

## Research Article

# Design and Implementation of Xinjiang Astronomical Observatory Astronomical Data Transmission Visualization System

Xinchen Ye <sup>1</sup>, Hailong Zhang <sup>1,2</sup>, Yan Zhu,<sup>1,3</sup> Jie Wang,<sup>1</sup>  
Tohtonur Ergesh,<sup>1</sup> and Huijuan Li<sup>4</sup>

<sup>1</sup>Xinjiang Astronomical Observatory, Chinese Academy of Sciences, Urumqi 830011, China

<sup>2</sup>Key Laboratory of Radio Astronomy, Chinese Academy of Sciences, Nanjing 210008, China

<sup>3</sup>University of Chinese Academy of Sciences, Beijing 100049, China

<sup>4</sup>Xinjiang University of Finance and Economics, Urumqi 830012, China

Correspondence should be addressed to Hailong Zhang; [zhanghailong@xao.ac.cn](mailto:zhanghailong@xao.ac.cn)

Received 1 January 2019; Accepted 19 March 2019; Published 2 April 2019

Guest Editor: Junhui Fan

Copyright © 2019 Xinchen Ye et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the development of astronomical observation technology, astronomical devices produce more data than ever. Astronomical telescopes are usually far away from city, so the long-distance data transmission between telescope and data center faces great challenges. Visualization system of astronomical data transmission with four-layer structure was built to manage data transmission. This visualization system has a four-layer structure: hardware layer, system layer, middle layer, and visualization layer. System function includes automatic data transmission, log recording of transmission process, and display of the transmission status in dynamic web pages. Besides, the middle layer contains an alarm subsystem that can automatically send system exceptions to administrator. We also design corresponding mechanisms to ensure the high stability of the system and to control the data transmission when the network is unstable through adaptive algorithms. In test, this visualization system can run stably for a long time in unmanned manner. This system also provides a solution for the astronomical observation bases to automatically transmit data to the data center.

## 1. Introduction

With the development of astronomical observational technology, the data quality of telescope receiving equipment is improved. In the meantime, the volume of data generated by telescopes is increasing exponentially [1, 2]. For example, the world's largest fully steerable radio telescope GBT (Robert C. Byrd Green Bank Telescope)[3] generates more than 1.4PB data per year (<http://data.xao.ac.cn/static/GBTArchiveProcess.pdf>). The world's largest radio telescope FAST (five-hundred-meter aperture spherical radio telescope)[4]'s 19-beam receiver[5] produces  $8 \text{ bit} \times 10^4 \times 2 \times 4 \times 19$  data per second; more than 10PB of data will be stored per year. After SKA (Square Kilometre Array) [6, 7] is built, it is going to produce 1PB data per day [8].

Due to the specificity of astronomical observations, observatory sites are usually far from data centers. Data needs to be transmitted over a leased line from observatory site to data center owing to the instability in transmission in outdoor data lines over a long distance. Astronomical data transmission requires a complete management system [9] which meets the following conditions: complete logging, user-friendly visual interface for administrators to control data transfer process, high stability to guarantee system running in unattended state for a long time, and automatic sending of alert email to administrator when data transfer process fails.

NGAS (Next Generation Archive System) [10, 11] is the most commonly used archiving software in the field of radio

astronomy. NGAS is for astronomical data archiving, processing, searching, and synchronization. Nowadays, NGAS is used in data archiving of multiple telescopes. MWA (Murchison Widefield Array) [12] is a precursor for SKA which uses NGAS to synchronize data from Massachusetts Institute of Technology and Victoria University of Wellington. ALMA (Atacama Large Millimeter/submillimeter Array) [13] also uses NGAS for data synchronization [14, 15].

NGAS is already a relatively complete astronomical data archiving system. However, as NGAS is a software developed more than 10 years ago, there are also some problems [16].

- (1) NGAS uses HTTP-based methods to transmit data. It was uncertain whether the existing NGAS architecture scales up to cope with a larger amount of data.
- (2) Sometimes the dataflow may saturate the transmission bandwidth, and NGAS lacks an effective mechanism to solve this problem.
- (3) Users cannot intuitively understand the status of data transmission through NGAS.

This paper designs and develops an astronomical data visualization transmission system, which is based on the actual requirement of data transmission of Xinjiang Astronomical Observatory (XAO) of Chinese Academy of Sciences (CAS). This system contains functions including astronomical data transmission control, log recording during transmission, autoalarms, and visual interface. It is able to efficiently help administrators to control data transmission and it can run steadily for a long time unattended. The total transmission will be recorded in detail for later troubleshooting. The visual interface can show situation of data transmission intuitively. The adopted modular developing technique will make it easier in later transplanting to central controlling systems or large screen display.

## 2. System Architecture Design

XAO's Nanshan 26m Radio Telescope (NSRT) [17] is about 100km away from the data center of XAO; observation data need to be sent to the data center through a dedicated line every day. At present, there is no systematic management system for data transmission. The 110-meter radio telescope [18] to be built by XAO, Qitai, Xinjiang, will be the world's largest fully steerable radio telescope, and its data transmission line will exceed 200 km. Its data transmission process will be displayed in the large screen system in the future.

The system architecture was designed based on the actual needs of XAO. Astronomical data visualization transmission system adopts a four-layer architecture. The four layers are hardware layer, system layer, middle layer, and visualization layer. The system architecture diagram is shown in Figure 1.

- (1) The hardware layer provides a hardware environment for data transfer. The system design and development described in this paper are based on a test hardware environment.
- (2) The system layer includes a log subsystem and a data transfer subsystem. The log subsystem is used

to record the log of transfer processes and provides management program for administrator. The core of the data transfer subsystem is the rsync transport framework. The data transfer encapsulates the shell commands to call the rsync command.

- (3) The middle layer is mainly composed of control programs. These programs are responsible for controlling the subsystems of the system layer and managing the log files and database. The middle layer is also responsible for receiving instructions from and transferring data to the visualization layer. When the transmission process is abnormal, the alarm program will automatically respond and send an alert message to the administrator.
- (4) The visualization layer is developed based on web technology and the data transmission situation is visually displayed by visual charts. The system administrator can quickly grasp the data transmission situation information and quickly solve the problem.

