University of Derby
School of Computing & Mathematics

A project completed as part of the requirements for the
BSc (Hons) in Computer Networks and Security

entitled
OctoPi: a low cost, energy efficient console server alternative.

by
James Brunt

In the Year 2014 - 2015
Contents

1. Introduction ........................................................................................................................................... 5

2. Literature Review ................................................................................................................................. 6
   2.1. Raspberry Pi as a Console Server ................................................................................................. 6
   2.2. Comparison of Perle Console Servers ......................................................................................... 7
   2.3. Comparison of Opengear 3G Console Servers ............................................................................ 8
   2.4. Comparison of Lantrinx Modular design Console Servers .......................................................... 8

3. Methodology .......................................................................................................................................... 9
   3.1. Considerations .............................................................................................................................. 10
   3.2. Software considerations .............................................................................................................. 10
      3.2.1. Ser2net ............................................................................................................................... 10
      3.2.2. Hostapd ............................................................................................................................. 11
      3.2.3. DHCP ............................................................................................................................... 11
      3.2.4. LAMP ............................................................................................................................... 11
      3.2.5. OpenVPN – OpenSSL ........................................................................................................ 12
   3.3. Hardware considerations .............................................................................................................. 12
      3.3.1. Raspberry Pi ......................................................................................................................... 12
      3.3.2. PiFace Control and Display ................................................................................................. 12
      3.3.3. AdafruitPowerBoost 500 ................................................................................................. 13
      3.3.4. PiJuice ............................................................................................................................... 14
      3.3.5. FTDI USB-COM232-PLUS-4 MOD .................................................................................. 15

4. Case Studies ......................................................................................................................................... 15
   4.1. Case study 1: University of Derby Network Department ............................................................. 15
      4.1.1. Introduction ......................................................................................................................... 15
      4.1.2. Background ........................................................................................................................ 16
      4.1.3. Physical Security .................................................................................................................. 16
      4.1.4. Dual Power System ............................................................................................................. 17
      4.1.5. Integration with LDAP ........................................................................................................ 17
      4.1.6. Centralization ....................................................................................................................... 18
      4.1.7. Wireless Interface ............................................................................................................... 19
   4.2. Case Study 2: University of Derby Students ............................................................................. 19
      4.2.1. Introduction ......................................................................................................................... 19
      4.2.2. Background ........................................................................................................................ 19
      4.2.3. Feedback ........................................................................................................................... 19
   4.3. Case Study 3: Ashfield School Networks Team ........................................................................ 20
      4.3.1. Introduction ......................................................................................................................... 20
4.3.2. Background............................................................................................................. 20
4.3.3. Feedback .............................................................................................................. 20
4.3.4. KVM support........................................................................................................ 20
4.3.5. Centralisation ...................................................................................................... 21
4.3.6. LDAP integration................................................................................................. 21
4.3.7. Out-of-band access.............................................................................................. 21

4.4. Case Study 4: Nowcomm Network Engineers ...................................................... 21
4.4.1. Introduction ......................................................................................................... 22
4.4.2. Background ........................................................................................................ 22
4.4.3. Feedback ............................................................................................................ 22
4.4.4. VPN advantages ................................................................................................. 22
4.4.5. Mobility ............................................................................................................. 22
4.4.6. Device logging ................................................................................................... 23
4.4.7. GSM/3G Module ............................................................................................... 23

4.5. Case Study 5: Courtaulds (UK) Limited Network team ......................................... 24
4.5.1. Introduction ........................................................................................................ 24
4.5.2. Feedback ............................................................................................................ 24
4.5.3. Physical power control ....................................................................................... 24
4.5.4. KVM support ..................................................................................................... 24
4.5.5. Monitoring ......................................................................................................... 25

5. Results & Analysis .................................................................................................... 25

