Bareos version 17 and roadmap

bareos
open source data protection

- Maik Außendorf
- Philipp Storz

What's new in Bareos 17.2?

- Database optimizations
  - Great performance enhancements for large backups (filetable denormalization)
  - Most queries are now consolidated in own file
- NDMP 2-Way backup
- LanAddress feature
- Web UI with new features and more languages

Upcoming with first 17.2 maintenance release

- Amazon S3 backend connector
  - Sponsored by customer and running
  - Documentation and integration tests missing

filetable denormalization

- Problem: insert of large amount of files / filenames needs increasing amount of time
  - Normalized File table, filename in different table with ID reference
- Denormalized file table brings significant performance enhancements

filetable denormalization

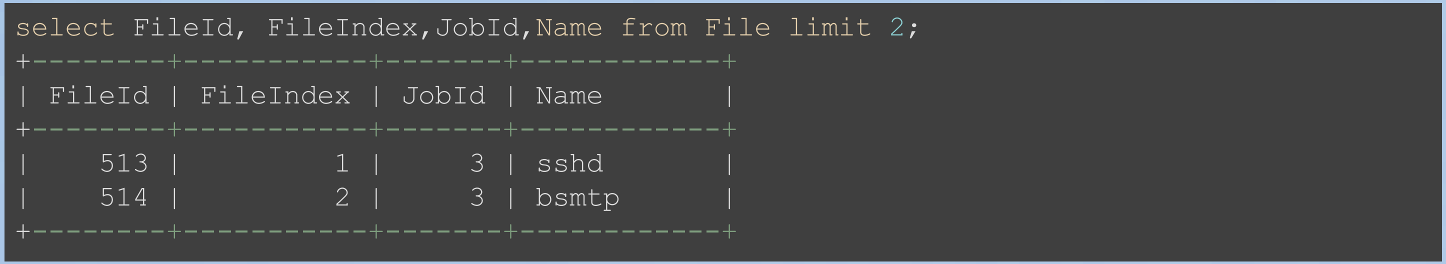
- Situation with Bareos < 17.2

```
select FileId, FileIndex,JobId,FilenameId from File limit 2;
+--------+-----------+-------+------------+
| FileId | FileIndex | JobId | FilenameId |
+--------+-----------+-------+------------+
|    513 |         1 |     3 |        218 |
|    514 |         2 |     3 |        246 |
+--------+-----------+-------+------------+
select * from Filename where FilenameId=218;
+------------+------+
| FilenameId | Name |
+------------+------+
|        218 | sshd |
+------------+------+
```

- Every File-Insert implies a FilenameId lookup, which consumes *O(log n)* time
- Steadily increasing with *n* number of filenames in catalog
- Updating filenames requires exclusive lock on filename table
- Backup too slow on systems with very high *n*

