

Table of Contents

5 Bittorrent control option	3
BTRT - BitTorrent Proxy Tracker	3
Description	3
Hardware requirements	3
Software requirements	3
Use cases	3
Local retracking	3
"Adjacent" retrackers data using	3
Handling of requests to external trackers	4
Установка	4
Обновление	5
Конфигурация	5
Формат YAML	5
The main configuration file /etc/btrt/btrt.conf	6
Управление	7

5 Bittorrent control option

BTRT - BitTorrent Proxy Tracker

Description

BTRT is a BitTorrent tracker allowing to optimize the network traffic distribution in favor of reducing traffic to external networks by increasing the amount of intranet traffic. Reducing the network traffic volume to external networks is achieved by prioritization of peer IP addresses in response to *announce* requests: according to the [configuration](#) IP addresses of the operator's network peers are replied in the first place.

BTRT replies to standard BitTorrent client HTTP requests: *announce* and *scrape*, also it designed to handle non-standard *forwarded_announce* requests, more details on it see [here](#).

Hardware requirements

The main requirement is imposed to the RAM size. BTRT holds the data about all torrent files and peers and more. To calculate the required amount of RAM you can use as basis of 512 bytes per peer.

Software requirements

BTRT runs on CentOS 6. To install the BTRT the following packages should be installed on the system:

- `mysql-server`, version 5.1.73 or later
- `mysql-connector-c++` version 1.1.5 or later

Use cases

Local retracking

The simplest way to use BTRT is to use it as a local retracker. Several trackers that the BitTorrent client should access by sending *announce* requests can be specified in torrent-files. Special name `retracker.local` is used for local retracking. Local retracking is provided by forwarding requests to the `retracker.local` on BTRT.

"Adjacent" retrackers data using

Local retracking allows to redistribute traffic only within the operator's network and provides a

reduction in volumes of traffic to external networks only through operator's customers. The operator often has a contractual relationship with partners which implies a lower traffic cost in the networks of partners. If the tracker is launched in the partner's network, the use of peers registered on this server allows to reduce the traffic volumes in external networks. When processing *announce* requests BTRT enables to use not only local peers data but also to request the data from retrackers operating in partner networks. In response to *announce* request in this case BitTorrent client is sent the IP addresses of peers registered both in operator's network and in partner networks resulting in redistribution of traffic generated by BitTorrent clients.

Handling of requests to external trackers

BTRT handles the *forwarded_announce* requests generated by DPI platform. *Forwarded_announce* requests stand for *announce* requests sent to external BitTorrent trackers, which in turn are intercepted by DPI platform and forwarded to the BTRT.

When the *forwarded_announce* request is processed, BTRT, if necessary, sends the request to an external (third-party) tracker for the list of peers, but a list of peers is formed according to the peers priorities specified in the configuration (local peers, partner peers, etc.) and then the list is sent to the BitTorrent client. So having enough peers in operator and partner networks, it's possible to significantly limit the bandwidth when communicating with the external peers, so to redistribute the traffic in such a way that the intranet traffic will be increased.

Установка

Для первоначальной установки приложения необходимо выполнить установку пакета и зависимых модулей при помощи команды:

```
sudo yum install btrt
```

BTRT при работе использует БД MySQL. Для выполнения дальнейших действий по установке необходимо запустить сервис **mysqld**:

```
sudo service mysqld start
```

Перед первым запуском BTRT необходимо создать пользователя (или использовать существующего) и схему в БД MySQL. Рекомендуется создавать отдельного пользователя MySQL для работы BTRT. Чтобы сделать это, необходимо подключиться к серверу MySQL с правами суперпользователя:

```
mysql -hlocalhost -uroot -p
```

и выполнить команды создания пользователя и схемы, указав пароль пользователя:

```
CREATE USER 'btrt'@'localhost' IDENTIFIED BY 'password_for_btrt_user';  
CREATE DATABASE btrt;  
GRANT ALL PRIVILEGES ON btrt.* TO 'btrt'@'localhost';
```

После того, как пользователь будет создан, необходимо завершить привилегированный сеанс MySQL, нажав сочетание клавиш **Ctrl-D**. Создать объекты в схеме данных для BTRT, указав имя и пароль ранее созданного пользователя MySQL:

```
mysql -hlocalhost -ubtrt -p < /etc/btrt/btrt.db_creation.sql
```

Перед первым запуском необходимо создать файл конфигурации [/etc/btrt/btrt.conf](#) на основе шаблон конфигурации [/etc/btrt/btrt.conf.default](#). Более подробно о конфигурации написано [здесь](#).