5.1. Web Interface ......................................................................................................... 25
5.1.1. Home Page and logging in ............................................................................... 25
5.1.2. Interface Settings Page ...................................................................................... 27
5.1.3. Wireless Settings Page ....................................................................................... 27
5.1.4. Node Settings Page ............................................................................................ 28
5.1.5. Network to serial Settings Page ....................................................................... 29
5.1.6. DHCP Settings page ........................................................................................ 30
5.1.7. VPN Settings Page ............................................................................................ 31
5.1.8. Client VPN Certificate Page ............................................................................ 31

5.2. LCD Menu Display ............................................................................................... 32

5.3. Power consumption of OctoPi ............................................................................. 33
5.3.1. Test 1 basic setup .............................................................................................. 34
5.3.2. Test 2 Ethernet .................................................................................................. 34
5.3.3. Test 3 Wireless Interface .................................................................................. 34
5.3.4. Test 4 LCD display .......................................................................................... 35

James Brunt OctoPi: a low cost energy efficient console server alternative.
5.3.5. Test 5 Console module ................................................................. 35
5.3.6. Test 6 every module connected .......................................................... 35
5.3.7. Results ......................................................................................... 35
5.4. Roadmap ......................................................................................... 39
  5.4.1. KVM support .............................................................................. 39
  5.4.2. Centralisation ............................................................................ 39
  5.4.3. LDAP Integration ....................................................................... 40
  5.4.4. Out-Of-Band access ................................................................... 40
  5.4.5. Monitoring .................................................................................. 40
  5.4.6. Physical Security ......................................................................... 40
6. Conclusion ......................................................................................... 41
  6.1. Further Development ..................................................................... 42
  6.2. Limitations .................................................................................... 43
7. References ......................................................................................... 44
Abstract This paper puts forward an alternative to console/terminal servers, the OctoPi. The OctoPi uses a Raspberry Pi and the Raspbian operating system as the foundation hardware and software. The Raspberry Pi has a number of advantages over conventional terminal servers including its low total power consumption, it is easily portable due to the low profile and weight and its affordability. This paper proposes that these attributes will allow the Raspberry Pi to be an ideal alternative to current console/terminal servers. It will create a low-cost starting point to allow network engineers to support network infrastructure in a flexible and affordable way. This paper presents the results of the energy efficiency of the OctoPi compared to current implementations as well the features that facilitates in making a Raspberry Pi become a console/terminal server. A prototype is used with case studies to gauge the relevance and potential of the project.

1. Introduction

In the paper the OctoPi is introduced as a low cost, energy efficient alternative to console/terminal servers. It is assembled using a Raspberry Pi for the base hardware and runs Raspbian as the Operating system. This creates a powerful platform to create a low powered console/terminal server that costs very little.

In this paper several Console/Terminal server vendors are reviewed taking into consideration the available models and features supported. The paper will analyse useful features that are desirable for a console/terminal server. It will then implement these features into the OctoPi prototype and describe how each feature was achieved. Features such as multiple console connection support, management interface, wireless interface support and VPN for secure access.

The paper will also describe what hardware and software considerations were made in implementing the design and explain the purpose behind each one. Detailed case studies were conducted using the prototype and the feedback from participants was recorded. It will discuss the criticisms and recommendations raised in each case study and propose possible solutions for each.

Quantitative evaluation of the energy consumption of the prototype will be gathered and this will be compared to the energy consumption of console/terminal servers surveyed within the literature review section. The tests will determine the Watts used by the prototype and show the difference against comparable models. The results will be used to evaluate the energy efficiency of the prototype as a more sustainable and affordable alternative.
The paper will then conclude with recommendations for future development of the prototype. Implementing these recommendations should increase the functionality of the prototype and allow the prototype to support a wider range of devices and be deployed in larger environments.

2. Literature Review

Consoles servers are a great tool for network engineers allowing access to the management port of routers, switches and servers from across the network. A Raspberry Pi can provide the same services as a Console Server. This will not only reduce the overall cost of deployment but will provide greater flexibility and will offer a more modular design. The Raspberry Pi does not require a vast amount of energy to power it which makes it an ideal candidate for an always on console server. This can potentially reduce the cost of powering the device compared to console servers offered today.

