BEAM

Blacknest Data System (BDS)

Prototype Overview - Preliminary

Project	Blacknest
Date	2008-09-03
Reference	blacknest/bdsPrototypeOverview-2
Author	Dr Terry Barnaby

Table of Contents

1. References

- The BEAM Blacknest support website at: https://portal.beam.ltd.uk/support/blacknest. This provides detailed information on the BDS system and the current AutoDRM, information on alternative AutoDRM implementations and information on data formats.

BEAM

2. Introduction

This document provides an overview of the Prototype Blacknest Data System (BDS). The purpose of the system is to provide storage and access to seismic data and associated information. Development of the BDS system will be an on-going process adding features and working around issues in the seismic data and meta data as they arise.

Seismic data is obtained from seismic stations and arrays of stations situated at various places in the world. Each seismic station consists of one or more seismometers detecting earth movement in one or more directions. The low frequency analogue data from each seismometer is digitally sampled, at a low sample rate, and stored as a channel in a data file. There is also an archive of analogue data on tapes which will be digitised for inclusion into the system.

The BDS system provides a core platform that is modular in design and integrates the raw seismic data with its meta-information.. The core defines API's and data formats that can be used and an overall system structure. The basic core system can be extended with additional modules as required.

3. Original Data

The original raw seismic data is stored in files in a number of different data formats. In general each data format contains a header, providing information on the contained data, such as the array/station name and the sample rate. There are then one or more channels of time domain samples from the array's or station's seismometers. The definition of which instrument the channels data is from is either in the header or in a separate Instrument Response Database. Blacknest has a large robotic tape data storage archive containing a large set of seismic data from local as well as remote seismic arrays and a preliminary MySQL database containing the Instrument Response information.

4. BDS Features

The systems core features are as follows:

1. To archive seismic data on data storage systems.
2. To archive seismic meta information such as instrument settings and calibration information.
3. To index the seismic data and meta information stored in the system.
4. To provide access to the data in the data storage system both locally and remotely.
5. To provide data format converters to allow access to the data in the format required.
6. To provide various data front-ends including an AutoDRM and Web based interfaces.
7. Can use multiple data storage systems such as tape archive and disk archive systems.
8. To provide data input systems: Manual, TapeDigitiser and Satellite.
9. Data access security by userid/password.
10. To provide access to data from real-time incoming data feeds.
11. Bandwidth management for remote links.
12. Uses a RAID disk array as the primary data storage archive.

BEAM

13. Provides a 'C++' and PHP API for easy and direct access to the data.

Future system features could include:

1. Provide additional program API's to allow easy and direct access to the data. This could be implemented for: Python, Octave, MatLab, (SAC, GeoTool), etc.

2. Provide in-built algorithms for data manipulation. This would allow, for example, sample rate changing etc.

3. Provide robot programs to look for data features etc.

4. Data backup ability to remote archive. This could just be the data base and meta-information but could also include the main data.

