BEAM BDS Remote Access

Project	Blacknest/Bds
Date	2021-05-28
Reference	BdsRemoteAccess
Author	Dr Terry Barnaby

Table of Contents

1. Introduction

This short document covers a brief investigation in methods of providing remote access to the BDS (Blacknest Data System) system for users of the system. The idea is to allow easy and quick access to the seismic data and Metadata held on the BDS for home working and other such purposes as well as acess to the new Event data/metadata API's that will be implemented..

Backnest have a number of, Linux based, servers and workstations on site that are used to store and process the seismic data. They have various home and remote working computers, including ageing Apple Mac laptops and desktop computers.

The BDS supports four core methods of accessing the BDS:

- Simple command line programs.

- X11 GUI applications.

- HTTP/HTML WEB interface.

- Event data/meatada access.

- Direct API access from C++, Python and PHP user programs.

We cover some possible methods of remote access here.

2. Remote SSH access

This is the commonly used method to access the remote systems and data. The user can login over the Internet and possibly through a VPN gateway to one or more Blacknest computers using the secure SSH protocol. This allows them to run remote command line applications. It is also possible to transfer files

using this protocol using command line and, in some cases, GUI file manager programs. It is also possible to run remote graphical GUI programs, such as the bdsAdminGui program over a SSH network socket tunnel. However the current generation of Linux X11 GUI applications have relatively low performance over a lowish bit rate Internet interface and so are difficult to use. Also with the coming of the Linux Wayland GUI interface, this will not be possible in the future.

It is also possible to tunnel network socket connections over a SSH connection allowing remote locally running, BDS command line and GUI client applications to communicate with the BdsServer at Blacknest to provide the interactive speed of a local application although with slow data transfer. However this method requires the BDS client programs to be able to run on the remote workstations. This will be fine if Linux systems are used, but will be difficult for Apple MAC computers as the BDS would need to be ported and maintained on this platform.

So the SSH method provides a good, standard, base level access. However it is not suitable for running remote GUI applications or providing direct program access to the BDS data with users Python or C++ programs.

3.   BDS Clients Running Locally With Docker

If the remote systems are running a Blacknest supported Linux system, it is easy to install and run the BDS command line and GUI applications locally. They would communicate with the BdsServer at Blacknest over an SSH tunnel and/or VPN connection. Also any user programs in C++, Python or PHP would be able to directly communicate with the remote BdsServer.

However, for an Apple MAC computer we would need to port and maintain the BDS programs and any other seismic programs on a Apple MAC computer. One possible method to make this easier would be to use the Docker system to run these applications on an Apple MAC.

We have done a quick test of building a docker-ce image with the BDS client application RPM's and it is simple to build and does work. You can run clients under Fdeora33 with commands like "xhost +; docker run --net=host --env="DISPLAY" bds1 /usr/bds/bin/bdsAdminGui5" or start a command line shell with "docker run -it bds1 /bin/sh".

The BDS docker image is about 1GByte and 500 MByes when compressed with gzip.

```
REPOSITORY              TAG        IMAGE ID       CREATED         SIZE
bds1                    latest     3e313f2453da   17 minutes ago  918M
```

We can provide the scripts to create a docker_ce image and/or the BDS client image itself if wanted.

However it will have these issues:

- Some of the Apple MAC's in use are old with very limited capabilities: 2 CPU cores, 2 GBytes of RAM and 60 GBytes of disk space. To run a suitable Linux Docker image on an Apple MAC it would effectively run a virtual machine and this would need significant resources. This would likely mean the system would need to swap and be very slow.

- The user experience of starting the applications may need some work so they can be run from the GUI menu's and command line in an easy way.

4.   BDS HTTP/HTML and HTTP/JSON WEB API

The BDS supports a HTTP/HTML WEB interface for data and Metadata access. It is possible to run a suitable BdsWeb server at Blacknest, on a suitable host and provide access to this over an SSH tunnel and/or VPN connection. This would not allow for the full features of the BDS GUI applications, but would provide access to the seismic data and Metadata.

However, local C++, Python or PHP applications could not access the BDS system directly. One thing we could develop is a BDS HTTP/JSON API interface. This would allow a, say Python program, to access the full features of a BdsServer using a HTTP/JSON ASCII type interface over the network port's 80 or 443. We believe it would be about 5 man days work to provide the BdsServer side support for this (The API can be mainly automatically generated based on an updated BIDL system). This would allow Python programs to access the system using generic JSON capabilities. We could then go further and generate a small Python library that would match the BDS's API library to ease usage of this system. That aspect would probably take another 5 man days work.

This API might be useful for local as well as remote Python applications as being ASCII based it will be less reliant of BDS Python libraries compiled for particular Python versions. Note however, that being ASCII based it will be less efficient that the current binary BDS API and could have issues with numeric resolution of the data and Metadata.

