td-hostserver.my.cnf
  YvvaSet log-level=error
</Location>
```

After changing these values, you need to restart the Apache http Server service using `service httpd restart`.

/var/log/td-hostserver.log: The log file for the `td-hostserver` background task. Check this one to verify that background tasks are being processed without errors. The log file location can be configured by changing the file name passed to the `log-file` option in the configuration file `/etc/td-hosting.conf`. The log level can be increased by changing the default value `error` for the `log-level` option to `trace` or `debug`. Changing these values requires a restart of the `td-hostserver` background process using `service td-hostserver restart`.

/var/log/mod_pspace.log: This log file contains error messages related to the `mod_pspace` Apache module, particularly when using TSHS. It needs to be writable by the user that the Apache http Server runs under (`apache` by default). The log file location is configured by the server setting `ModuleLogFile` and the amount of logging can be changed by adjusting the server setting `ModuleLogLevel` via the Host Server Administration Console. The value defines the maximum level of logging of messages logged: 1 = Protocol, 2 = Error, 3 = Warning, 4 = Trace, 5 = Debug. Changing these value requires restarting the Apache http Server.

/var/log/httpd/: The Apache httpd Server's log files (e.g. `error_log`) might also contain additional relevant error messages (e.g. from `mod_pspace`) that should be checked. The amount of logging is affected by the `ModuleLogLevel` setting described above.

/var/log/tshs.log: This log file contains errors and other messages generated by the `tshs` background service. The log file location and amount of output are defined in file `/etc/tshs.conf`, via the options `log-file` and `log-level`. Possible values in the order of verbosity are `protocol`, `error`, `warning`, `trace`, `debug`. The default is `warning`.

/var/log/s3d.log: This log file is written by the TeamDrive S3 daemon `s3d` and provides log messages and errors specific to the `s3d` background service. The log file location is defined in the init script `/etc/init.d/s3d`.

10.3 Tracing Client Accesses to a Single Space

For debugging issues with a specific Space, it might be useful to enable more verbose tracing of activity between the Host Server and the TeamDrive Clients accessing this Space.

For this purpose, access to that Space can be traced by providing the Space's ID to the option `watched_space_id` in `/etc/httpd/conf.d/td-hostserver.httpd.conf` as follows:

```
<Location /primespace>
      SetHandler pspace-handler
      MySQLCnf /etc/td-hostserver.my.cnf

      watched_space_id <space ID>

      # Necessary to ignore the extra Range-header
      # (see Range-header note in the documentation)
      RequestHeader unset Range
</Location>
```

Restart the Apache HTTP Server with `service httpd restart`. Any activity on the selected Space will now be logged into the log file `/var/log/mod_pspace.log`.

Note: Remove this option and restart the Apache HTTP Server once you've finished analyzing the problem, to avoid uncontrolled growth of the log file.

10.4 Common errors

10.4.1 Errors When Registering the Host Server

If the Host Server Registration fails, check `/var/log/mod_yvva.log` on the Host Server as well as `/var/log/pbt_mod.trace` on the Registration Server for hints. See the Troubleshooting chapter in the Registration Server Installation Manual for details.

10.4.2 MySQL Errors When Upgrading From an Older Host Server Version

If you observe `Access denied` or `Unknown database` errors from the MySQL server like the following ones after starting the updated TeamDrive Host Server using an older MySQL table structure:

```
140618 10:56:37 [Note] DROP DATABASE pbpg;
140618 10:56:37 [Error] -12036 (1044): Access denied for user
'teamdrive'@'localhost' to database 'hostapilog'
140618 10:56:37 [Error] "p1setup.pbt" P1Setup:upgradeSettings(328)
140618 10:56:37 [Error] "p1setup.pbt" P1Setup:setupDatabase(14)
140618 10:56:37 [Error] "p1setup.pbt" (506)
```

`Unknown database`:

```
140618 10:24:31 [Error] -12036 (1049): Unknown database 'hostapilog'
140618 10:24:31 [Error] "p1setup.pbt" P1Setup:upgradeSettings(328)
140618 10:24:31 [Error] "p1setup.pbt" P1Setup:setupDatabase(14)
140618 10:24:31 [Error] "p1setup.pbt" (506)
140618 10:24:31 [Error] "p1_shared.pbt" (2)
```

Double check that the `hostapilog` database actually exists and that the `teamdrive` user has the required privileges to access it.

Create the database using `CREATE DATABASE hostapilog;` and grant the required privileges using `GRANT ALL PRIVILEGES ON 'hostapilog'.* TO 'teamdrive'@'localhost';`. Restart the TeamDrive Service again using `service td-hostserver restart`, it should now conclude the schema conversion.

If you observe a `Can't connect to local MySQL server` error like the following one in `/var/log/httpd/error_log`:

