# Night Eye Guardian Design Document

PROJECT MAY14-29

**Client**
Adan Cervantes
*contact@sidsknowmore.org*

**Advisor**
Diane Rover
*drover@iastate.edu*

**Project Members**
Nicole Bruck
*nicole.a.bruck@gmail.com*

Jeremy Dubansky
*dubansky@iastate.edu*

Daisy Isibor
*isibord@iastate.edu*

Eric Woestman
*eric.woestman@gmail.com*

# Table of Contents

# Introduction

## *Executive Summary*

Doctors that are performing research in Sudden Infant Death Syndrome (SIDS) need better tools for collecting and analyzing data from infant sleep studies. Our client's theory is that a major factor in determining the risk of SIDS is the percent of time the infant spends in the Slow Wave Sleep (SWS) sleep state versus the Rapid Eye Movement (REM) sleep state and the sleeping pattern overall.

The issue with analyzing when the infant is in these sleep states is a challenging one. Parents do not want intrusive tools in their child's sleep area and they want their child to be safe. Reliability and effectiveness of the monitoring product is essential also. Researchers are looking for a way to gather the sleep data of infants without inconveniencing the parents or child.

## *Statement of Purpose*

We created a system built upon previous work from another senior design team to record a sleeping infant and use the video to determine the percent of time the infant is in each sleep state. There is also charting of the detected activity level, heart rate and breathing rates that is displayed as a function of time.

This project consists of developing a web enabled tool by integrating a web camera with special software which graphically displays infant sleep data to researchers conducting Sudden Infant Death Syndrome (SIDS) Polysomnography (sleep studies). The project requires integration of existing open source components to detect heart rate and customizing a user interface to display the video, activity and heart rate graphs. This also includes developing an algorithm to combine the activity and heart rate data to determine sleep states as a function of time.

The team made use of special software developed by MIT called, Eulerian Video Magnification (EVM), which is available for non-commercial research purposes and can be downloaded from: http://people.csail.mit.edu/mrub/vidmag/#code. This software allowed for video enhancement, making motion and blood pressure (redness pulsing through visible skin) much more evident.

The system hardware components consists of a motion detection camera, remote server, user's website and an optional wireless mobile device, to monitor infants as they sleep inside their cribs. Figure 1 in Appendix B shows the high-level system overview. The video data is currently uploaded to a server at SIDSKnowMore.net (hosted by 1and1.com). The researchers will use the system and data to evaluate the infant's SWS/REM sleep states and cycles.

## *Definitions*

### SIDS

Sudden Infant Death Syndrome: The sudden death of an infant that is not predicted by medical history and remains unexplained after a thorough forensic autopsy and detailed death scene investigation. As infants are at the highest risk for SIDS during sleep, it is sometimes referred to as cot death or crib death.

**SWS**

Slow Wave Sleep: The deepest stage of sleep, and is often referred to as deep sleep. It is characterized by shallow breathing and little or no eye movement during this stage.

**REM**

Rapid Eye Movement Sleep: The lightest stage of sleep. It is characterized by quick, random movements of the eyes and paralysis of the muscles.

**MIT EVM**

Massachusetts Institute of Technology's Eulerian Video Magnification Software: An open-source software to reveal temporal variations in videos that are difficult or impossible to see with the naked eye and display them in an indicative manner. Their method, which they call Eulerian Video Magnification, takes a standard video sequence as input, and applies spatial decomposition, followed by temporal filtering to the frames. The resulting signal is then amplified to reveal hidden information. Using this method, we are able to visualize the flow of blood as it fills the face and also to amplify and reveal small motions. This technique can run in real time to show phenomena occurring at temporal frequencies selected by the user.

**FOSCAM IR IP camera**

The Foscam Wireless IP Camera features high definition 1280 x 720p video resolution, h.264 video compression, built-in DVR via SD card (up to 32gb), two-way audio, wifi-N capability, pan/tilt, remote internet viewing, motion detection, night-vision as well as network video recording capability.

**1and1.org**

A web and server hosting company owned by United Internet, a German Internet company.

**ZoneMinder**

A free, open source closed-circuit television software application developed for Linux.

## *Operating Environment*

The system is designed to keep track of the duration and number of the infant's SWS and REM sleep states. The system determines the SWS and REM sleep states and cycles with the help of the MIT EVM software and an algorithm developed based on the motion activity and heart rate of the infant. A web camera collects the infant's video data and uploads the data to a secure web site. The EVM software enhances the video which is then detected for motion. The team can then correlate the movement rates and heart rate (simulated) to derive the SWS and REM sleep states and cycles. The duration and time for each sleep state and cycle is plotted graphically for an entire night and is presented to SIDS researchers via the internet.

### *Intended Users*

From our specifications there are three user cases, each one displaying different data for research or just video output. Depending on who needs to see specific information a user might be granted access to view more than others. Figure 4 in Appendix B shows the page to access archived data and review it. The users are defined as follows:

#### Admin

Has the most options available to any user, used to control most of the website and other user accounts. The webmaster and software developers would be the primary administrators.

#### Researcher

This user will be able to view all the video and the data that is necessary, including the archived videos, appropriate graphs, data tables, etc. The researcher needs to be able to view everything on the site but does not need to change other user account statuses.

#### Parent

Allows the parents to view what the researchers see but without the actual data they are collecting. This is a simplistic view just so the parents get a basic understanding of what the researchers are doing and to have full access to what information is being accessed.

## Requirements

### *Technical Requirements*

- ❖ Using the FOSCAM IR IP camera provided by our client, a video of the infant in focus should be captured in real-time.
- ❖ The video stream displayed on our webpage needs to be processed through the MIT EVM Software to enhance color and motion changes.
- ❖ The amplified video will be live streamed to a website for the target users to view at all times.
- ❖ The REM/SWS sleep states would be determined based on the IR motion detection (activity) on the FOSCAM IPcam and the heart-rate, which would be a simulation not the infant's actual heart-rate.
- ❖ Graphs of REM/SWS sleep states versus time, motion (activity), and heart rate would be displayed on the website.

## Objectives and Approaches

### *Objectives*

- ❖ Manage remote video cameras.
- ❖ Provide a video stream to the server over a network.

- Import video feeds provided from IP Cameras to the server.
- Transform the raw video into a clearer image of motion, provided from MIT EVM software.
- Store motion captured information into a database.
- Provide a clean user interface for each user case.
- Display the edited video of motion for the users to watch.
- Display a graph of the motion sensing data synced with the edited video times.
- Display a graph of heart rate data being streamed (simulated for now)
- Determine Sleep States based on activity and heart rate
- Display a graph and overall summary of mapped sleep states

### *Design Approach*

During the design process of the Night Eye Guardian, we started out by determining the objectives of the project and then went ahead to compile the necessary functionality. Based on the functionality of the project, we broke it down into several modules.

### *Development Approach*

The project modules will be implemented independently with but with focus on integration with other relevant modules (see Figure 9 in Appendix B).

### *Validation Approach*

Before proceeding, all the major design decisions will be verified with the client. Each module was tested individually by the team member assigned to it for functionality and performance. We will then test how each module interacts with the others.  Lastly, we had an overall testing for the system's end-to-end flow. This final testing was thorough and attempted to cover edge cases where reasonable and possible.

In addition to performance and functionality testing, we kept constant contact with our client and had demos after every feature was added. The client confirmed the changes and offered suggestions for the next iteration.

## Risks and Mitigations

The following is a list of potential risks to this project, and our strategies which we proposed to mitigate those risks.

#### Lack of Fully Formalized Requirements with Client

Since there are a lot of unknowns in this project and we do not fully know what features need to be implemented to make a successful product, it is likely that the requirements will change drastically in the future. Therefore, we will use an agile development methodology so that we can rapidly react to changing customer demands.

Due to issues with the previous design of the system, halfway through the project the team made a decision to complete redesign the overall structure of the software. Our client was made aware and encouraged our changes. We evaluated the risks of staying with the current design and possibly failing to complete the system or attempting a new design which could also fail. Due to the struggles and results that we had been experiencing with the previous design, it was deemed that the software would not be successful as is and the changes were made.

#### Lack of Availability

Our entire system depends on a hosted server and if, for whatever reason, the server is unavailable, we will be unable to make any progress or effect on the project. Throughout development, the server was always available and responsive. There were also periodic backups to ensure that our system was reinforced and reliable.

#### Lack of Familiarity with Web Development and Mobile Development

Our team is fairly unfamiliar with development of websites and mobile development. Fortunately, there is plenty of documentation available for us to learn how to use the products. However, this unfamiliarity inevitably slowed us down. The softwares required to be familiar with after changing our design were well-documented and easy to understand. The team was able to make progress with a smaller learning curve.

## System Design

The system is run on a remote server that will be taking care of most of the actual processing going into the project. The server is running an open source Linux operating system called CentOS. This operating system allows is to run multiple utilities to host a web platform that we developed for.