filetable denormalization

- Drop filename table and use filename instead in file table

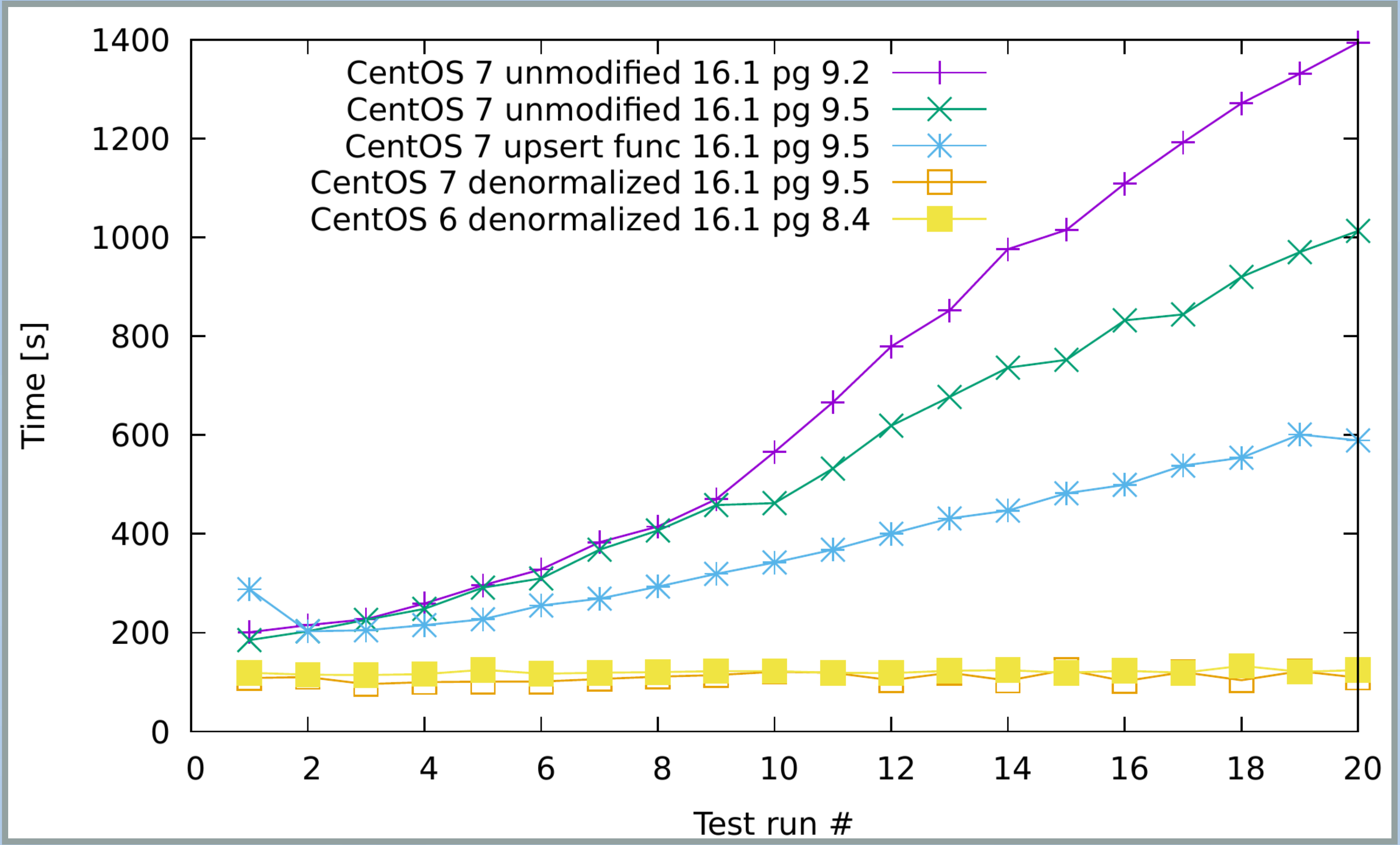


```
select FileId, FileIndex,JobId,Name from File limit 2;
+--------+-----------+-------+------------+
| FileId | FileIndex | JobId | Name       |
+--------+-----------+-------+------------+
|    513 |         1 |     3 | sshd       |
|    514 |         2 |     3 | bsmtp      |
+--------+-----------+-------+------------+
```

- Simple insert in file table without id lookup
- Supposed to require *O(1)* time - independent from number of filenames
- Cons: a little bit more space needed

filetable denormalization: evaluation

- Comparison of different database versions with / without denormalization on large sets of files (1 Mio files with distinct filenames)



filetable denormalization

- Situation now with Bareos >= 17.2
- Much faster *O(1)* aka constant time for inserts
- No table locks for filenames necessary
- Update process may take some time and needs some space

NDMP 2-Way backup

- What is NDMP?
- What is 2-way backup?

What is NDMP

- protocol for controlling backup and recovery
- Uses TCP/IP Port 10000
- supported by storage system vendors
- supported by backup applications

Elements of a NDMP System

- DMA
    - Data Management Application
- Data Agent
    - "Backup Client" running on primary storage
- Tape Agent
    - Handles read and write of data on a tape drive
- Robot Agent
    - Load and unload of Volumes, read barcode volume labels

How do the elements interact?