```
[Thu Jun 19 16:20:50 2014] [notice] mod_pspace 1.5.04 Loaded; Build Jun 19
2014 13:24:58;Crash-Reporting-Disabled
[Thu Jun 19 16:20:50 2014] [error] Failed to boot Admin API: MySQL 2002:
Can't connect to local MySQL server through socket
'/var/lib/mysql/mysql.sock' (2)
```

or in `/var/log/td-hostserver.log`:

```
140619 16:42:06 [Error] -12036 (2002): Can't connect to local MySQL server
through socket '/var/lib/mysql/mysql.sock' (2)
```

Double check that the MySQL Server is up and running and that the `socket` configuration setting in the `[mysqld]` group in `/etc/my.cnf` matches the one in `/etc/td-hostserver.my.cnf`.

The default value is `/var/lib/mysql/mysql.sock`. If the value in `my.cnf` is different, e.g. `/tmp/mysql.sock`, we suggest to revert back to the default value there instead of changing it in `td-hostserver.my.cnf` (unless you have an explicit reason to change the default socket path, of course). Restart MySQL and the TeamDrive Hosting Services after changing this value.

10.4.3 Admin Console: Clicking on "Host" Results in a "500 Internal Server Error"

If you observe an error message like the following when clicking on **Host** in the Host Server Administration Console:

```
500 Internal Server Error
ERROR -1: TshsMain: void CSDBConn::connect(CSDB.cc:1116) MySQL 1044: Access
denied for user 'teamdrive'@'localhost' to database 'tshs_admin'
```

Or:

```
500 Internal Server Error
ERROR -1: TshsMain: void CSDBConn::connect(CSDB.cc:1116) MySQL 1049: Unknown
database 'tshs_admin'
```

You likely changed the setting `TSHSEnabled` to `True`, but did not configure the MySQL settings for accessing the `tshs_admin` database in `/etc/td-hostserver.my.cnf`.

If you changed the setting by accident, simply set `TSHSEnabled` back to `False`.

Otherwise, consult the chapter *TeamDrive Scalable Hosting Storage* in the Team Drive Host Server Administration Guide for details on how to enable and configure TSHS properly.

10.4.4 "Duplicate key" MySQL errors when updating the database

If you observe "Duplicate key" errors in the `Traffic` or `Owner` tables when upgrading these to the latest schema version, you first need to manually remove the duplicates via the MySQL client or another tool like MySQL Workbench. Older versions of the Host Server database schema did not have UNIQUE constraints on some columns, which caused the creation of duplicate entries. For the `Traffic` table, this usually only affects older traffic accounting information that can safely be removed.

Duplicates in the `Owner` table are likely caused by user names or email addresses that refer to the same user account, but using different capitalization. In this case it helps to cross-reference the affected users with their information in the Registration Server Database - likely one of these accounts has not been actively used and can be deleted. Please contact support@teamdrive.net if you need assistance in resolving these conflicts.

10.4.5 Admin API Error: MySQL 1040: Too many connections

On a busy server, you might observe one of the following error messages in the Apache http Server's error log file from time to time:

```
[error] Failed to boot Admin API: MySQL 1040: Too many connections
[error] [client xxx.xxx.xxx.xxx] (500)Unknown error 500: Admin API Error:
MySQL 1040: Too many connections
```

In `/var/log/mod_yvva.log` you might observe a similar error:

```
141107 12:11:44 [Error] -12036 (1040): Too many connections
141107 12:11:44 [Error] "startup.yv" (80)
```

This error indicates that the number of child processes spawned by the Apache http Server (e.g. when many TeamDrive Clients attempt to connect to the Host Server concurrently), causes the MySQL Server to run out of threads for handling the incoming database connections.

By default, the MySQL Server is configured to accept 151 concurrent connections. Each Apache child process can establish up to two MySQL connections (one for `mod_pspace` and one for `mod_yvva`, depending on what kind of requests it needs to serve). Therefore, the maximum number of connections should be adjusted to be at least 1.5 times the maximum number of child processes spawned by the Apache http Server (defined by the `MaxClients` directive in the Apache http Server configuration file `/etc/httpd/conf/httpd.conf`).

The value can be changed by adding the system variable `max_connections` to the `[mysqld]` configuration group in the MySQL Server configuration file `/etc/my.cnf`, e.g.:

```
[mysqld]
datadir=/var/lib/mysql
max_allowed_packet=4M
max_connections=350
socket=/var/lib/mysql/mysql.sock
user=mysql
```

You need to either restart the MySQL server in order to apply this change, or change the value at run-time, by running the following SQL statement as the MySQL root user:

```
mysql> SET GLOBAL max_connections=350;
```

Keep in mind that increasing the maximum number of connections also increases the memory requirements of the MySQL Server. For more details, please consult the MySQL Server and Apache http Server documentation:

https://dev.mysql.com/doc/refman/5.6/en/too-many-connections.html

https://httpd.apache.org/docs/2.2/mod/mpm_common.html#maxclients

http://fuscata.com/kb/set-maxclients-apache-prefork

CHAPTER

ELEVEN

RELEASE NOTES - VERSION 3.0.013

Host Server Version 3.0.013 is the next major release following after version 3.0.011 (Version 3.0.012 was an internal release that has not been published).

Version 3.0.013 contains the following features and notable differences to version 3.0.011:

- The TeamDrive Host Server installation can now be performed via RPM on Red Hat Enterprise Linux 6 and derivative distributions, which significantly improves the installation procedure and the process of applying updates.
- The initial setup and registration of a Host Server is now fully web-based. It's no longer necessary to provide a `hosting.txt` or `properties` file. Instead, all the required information can be entered in a web form.
- The entire Host Server configuration is now stored in the MySQL database. This includes configuration settings for S3 daemon and TSHS.
- The web-based TeamDrive Hosting Service Administration Console has been improved significantly, by simplifying the work flows for common administration tasks and fixing several usability issues.
- TSHS, the TeamDrive Scalable Hosting Storage and the TeamDrive S3 Daemon provide additional scalability options to expand the storage capabilities of a TeamDrive Hosting Service.
- It's now possible to generate a monthly report that contains detailed statistics about all existing Depots and Spaces within these depots, including the monthly traffic and disk usage.
- The Host Server no longer depends on the PrimeBase Application Environment. Instead, it now uses the Yvva Runtime Environment, which replaces the following components:
  - `mod_yvva` replaces `mod_pbas` for providing the web-based Administration Console and API. The stand-alone `pbas` instance is no longer required. As a consequence, the `pbur` MySQL database which was used by PBAS to manage user accounts and privileges is no longer required and has been removed.
  - `yvvad` replaces `pbac` for running background tasks. The former `p1_autotask` background task PBAC instance is now provided by the service `td-hostserver`, which uses `yvvad`.
  - `yvva` replaces `pbac` for command line operations that involve executing PBT code on the shell.
- The installation location of the TeamDrive PBT code has been changed from `/home/teamdrive/pbas` to `/opt/teamdrive/hostserver/`.
- The `sakgen` binary that used to be installed in `/home/teamdrive/sakh` is no longer required. Instead, the functionality to encrypt Space Depot access keys is now provided by the `tshs` binary.
- All TeamDrive Host Server processes now run under the user ID used by the Apache http Server (`apache`). A dedicated `teamdrive` user account is no longer required.
- By default, the MySQL databases are now installed in the default location `/var/lib/mysql` instead of `/spacedb`, which made it difficult to enable SELinux on the MySQL instance.
- For security reasons, the MySQL credentials required for accessing the MySQL Database are no longer stored in the default MySQL configuration file `/etc/my.cnf`. Instead, the `[p1db]` options group has now been moved into a dedicated configuration file `/etc/td-hostserver.my.cnf`, only readable by the `apache` user.

- The Apache httpd Server configuration file has been renamed from `teamdrive.conf` to `td-hostserver.httpd.conf`.
- The overall robustness of the TeamDrive Host Server has been improved by issuing more meaningful error messages and performing more safety and consistency checks.
- Each Space Volume now contains a file `teamdrive-volume-id` that contains a unique global volume ID, to ensure that multiple volumes are mounted to the correct location.

11.1 Change Log - Version 3.0.013

11.1.1 3.0.013.13 (2015-05-11)