The four-layer architecture adopted by the system meets the construction needs. In the development process, there are some problems in the original architecture design. In this paper, we modify the original architecture to get rid of these problems. The layered architecture design is convenient for development and management of this system. Problems in the system testing can be layered. At the same time, such layered architecture is easy for system reuse or porting in future.

## 3. System Function Realization

**3.1. Hardware Layer Test Environment.** We used three servers for building the hardware environment. The servers are interconnected through a Gigabit switch. The servers for data sending and receiving are both HP P4300 G2 data server with 2 Intel E5520 CPU, 20 GB RAM, and 6.4TB hard disk. Control server uses the DELL PowerEdge R710 with 2 Intel Xeon 5680 CPU, 32GB RAM, and 3.6TB hard disk.

Because the control server load is low, it is recommended to configure the control program on the nondedicated server to reduce the cost of equipment and energy in a real environment.

### 3.2. System Layer

**3.2.1. Log Subsystem.** The log subsystem includes log collection, log storage, log management, and management program. The log subsystem is an independent development module and it has a complete set of data processing flow. So it can be split and used separately. The log subsystem structure diagram is shown in Figure 2.

The log content stored in the database is designed mainly for the convenience of visualization layer invoking; it contains 6 database tables.

- (1) File table (files): it is used to record the specific information of each file.

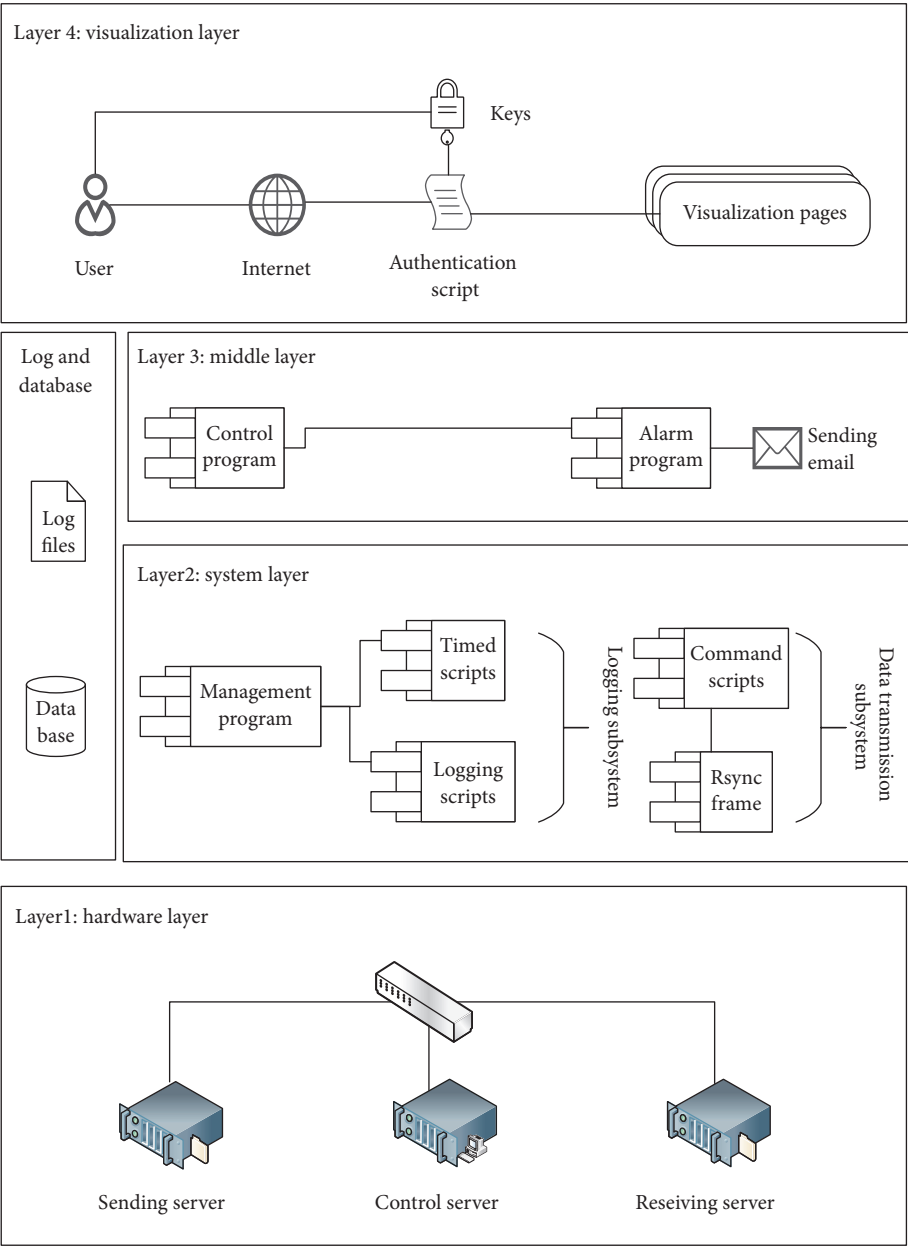


FIGURE 1: System architecture diagram.

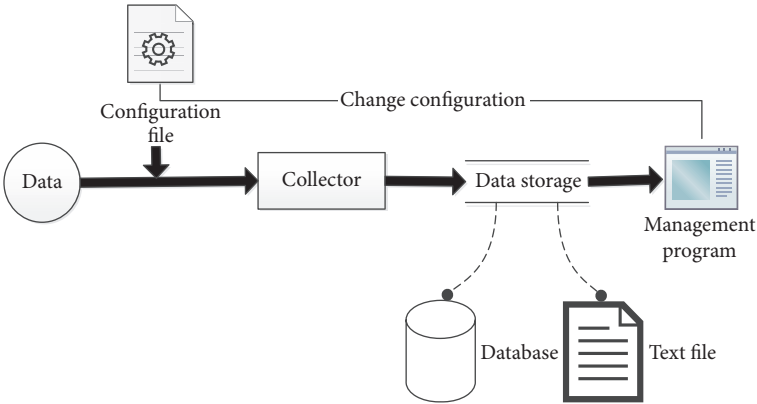


FIGURE 2: Log subsystem structure diagram.

- (2) Astronomical data table (data): it is for recording data storage information.
- (3) Folder table (folder): it is for recording information of subfolders of root directory.
- (4) Daily data delta table (dayData): it is for recording information of daily data increments.
- (5) Daily folder data delta table (dayFolderdata): it is for recording daily data increments of subfolders in root directory.
- (6) Scripts monitoring table (proc\_tatus): it is used to record the running of the scripts.