- DMA
  - controls the backup operation
- Data Agent
  - reads data from the primary storage system
  - sends the data to the Tape Agent
- Tape Agent
  - receives the data and stores it on tape.
  - Reports tape full and error conditions to DMA
- Robot Agent
  - handles media changer on request by DMA

NDMP 3-way Backup

- Data Agent and Tape Agent run separate
    1. Disk to Data Agent
    2. Data Agent to Tape Agent via Network
    3. Tape Agent to Tape

```
--+---------------- NETWORK ----+--------------------+----
  |    -->----->---->-->-->-->\ | //==>==>==>==>\\   |
  |   /                       | | ||      (2)   ||   |
  |   |                       | | ||            ||   |
/-----------\                 /-----------\    /-----------\
|           |                 |           |    |           |
|   DMA     |        DISK====>|   DATA    |    |   Tape    |====>TAPE DRIVE
|           |            (1) |   Agent   |    |   Agent   | (3)
\-----------/                 \-----------/    \-----------/
```

NDMP 2-way Backup

- Data Agent and Tape Agent are on same machine
  1. Disk to Data Agent
  2. Tape Agent to Tape
- No network involved

```
   --+--------------- NETWORK ----+--------
     |    -->---->---->-->-->-->\ |
     |   /                      | |
     |   |                      | |
/----------\                 /----------\
|          |                 | Data     |
|   DMA    |        DISK====>| Agent    |
|          |             (1) |          |
\----------/                 |    Tape  | (2)
                             |    Agent |====>TAPE DRIVE
                             \----------/
```

Properties of different NDMP Topologies:

- NDMP 3-way backup:
  - Data can be send over network to other location
  - Storage system does not need tape drive
  - Speed usually slower because of network transfer
- NDMP 2-way backup:
  - Data goes directly from system to tape
  - Fastest way to do a NDMP backup
  - storage system needs tape drive

What is DAR?

- DAR stands for Direct Access Recovery
- Data Agent sends File Information to the DMA.
- contains the FileName and File stat() info
- contains FileHandle Information
- FHInfo is the address in backup stream
- FHInfo used to directly access the right blocks

single file restore without DAR

- FileName List to restore is sent to Data Agent
- All data backed up is sent to Data Agent
- Data Agent scans **all data** and extracts only Files in list
  - *very slow*
  - *size of backup should not become too big*

single file restore with DAR

- FileName List with FHInfo is sent to Data Agent
- Data Agent **directly** forwards to blocks where Files are stored
- Data Agent extracts files
  - *very fast*

Directory DAR (DDAR)

optimization of DAR

- FileName List can become very big
- Handling of big list can slow down Data Agent
- DDAR selects only a directory
- Data Agent recovers directory and all content

NDMP in Bareos < 17

- special case of the 3-Way NDMP topology
- SD implements a Tape Agent with **infinitely long NDMP tape**
  - No volume handling needed so no robot agent needed
- Backup Stream is stored single Bareos File
- File Names link to this Bareos File
- Filehandle Info is not stored

This is still available and now referred as **NDMP_BAREOS**

Properties of NDMP_BAREOS

- No special NDMP media handling needed
- Data is stored into the same volumes as other Backup Data
- Copy and Migration is possible, as backup is a "normal Bareos Backup"
- File Name Information is stored in Bareos Database as link to Backup Image
- Recovery of single files is supported
- No support for direct access recovery
- Slow single file recovery

New Implementation is called NDMP_NATIVE

properties

- Bareos Director is only DMA
- Bareos Storage Daemon is not involved
- Director does the NDMP media handling and controls tape, robot and data agents
- Director stores FHInfo and FHNode for every file in the database
- Single file recovery with DAR is now supported