Running on the server is an open source platform to monitor video streaming called Motion. The original system design depended upon ZoneMinder yet after our redesign in May, the system was recreated to be centered on Motion. This software provides access to remote IP cameras that can provide video streaming back to the server. Motion takes care of most duties when it comes to filling videos and allows easy access to all the data associated with the videos. Motion was well-documented enough and simple to integrate into our website.

### *User Interface*

The user interface is an elegant website that is designed to display the data in a useful way. This project required an intuitive front end that allows users to view the necessary data for the specific user case. There are a few features that are very important to the project for instance, video display and related data.

### *Database*

The database module stores data received from the cameras and then retrieves that data for processing and viewing. It also stores user information, such as, parent accounts, researcher accounts, etc.

### *Server*

The server module controls all of the data processes run on the server. It receives video data from the cameras and stores it in the database. The video is then run through the processing and data extraction algorithms. It also displays video and data to the user through the user interface.

### *Architectural Diagram*

See Figure 8 in Appendix B.

### *Design Issues*

Below are mention the major issues that were encountered in our design. More are mention in the Informal Notes for Future Senior Design Groups.

#### ZoneMinder

One of the major issues was the software we had originally intended to use for capturing video and processing. We were unable to properly configure the cameras to work with ZoneMinder and in the middle of implementation, made a major change to our design to then work with Motion, another open-source software video capturing.

#### Iowa State Network

Another issue the use of our cameras on the Iowa State network. We had to work with our Systems Support Group to register the MAC address of our cameras so that we could connect them wirelessly to the network.

## Module Decomposition

### *Video Capturing Module*

This module consists of an IR Webcam camera. It records the infant during the sleep study. Despite the ability to stop recording when not detection motion, the camera is always recording in order to collect data throughout the study. The camera takes a high quality video that can be processed to determine data. The camera is connected to a network router in order to send the data to our remote server.

#### Validation Approach

The IR Webcam is able to hold a continuous connection to our servers and upload video data while recording. Tests can be performed to ensure that the camera provides a continuous video stream to our server.

### *Video Processing Module*

This module uses the MIT Eulerian Video Magnification software to process incoming video to enhance small changes of color and motion. By enhancing the color and motion we can better detect levels of activity of the subject.

**Validation Approach**

The MIT software we are using helps us to detect motion from our cameras, which helps in providing a higher rate of motion detection than what the camera can provide currently.

## *Server and Database Module*

This module consists of the server hosted by 1and1.com that contains the database for storing video input. The server receives the video data from the camera used in the study, stores it, and runs it through the Data Extraction and Sleep State Detection Modules. The resulting data is also stored in the database.

**Validation Approach**

The server is reliable and holds a good connection to the cameras while they are connected. Tests were performed on the current server with the provided cameras from the client, with the requirement that the tests needed to hold for over 99% of the time. They passed.

## *Data Extraction Module*

This module takes the processed video and extracts data from it. This data includes the infant's motion and heart rate. An algorithm will be used to extract this data based on the enhanced color and motion from the Video Processing Module.

**Validation Approach**

By using multiple data inputs, the algorithm is accurate in reading the data we need to display. We will used examples from previous research where they determined sleep states based on given activity and heart rate.

## *Sleep State Detection Module*

This module consisted of an algorithm that takes the data from the Data Extraction Module and uses it to determine the sleep states of the infant. This compared the relative changes in activity and heart rate to determine when the infant begins each sleep state.

**Validation Approach**

This module was validated by passing in a pre-defined variation of data. We would input data that implies high activity and heart rate, input some with low activity and heart rate, as well as data with a combination of high and low activity and heart rates. Based on the input we would observe the sleep state determined by this module and compare it with the expected sleep state.

## *User Interface Module*

The user interface module allows researchers and the parents of infants to view the data from sleep studies. Users are able to view the video received from the Video Capturing Module and the data from the Data Extraction Module and Sleep State Detection Module (See Figure 4 in Appendix B).

### Validation Approach

UI Mockups and screen flows were approved by client and advisor. We then created a general prototype that simulated the graphs and video streaming (plain images) and confirmed with the client that he approved of the overall website page navigation. For each component of the data page, we prototyped and demonstrated to the client.

## *Module Diagram*

See Figure 9 in Appendix B.

# Standards

Our project transmits a lot video and raw data over the internet and the web so we have to use several web and internet protocols. Below are the standards we followed and how we used them.

### Ethernet – IEEE 802.3[1]

The major reason we are using Ethernet is because of the high performance and security involved in using this protocol. IEEE standards committee for Ethernet has introduced new standards to define higher performance variants. Each of the Ethernet IEEE standards can be uniquely identified using its reference. We believe the features of the IEEE 802.3 standards fit our needs to connect the I/O system to the router using an Ethernet cable.

### Wireless – IEEE 802.11 b/g[2]

The Night Eye Guardian utilizes wireless communications between the routers or any access point, the FOSCAM IR IPcam, and possibly the users that would be watching over the infant using other Internet enabled such as PC stations and handheld devices.

For this implementation we would be following the IEEE 802.11 standards. These set of standards are used for implementing wireless local area network (WLAN) computer communication in the 2.4, 6.3 and 5 GHZ frequency bands. The 802.11 family consists of a series of half-duplex over-the-air modulation techniques that use the same basic protocol. The client provided us with a FOSCAM IR IPcam. This camera uses the 802.11/b/g standards which support bandwidths from 11 Mbps up to 54 Mbps. Some of the benefits of these standards are that they produce fast maximum speed and a signal range which is good and not easily obstructed. This perfectly suits our needs of streaming live video over the web.

### Security – TLS, WPA/WPA2[3]

[1] http://standards.ieee.org/about/get/802/802.3.html

[2] http://www.enhancedwirelessconsortium.org/wlan-ieee-802-11n-standards/

[3] http://en.wikipedia.org/wiki/Wi-Fi_Protected_Access

Another important aspect of this project was how to deal with the signal security. When a user uses the NEG to watch over an infant over the Internet, they had to ensure that it's done over a secure and reliable signal amongst parties – the IPcam, the server, and the device being used by the user.

The Transport Layer Security (TLS) Protocol provides enough information to make sure that communications between the clients (users) and server's applications are designed to prevent eavesdropping, tampering, or message forgery. Unlike WEP, Wired Protected Access (WPA) encrypts access. Thus, anybody can guess key to get access to network but cannot see traffic between another computer and access point. WPA secures individual connection for its duration and recalculates connection encryption each time same computer access the network. In 2002, it was found that WPA has flaws in its design and still could be decrypted, although it takes a long time. Wi-Fi alliance went and created newer version of WPA called WPA2 with some more features.

#### Video – MJPEG[4][5]

The FOSCAM IR IPcam supports compressed MJPEG standard for video recording process. The Moving Picture Experts Group is a working group of experts that was formed by ISO and IEC to set standards for audio and video compression and transmission. There are many version of MJPEG standards, the one that we would use is the MJPEG-7, which is a multimedia content description standard. It was standardized in ISO/IEC 15938. This description was associated with the content itself, to allow fast and efficient searching for material that is of interest to the user. MJPEG - 7 is formally called Multimedia Content Description Interface. Thus, it is not a standard which deals with the actual encoding of moving pictures and audio. It uses XML to store and time code in order to tag particular events.

#### W3C standards[6]

W3C standards defines an Open Web Platform for application development that has the unprecedented potential to enable developers to build rich interactive experiences, powered by vast data stores that are available on any device. Although the boundaries of the platform continue to evolve, industry leaders speak nearly in unison about how HTML5 will be the cornerstone for this platform. But the full strength of the platform relies on many more technologies that W3C and its partners are creating, including CSS, SVG, WOFF, the Semantic Web stack, XML, and a variety of APIs. We will be developing a backend that will rely heavily on these technologies. It is important to be consistent with our methods of developing secure, reliable, extensible code.

## Customer Usage

Explanation of how a standard user (parent or researcher) uses the system.

### *Account Setup*

[4] http://webstore.ansi.org/RecordDetail.aspx?sku=ISO%2fIEC+15938-2%3a2002
[5] http://en.wikipedia.org/wiki/MPEG-7
[6] http://www.w3.org/standards/

The user can select the ‘Create Account’ link on the page and follow the directions. The authentication will then email the provided address and verify the user. Once the account has been verified, the user may log in.

### *Camera Registration*

By selecting the ‘Set-Up’ tab, the user may enter the url for a video stream (location of camera) and provide a name. This will immediately redirect the user to the stream page where the camera will begin to record.

### *Begin Recording*

If the user has already set-up a camera, the user may select the option under their set-up page and click on ‘Begin Recording’ to start a new video. Once the user has finished and would like to end the recording, select ‘Stop Recording’ and the video will be automatically stored in Archived Video.