The information about specific field is shown in Table 1.

In the traditional log management system, the administrator's manipulation on the log file is usually performed in the form of command line, which is not convenient and not intuitive. Log query and control management interface developed in Qt creator [19] using Qt language facilitate the management of log system. Its functional structure diagram is shown in Figure 3. Through the management interface, log retrieval within a specified time range and various log queries can be realized; one-click backup of log files for a specified date range (3 months, half a year, one year) is also enabled. The log retrieval interface is shown in Figure 4.

**3.2.2. Data Transmission Subsystem.** The core of the logging subsystem is the remote synchronization tool rsync (Remote Sync). rsync is a mature mirror backup tool for Linux. It is used as a basic framework in a variety of data synchronization software [20]. Its main features are as follows.

- (1) Rsync can mirror the entire directory and file system, and its transfer process can maintain the permissions, time, soft connection, and other information of the original file.
- (2) Rsync supports incremental backup which can compress and decompress data in real time during transmission, so the transmission rate is faster. Besides, rsync can run on low bandwidth and high latency communication lines [21].

Rsync uses scp and ssh for data transmission. It will establish a virtual pipeline during transmission to ensure the security of data transmission. The rsync authentication process is shown in Figure 5.

The installation and configuration of rsync are more complicated. In addition to installation of the xinetd and rsync packages, it also needs to set configuration files and synchronization folder permissions and configure the system firewall. We have packed up the rsync installation packages and the required configuration files for easy installation and use. rsync requires a manual authentication password during the transfer process. Expect tool is used to automate the authentication process; expect is a tool built on tcl to automate processes that require interaction.

Shell scripts are used to make the server use rsync automatically to synchronize the data in the specified folder. Some of the rsync statements are encapsulated in shell scripts,

such as running, logging, and transferring. Transfer control can all be performed at the visual layer without having to operate on the command line. Specific package commands are shown in Table 2.

### 3.3. Middle Layer

**3.3.1. Control Program.** The control program is responsible for ensuring the normal running of scripts program, receiving commands from visual layer, sending commands to the system layer, and providing filtered log information to the visual layer. The control program is mainly composed of a set of shell scripts. A triangle daemon script architecture is designed to ensure stable running of unattended visual transmission system, as shown in Figure 6.

Two daemon scripts are used to monitor the core control scripts which also monitor each other. Under this architecture, system runs normally unless both daemon scripts and core control scripts are suspended at the same time. Except such situation, any script will be restarted when it fails. In the test, data transfer was sometimes suspended because of the rsync tool exception. A new monitor was developed to check the status of the rsync tool automatically. The rsync tool will be restarted if the monitor program finds an exception. In the last 1000 hours of testing, there was no manual intervention during system running.

Unstable transmission is likely to occur during long-distance data transmission. In this paper, the VSAN algorithm is designed in the control program to prevent the rsync from being repeatedly restarted when the network is not good and to ensure the stable operation of the system when the transmission quality is poor. The core idea of the VSAN algorithm is to transmit data normally when the network is unobstructed. When there are multiple small amounts of data, unified transmission will be sent after accumulating enough amounts of data. When the network delay is too high, the data transmission period will be extended. The flowchart of the VSAN algorithm is shown in Figure 7.  $V_n$  is the amount of data to be transmitted,  $S_d$  is the standard deviation of the transmission rate in 10 minutes,  $A_d$  is the average transmission rate in 10 minutes, and  $N_d$  is the transmission delay.

In the control program, the control interface can be used to start, shut down, and restart the system. It can also configure the system log storage directory, the size of a single log file, the log polling mode, and the system scan interval. The control interface is shown in Figure 8.

**3.3.2. Alarm Program.** The data transmission process will encounter various abnormal conditions. The alarm program periodically analyzes the log files to discover abnormal conditions in the system. Then the alarm program will automatically generate an exception report file and send an email to administrator for timely processing. The alarm program works by periodically analyzing specific log fields on the receiving server and the sending server to determine whether an exception has occurred and automatically writing the exception code value to the specified file on port 80. The control server periodically obtains the code value through the

TABLE 1: Database table.

| Table name    | Fieldname         | Field type | Field description                                       |
|---------------|-------------------|------------|---|
| files         | File_name         | char       | File name   |
|               | File_size         | int        | File size   |
|               | storeDBtime       | timestamp  | File storage time                                       |
|               | File_time_last    | timestamp  | Last time the file was modified                         |
|               | Md5value          | char       | Checksum  |
| data          | data_no           | int        | Order number  |
|               | data_volume       | double     | Total amount of data                                    |
|               | data_number       | int        | Total number of files                                   |
|               | data_time         | timestamp  | The time of the data is written to the database         |
|               | data_time_last    | timestamp  | The time of the data was last modified                  |
|               | data_add          | char       | Alternate field   |
| folder        | auto_no           | int        | Order number  |
|               | folder_no         | int        | Folder number   |
|               | foldet_cycle_no   | int        | The time of the folder has been accessed                |
|               | folder_name       | Char       | Folder name   |
|               | folder_volume     | Double     | The size of the folder                                  |
|               | folder_number     | Int        | Total number of files in the folder                     |
|               | folder_time       | Timestamp  | The time of the data was written to the database        |
|               | folder_time_last  | timestamp  | The time of the folder was last modified                |
| dayData       | folder_add        | char       | Alternate field   |
|               | auto_no           | int        | Order number  |
|               | dayDate           | date       | Date of the day   |
|               | data_volume       | double     | Total historical data                                   |
|               | data_number       | int        | Total number of historical files                        |
|               | incre_volume      | double     | Data increase on the day                                |
|               | incre_number      | int        | Increase in the number of files on the day              |
| dayFolderData | dayData_add       | char       | Alternate field   |
|               | auto_no           | int        | Order number  |
|               | dayDate           | date       | Date of the day   |
|               | folder_name       | char       | Folder name   |
|               | folder_volume     | double     | The size of the folder                                  |
|               | folder_number     | int        | Total number of files in the folder                     |
|               | incre_volume      | double     | Data increase of the folder on the day                  |
|               | incre_number      | int        | Increase in the number of the folder's files on the day |
| proc_status   | dayFolderData_add | char       | Alternate field   |
|               | auto_no           | int        | Order number  |
|               | writeData         | tinyint    | Script writeData  |
|               | control           | tinyint    | Script control  |
|               | oneDayStat        | tinyint    | Script oneDayStat                                       |
|               | rsync             | tinyint    | Script rsync  |
|               | update_time       | timestamp  | Last updated  |

“heartbeat” method and sends an email of the corresponding content to the system administrator according to the different code values. The exceptions and exception codes are shown in Table 3.