5. Ability to feed data to other sites

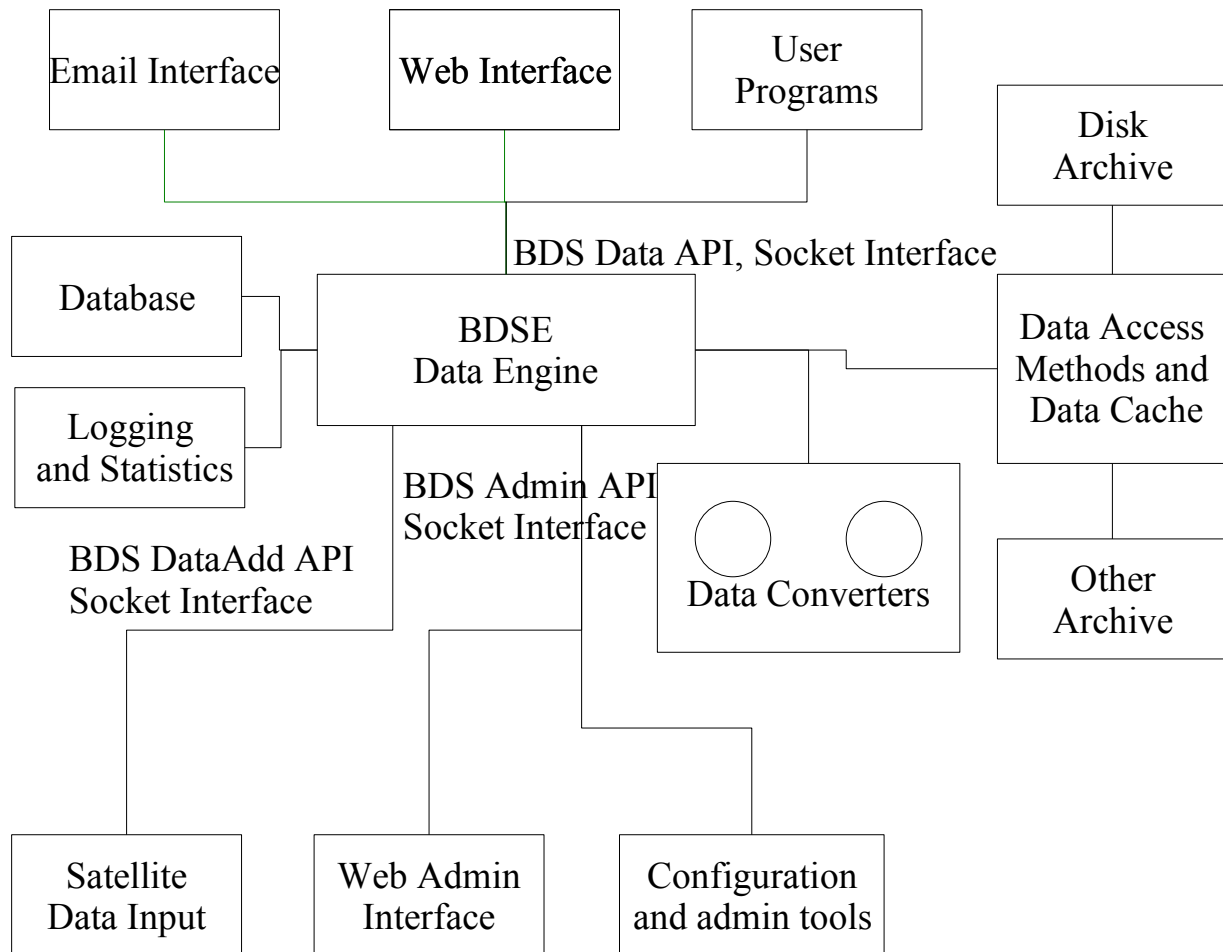6. Event system for errors such as satellite data feed errors.

4.1. Notes

- The data access front-ends are separated from the data gathering and data conversion systems. This allows other front ends to be easily added and modularises the system. Examples of different front ends are: email, web and direct access from local data processing programs.

- The data converters are implemented as separate modules. This will allow them to be used for other purposes and allow better testing of their functionality. It will also allow other data converters to be added to the system easily. The data converters will convert between their own format and a special internal format. This internal format will be chosen so as to efficiently store the data present in all data formats. It will not be a published format so that it can be changed to suit data conversion requirements.

- The system uses a MYSQL Database for storing the user, indexing and instrument changes and calibration information.

- The method used to retrieve the data is implemented in a modular way. This modularity allows other data access modules to be added as required in addition to the standard disk archive access method.

- The system implements a simple priority queue system. This will allow local access to occur while external requests are being serviced.

- The system has the concept of security groups. This allows certain users to access data that is not accessible to all.

- Full logging and statistical information is available.

- A password protected, administration interface is provided for ease of administration and the viewing of major system statistics.

5. Design Overview

This section describes the overall BDS design. The design is modular and allows extension as well as use of its components by external systems. The fundamental split is between the input/output modules and the main Data Processing Engine (BDSE).

BEAM

The BDS modules communicate with the main "Data Engine" using the BOAP Object Orientated RPC

BDS Overall Design

protocol over an network socket interface.

The "Data Engine" (BDSE) is the heart of the system. It is implemented in the "bdsServer" program. It queues requests for data, prioritises them, gathers the data and converts it into the appropriate format using the converters available. All accesses to read or write data go through the BDSE. It is responsible for keeping the data protected and valid.

It provides three separate API's. These have independent protected access:

- BDS Data API. This provides "read only" functions for accessing the seismic data and meta-data. It is used by the AutoDRM email interface, the web interface and for direct program access.

- BDS DataAdd API. This provides the ability to add data to the system. It is used by the Manual and Automated data entry programs.

- BDS Admin API. This provides the ability to add and change the data stored in the system. It is

BEAM

used by local administrators.

The "Database" is used to store Meta Data. This includes the Instrument Response database, the data files database, authorised users database and other information. This database is implemented using MySQL either running on the same platform as the "bdsServer" or on another system.