### *Accessing Archived Video*

By selecting the ‘Archived Video’ tab, the user can select a previous recording to view. A page similar to the stream page will be displayed until navigated to a different page.

## Testing

Please see documentation attached at the end of this report.

# Appendix A

## *Sleep States and Cycles*

We introduce a proposition regarding the biogenetic evolution of sleep cycles and states. Our hypothesis considers that our need for sleep evolved through millions of years and was shaped by its survival benefits. Human sleep behavior that we observe today can be directly attributed to these inherited survival traits.

Millions of years ago, when humans were still part of the animal world, the traits which we adopted were based on giving humans some form of survival advantage over other animals. The animal world is based on a predator and prey scenario. A plausible reason for the function of sleep, from a human evolutionary standpoint, may be that sleep served to disable our physical movement at night, so as to minimize the risk of detection by predatory animals. However, total lack of physical movement with sensory shutdown while sleeping has the disadvantage of inhibiting our ability to detect a nearby approaching predator and escape, should the need arise.

A likely evolutionary compromise would be to minimize physical movement, but also maintain some form of sensory detection to alert us of possible danger while sleeping. Enter the two states of sleep: 1) Rapid Eye Movement (REM), or active sleep and 2) Non REM (NREM), or Slow Wave Sleep (SWS). These two states together make up one sleep cycle. We propose that sleep evolved with multiple sleep cycles, due to two major reasons. First, sleep allows our body the obvious benefits of health regeneration, and second, our genetics are preprogrammed with millions of years of survival traits, including sleep with alertness standby (REM State) to survive the predator and prey animal world.

Today, we observe sleep in two states. One state is the SWS sleep state and the other is the REM sleep state. Throughout the night, these states occur in multiple cycles. Both the SWS sleep and REM sleep states cause the body to be physically immobile to minimize the risk of detection by predatory animals, but during the REM sleep state, our senses are on standby mode with a higher level sense of alertness to improve the probability to detect movement and sound as an inherited survival trait against predatory nocturnal animals.

During the SWS state, the alertness levels decrease as seen by lower body temperature and cardiac and respiratory rates, which improves the probability to go undetected by predatory animals. However, the decrease in cardiac and respiratory rates must not go so low as to become unable to maintain proper homeostasis. Therefore, when the minimum homeostasis threshold level is reached, our neurological survival mechanism triggers a higher level of cardiac and respiratory rate to maintain proper oxygenation levels and stable homeostasis. This transition from lower cardiac and respiratory rates to higher cardiac and respiratory rates occurs from the SWS sleep state to the REM sleep state. Therefore, by going through alternate REM and SWS sleep states during the night, this scenario most likely provided humans the best of defense against predatory animals while asleep. We proposed that this maybe a plausible explanation for the REM and SWS sleep states.

## *REM/SWS and Thalamocortical Neurons*

SIDSKnowMore believes that a key factor associated with SIDS, relates to sleep states and cycles. At birth, infants have chaotic EEG waveforms due to exponentially extreme neuronal growth. As part of the infant's adaptation to its environment, the wake/sleep cycles are attempting to synchronize with light as well as auditory and other environmental clues.  During sleep, the infant's neurological system is still coupled to electromagnetic environment and the thalamocortical neurons are in the intrinsic oscillatory state. Changes between Slow Wave (SWS) and Rapid Eye (REM) states are dependent on frequency and amplitude changes in the thalamocortical neurons and can be measured with EEG charts. The membrane potential is -85 mV during the SWS state and -65 mV during the REM sleep state, as detailed in Figure 6 in Appendix B.

During the REM sleep state, the infant's cardiorespiratory rate increases and more oxygen flows to the brain. Also, the infant's temperature increases slightly during REM sleep. During sleep, the thalamocortical neurons are in an intrinsic oscillatory state. In the oscillatory/bursting mode, the neurons in the thalamus become synchronized with those in the cortex, essentially "disconnecting" the cortex from the outside world. This disconnection of the cortex translates into the body, becoming physically immobile during sleep. During the SWS sleep state, the EEG recordings show the lowest frequency and the highest amplitude. When the infant is awake, the thalamocortical neurons are in tonic active state.  Figure 7 in Appendix B illustrates the thalamocortical neurons showing oscillatory mode in a sleep state and tonically active mode in an awake state. In the tonic state, the thalamocortical neurons transmit information to the cortex that matches the spike trains encoding peripheral stimuli.

The system makes use of these medical findings to derive epochs of SWS and REM sleep states and cycles and proposes that infants are at their highest risk to die due to SIDS during the Slow Wave Sleep (or non-REM) stages of sleep due to lower Inter Spike Intervals which are dependent on complex bursts of neuronal network behavior associated with limit cycle functions, which become unstable due to neuronal bifurcations associated with non-linear systems, such as an infant's brain.

## Appendix B: Figures

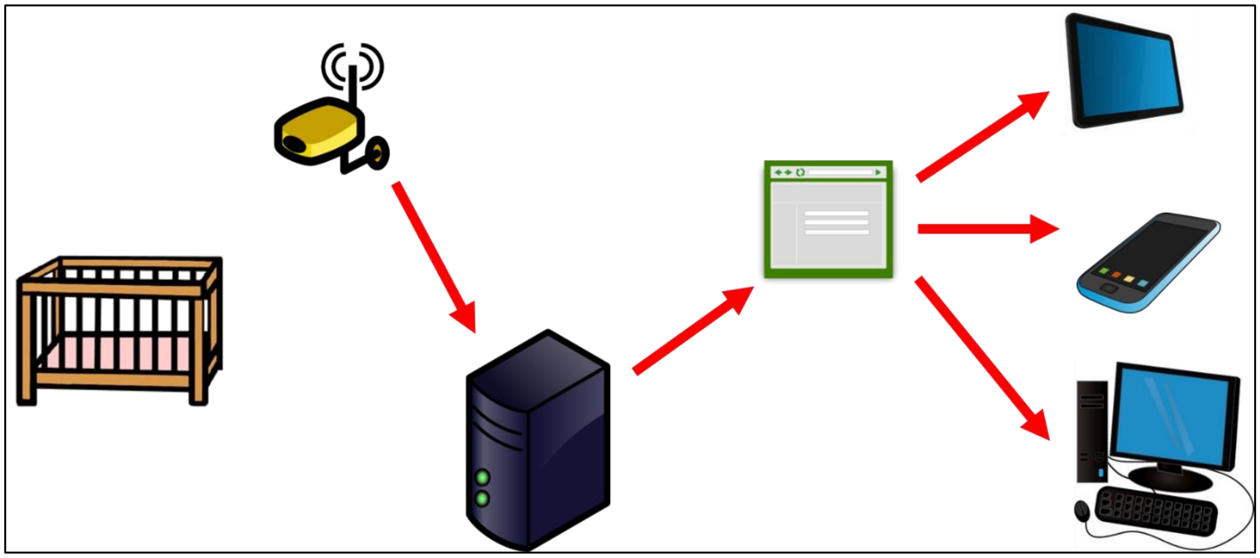


*Figure 1: System Overview*

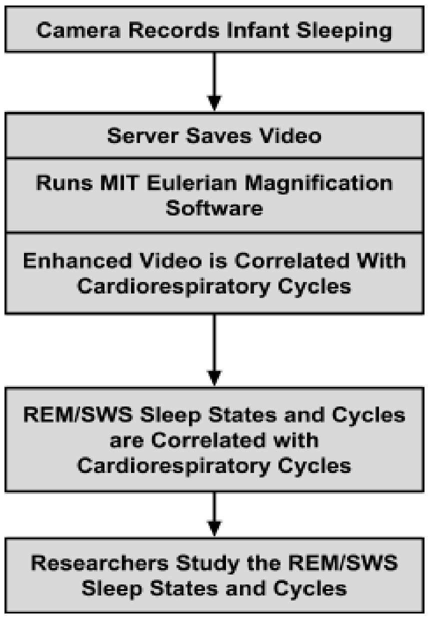

Camera Records Infant Sleeping
Server Saves Video
Runs MIT Eulerian Magnification Software
Enhanced Video is Correlated With Cardiorespiratory Cycles
REM/SWS Sleep States and Cycles are Correlated with Cardiorespiratory Cycles
Researchers Study the REM/SWS Sleep States and Cycles