- mod_pspace/s3d: Added workaround to handle a deviation in the Ceph 0.8 Object Store S3 API: the "list multipart upload parts" API request returns `ListMultipartUploadResult` instead of `listpartsresult` (see BUG#11494 in the Ceph bug tracker for details). (HOSTSERVER-484)
- mod_pspace: Added missing call to `s3d_delete()` when an "Upload to file that has already been transfered to S3" is detected. Due to the missing call, Clients could end up in an endless loop, showing a "wrong md5" error in the log file. (TDCLIENT-2045)
- mod_pspace: Added new module option `watched_space_id` that can be used to trace Client accesses to a specific Space for debugging purposes. See tracing_client_accesses_to_a_single_space for details. (HOSTSERVER-486)

11.1.2 3.0.013.12 (2015-04-14)

- s3d: Uploading the `last.log` file failed with a checksum error if the log was written to before the upload was complete. `s3d` now only transfers the data size used when calculating the checksum. This will allow the `last.log` file to grow while being uploaded to S3. (HOSTSERVER-474)
- s3d: Fixed unsafe object references during multi-part uploads which may have lead to `s3d` crashes. (HOSTSERVER-454)
- Installation: The `td-hostserver` RPM package will no longer reset the permissions and ownerships of the `/spacedata` and `/spacedata/vol01` directories to `700` and `apache:apache` during an update, if they had been changed by the administrator after the initial installation. Depending on how the Space Volume is mounted, the RPM installation could fail with an error like `error: unpacking of archive failed on file /spacedata`. A new installation will still create the directories using these permissions/ownerships by default. (HOSTSERVER-401)
- Host Server: Converted the type of the `StatisticRest` setting from `INT` to `DATE`, to avoid an error that could occur when updating from very old Host Server Versions (the `resetTraffic()` auto task failed with an `Invalid integer literal` error). This also fixes a potential issue that could result in the reset routine being run multiple times on the day the traffic is reset. (HOSTSERVER-478)
- Documentation: Fixed link structure in the HTML documentation so that clicking **Next** and **Previous** within a document works as expected. (HOSTSERVER-471)

11.1.3 3.0.013.11 (2015-03-30)

- Administration Console: Updated logo and favicon.
- Host Server: Updated some error messages by replacing "Repository" with "Depot". Ensure that a Space Depot that has been marked as "Deleted" no longer allows the creation of new Spaces. (HOSTSERVER-456)
- mod_pspace: Reduced logging of errors by only logging Client accesses to deleted Spaces as an error if the Space status is zero. (HOSTSERVER-449)

- mod_pspace: Fixed a crashing bug that could occur in rare situations. (HOSTSERVER-457)
- s3d: Fix unsafe access to the thread pool that may have caused `s3d` to crash in certain situations. (HOSTSERVER-454)
- s3d: Fixed a problem that caused a crash if a multipart upload was interrupted before completion and then restarted again. The parts list could have holes in it for the parts that were successfully uploaded in the first try.
- Documentation: Added section that instructs the user to perform a `yum update` after installing the VM image. Reformatted the 3.0.013 release notes and replaced the table with regular sections for improved readability.
- Documentation: Added Failover and Scalability chapter to the Administration Guide, added description of the startup sequence/dependencies to the Installation Guides. (HOSTSERVER-431)

11.1.4 3.0.013.10 (2015-01-26)

- s3d: Fixed a problem that caused a crash from time to time. The crash would occur if a request for an object's header timed out or was interrupted.
- Host Server: Fixed bug in the calculation of `DiskUsed` for Space Volumes that did not contain any Spaces. (HOSTSERVER-452)
- Administration Console: The Volume repair button now only appears if a repair is actually required (previously it appeared whenever there was an error on the volume).
- Installation: added a new RPM package `td-hostserver-doc-html` that contains the Host Server documentation in HTML format, installed in the Host Server's Apache document root `/var/www/html/td-hostserver-doc/`. Access to the documentation can be restricted by editing `/etc/httpd/conf.d/td-hostserver-doc.httpd.conf`. (HOSTSERVER-450)
- Installation: fixed bug in upgrading from older versions and the `hostapilog` database did not get created. (HOSTSERVER-446)

11.1.5 3.0.013.9 (2015-01-14)