The Email, Web, Administrator Web interfaces run as separate processes to the main Blacknest Data System Engine (BDSE) "bdsServer". This provides another layer of security against outside attacks as well as providing increased modularity. The Email and Web interfaces could be implemented on other systems if required and other systems can connect directly to the "Data Engine" to get data.

Local programs and utilities are able to access the system using the Data, DataAdd or Admin API's. This allows local network applications such as user analysis programs or robotic data analysis programs to access the data directly.

The Data Converters are implemented as separate programs that can convert their specific data format to and from a standard internal data format. This internal data format, BdsData, has been developed for flexibility and the ability to support all of the information in all of the data formats. Each data converter can convert to and from their appropriate data format and this internal data format. A simple data converter API library will be implemented which will make it easy to produce the data converters.

The "Data Access Methods" section provides a generic raw data data access interface for the BDSE. This provides the ability to fetch data from different sources using different methods. The primary method is a local file system access to a RAID disk archive, but NFS, FTP, HTTP or other access methods could be added later if required. This module also could be developed to provide caching of the data if required.

The Logging and Statistics module can be used by all other modules to log errors and statistical information.

There is a set of configuration and other administration tools available to handle configuration and management requirements.

6. BDS API's

The BDS API's form the heart of the system. All data and meta data access go through the API's. The BDS provides three separate API's. These will have independent protected access:

- BDS Data API. This provides "read only" functions for accessing the seismic data and meta-data. It is used by the AutoDRM email interface, the web interface and for direct program access.

- BDS DataAdd API. This provides the ability to add data to the system. It is used by the Manual and Automated data entry programs.

- BDS Admin API. This provides the ability to add and change the data stored in the system. It is used by local administrators.

The API's are implemented using the BOAP Object Orientated RPC mechanism over a network socket interface. Access to the API's are secured by a userid/password system. This could be extended to a public/private key system in the future.

The document **BdsApiOverview.html** describes the API in more detail.

BEAM

7. Software Packaging

The BDS software consists of a number of API libraries and a set of applications bother for the server and for client systems. The software is currently packaged in the following RPM modules:

Package	Description
bds-repo	Package to install the yum package repository information
bds	General BDS package, needed by all other packages
bds-server	The main BDS server package. All programs that are neded on a BDS server
bds-autodrm	The BDS Autodrm front end program
bds-web	The BDS Web interface front end program
bds-clients	A set of general BDS client applications
bds-devel	BDS API library include files and bits for developing BDS applications
Utility Libraries	
libgcf2	Gurlap GCF data file access library runtime
libgcf2-devel	Gurlap GCF data file access library development
libmailutils	Email manipulation library runtime
libmailutils-devel	Email manipulation library development

All of these packages may be installed on a single BDS server host. Alternatively individual packages may be installed on different hosts. The bds-autodrm package may, for example, be installed on a front end email host and the bds-clients package may be installed on any system that requires access to the BDS system

8. The BDS Server

The BDS server is the heart of the system. On this platform runs the main **bdsServer** program. The **bdsServer** program provides the BDS network API for all BDS client applications and implements the core of the system. It provides access to the seismic data and the associated meta-data. It is responsible for ensuring data security and validating data requests. On this system also runs the BOAP **boapns** daemon. The **boapns** daemon provides an Object name to IP Address and Service number lookup system that the client applications use to find the bdsServer's network API sockets.

The BDS Server will also normally run the MySql database system, although this can be situated on a separate host if required. The raw Seismic data files can also be situated on this host or on a separate system.

The **bdsServer** program has a single configuration file, */etc/bdsServer.conf*, that defines its main configuration settings. It is started automatically at system power up as the **bdsServer** service.

There are some MySql database setup scripts in the **bdsSql** directory.

BEAM

9. The BDS Autodrm

The **bdsAutodrm** program implements the BDS Autodrm email interface. It can be situated on any host that has access to the BDS Server. It implements the Blacknest Autodrm protocol to retrieve seismic data through an email interface and FTP interface. All data and meta data requests go through the main **bdsServer**. The **bdsServer** is also responsible for converting the data into the requested format. Thus the **bdsAutodrm** is solely responsible fro providing the email interface.

It has a configuration file, */etc/bdsAutodrm.conf*.

10. The BDS Web Interface

The BDS Web interface is implemented as as a PHP web application. It has been based on the BEAM WebSys PHP simple content management system. There is a single PHP module, named bdsdata, that provides the special BDS data access functionality.