Agent Location NDMP_BAREOS NDMP_NATIVE

Comparison of Bareos NDMP options

Agent Location	**NDMP_BAREOS**	**NDMP_NATIVE**
Data Agent	external	external
DMA	Director	Director
Tape Agent	Storage Daemon	external
Robot Agent	None	external
Features		
backup to tape	Yes	Yes
backup to other SD device	**Yes**	**No**
2-way backup	**No**	**Yes**
3-way backup	Yes	Yes(untested)
single file restore	Yes	Yes
Direct Access Recovery	**No**	**Yes**
Directory DAR	**No**	**Yes**
Copy and Migration Jobs	**Yes**	**No**

LanAddress Feature

Problem: Some FD or SD are behind a NAT device

```
/-------------------\
|                   |     LAN 10.0.0.1/24
|  FD_LAN   SD_LAN  |
|  .10         .20  |
\___________________/
          |
      NAT Firewall
      FD: 8.8.8.10 -> 10.0.0.10
      SD: 8.8.8.20 -> 10.0.0.20
          |
/-------------------\
|                   |      WAN / Internet
|        DIR        |
|     8.8.8.100     |
|                   |
| FD_WAN   SD_WAN   |
| .30         .40   |
\___________________/
```

- The director can access the FD_LAN via the IP 8.8.8.10, which is forwarded to the IP 10.0.0.10 inside of the LAN.
- The director can access the SD_LAN via the IP 8.8.8.20 which is forwarded to the IP 10.0.0.20 inside of the LAN.

LanAddress Feature

Problem: Some FD or SD are behind a NAT device

```
/-------------------\
|                   |    LAN 10.0.0.1/24          Client {
|  FD_LAN   SD_LAN  |                                Name = FD_LAN
|  .10         .20  |                                Address = 8.8.8.10
\___________________/                             }
          |                                       Storage {
      NAT Firewall                                   Name = SD_LAN
      FD: 8.8.8.10 -> 10.0.0.10                      Address = 8.8.8.20
      SD: 8.8.8.20 -> 10.0.0.20                   }
          |                                       Client {
/-------------------\                                Name = FD_WAN
|                   |     WAN / Internet             Address = 8.8.8.30
|        DIR        |                             }
|     8.8.8.100     |                             Storage {
|                   |                                Name = SD_WAN
| FD_WAN   SD_WAN   |                                Address = 8.8.8.40
| .30         .40   |                             }
\___________________/
```

- There is also a FD and a SD outside of the LAN, which have the IPs 8.8.8.30 and 8.8.8.40
- All resources are configured so that the "Address" directive gets the Address where the Director can reach the daemons.

LanAddress Feature

Problem: Some FD or SD are behind a NAT device

```
/-------------------\
|                   |    LAN 10.0.0.1/24          Client {
|  FD_LAN   SD_LAN  |                                Name = FD_LAN
|  .10         .20  |                                Address = 8.8.8.10
\___________________/                             }
          |                                       Storage {
      NAT Firewall                                   Name = SD_LAN
      FD: 8.8.8.10 -> 10.0.0.10                      Address = 8.8.8.20
      SD: 8.8.8.20 -> 10.0.0.20                   }
          |                                       Client {
/-------------------\                                Name = FD_WAN
|                   |     WAN / Internet             Address = 8.8.8.30
|        DIR        |                             }
|     8.8.8.100     |                             Storage {
|                   |                                Name = SD_WAN
| FD_WAN   SD_WAN   |                                Address = 8.8.8.40
| .30         .40   |                             }
\___________________/
```

- Problem: Backup from FD_LAN to SD_LAN does not work, as FD_LAN tries to connect to 8.8.8.10 instead of 10.0.0.10

LanAddress Feature

Solution: LanAddress Directive

- Devices being in the LAN get additionally the LAN address configured in the "LanAddress" Directive

```
Client {
   Name = FD_LAN
   Address = 8.8.8.10
   LanAddress = 10.0.0.10
}
Storage {
   Name = SD_LAN
   Address = 8.8.8.20
   LanAddress = 10.0.0.20
}
Client {
   Name = FD_WAN
   Address = 8.8.8.30
}
Storage {
   Name = SD_WAN
   Address = 8.8.8.40
}
```

LanAddress Feature

How does LanAddress work?