We assume that the control server is usually located in the data center and it is rare for the network to be abnormal. In addition, the data center usually has its own network situation alarm system. So we use a separate alarm system architecture.

The alert is not sent directly from the sending server or the receiving server.

**3.4. Visualization Layer.** The visualization layer provides web pages developed by HTML5, PHP, JavaScript, jQuery, and other web technologies to display astronomical data transmission. The visualization layer extracts the content from the middle layer and displays it in the web pages in the



TABLE 2: Package rsync command table.

| Packaged command option | Package command function  |
|-------------------------|---|
| -a, --archive           | Recursive mode, which means recursively transferring files and keeping all file attributes            |
| -r, --recursive         | Treat subdirectories in recursive mode  |
| -p, --perms             | Keep file permissions   |
| -t, --times             | Keep file time information  |
| -g, --group             | Keep file group information   |
| -z, --compress          | Compress when transferring files  |
| --progress              | Display the transmission process during transmission  |
| -v, --verbose           | Detailed output mode  |
| -q, --quiet             | Streamlined output mode   |
| -b, --backup            | Back up legacy files in the target directory when there are changes                                   |
| --ignore-existing       | Ignore files that already exist on the receiving server; only those newly created files are backed up |

TABLE 3: Exception code correspondence table.

| Exception codes | Exception                    |
|-----------------|------------------------------|
| 110             | Everything is normal         |
| 111             | rsync startup exception      |
| 777             | Control module exception     |
| 555             | Network delay is abnormal    |
| 444             | Storage space is running out |

form of charts. The adaptive layout of the web pages enables users to access the web pages through an authorized account using a computer or mobile browser. Here is a test web page <http://210.73.36.12/qttas/for-test.php>, which is open to all to browse parts of the visualization layer. The following part of this section will show the visualization web pages of data transmission between the Qitai Observatory and the Xinjiang Observatory headquarter.

The web pages are divided into five parts. The first part shows the running state of the scripts obtained from the table 'proce\_status' in the database on the sending and receiving servers. Any script that is not running will be intuitively displayed in this part, as shown in Figure 9.

The second part is shown in Figure 10. This part displays the amount of data through the column charts. The column charts can display the volume of data that has been transmitted on the current day and the past 7 days. The second part calls the 'dayData' and 'data' database tables.

The third part is shown in Figure 11. This part shows the storage status of the sending server and the receiving server through pie charts. The third part can help administrators to determine whether data storage needs to be expanded. When the free space of the data server is lower than the threshold, the third part will be displayed in red.

The fourth part is shown in Figure 12. This part displays the data storage in the past 56 days in the form of color blocks. The darker the color block, the more the data produced in this day and the lighter, the less. In order to ensure that the color block display is natural and can truly reflect the amount of data, we first get the data volume sorting in the past 56 days by the bubble sorting method. The maximum data volume is  $V_{max}$ , the minimum data volume is  $V_{min}$ , the data volume

interval is  $V_{di} = V_{max} - V_{min}$ , and the daily data volume is  $V_{day}$ ; the color value percentage of daily data is

$$C_{per} = \frac{(V_{day} - V_{min})}{V_{di}} \quad (1)$$

The fifth part displays the amount of the transferred data and the stored data per minute in the past 2 hours by the broken line charts. As shown in Figure 13, we simulate the real time in which the storage bandwidth is greater than the transmission bandwidth. The broken line charts can visually display the fluctuation of the data transmission rate, and the administrator can judge whether the data link is unblocked by these charts.

In addition to these five sections, the pages also display the servers and link status and the information of the data being transmitted in text. When the system fails, the alarm information will be displayed on the visualization pages.

The advanced query page is shown in Figure 14. The advanced query page requires an advanced authentication command to access. The advanced query page supports detailed information query about data storage for specifying day on data servers, results of file MD5 validation display, and even keyword retrieval.

## 4. Summary

This paper completes the construction and development of the astronomical data visualization system and provides a complete set of management system for transferring data from the astronomical observation site to the data center. We have completed the four-layer system architecture design based on the analysis of the advantages and disadvantages of the existing astronomical data transmission system and the actual needs of the Xinjiang Astronomical Observatory. During the development process, we have fixed the deficiencies in the original design, and the system was stable during the last 1000 hours of testing. This paper provides a feasible astronomical data transmission scheme, which assists administrators in managing the transmission process through the log system and the visual interface. As a newly developed system, the astronomical data visualization

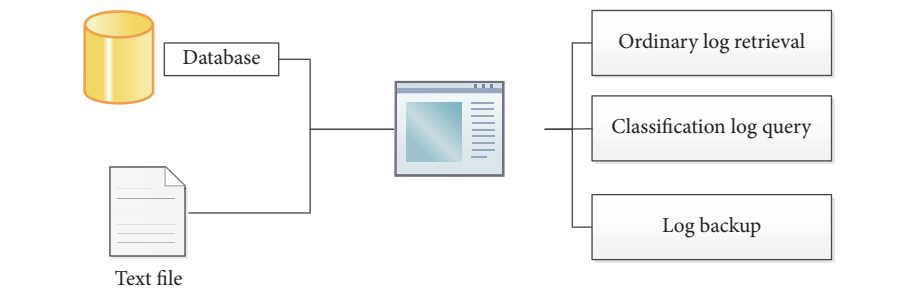


FIGURE 3: Log subsystem management interface functional structure.