The BDS Web interface can be installed on any system that supports an Apache web server and which has access to the BDS Server.

The BDS Web interface is just a demo system at the moment. It makes use of the **bdsDataAccess** command line client program to access the BDS Server. It can search for the data and return sections of the data in any of the BDS supported formats. It also has a primitive graphical data viewer built in.

11. The BDS Client Utility Programs

The BDS system has number of command line and GUI applications available to perform administrative functions as well as add data to the system. These use the BDS API's to access the information. The initial applications that have currently been developed include:

1. **bdsAdminGui**. This a simple GUI application that allow basic data and metadata information to be manipulated. It allows all of the available data and metadata to be viewed. The MetaData can be edited and verified. Siesmic data cane be viewed in graphical form.

2. **bdsDataAccess**. This is a simple command line program that allows access to the seismic data and associated meta data.

3. **bdsImportOldDatabase**. This program is able to import all of the existing instrument response database into the BDS system

4. **bdsImportData**. This command line program is able to import all of the existing seismic data files into the BDS system

Future client applications may include:

5. **bdsAddDataGui**. This GUI application would add seismic data to the system. It would present a GUI interface to check the data prior to importing into the BDS system.

6. **bdsAddMetaData**. This GUI applications would allow MetaData to be added to the system.

7. **bdsAddStreamData**. This command line, possibly daemon, application would add data and MetaData from the Guralp satellite feeds.

8. **bdsCheckGui**. This application could perform data and metadata consistency tests.

BEAM

9. **bdsDataViewGui**. This GUI application would enable the viewing and saving of seismic data from the BDS system.

12. BDS DataBase

The BDS system employs a MySQL database to store Seismic MetaData information such as Instrument responses and information on the available Seismic data. It also stores information on the users and is used for queuing user requests for data.

There is a single database, named **BDS**, that has a number of tables. The **BdsSchema.html** document will describe the prototype Database Schema in detail.

Normally only the **bdsServer** program accesses the database although it obviously is possible to directly access it. The **bdsServer** also provides an API call to directly access the database if required.

13. BDS Data Handling

Seismic data is available in many formats. In order to simplify data access the BDS employs an API that generalises access to the data. The BDS uses the following model for data access.

- All seismic data is stored in individual channels.

- Each channel provides a single set of data organised as a set of variable length blocks.

- Each data block has a start and end time stamp and a set of continuous data samples.

- A data block can contain any number of channels of data for the given time period. There can be a variable number of samples per channel.

- Data blocks may not be continuous in time, there can be missing data blocks. (There could be an option to introduce 0 valued blocks in this case as an option when reading the data).

- The format of each sample is retained through the system. The current Sample types supported include: Int16, Int32 and Float32. When passing the data within the system the samples may be stored in a Float64 for simplicity, but there original sample format is known and can always be re-created. Data is stored in the standard BDS format using the original data's sample format.

- In data files or streams the samples within a block are packaged in specific pack format. This may employ compression.

- In data files or streams, multiple channels can be multiplexed by sample or by channel.

- Meta data for each channel provides, at a minimum: network, station name, channel name, sampleRate and sampleFormat.

The BDS system retains the block sizes and thus time stamps of the original data source as well as the original data's sample format. Thus the full integrity of the data is retained.

13.1. BDS Seismic Data Files

The prototype system initially supports the following external data file formats:

- BKNAS 1.0 (BKNAS)

- IMS 2.0 (IMS)

BEAM

The prototype system supports import of the following data file formats:

- BDRS
- GCF: Guralp compressed format.
- TapeDigitiser The Blacknest TapeDigitser file format for Digitised Analogue Tapes.

The first production BBDS system will also support import of the following data formats:

- SEED Standard for the Exchange of Earthquake Data format. (Initial support for Blacknest stored SEED files).

Support can be later added for the following formats:

- WRA: WRA40, WRA64 and WRA-AGSO
- CD-1.1 IDC Streaming data format
- Better SEED file support.

The system store the seismic data using a single file format. This is named the **BdsData** format. Some more information on this is in the **BdsDataFile.html** document. This format is subject to change as the system evolves.

13.2. BDS Data Importing

The policy and procedure for importing data into the BDS system needs to be decided and implemented by Blacknest. However to first production BDS system will allow the following procedure to be carried out:

1. Data can be imported using the bdsImportData command line program. Initially this will handle: BDRS, GCF and TapeDigitiser data.