This section provides a review of features that are currently available for console servers. The review will assess different vendors and the models available and compare the functionality of them to a Raspberry Pi implementation. It will do this by comparing the features and functionality that are advertised on a vendors’ website to that which are available to a Raspberry Pi. This will demonstrate that using a Raspberry Pi can provide the same functionality while maintaining a lower overall deployment cost.

2.1. Raspberry Pi as a Console Server

This section discusses the features that are available to the Raspberry Pi enabling it to support the features of a Console Server.

A big weakness of console servers is the size. Console Servers supporting more than eight ports are rack mounted. This presents a problem as rack space is not always available. A Raspberry Pi can be implemented and not need to take up any rack space. Having 48 devices that need console access can be serviced by one rack mounted 48 port console server. However it is unlikely for those devices to be located in the same rack. Depending on the distance between devices, it can be impractical to cable all devices to one console server.
A Raspberry Pi can also support many console connections through the use of USB hubs. This allows for the Raspberry Pi to have a flexible and modular design. Being able to deploy a Raspberry Pi which supports only the required number of console ports required per server rack.

Another weakness of console servers is mobility. Network engineers that need to access a console port of a switch or router will typically use a USB to RS232 adapter to connect laptops to devices. If access to multiple devices is required then the connection is moved to the required device. This can cause a lot of time swapping cables about between devices. A Raspberry Pi can be easily taken to site to support access to console ports due to the small size. The Raspberry Pi can also be powered from a battery source instead of mains power. This will enable the Raspberry Pi to be deployed in areas that do not have any spare power outlets.

A wireless adapter can be added to the Raspberry Pi to enable an engineer to access the console port without having to physically wire the laptop to the Raspberry Pi. This will allow the engineer to efficiently access console ports without being in the cold aisle of a server room. The Raspberry Pi can also act as an access point. Another Raspberry Pi can then connect to this access point providing a mini network of Raspberry Pi Console servers. The network engineer could then access any number of console servers throughout a server room without having to physically be in any of the aisles.

Communication can be secured by using OpenVPN which can be used to create a VPN between a client and the Raspberry Pi. This will encrypt the communication between the devices.

2.2. Comparison of Perle Console Servers

Perle are a company that provide terminal and console servers. The Perle IOLAN DS1 and TS2 are models that provide one or two serial connections respectively, they provide basic features such as tunnelling raw serial data across Ethernet and are no larger than a Raspberry Pi. These devices retail at £150. (Perle 2015) The IOLAN STS4 D model supports either four or eight serial ports. This model comes with security features that will encrypt communication to the device. These models start at £325. The larger models support more ports and are rack mounted. These models start at £659 and rise above £1000.
The Raspberry Pi can provide the same functionality as these models but also provide a larger amount of console connections by utilizing a USB hub adapter. This can allow a Raspberry Pi to be easily customisable and remain a cost effective solution.

A Raspberry Pi B+ can be purchased for £19.09 excluding VAT.(Farnell 2015) This will support four native USB ports which a USB to serial adapter can be plugged into. Adding a USB hub to the Raspberry Pi can provide more serial connections.

By using the Raspberry Pi this way a network administrator can deploy multiple Raspberry Pis each supporting as many console devices as required. This can then easily be increased on each Raspberry Pi by adding a USB hub to support more devices if added later. This will give the Raspberry Pi the features and port availability of a high end console server, but a network administrator will have the flexibility of deploying the serial interfaces in racks where needed.

2.3. Comparison of Opengear 3G Console Servers

Opengear is another company that supply terminal and console servers. This company offers a similar range of products compared to Perle but also offer Cellular out-of-band access. This allows access to a console server by utilizing a 3G network.