- mod_pspace/s3d: fixed `unexpected object "vol01/..." starting with 'vol' was found in the bucket...` error , which prevented the Apache module from starting. This error could occur after updating from a previous version if S3 was already enabled, and the old object format (prefixed by volume name) was used on an S3 compatible object store. (HOSTSERVER-447)

11.1.6 3.0.013.8 (2015-01-13)

- API: Added missing `activatedepot` API command and added new tag `<changeinfo>` to add a free form comment to the change history of the following API commands: `activatedepot`, `assignusertodepot`, `createdepotwithoutuser`, `deactivatedepot`, `deletedepot`. Updated API version to 3.0.004. (HOSTSERVER-337)
- Installation: fixed typo in the installation script that adds the RewriteRules to `ssl.conf`. Added RewriteRule in preparation for accepting Client requests for Space data via SSL/TLS (not supported yet).
- Installation: the binary tarball distribution now includes debug versions of the Host Server binaries (`s3d-debug` and `tshs-debug`) and Apache module (`mod_pspace-debug.so`, to better support analyzing possible crashing bugs. (HOSTSERVER-445)
- Installation: fixed possible upgrade error from previous versions: moving the MySQL table `pbpg.Keys` to the `pspace` database failed if an empty `pspace.Keys` table already existed. (HOSTSERVER-441)

11.1.7 3.0.013.7 (2014-12-12)

- Fixed error in creating an index during the initial MySQL table creation (HOSTSERVER-440)

11.1.8 3.0.013.6 (2014-12-09)

- Installation: fixed possible upgrade error from 3.0.011 when the MySQL database `pbpg` still existed, but the `Keys` table was already moved to the `pspace` database (HOSTSERVER-427)
- Fixed bug in which failed Auto Tasks were not executed anymore (HOSTSERVER-407)
- mod_pspace: fixed possible crash when system settings are NULL (e.g. in an upgrade scenario from 3.0.011 to 3.0.013, when httpd was started before yvvad performed the required schema updates)
- mod_pspace: Fixed possible "Admin API: AES decode error- curruption detected" error when updating from older versions (timing issues could result in the generation of duplicate private keys) (HOSTSERVER-420, HOSTSERVER-422)
- Increased the size of the `S3Options` settings field from 200 to 2000 chars, to accommodate longer option strings required for certain OpenStack environments (HOSTSERVER-425)
- Installation: updated `RewriteRule` sets in the httpd configuration files (removed obsolete `/depot` rule, HOSTSERVER-424)

11.1.9 3.0.013.5 (2014-09-26)

- mod_pspace: fixed a Space corruption bug that could occur when updating from a previous Host Server version to version 3.0.013 and Space Volumes were using a non-standard naming scheme (not "volxxx")
- Admin Console: added "Repair" button that allows performing an automatic repair of Volumes affected by the corruption bug. Clients will be notified to perform a Space Restore operation on affected Spaces.

11.1.10 3.0.013.4 (2014-09-18)

- Admin Console: fixed 404 errors when opening the Admin URL without a trailing slash (HOSTSERVER-398)
- Admin Console: the input focus is now automatically set to the password field (HOSTSERVER-392)
- s3d: Fixed bug in path deletion on S3: if the path ended with '/' it wasn't being deleted.
- s3d: exceptions are now logged in `/var/log/s3d.log`

11.1.11 3.0.013.3 (2014-09-05)

- mod_pspace: Replaced the previously used MD5 implementation with calls to the MD5 routines provided by OpenSSL (yielding a 70% performance improvement when calculating MD5 checksums on large files) (HOSTSERVER-355)
- mod_pspace: consolidated brand-specific settings into one place and disabled multi-part uploads for OpenStack
- mod_pspace: Fixed bug where failed uploads (resulting in MD5 checksum failures) would still be accounted for as bytes written in the Space usage statistics (HOSTSERVER-352)
- Fixed autotask `resetTraffic()` to properly reset the traffic for Spaces that had the `SPACE_TRAFFIC_FULL` status flag enabled. (HOSTSERVER-353)
- Installation: security enhancement: set `ServerTokens` to `Prod` and `ServerSignature` to `Off` in `httpd.conf` to disable displaying the Apache Server version and OS version in the HTTP headers and on error pages (HOSTSERVER-357)

- mod_pspace: Disabled unnecessary buffering of files fetched from S3 object store and passed back to the client. (HOSTSERVER-356)
- tshs: `add-s3-host` will ping the S3 service before actually adding the host details.
- Admin Console: security enhancement: don't display the version and build number on the login page and https redirection page (HOSTSERVER-359)