2. The user will need to supply the file name, the data format and the network, station and channel names for each of the files data channels. (We could automate the channel names by either using a simple file database or adding a table to the BDS SQL. This would use the time and station name to decide the orientation of channels with the files).

3. The bdsImportData program will first validate the data. The following validations will be performed:

   - Generally check for data file corruption.
   - Check that the blocks time-stamps are continuous and within an expected range.
   - Check that the date and time encoded in the file name matches with the time stamps within the data blocks.
   - Check that the sample rate is correct.

4. If the validation passes the data is uploaded into the BDS system. If it fails an error message is reported to the user. In this case the user can use the "-force" flag to upload the data with an appropriate description message.

Further data validations can be performed by Blacknest staff. These may include:

BEAM

- Check if meta data exists for these data channels within the BDS system.
- Check if the channel allocation is correct.
- Check the polarity of the data channels.
- Check for periods where data channels do not contain useful data due to disconnected sensors etc.

13.3. BDS Real-time data feeds

Blacknest receives some data from real-time satellite data feeds. There is a problem with this data source in that it is currently sent using a UDP based broadcast system that can lose blocks of data. The current system for getting a retransmission of the missing data blocks means that it may require up to 5 hours before a complete set of data is available.

The best way of handling this issue would be to fix the satellite data reception system. However if this is not possible the satellite data reception system could provide two data streams to the BDS server. One would be the real-time data feed that would contain missing data blocks. The second stream would be the delayed and re-constituted data stream. The BDS system would treat the real-time stream as a temporary data stream and would only keep about 7 days of this data.

This system would allow direct access to the real-time data, although it may be missing data blocks.

Note that the current development contract for the prototype and version 1 system does not include supporting real-time data feeds.

14. BDS Server Hardware

Most of the BDS software will run on a single server. The following gives a basic overview of the hardware configuration. After the system has fully developed we may need to revise the system specification.

Item	Specification	Details
Processor	Dual Xeon Quad cores	
RAM	2GByte	
Networking	1 or 2 Gigabit Ethernet	
Disk	4 or 8 750G SATA drives	RAID 5 configuration

We recommend the following server platform:

- HP ProLiant DL380 G5

14.1. Operating System

We recommend one of the following, Linux based, operating systems to run on the server:

- CentOS5
- Redhat Enterprise 5
- Fedora 8

BEAM

15. Software Development

All code is produced under the GPL software license so that it can be donated to the open source community if required. The implementation OS is Linux, however the code is written in a portable manner to allow porting to other systems. The target systems are CentOS5 or Fedora 8. The software is packaged in the RPM package format to simplify system installation and maintenance.

The SVN version control system is used for source code management. This will be made available over the Internet.

The main language used for development is 'C++' and is used in an object orientated style. We make use of a number of BEAM developed 'C++' class libraries to ease development.

The GUI administration applications are written in 'C++' and use the QT GUI widget set and the BEAM 'C++' class library.

The web applications are written in PHP and use 'C++' extensions to implement the BDS BOAP API.

The AutoDRM is implemented in 'C++' .

There are three main software libraries used in the BDS system:

- **LibBeam**. This is the BEAM 'C++' class libraries. It provides low level classes for String, List and Array handling as well as system access classes. It also provides the core of the BOAP API system.

- **BdsLib**. This is the main BDS API library. It implements the BOAP BDS API as well as some other utility classes. For the most part it is automatically generated using the BOAP **bidl** utility from the *Bds.idl* API specification file.

- **BdsDataLib**. This library provides a set of classes for reading and writing to seismic data files in any of the supported formats. There is one class per file format. The **bdsServer** uses these classes to access and convert between data formats.

16. Documentation

The system's documentation is available on the BDS support website at: http://portal.beam.ltd.uk/support/blacknest.

There are three separate document categories:

- A users manual describing the operation of the system from a users perspective.

- An administrators manual describing how to administer the system.

- API Manual for software module writers describing the basic operation of the code and how to build and extend it.

17. BDS Testing

Testing of the system will be carried out mainly by comparing the data returned by the system with the data from the current Blacknest systems. The following sections list basic initial test schedules:

BEAM

17.1. Beam Testing

At BEAM we will perform the following basic testing:

1. General system stability. Run a set of automated test programs that will fetch random data and meta-data from the system.

2. Basic data integrity. We will load a small set (300G) of existing BDRS, GCF and TapeDigitiser data onto the test BDS system. We will then produce a program to fetch the data from the BDS system and original data files and compare the results.