*Figure 2: Functional Block Diagram*

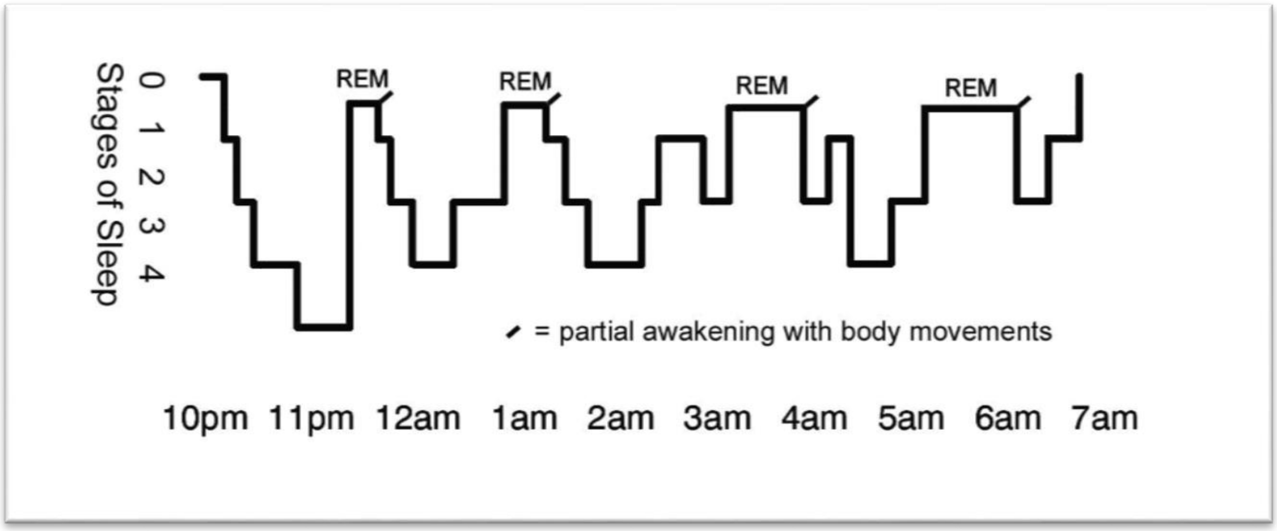


***Figure 3: Chart of Infant Sleep States and Cycles***

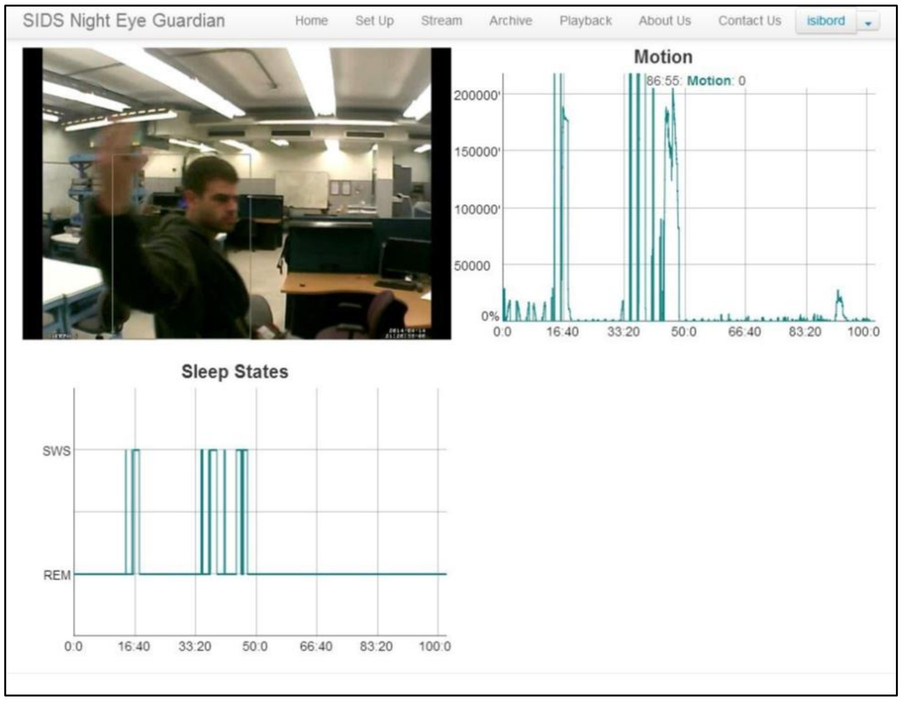
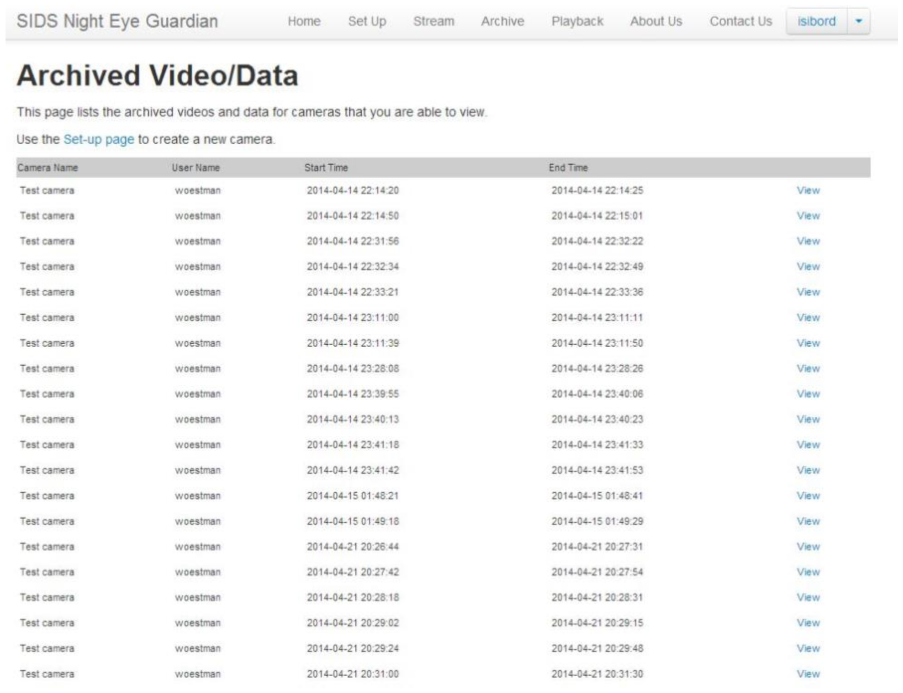


***Figure 4: Conceptual User Interface Main Page***



## Archived Video/Data

This page lists the archived videos and data for cameras that you are able to view.

Use the Set-up page to create a new camera.

| Camera Name | User Name | Start Time | End Time | |
|---|---|---|---|---|
| Test camera | woestman | 2014-04-14 22:14:20 | 2014-04-14 22:14:25 | View |
| Test camera | woestman | 2014-04-14 22:14:50 | 2014-04-14 22:15:01 | View |
| Test camera | woestman | 2014-04-14 22:31:56 | 2014-04-14 22:32:22 | View |
| Test camera | woestman | 2014-04-14 22:32:34 | 2014-04-14 22:32:49 | View |
| Test camera | woestman | 2014-04-14 22:33:21 | 2014-04-14 22:33:36 | View |
| Test camera | woestman | 2014-04-14 23:11:00 | 2014-04-14 23:11:11 | View |
| Test camera | woestman | 2014-04-14 23:11:39 | 2014-04-14 23:11:50 | View |
| Test camera | woestman | 2014-04-14 23:28:08 | 2014-04-14 23:28:26 | View |
| Test camera | woestman | 2014-04-14 23:39:55 | 2014-04-14 23:40:06 | View |
| Test camera | woestman | 2014-04-14 23:40:13 | 2014-04-14 23:40:23 | View |
| Test camera | woestman | 2014-04-14 23:41:18 | 2014-04-14 23:41:33 | View |
| Test camera | woestman | 2014-04-14 23:41:42 | 2014-04-14 23:41:53 | View |
| Test camera | woestman | 2014-04-15 01:48:21 | 2014-04-15 01:48:41 | View |
| Test camera | woestman | 2014-04-15 01:49:18 | 2014-04-15 01:49:29 | View |
| Test camera | woestman | 2014-04-21 20:26:44 | 2014-04-21 20:27:31 | View |
| Test camera | woestman | 2014-04-21 20:27:42 | 2014-04-21 20:27:54 | View |
| Test camera | woestman | 2014-04-21 20:28:18 | 2014-04-21 20:28:31 | View |
| Test camera | woestman | 2014-04-21 20:29:02 | 2014-04-21 20:29:15 | View |
| Test camera | woestman | 2014-04-21 20:29:24 | 2014-04-21 20:29:48 | View |
| Test camera | woestman | 2014-04-21 20:31:00 | 2014-04-21 20:31:30 | View |