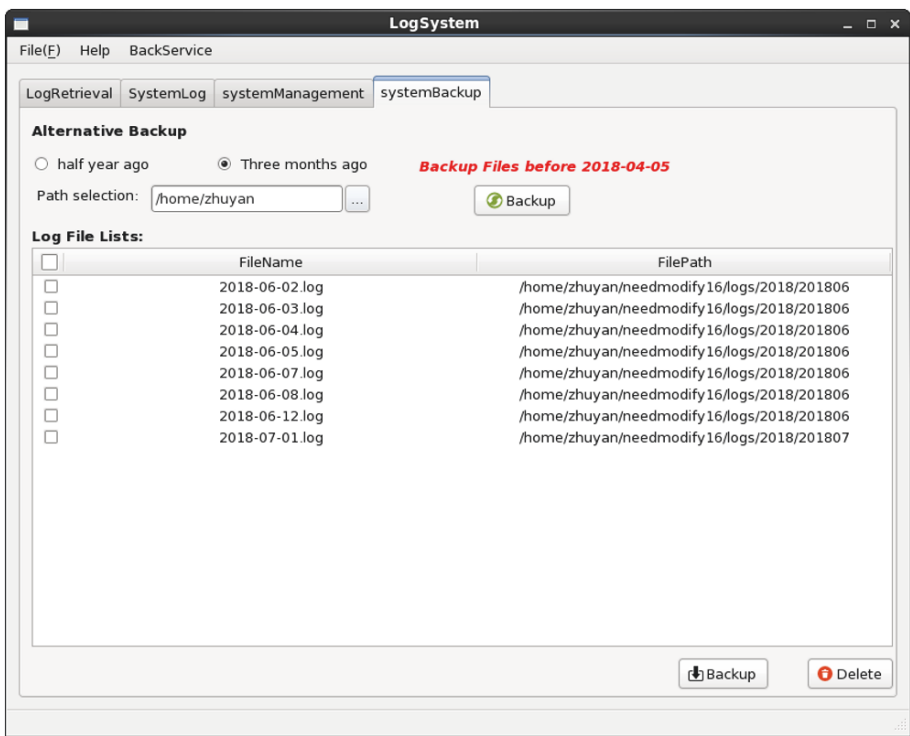


FIGURE 4: Log retrieval interface.

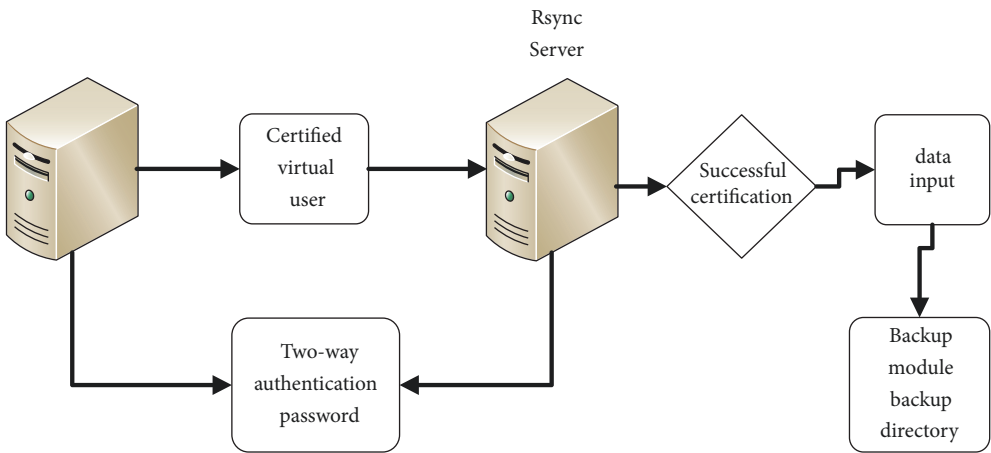


FIGURE 5: Rsync authentication process.

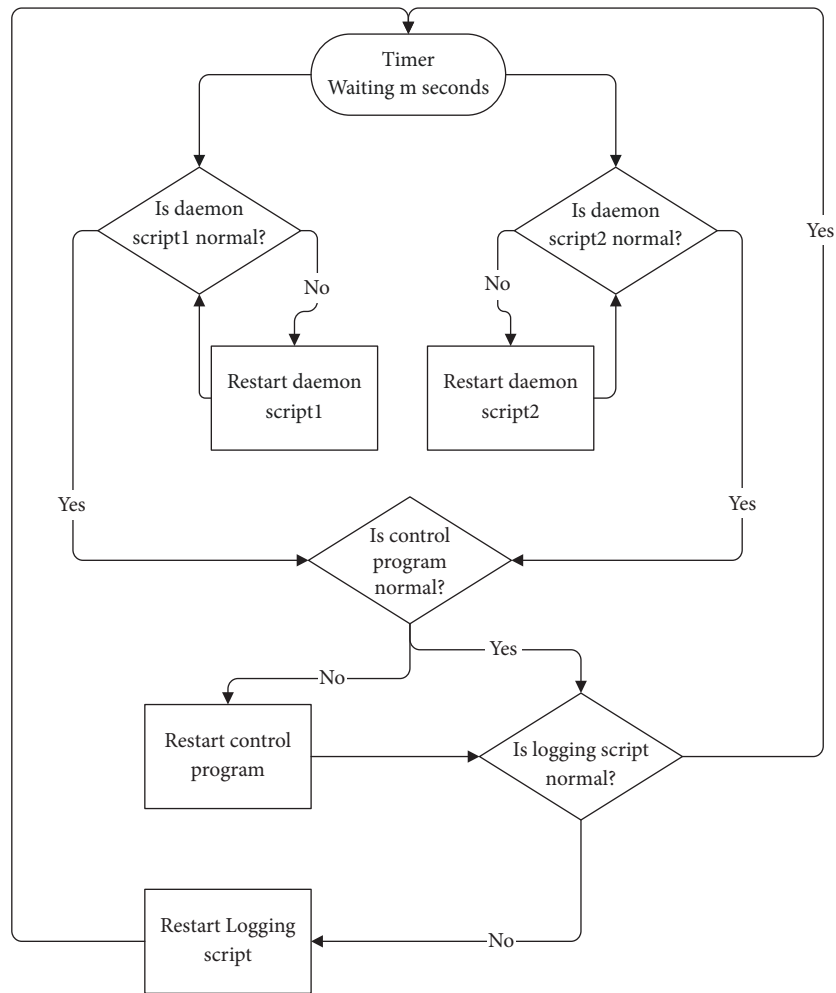


FIGURE 6: Daemon scripts running flowchart.