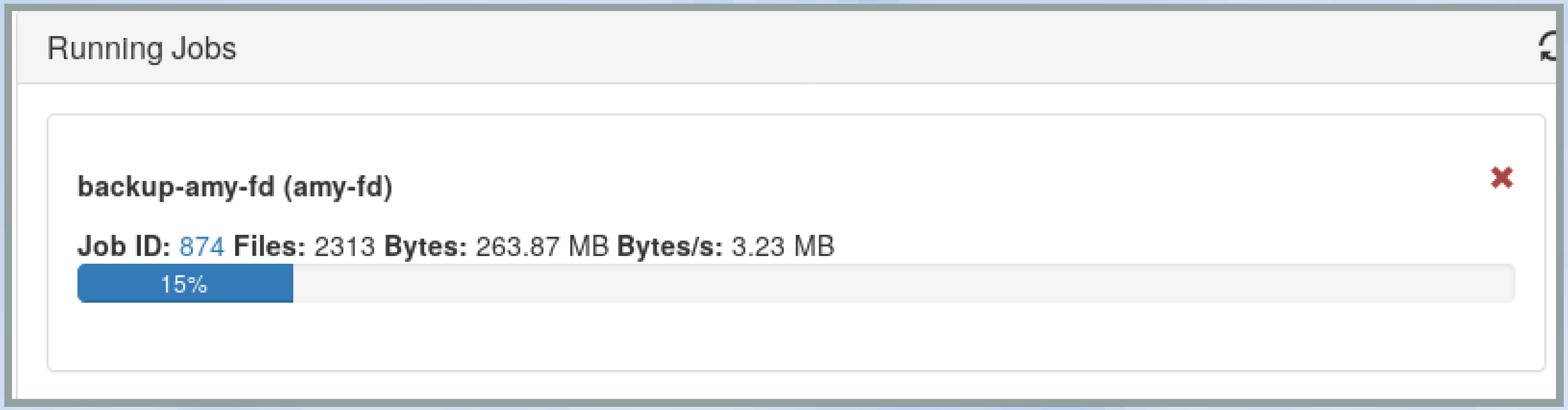
- The director simply checks, if both the involved client and storage both have a "LanAddress" configured.

  In that case, the initiating daemon is ordered to connect to the "LanAddress" instead of the "Address". (In active client mode, the FD connects to the SD, in passive client mode the SD connects to the FD).

  If only one or none of the involved client and storage have a LanAddress configured, the "Address" is used as connection target for the initiating daemon.
- Backups from each FD to each SD are possible.