The Raspberry Pi can have a 3G module installed.(Adafruit 2015) This Module can then provide a network engineer out-of-band access to the Raspberry Pi the same way as the Opengear 3G cellular model. The advantage of the Raspberry Pi supporting this is the 3G module can be easily added after deployment. If the need to access the console ports out-of-band over a 3G network is required then this can be provided as an add-on module rather than a network engineer having to purchase a Console server that only supports limited console interfaces.

2.4. Comparison of Lantrinx Modular design Console Servers

A company named Lantrinx offer Console servers that have a modular design. The Lantronix SLC 8000 can support up to forty eight ports.(Lantrinx 2015) Each expansion module comes with sixteen ports. This allows the user to have buy sixteen ports to start with and buy an expansion module later to add more ports.
While this is a great way to save money by adding ports when required. It does not offer as much flexibility as with a Raspberry Pi. Firstly the ports need to be purchased in multiples of sixteen. This means ports can be purchased and go unutilized. Also if the company stops supporting that particular model or goes out of business, then buying the required ports can be difficult as only proprietary ports will work with that model.

The Raspberry Pi can support greater flexibility as the amount of ports installed is not restricted. If a network administrator requires more ports then there are a number of options available. The network administrator can purchase any kind of USB hub and combine this with the amount of USB to Serial adapters required. As these are freely available from any vendor the network administrator is not restricted to proprietary interface modules that may become unavailable.

3. Methodology

Designing and building a prototype will allow for real world testing and evaluation of the potential reduction in cost and energy efficiency compared to serial servers currently on the market. Comparing the difference in energy consumption between the prototype and current console servers will allow for an analysis of the potential energy saving. From this cost savings can be calculated not just from the initial hardware purchase but also from the savings to power these types of devices.

For the prototype to be a variable alternative, it needs to provide the same services which can be already found in console servers. This will require testing of the prototype to determine if it can be used in real life situations and fulfil the requirements of a console server. Case studies will be used to evaluate whether the prototype will be useful in real world situations and will highlight areas in which improvements or modifications are required.

This chapter will describe the development of a Raspberry Pi console server. Detailing which services and features were considered including reasons why the services were chosen. It will provide methods for the design process and use case studies to evaluate the prototype as a valid alternative. Using Case studies will be conducted as a means of testing the effectiveness of the prototype. The use of case studies to continually evaluate the prototype will highlight areas where it can be improved and features that can be made available.
3.1. Considerations

There were a lot of different factors to consider when planning the development of the Raspberry Pi console server. Consideration over the services available in the Linux operating system which will provide and support the functionality of a console server. This section will investigate the available services in this area.

3.2. Software considerations

3.2.1. Ser2net

Ser2net is a service available in Linux that provides a proxy which allows Telnet/TCP connections to be made to serial ports on a machine. (ser2net 2015) This will allow the Raspberry Pi to divert incoming Telnet connections to a serial connection. Using a configuration file, Ser2net can configure different incoming port numbers to corresponding serial communication ports.

Once the service has been installed on the Raspberry Pi, Incoming Telnet sessions will be forwarded out of an attached serial port. A USB to console adapter connected to the Raspberry Pi will facilitate these connections. It is possible to assign different port numbers to different USB to serial connections by altering the Ser2net configuration file. This will allow for multiple Telnet-to-console connections.

The ser2net.conf file follows the following format:
<TCP port>:<state>:<timeout>:<device>:<options>

Below is an example of a Ser2net configuration file. It shows four configured console connections from port 5000 to 5003 that will forward incoming Telnet connections to ports USB0 to USB3 respectively:

5000:telnet:600:/dev/ttyUSB0:9600 8DATABITS NONE 1STOPBIT banner
5001:telnet:600:/dev/ttyUSB1:9600 8DATABITS NONE 1STOPBIT banner
5002:telnet:600:/dev/ttyUSB2:9600 8DATABITS NONE 1STOPBIT banner
5003:telnet:600:/dev/ttyUSB3:9600 8DATABITS NONE 1STOPBIT banner
3.2.2. Hostapd
The Hostapd service allows the Raspberry Pi to be configured into a wireless access point. This service will allow engineers to connect to the Raspberry Pi over a wireless network. This will allow network engineers to gain console access to devices without the need to be wired to a console server. Once connected to the Raspberry Pi wirelessly an engineer can Telnet to the Raspberry Pi which will then forward Telnet connection to a console connection.

