Block Exchange Protocol

Block Exchange Protocol

- Was ist BEP und wofür wird es benutzt?
- Quelle.
- Protokollerklärung.
  - Blockgröße
  - Vor-Authentifizierungs-Nachrichten
  - Post-Authentifizierungs-Nachrichten
- Nachrichtentypen (Message Types)
  - Cluster-Config
  - Index und Index-Update

Block Exchange Protocol

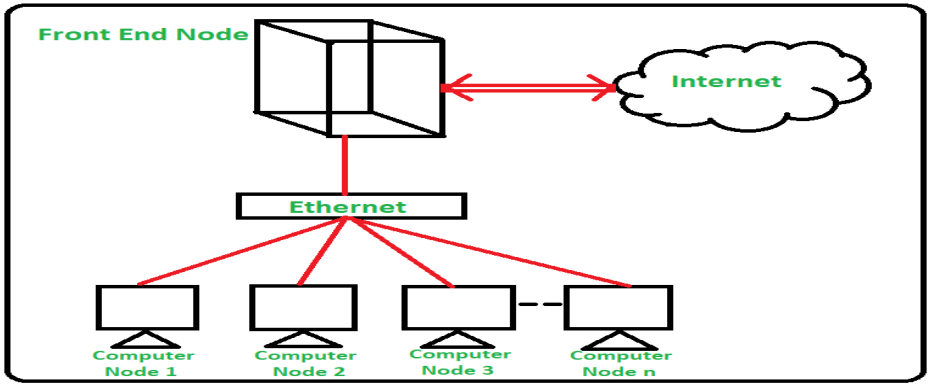
- Nachrichentypen (Message Types)
  - Request
  - Response
  - DownloadProgress
  - Ping
  - Close

Block Exchange Protocol

- ## Was ist BEP?

Das Block Exchange Protocol (BEP) wird zwischen zwei oder mehr Geräten verwendet, die einen Cluster bilden. Jedes Gerät verfügt über einen oder mehrere Ordner mit Dateien, die durch das lokale Modell beschrieben werden und Metadaten und Block-Hashes enthalten. Das lokale Modell wird an die anderen Geräte des Clusters gesendet. Die Vereinigung aller Dateien in den lokalen Modellen, wobei die Dateien mit der höchsten Änderungsversion ausgewählt werden, bildet das globale Modell. Jedes Gerät ist bestrebt, seine Ordner mit dem globalen Modell zu synchronisieren, indem es fehlende oder veraltete Blöcke von den anderen Geräten im Cluster anfordert.

Einfache Architektur eines Clusters:

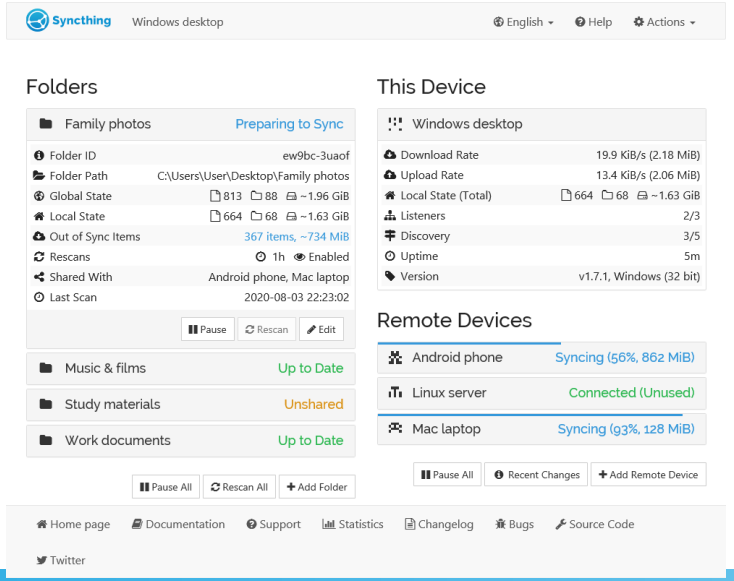


Block Exchange Protocol

- Quelle: Syncthing Docs.

Syncthing ist eine kostenlose Open-Source-Anwendung zur Peer-to-Peer-Dateisynchronisierung, die für Windows, macOS, Linux, Android, Solaris, Darwin und BSD verfügbar ist.[5] Sie kann Dateien zwischen Geräten in einem lokalen Netzwerk oder zwischen entfernten Geräten über das Internet synchronisieren. Datensicherheit und Datenschutz sind in das Design der Software integriert.

Syncthing GUI:



Block Exchange Protocol

- Protokollerklärung
  - Blockgröße

    Dateidaten werden in Einheiten von Blöcken beschrieben und übertragen, wobei jeder Block zwischen 128 KiB (131072 Bytes) und 16 MiB groß ist, in Zweierpotenzen. Die Blockgröße kann von Datei zu Datei variieren, ist aber in jeder Datei konstant, mit Ausnahme des letzten Blocks, der kleiner sein kann.
  - Vor-Authentifizierungs-Nachrichten

    NACH dem Aufbau einer Verbindung, aber VOR der Durchführung einer Authentifizierung, MÜSSEN die Geräte Hello-Nachrichten austauschen.