***Figure 5: Conceptual Image of Archived Data***

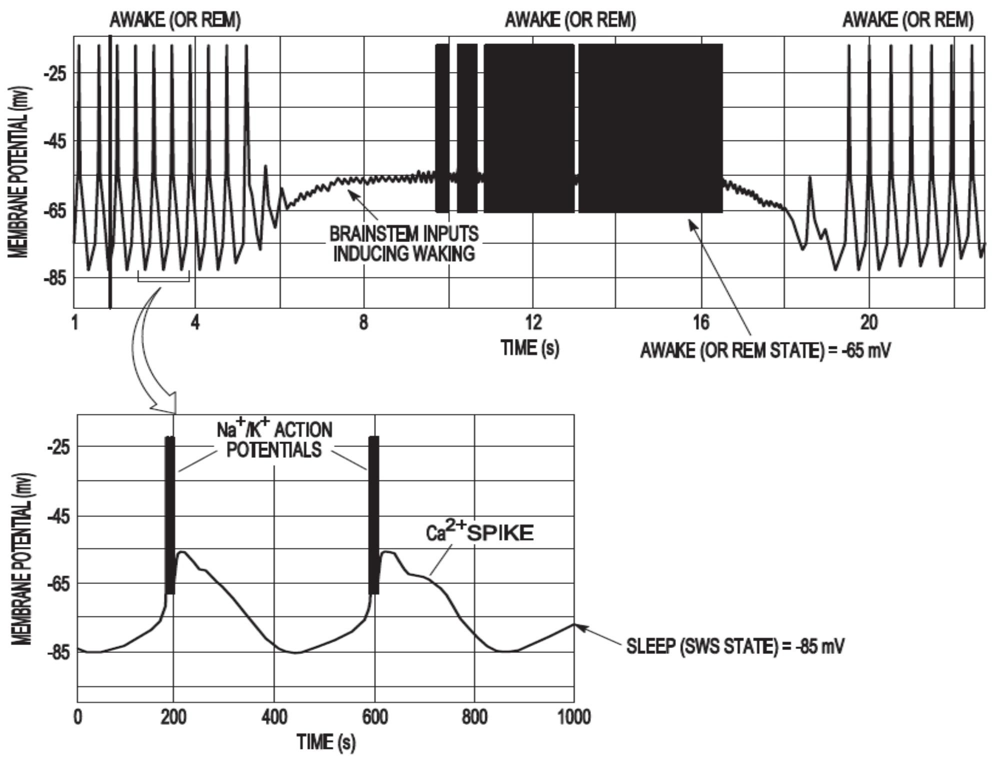


***Figure 6: Thalamocortical Neurons - During SWS and REM Sleep States***

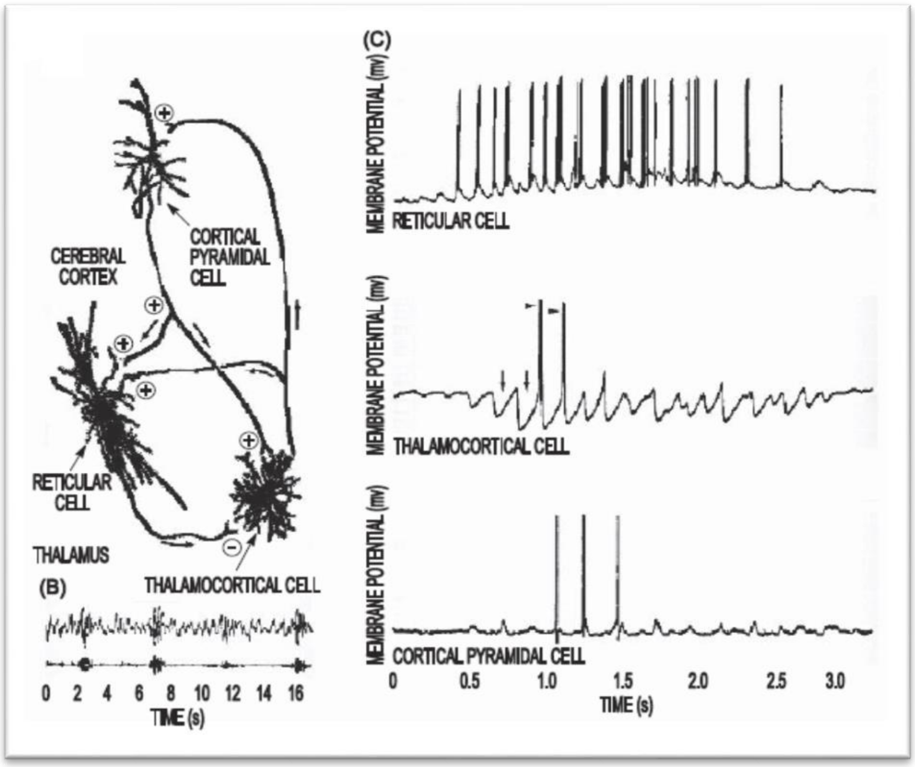


***Figure 7: Thalamocortical Neurons - Oscillatory Mode in a Sleep State and Tonically Active Mode***

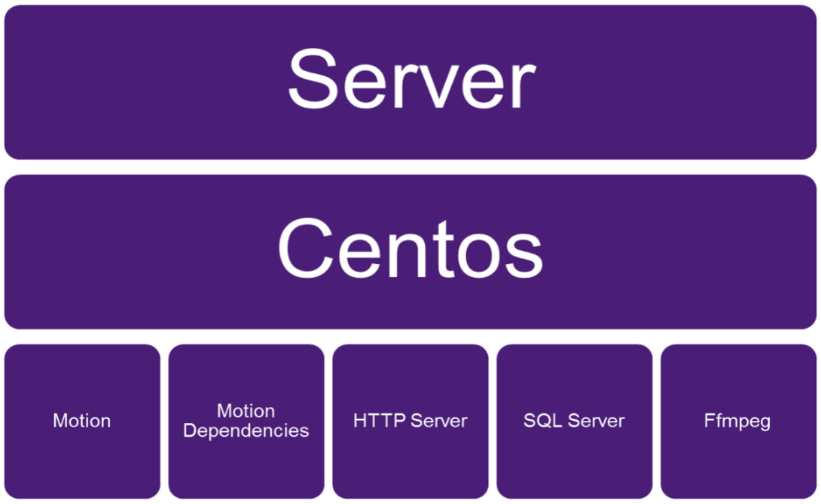


***Figure 8: Architectural Diagram***

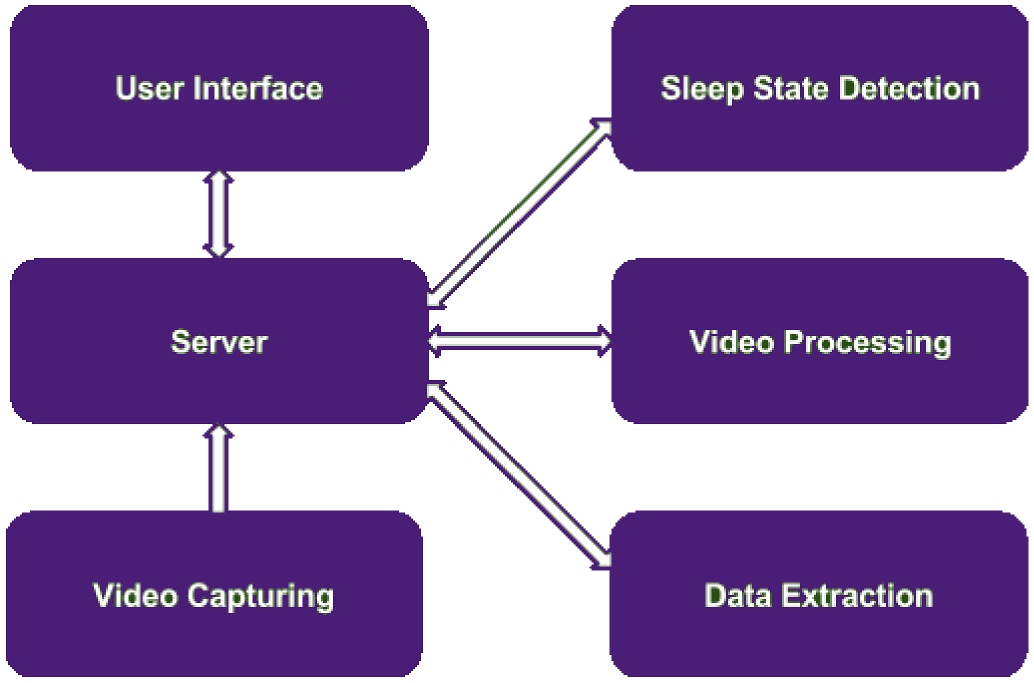


***Figure 9: Module Diagram***

# Appendix C: Informal Notes for Future Senior Design Groups

locker combo: [redacted]

FTP Host    : sftp://sidsknowmore.net
FTP username: [redacted]
FTP password: [redacted]

**Gmail:**
SIDSNightEyeGuardian@gmail.com
[redacted]

**Skype:**
SIDSNightEyeGuardian
[redacted]

may1429-cam-01-wl.ece.iastate.edu
00:0d:c5:d6:1a:ab
129.186.55.85

may1429-cam-01.ece.iastate.edu
48:02:2a:41:40:a3
129.186.55.84

may1429-cam-02-wl.ece.iastate.edu
00:0d:c5:d4:df:b5
129.186.55.89

may1429-cam-02.ece.iastate.edu
00:12:7b:5b:da:ca
129.186.55.88

Gateway IPv4: 129.186.55.254
Netmask: 255.255.255.0
DNS:
129.186.1.200
129.186.140.200
129.186.78.200

**UserPie (login auth) MySQL:**
User: [redacted]
Password: [redacted]

**UserPie (login auth) now uses the same MySQL database/user as ZoneMinder:**
User: [redacted]
Password: [redacted]

**Database for archives (MYSQL):**
User: [redacted]
Password: [redacted]

If you need to access PhpMyAdmin off campus you will need to add your IP address to the Allow list. Edit /etc/httpd/conf.d/phpmyadmin.conf on the server and add your IP address under the last "Allow from *" line. Then execute "service httpd restart" on the server so that the changes will take effect.