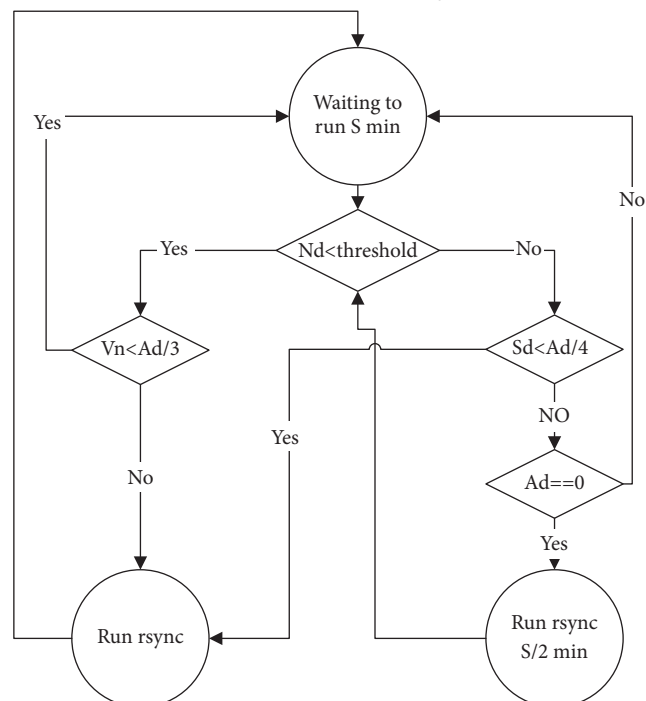


FIGURE 7: VSAN algorithm flowchart.



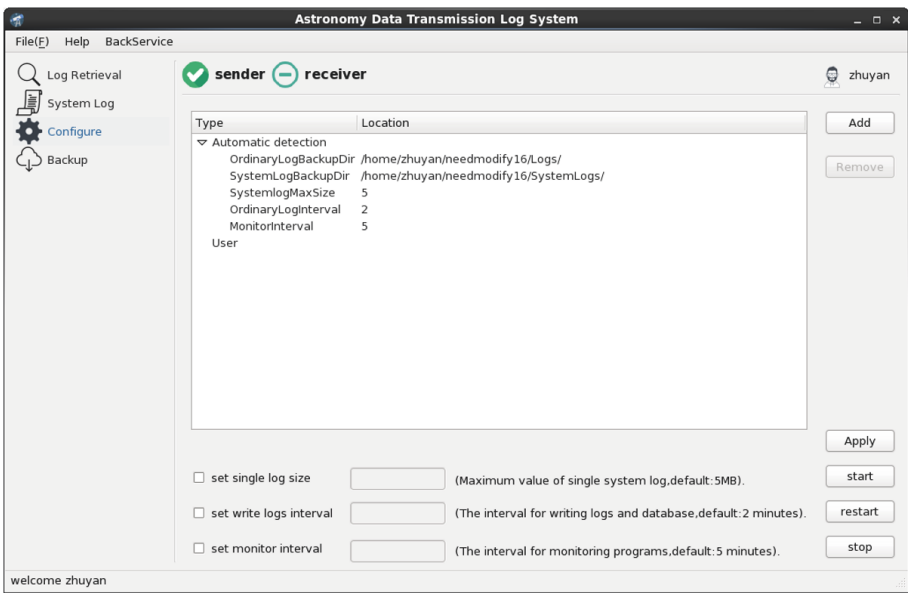


FIGURE 8: Control interface.

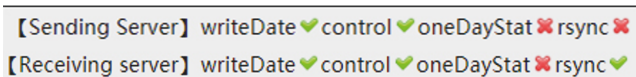


FIGURE 9: The first part of visualization web pages.

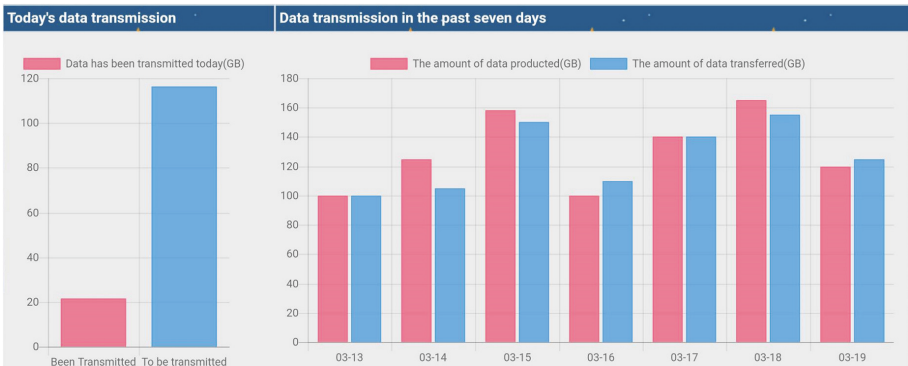


FIGURE 10: The second part of visualization web pages.

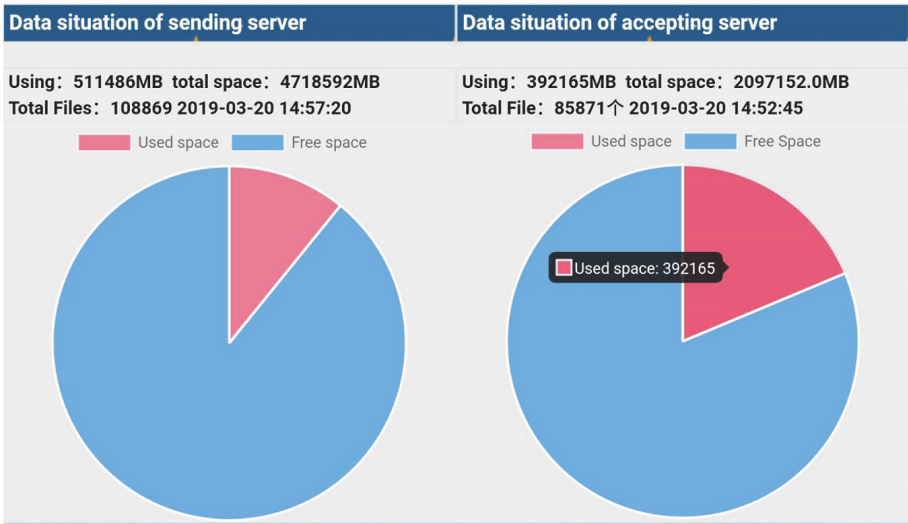


FIGURE 11: The third part of visualization web pages.