3. Validate BKNAS and IMS data format converters with test data set.
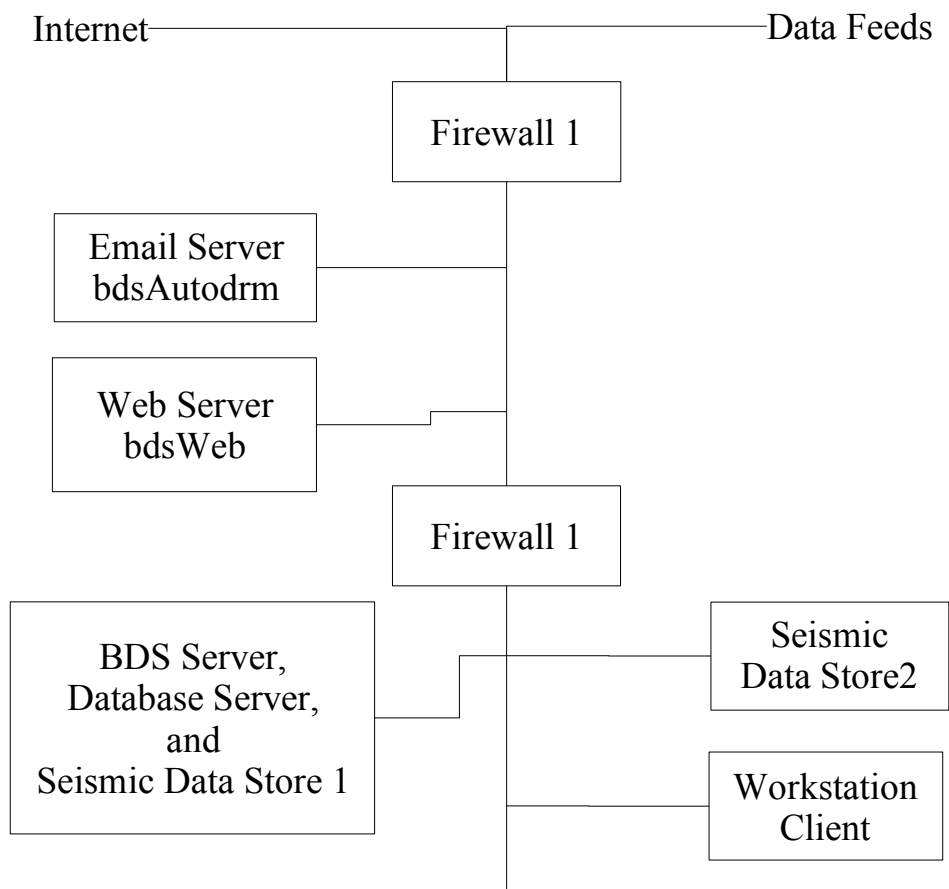
17.2. Blacknest Testing

At Blacknest the following tests will be performed:

1. General system stability. Run a set of automated test programs that will fetch random data and meta-data from the system.

2. Basic data integrity. We will load a small set (300G) of existing BDRS, GCF and TapeDigitiser data onto the test BDS system. Blacknest can then write a program to fetch the data from the BDS system and original Autodrm system in BKNAS or IMS format and compare the results.

3. Validate BKNAS and IMS data format converters with test data set.

When the production BDS system has been installed and Blacknest has imported all of the existing data then a full comparison of all data could be performed using an automated program.

18. Integration with Blacknest Systems

The diagram below gives a basic overview of the computer systems employed at Blacknest that would be involved with a final production BDS system. There is main Firewall system (Firewall 1) that protects Blacknest's external service servers (Email and Web) from the Internet and external data sources. The external data sources include Satellite seismic data feeds. A further Firewall (Firewall 2) protects Blacknest's internal servers and desktop clients. The BDS and Database servers sit on inner protected network. Emailed requests are accepted by the sites external Email server and, after checking for SPAM etc., are forwarded to the BDS server. The BDS server will initially implement the Database server and the Data file storage system. It is estimated that Blacknest has at present about 3 Terabytes of Seismic data plus the Analogue archive.

BEAM

Internet————————————————Data Feeds

Firewall 1

Email Server
bdsAutodrm

Web Server
bdsWeb

Firewall 1

BDS Server,
Database Server,
and
Seismic Data Store 1

Seismic
Data Store2

Workstation
Client

Blacknest Computer Systems