- Security enhancement: disabled unneded HTTP methods in `td-hostserver.httpd.conf` (ony allow GET, POST, PUT, disable HEAD, OPTIONS, TRACE) (HOSTSERVER-361)
- Virtual appliance security enhancement: set `ServerTokens` to `Prod` and `ServerSignature` to `Off` in `httpd.conf` to disable displaying the Apache Server version and OS version in the HTTP headers and on error pages (HOSTSERVER-357)

11.1.12 3.0.013.2 (2014-07-14)

- To avoid confusion, the S3-related configuration option `openStackAuthURL` was renamed to `openStackAuthPath`

11.1.13 3.0.013.1 (2014-07-11)

- Initial public release

CHAPTER

TWELVE

RELEASE NOTES - VERSION 3.0.011 AND OLDER

Table 12.1: Release Notes - Version 3.0.011 and older

Build Date	Version	Comment
YYYY-MM-DD	3.0.011.6	• HOSTSERVER-228: Add settings for `ClientPollFrequency` and `StatisticPollFactor` • HOSTSERVER-241: Moved `[p1db]` group from `my.cnf` to a dedicated configuration file `/etc/td-hostserver.my.cnf` to improve security and packaging.
2014-04-22	3.0.011.5	• HOSTSERVER-224: Added `SpaceStatisticEnabled` and `SpaceStatisticExportPath` • Updated `teamdrive.conf` Apache configuration file: wrapped long lines and updated `s3daemon` file locations to match the defaults suggested in the Installation Manual • HOSTSERVER-191: Fixed Magic Username problem with `sakgen` by enclosing them with single quotes to avoid the shell from expanding them as variables. Fixed "bad file descriptor error"
2014-03-12	3.0.011.4	• Fixed HOSTSERVER-99: created database migration script `mysql/v3.0.010_to_v3.0.011.sql` to update the table structures, move the Keys table from database `pbpg` to `pspace` and renamed database `td2apilog` to `hostapilog`. • Removed default `API_SALT` in sql script. • Improved `hosting.txt` value validation.
2014-03-03	3.0.011.3	• Updated version number in `pbstab` from "4546" to "4547" • Fixed HOSTSERVER-172: The default MySQL table definition file `mysql/p1space_schema.sql` contained a wrong value for the configuration variable `PathToSAKConverter`. Instead of `/home/teamdrive/sakh/sakgen` it should have been `/home/teamdrive/sakh/`.

Continued on next page

Table 12.1 – continued from previous page

Build Date	Version	Comment
2014-02-07	3.0.011.2	• Updated sample `hosting.txt` file: no trailing slash after `REGSERVERURL` • Updated and completed Translation files (grammar, typos, obsolete terms) • Set `PathToSAKConverter` configuration variable to `/home/teamdrive/sakh/sakgen` by default • Added S3Daemon config and script files to the installation package • Fixes to object store access log processing
2014-02-04	3.0.011.1	• Added parsing and error handling for `API_IP_LIST` and `API_SALT` from the `hosting.txt`. • pbstab: changed log file from `/home/teamdrive/pbas/setup/pbac.log` to `/var/log/p1_autotask.log` (Jira-Issue HOSTSERVER-145) • `pbstab`: fixed wrong path to `p1ctl.dal` • Fixed setting space status bit • Fixed autotask debug output • Fixed typos and obsolete reference to p1ctl from the translation files • Changed configuration variable 340 “Protocol Log File” in `pbas.env` from “<< Default Log >>” to “/var/log/pbas.log” - note that this file needs to be created and assigned to the user running the PBAS instance (`touch /var/log/pbas.log ; chown teamdrive:teamdrive /var/log/pbas.log`) • Fixed HOSTSERVER-150: removed reference to td2apilog database
2014-01-28	3.0.011.0	• First build of the 3.0.011 branch, using the scripted build
2012-08-22	3.0.009	• Fixed traffic LastReset bug
2012-08-03	3.0.008	• MySQL plugin with new reconnect; Fixed MySQL result set handling

CHAPTER

THIRTEEN

APPENDIX

13.1 Abbreviations

PBAC **P**rime **B**ase **A**utomation**C**lient

PBAS **P**rime **B**ase **A**pplication**S**erver

PBT **P**rime **B**ase **T**alk is an object oriented language specifically designed for the programming of "server-side" functionality common to intra- and internet Web sites. A large share of the TeamDrive Host and Registration Server functionality is implemented in PBT. The code is parsed and executed by the Yvva application server components.

SAKH **S**erver **A**ccess **K**ey **H**TTP for TeamDrive 2.0 Clients

TDES **T**eam **D**rive **E**nterprise **S**erver

TDNS **T**eam **D**rive **N**ame **S**ervice

TDRS **T**eam **D**rive **R**egistration **S**erver

TDSV Same as **SAKH**, but for TeamDrive 3.0 Clients: **T**eam **D**rive **S**er**v**er

TSHS **T**eam **D**rive **S**calable **H**osting **S**torage.