**Motion info:**
http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome

Motion works by running threads for each camera. The way it has been set up so far is a user (linux user so far but could be edited for a sql user) has an individual folder with a Config file and a folder to store the video.

**Config file specifics:**
netcam_url http://may1429-cam-02-wl.ece.iastate.edu/videostream.cgi?user=visitor&amp;pwd=visitor
-- the url to the webcam. this overwrites the other settings for video input.
-- Using the videostream.cgi provided better resolution than the snapshot (8 frames rather than2)
    -- might be able to enhance this some more?

output_normal off
-- if this is on it produces a bunch of image files that might not be needed. we really only want the avi file

locate on
-- places a box around the areas of motion

target_dir /home/dubansky/files
-- folder where to store the video/image files from the server root.

movie_filename %v-%Y%m%d%H%M%S
-- I didnt change this yet, but we could edit this to make it easier to find video files if need be.

webcam_port 12345
--the port to which we output the video stream to, need to have a web page pointing to this not a browser
<applet code=com.charliemouse.cambozola.Viewer
  archive=cambolo3.jar width="320" height="240" style="border-width:1; border-color:gray; border-style:solid;"> <param name=url value="http://www.sidsknowmore.net:12345"> </applet>

-- cambolo3.jar is the most recent cambozola version I had laying around, it needs to be in the same directory as the webpage containing this code, or point to another directory.
-- if we want to have multiple cameras we might need to set different ports!

control_port 8889
-- used for controlling Motion from a web browser, allows for restart/shutdown features. just point the browser there.

**Database Options for MySQL:**
# Mysql database to log to (default: not defined)
mysql_db archives

# The host on which the database is located (default: localhost)
mysql_host localhost

```
# User account name for MySQL database (default: not defined)
mysql_user [redacted]

# User password for MySQL database (default: not defined)
mysql_password [redacted]!
```

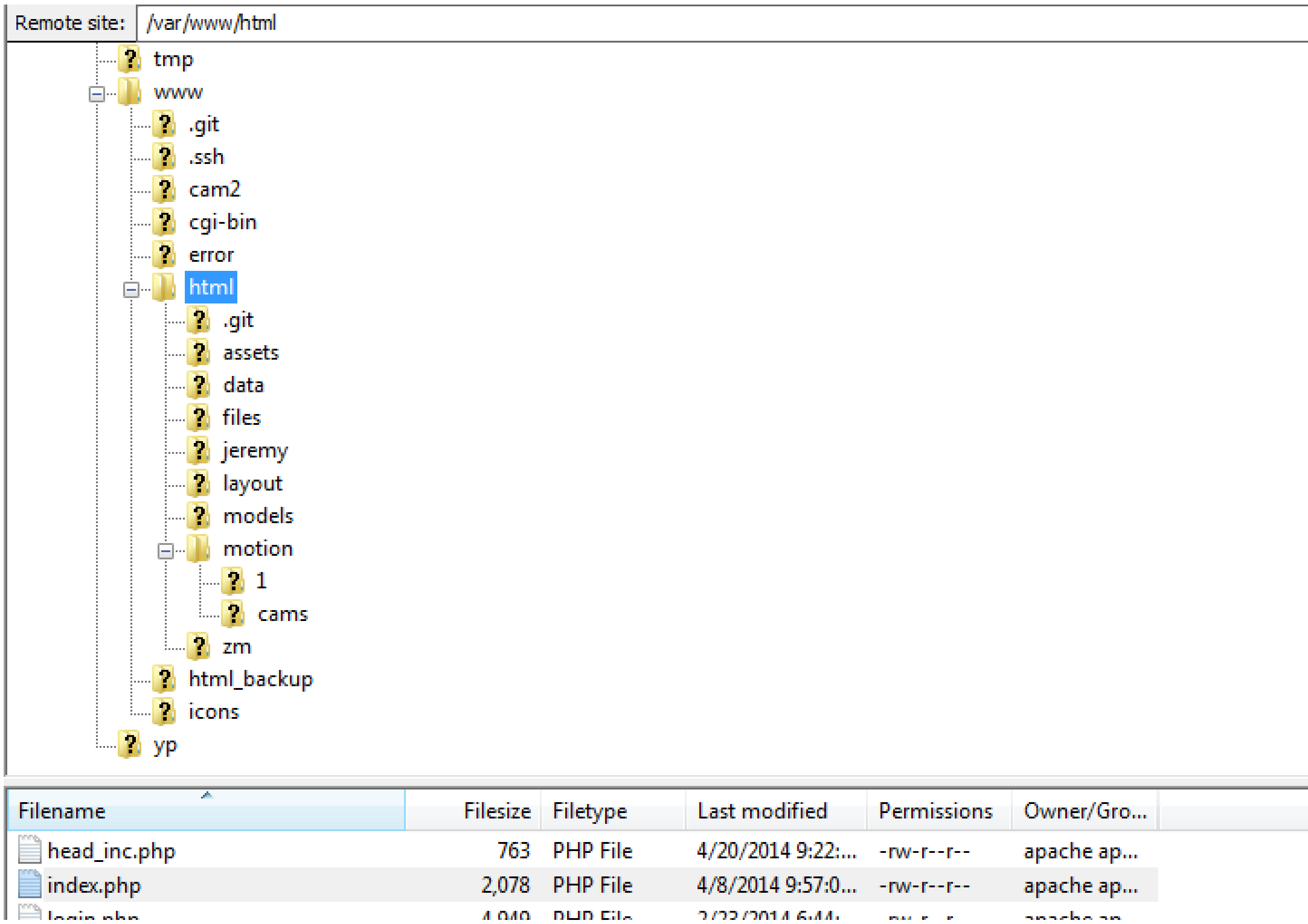
**Camboloza Issues**
there is a security risk using this and on some computers wont run, to fix this:
click Windows icon -> search  configure java -> run.
GoTo the security tab and add the sidsknowmore.net domain, and/or drop security level to medium.

**What to do from here:**
Setup a folder containing a Config file and Video folder for each new user,
Edit the main motion.config to add the new user config as a thread, (possibly restart?)
Track the new port number and set a webpage to stream video from that port,
Graph the data from the SQL DB, -- DONE

CAMBOZOLA ISSUE - sign it so it can access socket?
http://stackoverflow.com/questions/4169717/why-does-my-applet-get-a-java-security-accesscontrolexception-access-denied-ja

It is important to keep the video files in the www/html directory because they if they are outside, then the webserver has issues trying to locate them. So in order to keep everything organized we kept everything, even motion configuration files in the html folder. For each new user, when they add a camera it creates a directory inside motion/cams for that individual's new camera. When a new camera is created, the port for video streaming is also changed to add the streaming option.

Once motion is up and running it will push updates to our SQL server, we can view what camera recorded motion and at what times. When we are ready to display the data, it is simple request to a PHP page that grabs the motion data given a camera and time stamp.



Remote site: /var/www/html

```
tmp
www
  .git
  .ssh
  cam2
  cgi-bin
  error
  html
    .git
    assets
    data
    files
    jeremy
    layout
    models
    motion
      1
      cams
    zm
  html_backup
  icons
yp
```

| Filename | Filesize | Filetype | Last modified | Permissions | Owner/Gro... |
|---|---|---|---|---|---|
| head_inc.php | 763 | PHP File | 4/20/2014 9:22:... | -rw-r--r-- | apache ap... |
| index.php | 2,078 | PHP File | 4/8/2014 9:57:0... | -rw-r--r-- | apache ap... |
| login.php | 4,949 | PHP File | 2/23/2014 6:44:... | -rw-r--r-- | apache ap... |

# Testing Open Source Software: Motion

**Nicole Bruck**
Software Engineering
123 Sheldon Ave #27
Ames, Iowa 50014
(563) 599-9556
bruckna@iastate.edu

**Daisy Isibor**
Computer Engineering
233 Sheldon Ave B1
Ames, Iowa 50014
(515) 708-7197
isibord@iastate.edu

## 1. ABSTRACT

This paper describes the process and results from testing the open source software, Motion[1]. It will describe the tools used, Valgrind and Python Injection, and the steps followed. The test suite was created to be fully automated with a single command.