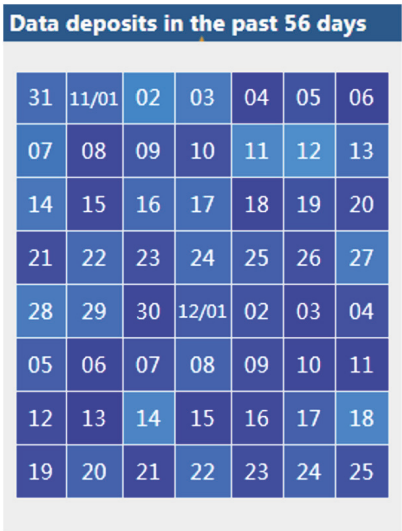


FIGURE 12: The fourth part of visualization web pages.

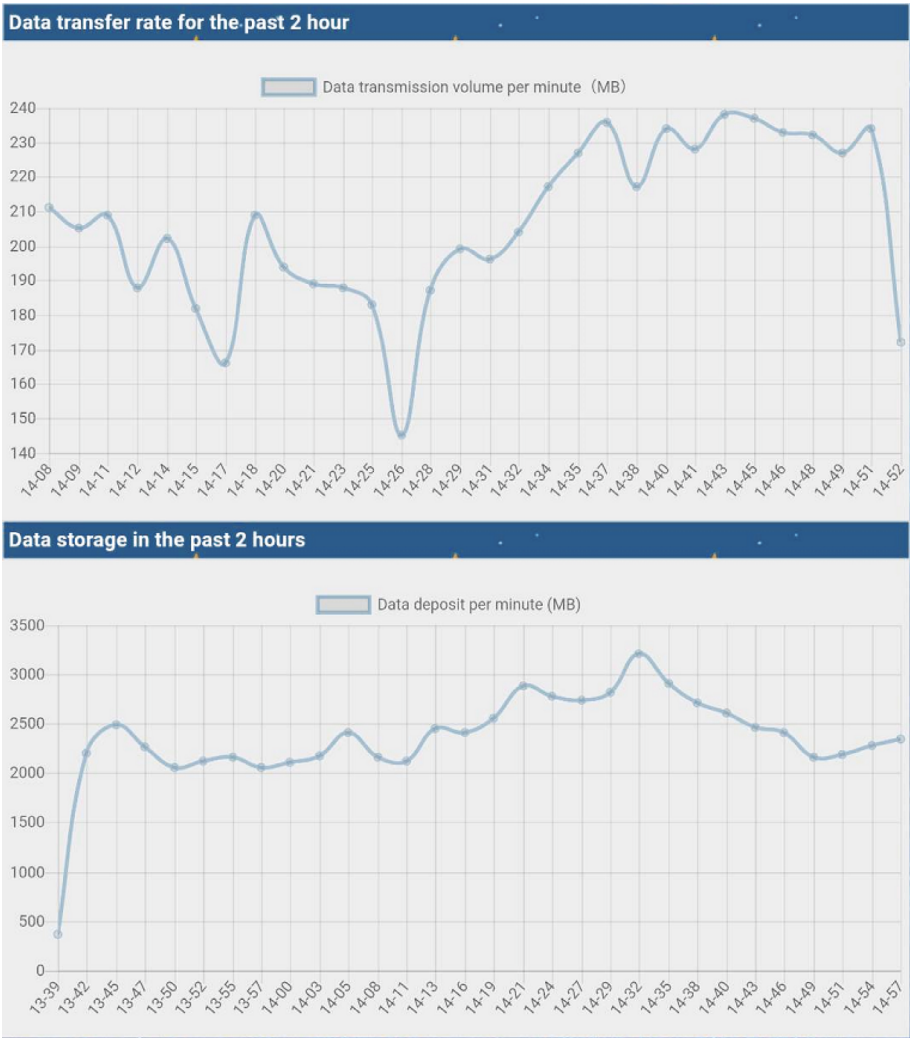


FIGURE 13: The fifth part of visualization web pages.

