James Keeler

How to run the FitNet system and basic troubleshooting tips.

Operations Manual

FitNet System

# Introduction

This manual is intended to be used when operating the FitNet system software. The scope of this document is strictly the operation of the system and basic troubleshooting. Failure to follow the procedure in this document could lead to unexpected results, if any at all.

## FitSend

This program is intended to read a stored high definition video file with a resolution of 1920x1080 and a playback frame rate of 30 fps. Still images of this video are captured to the sender’s hard drive, then packetized, and sent to a list of receivers. Once sent, each image is deleted from the sender’s hard drive.

## FitReceive

This program is intended to receive the packetized images from FitSend and report on packet loss and throughput. Video playback was not implemented in this version of the software due to the nature of the project.

# Prerequisites

The operating environment must be configured as described in the Installation Manual. There must be at least two network nodes, one which will be the sender and one which will be the receiver. The corresponding Python programs must reside on these computers.

For optimal results, it is suggested that the network be capable of utilizing OpenFlow and have at least 100 Mbps of available bandwidth. A flow should be created with quality of service (QoS) rules which maximize the reliability of UDP traffic.

The computer receiving the network packets must meet the minimum hardware requirements for the chosen operating system. It is highly recommended that the sending computer have at least the following hardware specifications:

* 4 processing cores
* 4 GB of RAM
* 1 Gbps of available network bandwidth
* Solid state drives (SSDs) or fiber channel connected storage area network (SAN) drives

# Program Execution

To run the software, you must first launch the FitReceive program on each of the receiving systems, and then launch the FitSend program on the sending computer. Once FitSend begins executing, it performs the following steps:

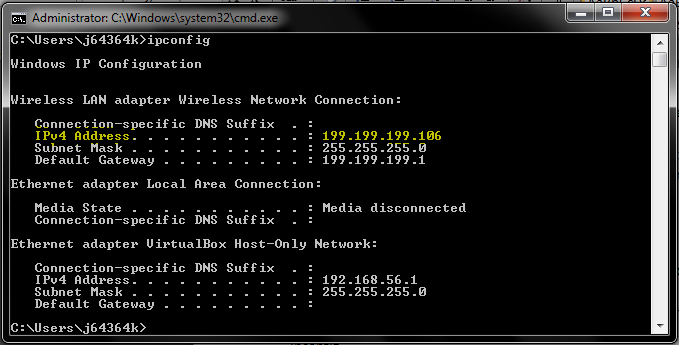
1. Open the video and captures still images to the hard drive
2. Open a UDP connection with the receiver(s)
3. Open an image captured from the video and divides it into 1081 equally sized packets, each containing a single line of resolution from the image (all images are 1920x1080)
4. Once the entire image has been sent, the file is deleted
5. Repeat steps 3 & 4 until no more images exist
6. Close the UDP connection

## IP Addresses

Prior to launching any of the software, you must gather the IP addresses for the computers acting as sender and receiver(s). On a Windows computer you can find the computer’s IP address by pressing Windows+R keys and typing cmd into the dialog box. At the command line, type the following:

ipconfig

Depending on the computer’s available and active network connections, the IP address information might look something like Figure 1. Once you have determined which adapter is being used, record the IPv4 information for later use.



Figure

The same idea is used in Linux. Open a console window, and type the following:

ifconfig

This command returns information similar to that in Figure 1. Again, we are only interested in the IPv4 address of the adapter being used.

## Launch the Receiver

To launch the receiver on a Windows computer type the following:

FitReceive.py

To launch the receiver on a Linux computer type the following:

python FitReceive.py

## Launch the Sender

Using the IP address in Figure 1 as an example, to launch the sender on a Windows computer and send data to the receiver IP address in Figure 1, type the following:

FitSend.py 199.199.199.106

In Linux it is very similar, to launch the sender type the name of the FitSend program followed by the IP address of the receiver:

python FitSend.py 199.199.199.106

# Troubleshooting

## Unable to Send/Receive

If you are unable to send or receive packets, verify connectivity between the network nodes by pinging from one node to another. For example, if we have two Windows computers and the sender fails to create a connection and send packets we can ping the receiver by pressing Windows+R keys and typing cmd into the dialog box. The type the following command:

ping 199.199.199.106

The exact same syntax can be used in a Linux console window.

## Disk Issues

If you get errors stating a file cannot be found, access was denied, or disk is full, you will need to investigate the disk subsystem. Be careful not to capture more images than you have available storage space. Each image captured is over 6 MB and the video plays at 30 fps, so you will use 180 MB of disk space for every second of video recorded. For example, if you capture images from 10 seconds of video, you will use 1,800 MB of disk space

# Known Issues

## GENI Linux Support for GitHub

For some reason, the Linux computers on GENI to not like unpacking GitHub files. The workaround for this is to copy the text from the GitHub repository using your favorite web browser, and paste the text into a blank file on your Linux computer in GENI. This copy/paste technique usually omits the first few characters of the code, so always to verify that the first few lines of code are correct before saving the file.

## Packet Dropping Omitted

The choice was made to omit the packet dropping portion of the system due to time constraints. This should be investigated and implemented by future teams. To effect this properly, these teams should be prepared to alter the OpenVSwitch codebase and add a module that allows a user to programmatically drop packets based on protocol. For example, when fully implemented the OpenVSwitch should perform the following concurrent actions on an incoming packet stream:

1. Allow all packets and route to port 1
2. Allow 1081 packets, then drop 1081 packets, routing all packets to port 2
3. Allow odd numbered packets while the total packet count is less than 1081, then drop 1081 packets, routing all packets to port 3

By performing these three actions on the incoming packets stream, the switch essentially multiplexes the packets across three ports where computers attached to port 1 receive all packets in the stream (full video frame rate and full video resolution), computers attached to port 2 receive every other group of 1081 packets (half video frame rate and full video resolution), and the computers attached to port 3 receive every other packet from every other group of 1081 packets (half video frame rate and half video resolution).