5.   BDS Cloud Access

All of the above methods require seismic data/metadata access and seismic processing applications to be ported to and maintained on the various remote computers in use. Keeping these programs up to date and matching those available on Blacknest's internal systems will be difficult. Also having program/library/system environments that can be used to completely replicate a previous processing analysis will be almost impossible. Another major aspect is that the seismic data and metadata would end up passing over the Internet to be stored on remote client systems. There are security aspects to this especially as some data may be sensitive. The seismic processing would also be carried out on capable processing servers rather than performance limited laptops computers. Updating the remote Apple MAC's to the newer ARM based equivalents would be much less of an issue.

Another method of remote access is to keep all programs and data at Blacknest and provide one or more seismic application servers (BdsAppServer). Remote access would be provided by a suitable GUI remote application/desktop access system. With this the user would just run all data processing on the servers at Blacknest and there would be no special programs that need to be installed and maintained on the remote laptop/desktop computers apart from the remote application/desktop access system.

This really simplifies the software installation and maintenance issue. All of Blacknest seismic processing software would be maintained on the one Linux OS and on one or more servers at Blacknest. It would also be possible to setup processing environments so a particular versioned set of libraries and applications could be used to repeat a previous analysis exactly as it was done and share that processing environment with other users.

BEAM   BDS Remote Access

There are various remote access systems available, some commercial and some OpenSource. One issue in the past with these is the latency and speed of operation and indeed many of these systems have that issue today especially over slow Internet links. However the more recent video compression protocols and codecs, such a H264 or H265/HEVC, can be used to provide latency and speed performances close to local access performance across reasonably slow Internet links. Hardware encoding/decoding capabilities aid in this. Remote gaming access engines are starting to use this method.

Unfortunately the OpenSource remote access protocols are only just starting to make use of this. One we have found is xpra that does have the ability to use hardware h264/h265 encodings, but is still not ideal in its usage of these.

Some information on in house, low-level tests of remote access using H264 are given in the next section.

5.1. Remote access Using H264

We did some latency and speed tests across an Internet connection using hardware H264 encoding to see what theoretically can be achieved with remote desktop access.

The server was a 4 x Core, 4 s HyperThreads, Intel(R) Xeon(R) CPU E3-1245 v5 @ 3.50GHz based server running Fedora33. This Xeon CPU has an Intel GPU inside.

The client system was a 6 x Core, 0 x HyperThreads, Intel(R) Core(TM) i5-9400 CPU @ 2.90GHz PC running Fedora33. This Core i5 CPU has an Intel GPU inside.

The Internet link was a 30 MBits/s both ways (actually 150/30 but as used in a bi-diectional manner equates to a 30/30 link) Openreach/Zen/FTTP based link and OpenVPN as well as the SSH was being used across this 20ms ping time link.

Using hardware H264 encoding, provided by the Intel CPU's GPU, we obtained the following figures for a 1024x768 GUI application being displayed across the network. This was done using gstreamer. For this we updated the display continuously at 30 frames per second which would provide a low latency GUI experience, like watching a video.

```
# One 1024x768 client
Host        Process     ssh    System
====================================
CPU Beam:   18%          1%    3%
CPU Study   7.6%         4%    5%
NetworkRate:      1.1 MByte/s

# Running two 1024x768 clients
Host        Process     ssh    System
====================================
CPU Beam:   2*18%       2*1%   6%
CPU Study   2*7.6%      2*4%   9%
NetworkRate:      2.2 MByte/s

# Running three instances
Host        Process     ssh    System
====================================
CPU Beam:   3*18%       3*1%   9%
CPU Study   3*7.6%      3*4%   12%
NetworkRate:      3.2 MByte/s
```

The "top" process CPU usage (based on one core) and the overall system CPU usage (based on 4 or 6 cores) is given. The SSH was using standard encryption protocol and the H264 encoder using default settings. The OpenVPN connection was run on separate gateway/firewall hosts.

This shows that with a modern system that has hardware H264 encoding/decoding capabilities it would be quite feasible to run remote applications over a reasonable Internet connection (10 MBits/s) with low latency and good speed. A basic server could easily support 3 concurrent users running intensive graphics applications with little CPU usage overhead and with more normal usage I would have thought 10 or more users could be supported from the GUI viewing side of things (seismic analysis program usage may require a lower user to server ratio).

In reality the systems would only send GUI updates when the screen actually changes so the overall CPU usage and data rates would be much lower than this. But at 30 frames per second it would be possible to watch videos displayed in the remote applications. Using H265/HEVC encoding would reduce the bandwidth for the same quality by about half.

So this looks doable and it looks like a suitable single server could support say 8 or more concurrent users very well depending on the seismic data processing usage.