    Hello-Nachrichten werden verwendet, um zusätzliche Informationen über den Peer zu übermitteln, die für den Benutzer von Interesse sein können, auch wenn der Peer aufgrund einer fehlenden Authentifizierung nicht kommunizieren darf.

Block Exchange Protocol

- ## Hello-Nachrichten

```
message Hello {
    string device_name    = 1;
    string client_name    = 2;
    string client_version = 3;
}
```

Ein Beispiel für einen Client-Namen ist "syncthing" und eine Beispiel-Client-Version ist "v0.7.2".

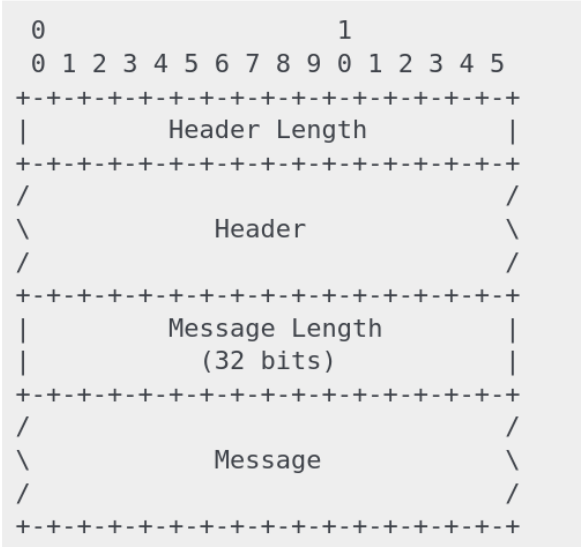
Der Gerätename ist ein vom Menschen lesbarer (konfigurierter oder automatisch erkannter) Gerätename oder Hostname für das entfernte Gerät.

Block Exchange Protocol

- Post-Authentifizierungs-Nachrichten

Header Length(16 bits)

Message Length(32 bits)



Block Exchange Protocol

- Nachricht-Header Schema

```
message Header {
    MessageType        type        = 1;
    MessageCompression compression = 2;
}

enum MessageType {
    CLUSTER_CONFIG    = 0;
    INDEX             = 1;
    INDEX_UPDATE      = 2;
    REQUEST           = 3;
    RESPONSE          = 4;
    DOWNLOAD_PROGRESS = 5;
    PING              = 6;
    CLOSE             = 7;
}

enum MessageCompression {
    NONE = 0;
    LZ4  = 1;
}
```

Block Exchange Protocol

- ## Cluster-Config:

  Eine Clusterkonfigurationsnachricht MUSS die erste Post-Authentifizierungsnachricht sein, die bei einer BEP-Verbindung gesendet wird. Weitere Cluster Config-Nachrichten MÜSSEN nach dem ersten Austausch NICHT gesendet werden.

Block Exchange Protocol

- Cluster-Config Schema

```
message ClusterConfig {
    repeated Folder folders = 1;
}

message Folder {
    string id                    = 1;
    string label                 = 2;
    bool   read_only             = 3;
    bool   ignore_permissions    = 4;
    bool   ignore_delete         = 5;
    bool   disable_temp_indexes = 6;
    bool   paused                = 7;

    repeated Device devices = 16;
}

message Device {
    bytes           id                          = 1;
    string          name                        = 2;
    repeated string addresses                   = 3;
    Compression     compression                 = 4;
    string          cert_name                   = 5;
    int64           max_sequence                = 6;
    bool            introducer                  = 7;
    uint64          index_id                    = 8;
    bool            skip_introduction_removals = 9;
    bytes           encryption_password_token  = 10;
}

enum Compression {
    METADATA = 0;
    NEVER    = 1;
    ALWAYS   = 2;
}
```

Block Exchange Protocol

- ## Index und Index-Update

  Die Index- und Index-Update-Nachrichten definieren den Inhalt des Absenderordners. Eine Indexnachricht stellt den gesamten Inhalt des Ordners dar und ersetzt somit jeden früheren Index. Eine Index-Aktualisierung ergänzt einen bestehenden Index um neue Informationen, wobei Einträge, die nicht in der Nachricht enthalten sind, nicht betroffen sind. Eine Indexaktualisierung DARF NICHT gesendet werden, wenn ihr kein Index vorausgeht, es sei denn, das Peer-Gerät hat für den betreffenden Ordner eine Max Sequence ungleich Null angekündigt.