This will give engineers the ability to access several different devices connected to the Raspberry Pi without having to physically be at each device. Engineers can configure devices without having to be located within cold server isles and will not have to keep unplugging console connections. This will be beneficial in situations where connecting to devices over a network is not possible such as new installations or connectivity to a device has been lost.

3.2.3. DHCP
Dynamic Host Configuration Protocol (DHCP) is a standardized network protocol which provides dynamic distribution of configuration parameters such as IP address, subnet mask, and default-gateway to devices on a network. This protocol helps network administrators to assign network address information to devices from a centralized server. When using the Raspberry Pi as an access point it will be advantageous for devices which connect to gain an IP address within the same range as the wireless interface of the Raspberry Pi.

3.2.4. LAMP
Linux Apache MySQL PHP (LAMP) are services that allow the Raspberry Pi to be turned into a web server. A LAMP server can serve as a GUI interface for users to configure options. Using LAMP, a web page can be displayed when connecting to the Raspberry Pi displaying configuration options.

Apache is a web server application which processes HTTP requests. This will allow the Raspberry Pi to host a webpage which present options for configuring the Raspberry Pi such as interface addresses, port configurations and wireless network settings.

PHP is a server-side scripting language. It can be implemented within HTML code to facilitate interactions between user input into a web browser and the Raspberry Pi. This can allow for the manipulation of configuration files such as changing the Ethernet IP address of the Raspberry Pi.
MySQL is a relational database management system. This can be used to store login credentials to restrict access the Raspberry Pi. PHP can then interact with the database to validate usernames and passwords submitted by users. Restricting access is important as the Raspberry Pi will potentially be connected to the console ports of Routers, switches, servers and other essential IT infrastructure.

### 3.2.5. OpenVPN – OpenSSL

If the network is properly configured the Raspberry Pi can be accessed from a remote location through the internet. This presents a security issue as the internet is not a secure medium and eavesdroppers can potentially intercept data being transmitted between an engineer and the Raspberry Pi such as server credentials. To reduce the chances of nefarious eavesdroppers, the Raspberry Pi can be configured as a VPN server allowing engineers to establish a VPN connection to the Raspberry Pi which will encrypt the traffic between the two end devices.

This service may not be required if the Raspberry Pi is deployed in a network which an engineer can gain a VPN connection such as a site to site VPN or remote access VPN. However it may be advantageous to provide VPN access directly to the Raspberry Pi.

### 3.3. Hardware considerations

#### 3.3.1. Raspberry Pi

The raspberry Pi comes in a few different models. However the Raspberry Pi model B+ is a better choice for this project. This model has four USB ports which will provide for more console connections compared to the model A which only has two. The Raspberry Pi model B+ also has 512MB of RAM, twice as much as the model A. While this model seems to be initially the better choice, tests can be conducted using both models to evaluate which will offer the most benefit.

#### 3.3.2. PiFace Control and Display

It will be important to display configuration as status messages to a user without the user having to open a web browser. Connecting a HDMI monitor, keyboard and mouse to the Raspberry Pi will not be an efficient way to display configuration and status messages. This will limit the mobility of the Raspberry Pi and use up extra USB ports. The PiFace control and display board allows users to interact with the Raspberry Pi through the use of programmable buttons.
2015) It also has a 16x2 LED display which can be programmed to display text. This will enable the Raspberry Pi to display information such as interface IP addresses and wireless configuration settings without the need for a monitor, keyboard and mouse. This will not only ensure the Raspberry Pi remains a mobile device but reduces the energy required to power a HDMI display.