The infrastructure of the test suite is mainly composed of commands to the tool Valgrind to generate and run tests to detect many memory management and threading bugs, and profile the software in detail. The other major component is using commands containing python code to inject malicious input to the program.

To correctly run the software, first install and verify Motion by following the instructions that can be found on their homepage guide[1]. Then install the software Valgrind, which can be found on their homepage with instructions[2].

## 2. CATEGORIES AND DESCRIPTORS

D.1 [**SIDS**]: Sudden Infant Death Syndrome[3] – also known as cot death or crib death is the sudden death of an infant that is not predicted by medical history and remains unexplained after a thorough forensic autopsy and detailed death scene investigation.

D.2 [**ffmpeg**]: FastForwardMPEG[4] – a free software project that produces libraries and programs for handling multimedia data. ffmpeg includes libavcodec, an audio/video codec library used by several other projects, libavformat, an audio/video container mux and demux library, and the ffmpeg command line program for transcoding multimedia files.

## 3. GENERAL TERMS

Measurement

## 4. KEYWORDS

Testing, Motion

## 5. INTRODUCTION

An Iowa State Electrical and Computer Engineering Department Senior Design Project, SIDS Night Eye Guardian[5], utilizes the open source software, Motion. The two authors of this document on currently on the Senior Design team. Testing is a required portion of their project and it serve as a good option to test one of their primary softwares.

Motion, a software motion detector, is a free, open source software application developed for Linux. It can monitor video signal from one or more cameras and is able to detect if a significant part of the picture has changed saving away video when it detects that motion is occurring (it can also do time lapse videos, et al.).

The program is written in C and is made for Linux (exploiting video4linux interface[6]). Motion is a command line based tool whose output can be either jpeg, netpbm files or mpeg video sequences. It is strictly command line driven and can run as a daemon with a rather small footprint and low CPU usage.
Some of motion's features include:

- Taking snapshots of movement
- Watch multiple video devices at the same time
- Watch multiple inputs on one capture card at the same time
- Live streaming webcam
- Real time creation of mpeg movies using libraries from ffmpeg
- Take automated snapshots on regular intervals
- Take automated snapshots at irregular intervals using cron
- Execute external commands when detecting movement (and e.g. send SMS or email)

[1]http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome

[2] http://valgrind.org/docs/manual/quick-start.html

[3] http://www.sids.org/ndefinition.htm

[4] http://www.ffmpeg.org/documentation.html

[5] http://seniord.ece.iastate.edu/may1429/

[6] http://www.exploits.org/v4l/

- Motion tracking (camera follow motion - special hardware required)
- Feed events to a MySQL or PostgreSQL database.
- Feed video back to a video4linux loopback for real time viewing
- Lots of user contributed related projects with web interfaces etc.
- User configurable and user defined on screen display.
- Control via browser (older versions used xml-rpc)
- Automatic noise and threshold control

The primary use of the motion with respect to the Senior Design project is the video streaming and motion tracking. Therefore, these areas will be the main focus of testing.

## 6. TESTING

### 6.1 Bug Database

Motion actually has a very well-kept and active bug database that is available to all users (open source) to report t[7]. As of April 24th, the Current Release 3.2.12 contains 44 new, 4 assigned and 42 resolved bug reports.

The process of reporting the bug is to provide a title, description any available messages or output, environment information and Motion version number.

### 6.2 cppcheck[8]

The first bug tested for was reported June 1, 2012 by the user FolkertvanHeusden. It was described as below.

"*Ran motion 3.1.2 through cppcheck. Motion has 2 bugs:*

*[video.c:457]: (error) Resource leak: vloopbacks*

*[webhttpd.c:736]: (error) Memory leak: text_help*"

A design choice was made and through a recommendation by the professor of this course, Valgrind was used to detect and replicate this bug.

### 6.3 Multiple Vulnerabilities[9]

The first bug tested for was reported March 7, 2013 by the user RvH. It was described as below.

"*Motion 3.2.12 is prone to multiple vulnerabilities. These vulnerabilities are Buffer Overflows, Cross Site Scripting and Cross Site Request Forgery*"

Attached to the bug report is multiple screenshots of python injection attacks into the command arguments when running Motion. A design choice was made to do a simple example of a Python attack by printing to the console. This will be covered in Section 9.2 under Python Injection.

### 6.4 Segmentation Fault[10]

The first bug tested for was reported November 21, 2012 by the user RzTen1. It was described as below.

"*I'm getting a segmentation fault on startup when trying to grab frames from an iViewHD SD camera.*"

To replicate the bug, testing utilized the cameras made available by the Senior Design team, FOSCAM IR cameras. On launching Motion during the course of testing, we encountered a Segmentation Fault on startup, as in the case of this bug report. Therefore, there is no section devoted to describing this portion of the test suite but results are included at the end.

## 7. VALGRIND[2]

To actually utilize Valgrind to test Motion, it is command-line driven and an example is below. There is a snapshot of the command being run from the terminal in Figure 1.

```
% valgrind --tool=memcheck motion
--leak-check=yes --show-reachable=yes
--log-file=/home/logfile ./motion –c
/home/user/motion.conf
```

[**--tool=memcheck**] specifies which tool within Valgrind to use.

[**--leak-check=yes**] flags that the tests should be aware of memory leaks while the software is running.

[**--show-reachable=yes**] flags that when a set of memory has been appointed in the software, if it is still reachable after that function returns.

[**--log-file=/home/logfile**] specifies where the generated log file should be saved with the test results.

[**./motion –c /home/user/motion.conf**] is the command arguments for the program to run.

## 8. PYTHON INJECTION[1]

To actually utilize Valgrind to test Motion, it is command-line driven and an example is below.

```
% motion –c 'python –c 'print
"\x41"*1000'
```

[7]http://www.lavrsen.dk/foswiki/bin/view/Motion/BugReports

[8]http://www.lavrsen.dk/foswiki/bin/view/Motion/BugReport2012x06x01x093100

[9]http://www.lavrsen.dk/foswiki/bin/view/Motion/BugReport2013x03x07x071831

[10]http://www.lavrsen.dk/foswiki/bin/view/Motion/BugReport2012x11x21x225754

**[python –c 'print "\x41"*1000]** is a basic python command to print 1000 times the character 'A'

## 9. RUNNING THE TEST SUITE

To execute the entire test suite, simply copy and paste the code from the test in Figure 2 into a file located in the same directory as Motion's installation and save it as MotionTest.bat then simply call the file as below.

```
% ./MotionTest.bat
```

## 10. RESULTS

### 10.1 Valgrind

After running the command from section 3 above, Valgrind provides a log file with results based on the parameter values that were set. The results mainly consists of two parts. The small snippet of the results are shown in Figure 3.

The first one is "Conditional jump or move depends on uninitialized value(s)". This means that there is a path in the code that starts out with an un-initialized variable which is later used somewhere without a value assignment in the path. This is dangerous because the value of the variable could be anything, and if the variable is used in any computation, the results may be wrong.

The second part of the result looks like this "16,384 bytes in 1 blocks are definitely lost in loss record 22 of 25". This means that there is a block of code, 16,384 bytes long that leaks 22 out of 25 times that it is executed. Sometimes, this result could show that a block of code is lost as a result of another block.

One advantage to using Valgrind is that it shows the line(s) of code where the memory leaks are, as well as the memory location when it was ran. This allows a developer to have a much smaller window to discover the error within the software.

What was found from running Valgrind is the immense amount of memory leaks that exists in motion. Although some of the test results were repetitive because Valgrind would run the same code multiple times, there were over 10000000 errors found and 31 unique instances of these errors. Some cases were more critical than others, especially the case where leaks were happening in large code blocks more than 90% of the time. Those are the kind of scenarios that need to be attended to in order to avoid incorrect behavior in the program.

### 10.2 Python Injection

The python injection resulted from inserting python code as a parameter for running motion. The example used as explained in Section 4 was intended to display the character A, 1000 times. The result obtained is shown in Figure 4. Motion executed the python code by displaying the character 'A', 1000 times before it started the execution of Motion itself.

We found out from this test case that there is no sanitization of user input. In more extreme cases, it is possible for the user to include scripts with malicious code that can do some damage. It is important to maintain control of the inputs to the program and while this is not an error within the code, it is a feature that should be expected of the software.