The only issue is that the remote desktop access systems don't fully support using H264/H265 in this mode as yet. We believe with a small amount of work one of these systems, especially xpra, could be extended to provide this or indeed might do this with suitable configuration options.

5.2. Remote Desktop Access Software

This lists some of the remote desktop access software that is available. The common VNC and RDP protocols are supported as well as some custom ones. Some of these show a complete remote desktop, but some, notably xpra supports running a single application remotely that is shown on the local display just like a local application (like an X11 over SSH GUI application).

Xpra	https://xpra.org/ https://hasanyavuz.ozderya.net/?p=503 XPRA does work under Fedora33. Seems fast but has noticeable latency. Support's H264/H265 and hardware encode/decode, but not sure how well this is implemented.
KRDC	KDE VNC/RDP client
vncserver	Remove VNC/RDP X11 server
X2GO	
TigerVNC	Does this support H264 ?
libfreerdp	
Google Chrome	https://remotedesktop.google.com/?pli=1

apache guacamole	
teamviewer	Commercial application

5.3. XPRA

XPRA uses a custom protocol and has clients for Apple MAC, MsWindows and Linux systems. One nice feature is that it supports single application operation as well as full desktop operation. With single application operation a single remote application can be started from a provided GUI menu or via the command line and is displayed and interacted with just as if it were a locally running application. The system also supports cut and paste, sound, files and even webcam access if enabled.

5.3.1.XPRA With Fedora33

Install "dnf install xpra Xvfb xpra-codecs-freeworld xpra-html5 xpra-udev python3-pyxdg"

Test1
Run on remote desktop: "xpra start ssh:terry1@beam –dpi=96 --start=firefox"

Test2
Start on server: "xpra start ssh:terry1@beam --encodings=h264 --start=firefox"
Run on remote desktop: "xpra attach ssh:terry1@beam --encodings=h264"

Other
Stop xpra servers: "xpra stop all"

When started a toolbar menu of the remote applications and xpra running status is provided.

5.3.2.X2Go With Fedora33

Install: "dnf install x2goserver x2goclient"

Start client with x2goclient and configure a session. Works as whole desktop, about same speed as xpra. No H264 support that I can see.

6.   Conclusions

For now the remote access using SSH works, although it is awkward and especially slow with GUI applications. Providing access to a Blacknest BdsWeb server over the VPN or and SSH tunnel will provide easier access to the data and metadata and can be setup either on the main BdsServer or a separate server as desired. This just needs configuration and setup and would have the benefit of allowing usage testing of the BdsWeb interface prior to the general external users BdsWeb interface being implemented at Blacknest.

However, from the quick investigation of the needs especially those going forward, I believe the simplest and best solution is to use a suitable remote desktop access system and run all of the applications at Blacknest on an application server (BdsAppServer). This would mean that no special software, apart from

the remote desktop access program, is needed on the Apple MAC or other remote computer and as long as that computer has the capabilities to show a full screen H264 video (which most would have these days) no great Laptop/PC capabilities are required. The Internet connection would need a greater than 10 MBits/s internet (5 MBits/s for H265) which again should be possible in most cases these days.

All data and processing would be at Blacknest and there would be a single encrypted and SSH Key + password connection to Blacknest required. So security of the system would be relatively easy to achieve.

Some possible problems:

1. The remote access display system will need to support low latency and good speed to make the system usable. There will have to be some work done to improve one of the remote desktop systems or at least experimenting with configurations to achieve this.

2. The Blacknest server used could be a normal workstation for now, but setting up a dedicated BDS application server (BdsAppServer) for both internal and external use would likely be a better approach. This server would need to support H264/H265 hardware encoding so would need some form of graphics GPU that Linux supports H264/H265 encoding on.

3. The Blacknest Internet connection would need to support a bit rate for the number of remote users expected. With H265 say 2.5 MBits per user ?

4. The Blacknest power supply to the BdsServer, BdsAppServer and internet connection would have to be stable to provide good uptime reliability. The use of UPS's and perhaps backup generators could be considered. Looking at lowering the power usage of these servers would help along with helping Blacknest's Green credentials.

Setting up a test using the default xpra to an existing workstation or new workstation system at Blacknest would be relatively easy. With a bit more work xpra access can be improved by experimenting with configuration options and/or working with the developer to provide better h264/h265 support. Alternatively we can setup a BdsAppServer at Beam for Blacknest users to try out. We think 5 man days work would achieve working on improveing the xpra application and testing this on a Beam BdsAppServer.

A second option is to use the BdsWeb server to provide access to the data and Metadata at Blacknest. This does have access and data security issues though and Apple MAC seismic applications would need to be supported. We can extend the BdsWeb system to provide more features as wanted/needed.