Обновление

Для обновления ранее установленного приложения необходимо в первую очередь остановить сервис трекера:

```
sudo service btrt stop
```

установить обновленный пакет:

```
sudo yum update btrt
```

При обновлении с версии ниже 2.1 необходимо выполнить обновление структуры БД:

```
mysql -hlocalhost -ubtrt -p < /etc/btrt/btrt.db_upgrade_2.1.0.sql
```

Так как при переходе от версии к версии возможны изменения в [конфигурации](#) необходимо проверить ее корректность, а после запустить сервис трекера:

```
sudo service btrt start
```

Конфигурация

Основным файлом конфигурации является файл [/etc/btrt/btrt.conf](#). Конфигурация в этом файле задается в формате [YAML](#).

Формат YAML

Базовыми элементами формата [YAML](#) являются:

- пара "Ключ-Значение", например:
level: "info"
- последовательность значений, например:
 - "127.0.0.0/8"
 - "192.168.0.0/16"

На основе базовых элементов возможно создавать сложные структуры данных.

Одним из принципов для описания сложных структур заложенных в формат [YAML](#) является правило форматирования вложенных элементов при помощи отступов от начала строки. Элементы лежащие на одном уровне иерархии должны иметь одинаковое количество ведущих пробелов, например, следующий текст:

```
#
# Logging parameters
logging:
  path:          "/var/log/btrt"      # Path to log files, default:
/var/log/btrt
  level:         "info"              # Possible levels are:
                                     # critical, error, warning, info,
diagnostic, debug
                                     # default logging level is "info"
  switch_time:   "01:00:00"          # Log switching time in format
"HH:MM:SS", default: "01:00:00"
```

можно трактовать как структуру данных logging, у которой есть три поля - path, level, switch_time.

Учитывая выше сказанное, главным правилом при редактировании документов в формате [YAML](#) является **использование определенного количества пробельных символов для формирования отступов вместо символов табуляции.**

The main configuration file /etc/btrt/btrt.conf

Основной файл конфигурации содержит описание большинства параметров используемых при работе BTRT. Первоначально при установке пакета создается файл шаблона конфигурации [/etc/btrt/btrt.conf.default](#), который содержит описание всех параметров с комментариями и который может использоваться для создания реальной конфигурации.

Для создания рабочей конфигурации минимально необходимо выполнить модификацию следующих параметров:

Описание подключения к БД MySQL:

```
database:
  url:           "tcp://127.0.0.1:3306" # URL for connection to MySQL,
default: "tcp://127.0.0.1:3306"
  user:         "root"                 # User name, default: "root"
  password:     "root"                 # Password
  schema:       "btrt"                 # Schema name, default: "btrt"
```

Описание групп диапазонов IP адресов:

```
peer_groups:
  # It is useful to have definition for peers in local network.
  # It can be used when peer from the local network is registered on
```

```

non-local tracker (neighbor, forwarded tracker)
-   name:          "local peers"
    priority:      0
    cidr_file:     "/etc/btrt/local_cidrs.conf"
    cidr_list:
        -

    # neighboring peers is used to identifying range of IP-addresses of
    # peers connected to neighboring trackers
-   name:          "neighboring peers"
    priority:      1
    cidr_file:     "/etc/btrt/neighboring_cidrs.conf"
    cidr_list:
        -

```

Описание параметров "соседних" трекеров:

```

neighbors:
-   name:          "the nearest retracker"
    refresh_interval: 1800
    peers_expiry_interval: 7200
    announce_request:
"\"http://retracker.mgts.by/announce?info_hash={info_hash}&peer_id=0123456789012345678A&port=6882&downloaded=0&uploaded=0&left=4194304&event=started&compact=1&numwant=1000\""

```

Управление

Основной исполняемый файл трекера - /usr/sbin/btrt.

Для вывода встроенной подсказки необходимо выполнить следующую команду:

```
btrt help
```

Трекер разработан для работы в качестве сервиса. Для старта сервиса необходимо выполнить следующую команду:

```
sudo service btrt start
```

для остановки сервиса:

```
sudo service btrt stop
```

При работе трекера создаются журнальные файлы в соответствии с настройками в конфигурационном файле. По умолчанию журнальные файлы создаются в каталоге /var/log/btrt.