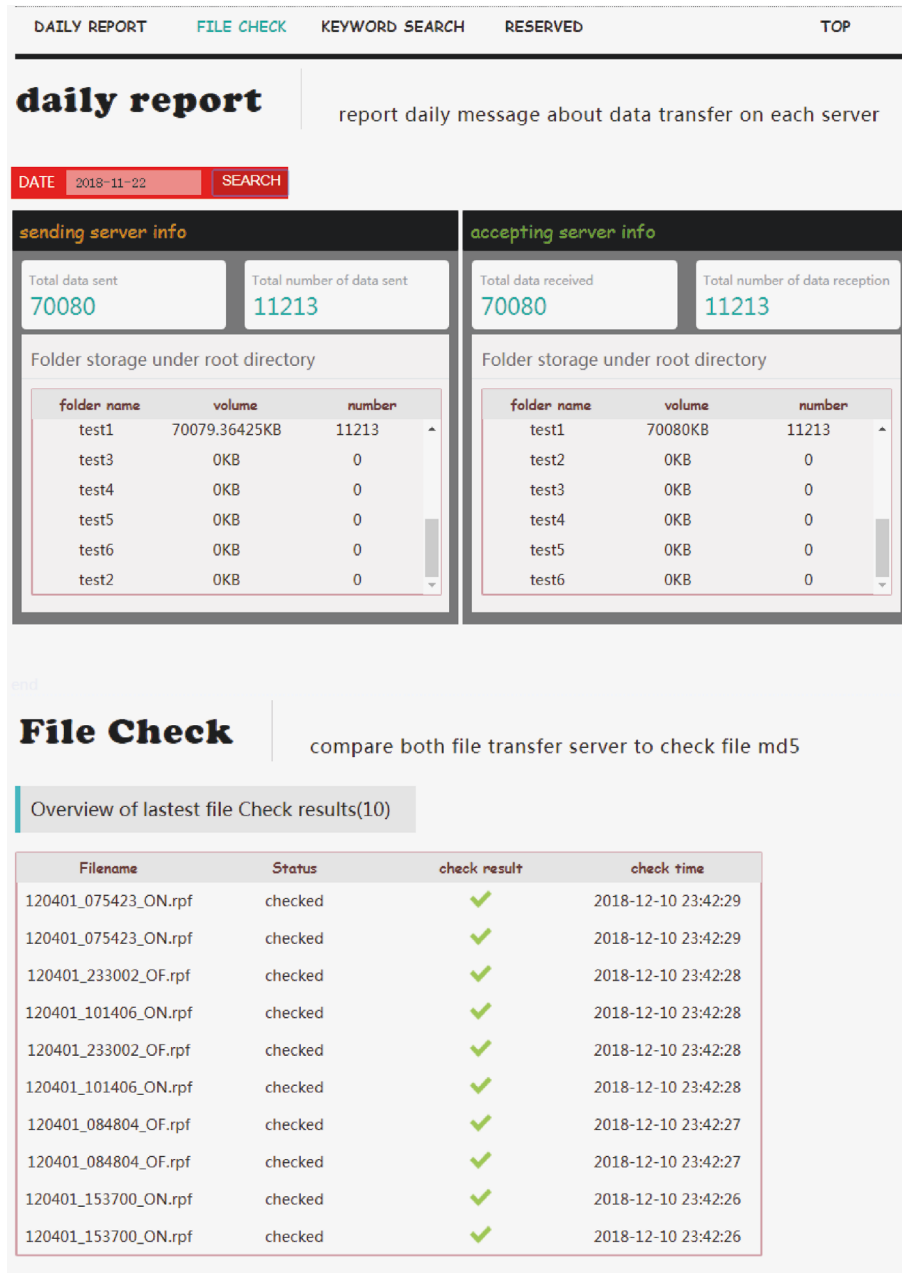


FIGURE 14: The advanced query page.

transmission system is still insufficient and will be further improved in the future work.

### Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

### Conflicts of Interest

The authors declare that they have no conflicts of interest.

### Acknowledgments

The authors gratefully acknowledge the support of the National Natural Science Fund of China (11873082, U1531125, 11803080, and 11503075), National Key Basic Research Program of China, 973 Program 2015CB857100, National Key Basic Research and Development Program 2018YFA0404704, and Youth Innovation Promotion Association CAS.

### References

- [1] P. Rosen, B. Wang, A. Seth et al., "Using contour trees in the analysis and visualization of radio astronomy data cubes," 2017, <https://arxiv.org/abs/1704.04561>.

- [2] Z. D. Stephens, S. Lee Y, F. Faghri et al., “Big data: astronomical or genomic?” *PLoS Biology*, vol. 13, no. 7, Article ID e1002195, 2015.
- [3] R. Prestage, K. Constantikes, T. Hunter et al., “The green bank telescope,” *Proceedings of the IEEE*, vol. 97, no. 8, pp. 1382–1390, 2009.
- [4] R. Nan, D. Li, C. Jin et al., “The five-hundred-meter aperture spherical radio telescope (FAST) project,” *International Journal of Modern Physics D*, vol. 20, no. 6, pp. 989–1024, 2011.
- [5] D. Li, P. Wang, L. Qian et al., “FAST in space: considerations for a multibeam, multipurpose survey using china’s 500-m aperture spherical radio telescope (FAST),” *IEEE Microwave Magazine*, vol. 19, no. 3, pp. 112–119, 2018.
- [6] D. H. Schaubert, A. O. Boryssenko, A. Van Ardenne et al., “The Square Kilometer Array (SKA) antenna,” in *Proceedings of the 6th IEEE Phased Array Systems and Technology Symposium, Array 2003*, pp. 351–358, USA, October 2003.
- [7] C. Carilli and S. Rawlings, “Motivation, key science projects, standards and assumptions,” *New Astronomy Reviews*, vol. 48, pp. 979–984, 2004.
- [8] G. Rhee, *Looking Ahead in Wonder: Telescopes at the Cosmic Frontier*, Springer, New York, NY, USA, 2013.
- [9] E. Dovgan, C. Knapic, M. Sponza, and R. Smareglia, “A new archival infrastructure for highly-structured astronomical data,” *Experimental Astronomy*, vol. 45, no. 1, pp. 41–55, 2018.
- [10] A. Wicenec, J. Knudstrup, and S. Johnston, “ESO’s next generation archive system,” *Messenger*, vol. 129, pp. 27–31, 2002.
- [11] A. Wicenec and J. Knudstrup, “ESO’s next generation archive system in full operation,” *The Messenger*, vol. 129, pp. 27–31, 2007.
- [12] S. J. Tingay, R. Goeke, J. D. Bowman et al., “The Murchison widefield array: the square kilometre array precursor at low radio frequencies,” *Publications of the Astronomical Society of Australia*, vol. 30, 2013.
- [13] A. Wootten and A. Thompson, “The atacama large millimeter/submillimeter array,” *Proceedings of the IEEE*, vol. 97, no. 8, pp. 1463–1471, 2009.
- [14] A. Wicenec, S. Farrow, S. Gaudet, N. Hill, H. Meuss, and A. Stirling, *Astronomical Data Analysis Software and Systems (ADASS) XIII*, vol. 314, 2004.
- [15] A. Manning, A. Wicenec, A. Checcucci, and J. A. Gonzalez Villalba, *Astronomical Data Analysis Software and Systems XXI*, P. Ballester, D. Egret, and N. P. F. Lorente, Eds., vol. 461, Astronomical Society of the Pacific Conference Series, 2012.
- [16] C. Wu, A. Wicenec, D. Pallot, and A. Checcucci, “Optimising NGAS for the MWA archive,” *Experimental Astronomy*, vol. 36, no. 3, pp. 679–694, 2013.
- [17] Q. Xu, L. Yi, L. Li, M. Chen, N. Wang, and Y. Li, “A rapid feed switching mechanism design for NSRT,” in *Ground-Based and Airborne Telescopes VII*, vol. 10700, International Society for Optics and Photonics, 2018.
- [18] N. Wang, *Scientia Sinica Physica, Mechanica & Astronomica*, vol. 44, 2014.
- [19] R. Rischpater, “Application development with Qt creator,” 2014.
- [20] Y. J. So, *SyncML Data Sync System and Data Exchanging Method for Data Exchange between Clients: U.S. Patent 7,917,653*, 2011.
- [21] D. Rasch and R. C. Burns, “In-place rsync: file synchronization for mobile and wireless devices,” in *Proceedings of the USENIX Annual Technical Conference, FREENIX Track*, p. 100, 2003.