Block Exchange Protocol

- Index und Index-Update Schema

```
message Index {
    string            folder = 1;
    repeated FileInfo files  = 2;
}

message IndexUpdate {
    string            folder = 1;
    repeated FileInfo files  = 2;
}

message FileInfo {
    string       name           = 1;
    FileInfoType type           = 2;
    int64        size           = 3;
    uint32       permissions    = 4;
    int64        modified_s     = 5;
    int32        modified_ns    = 11;
    uint64       modified_by    = 12;
    bool         deleted        = 6;
    bool         invalid        = 7;
    bool         no_permissions = 8;
    Vector       version        = 9;
    int64        sequence       = 10;
    int32        block_size     = 13;

    repeated BlockInfo Blocks         = 16;
    string             symlink_target = 17;
}
```

```
enum FileInfoType {
    FILE              = 0;
    DIRECTORY         = 1;
    SYMLINK_FILE      = 2 [deprecated = true];
    SYMLINK_DIRECTORY = 3 [deprecated = true];
    SYMLINK           = 4;
}

message BlockInfo {
    int64 offset     = 1;
    int32 size       = 2;
    bytes hash       = 3;
    uint32 weak_hash = 4;
}

message Vector {
    repeated Counter counters = 1;
}

message Counter {
    uint64 id    = 1;
    uint64 value = 2;
}
```

Block Exchange Protocol

- ## Request

  Die Request-Nachricht drückt den Wunsch aus, einen Datenblock zu erhalten, der einem Teil einer bestimmten Datei im Ordner des Peers entspricht.

- ## Request-Nachricht Schema

```
message Request {
    int32  id             = 1;
    string folder         = 2;
    string name           = 3;
    int64  offset         = 4;
    int32  size           = 5;
    bytes  hash           = 6;
    bool   from_temporary = 7;
}
```

Block Exchange Protocol

- Response

Die Response-Nachricht wird als Antwort auf eine Request-Nachricht gesendet.

- Response Protocol-Schema:

```
message Response {
    int32     id   = 1;
    bytes     data = 2;
    ErrorCode code = 3;
}

enum ErrorCode {
    NO_ERROR     = 0;
    GENERIC      = 1;
    NO_SUCH_FILE = 2;
    INVALID_FILE = 3;
}
```

Block Exchange Protocol

- ## DownloadProgress

  Die DownloadProgress-Nachricht wird verwendet, um entfernte Geräte über die teilweise Verfügbarkeit von Dateien zu informieren. Standardmäßig werden diese Nachrichten alle 5 Sekunden gesendet, und zwar nur in den Fällen, in denen ein Fortschritt oder eine Statusänderung festgestellt wurde. Jede DownloadProgress-Nachricht ist an einen bestimmten Ordner gerichtet und MUSS null oder mehr FileDownloadProgressUpdate-Nachrichten enthalten.

Block Exchange Protocol

- DownloadProgress Protocol Schema

```
message DownloadProgress {
    string                                folder  = 1;
    repeated FileDownloadProgressUpdate updates = 2;
}

message FileDownloadProgressUpdate {
    FileDownloadProgressUpdateType update_type   = 1;
    string                         name          = 2;
    Vector                         version       = 3;
    repeated int32                 block_indexes = 4;
}

enum FileDownloadProgressUpdateType {
    APPEND = 0;
    FORGET = 1;
}
```

Block Exchange Protocol

- PING

Die Ping-Nachricht wird verwendet, um festzustellen, ob eine Verbindung noch besteht, und um Verbindungen durch statusverfolgende Netzwerkelemente wie Firewalls und NAT-Gateways aufrechtzuerhalten. Eine Ping-Nachricht wird alle 90 Sekunden gesendet, wenn in den vorangegangenen 90 Sekunden keine andere Nachricht gesendet wurde.

Protocol Buffer Schema

```
message Ping {
}
```

Block Exchange Protocol

- ## Close

Die Close-Nachricht KANN gesendet werden, um anzuzeigen, dass die Verbindung aufgrund eines Fehlers abgebaut wird. Auf eine Close-Meldung MUSS KEINE weitere Meldung folgen.

Protocol Buffer Schema

```
message Close {
    string reason = 1;
}
```

Block Exchange Protocol

- Sharing Modes
  - Trusted:

Trusted mode is the default sharing mode. Updates are exchanged in both directions.

  - Send Only:

In send only mode, a device does not apply any updates from the cluster, but publishes changes of its local folder to the cluster as usual.

  - Recieve Only:

In receive only mode, a device does not send any updates to the cluster, but accepts changes to its local folder from the cluster as usual.

Block Exchange Protocol

- Example Exchange

Example Exchange

#	A	B
1	ClusterConfiguration->	<-ClusterConfiguration
2	Index->	<-Index
3	IndexUpdate->	<-IndexUpdate
4	IndexUpdate->	
5	Request->	
6	Request->	
7	Request->	
8	Request->	
9		<-Response
10		<-Response
11		<-Response
12		<-Response
13	Index Update->	
…		
14		<-Ping
15	Ping->	