Web UI: Running Jobs Widget

- Shows progress bar for each running job



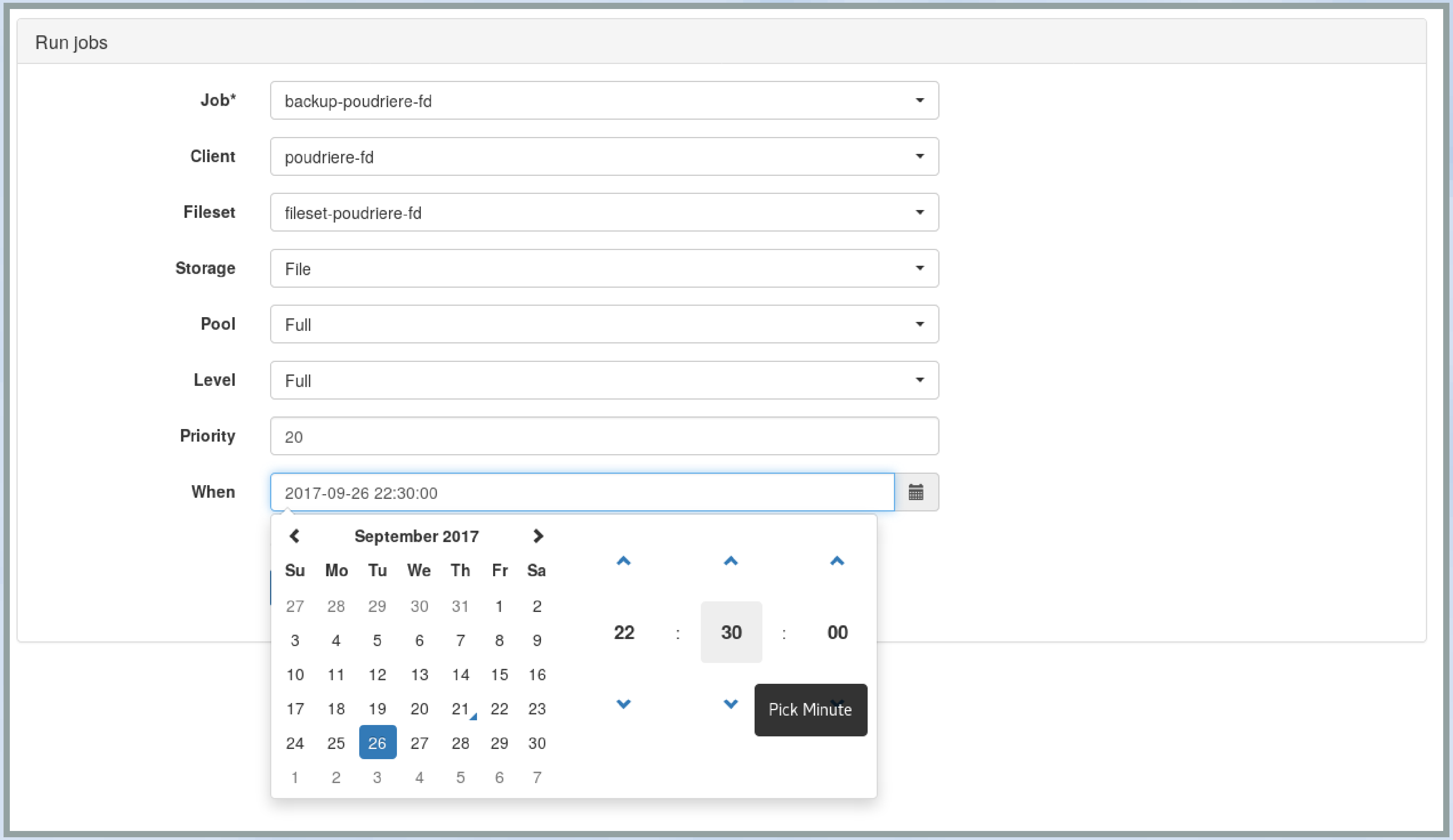
Running Jobs

backup-amy-fd (amy-fd)

Job ID: 874 **Files:** 2313 **Bytes:** 263.87 MB **Bytes/s:** 3.23 MB

15%

Web UI: Run customized jobs with option overrides


Run jobs

Job*	backup-poudriere-fd
Client	poudriere-fd
Fileset	fileset-poudriere-fd
Storage	File
Pool	Full
Level	Full
Priority	20
When	2017-09-26 22:30:00

Web UI: New languages added

- Slovak
- Turkish
- Czech

For completeness - already implemented before:

- Chinese
- Dutch
- English
- French
- German
- Italian
- Russian
- Spanish

Roadmap for 2018

- Encryption
  - Enable TLS between DIR-SD-FD with pre-shared-keys by default
  - Use passwords from configuration as default keys
  - Get encryption on the fly without any configuration hassle
- Web UI
  - Version browser
  - Better view of available backups for a client
  - Performance graphs
- Change the buildsystem from old-fashioned autotools to cmake
  - Easier and more flexible to handle
- Extended support for Unix binaries
  - Solaris, HP-UX, AIX, FreeBSD, MacOS - thanks to customer requests and support
- Increased subscription sales enable us to hire more staff for development and support