### 10.3 Segmentation Fault

The error from the Segmentation Fault issue as explained in Section 2.4 is a scenario where a user is left with a segmentation fault on startup of Motion. In testing, a different camera than what the user reported was used but still able to reproduce the error. On an instance of running Motion while generating tests for these experiments, there was an issue of a Segmentation Fault at startup. See Figure 5. It was not the exact same command that the user reported but it correlates in the sense that it happens on startup with a specific command. Also as in the case of the reported bug, the segmentation fault is consistent with the command.

This issue is a more complex issue to directly find out what is going on because Motion does not always produce segmentation faults on startup. It will require more time and research to find out why the segmentation fault only comes up with certain commands. Looking into the core dump may produce more information on this behavior.

## 11. ACKNOWLEDGMENTS

Our thanks to our wonderful, beautiful and intelligent professor, Dr. Kathryn Stolee, for being our inspiration and drive in this endeavor.

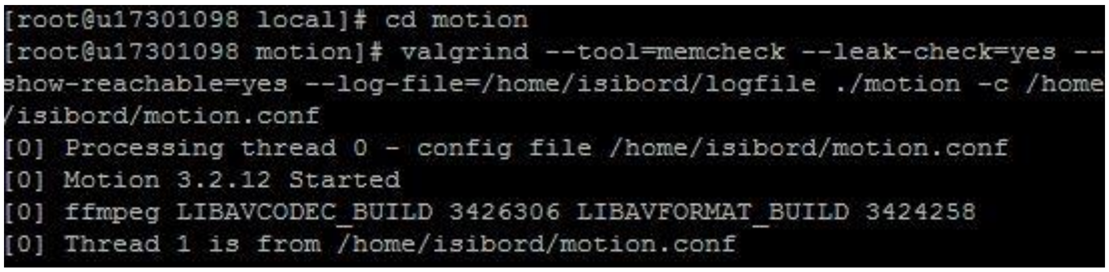
## 12. REFERENCES

[1] TosiaraT. "Motion Guide." WebHome. Motion, 15 Feb. 2014. Web. 27 Apr. 2014.

[2] Allain, Alex. "Using Valgrind to Find Memory Leaks and Invalid Memory Use."Using Valgrind to Find Memory Leaks. Cprogramming.com, n.d. Web. 24 Apr. 2014.

[3] "The Valgrind Quick Start Guide." Valgrind Developers, 2013. Web. 27 Apr. 2014.

[4] "What Is SIDS?" American Sudden Infant Death Syndrome Institute. N.p., 2009. Web. 27 Apr. 2014.



```
[root@u17301098 local]# cd motion
[root@u17301098 motion]# valgrind --tool=memcheck --leak-check=yes --
show-reachable=yes --log-file=/home/isibord/logfile ./motion -c /home
/isibord/motion.conf
[0] Processing thread 0 - config file /home/isibord/motion.conf
[0] Motion 3.2.12 Started
[0] ffmpeg LIBAVCODEC_BUILD 3426306 LIBAVFORMAT_BUILD 3424258
[0] Thread 1 is from /home/isibord/motion.conf
```

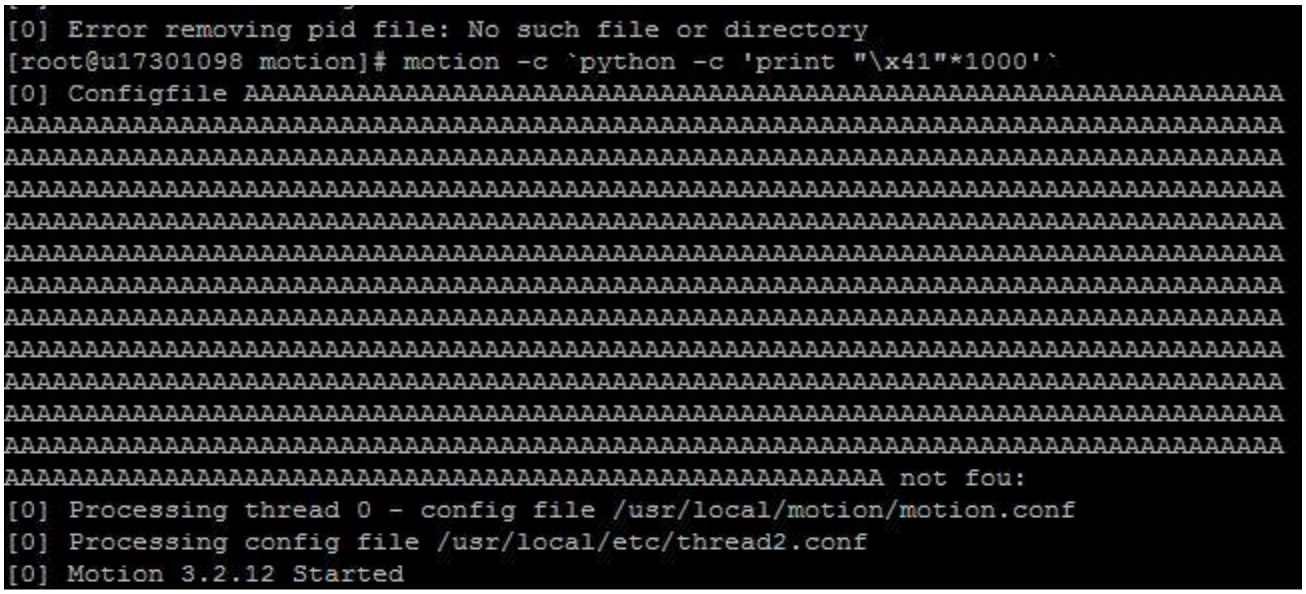
Figure 1: Valgrind Commands

```
#!/bin/bash
valgrind --tool=memcheck motion --leak-check=yes --show-reachable=yes
--log-file=/home/logfile ./motion –c /home/user/motion.conf
PID=$!
sleep 60
kill -INT $PID
motion –c ‘python –c ‘print “\x41”*1000’
PID=$!
sleep 60
kill -INT $PID
```

Figure 2: MotionTest.bat

```
...
==11887== Thread 1:
==11887== 24 bytes in 1 blocks are still reachable in loss record 1 of 16
==11887==    at 0x4A069EE: malloc (vg_replace_malloc.c:270)
==11887==    by 0x31DBE29941: my_malloc (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x31DBE2A78A: my_error_register (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x31DBE2842B: mysql_server_init (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x31DBE4FD5E: mysql_init (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x4058F6: motion_loop (motion.c:668)
==11887==    by 0x3BA40079D0: start_thread (in /lib64/libpthread-2.12.so)
==11887==    by 0x3BA3CE8B6C: clone (in /lib64/libc-2.12.so)
==11887==
==11887== 24 bytes in 6 blocks are indirectly lost in loss record 2 of 16
==11887==    at 0x4A069EE: malloc (vg_replace_malloc.c:270)
==11887==    by 0x31DBE29941: my_malloc (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x31DBE29A6A: my_strdup (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==   by 0x31DBE51B11: mysql_real_connect (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x405932: motion_loop (motion.c:670)
==11887==    by 0x3BA40079D0: start_thread (in /lib64/libpthread-2.12.so)
==11887==    by 0x3BA3CE8B6C: clone (in /lib64/libc-2.12.so)
==11887==
==11887== 54 bytes in 6 blocks are indirectly lost in loss record 4 of 16
==11887==    at 0x4A069EE: malloc (vg_replace_malloc.c:270)
==11887==    by 0x31DBE29941: my_malloc (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x31DBE29A6A: my_strdup (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==   by 0x31DBE5243B: mysql_real_connect (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x405932: motion_loop (motion.c:670)
==11887==    by 0x3BA40079D0: start_thread (in /lib64/libpthread-2.12.so)
==11887==    by 0x3BA3CE8B6C: clone (in /lib64/libc-2.12.so)
==11887==
==11887== 72 bytes in 6 blocks are indirectly lost in loss record 5 of 16
==11887==    at 0x4A069EE: malloc (vg_replace_malloc.c:270)
==11887==    by 0x31DBE29941: my_malloc (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x31DBE29A6A: my_strdup (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==   by 0x31DBE51B2D: mysql_real_connect (in /usr/lib64/mysql/libmysqlclient.so.16.0.0)
==11887==    by 0x405932: motion_loop (motion.c:670)
==11887==    by 0x3BA40079D0: start_thread (in /lib64/libpthread-2.12.so)
==11887==    by 0x3BA3CE8B6C: clone (in /lib64/libc-2.12.so)
...
```

Figure 3: Example of Valgrind logfile



```
[0] Error removing pid file: No such file or directory
[root@u17301098 motion]# motion -c `python -c 'print "\x41"*1000'`
[0] Configfile AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA not fou:
[0] Processing thread 0 - config file /usr/local/motion/motion.conf
[0] Processing config file /usr/local/etc/thread2.conf
[0] Motion 3.2.12 Started
```

Figure 4: Python Injection