![Figure 1: PiFace control and Display](image)

### 3.3.3. AdafruitPowerBoost 500
This device allows the Raspberry Pi to be powered using a battery source. This will allow the Raspberry Pi to be deployed where powering the device from a wall socket is not possible. This will allow engineers to temporarily deploy the Raspberry Pi. It can also be charged using the same power adaptor used to power a Raspberry Pi.
3.3.4. PiJuice

PiJuice is a battery power pack designed with the Raspberry Pi in mind. It is a project that is currently on Kickstarter (Kickstarter 2015) and is potentially a better choice of supplying power due to being designed for the Pi.

Features

- On-board 1400 mAhLipo / LiIon
- Full UPS (Uninterrupted Power Supply)
- On/off switch
- API for power management with auto shutdown capability when running low on power
- Designed, to fit existing cases for the Raspberry Pi
3.3.5. FTDI USB-COM232-PLUS-4 MOD

![Image of FTDI USB-COM232-PLUS-4 MOD]

*Figure 4: USB to Serial module.*

The USB-COM232-Plus-4 is a USB to four console port module. It can be connected to a USB port on the Raspberry Pi to allow console connections through the Raspberry Pi. Multiple modules can then be used to expand the console ports available. Connecting a USB hub to the Raspberry Pi will expand the amount of modules that can be added to the four USB ports native to the Raspberry Pi.

4. Case Studies

To evaluate the potential of the prototype to become integrated equipment that will support multiple network scenarios. Case studies have been conducted to gain insight into current implementations in industry as well as opinions of individuals working in networking. Interviews have taken place at the University of Derby Network team and students, Ashfield School’s Network Team, Courtaulds (UK) Limiteds network team and network engineers working at Nowcomm. These case studies have gained feedback from over 30 participants.

4.1. Case study 1: University of Derby Network Department

4.1.1. Introduction

This case study interviewed four members of staff that work at the university about the deployment on console/terminal servers at the University of Derby. They work as a part of the Server and Storage team and the datacentre team. They work with servers on a daily basis and use a keyboard, video and mouse (KVM) switches to support and manage the University’s server and network environment.
4.1.2. Background
The current installation has several Raritan Domian KXII KVMs. These KVMs support between 30 to 40 servers and other devices. They estimated cost of deployment is between £15000 and £20000. This does not take into consideration the cost of support from Raritan as none has been purchased. They are currently looking to replace the KVMs as they are getting old and outdated. They have also come across problems in the system due to Java conflicts. A recent Java update has rendered the KVMs unusable and they have turned many of the KVMs off due to these issues. They are looking into alternative options to the current KVM system.

The current KVMs consume between 16-17 Watts of power each at rest. This will be more if they were in use but due to Java issues a demonstration of how they worked was not possible. The space the KVMs take up in the datacentre is about 8 rack unit spaces.

The prototype was demonstrated in one of the meeting rooms located at the University of Derby. The Demonstration showed how the prototype would connect to a network to provide console connections of TCP. The participants could connect wirelessly to view the management interface of the prototype. From here they could also use a Telnet session to connect to a serial connection. The serial connections were not connected to any devices due to security concerns.

4.1.3. Physical Security
There were many comments after demonstrating the prototype. An issue about securing the device in the rack space was brought up. The current KVMs are rack mounted and are screwed into position. The racks have doors that can be locked. The prototype will have to be secured in some way without anybody having physical access to it without first opening the rack door, this is for security purposes. Due to the size of the prototype, it can fit in a smaller space than traditional servers. This allows the prototype to utilise space which would otherwise be left empty.

In racks, servers and networking equipment are mounted centrally with space either side of the rack. This generally is only used for cabling as the space either side is limited and not designed to store equipment. Due to the size of the prototype this space can be utilised instead of the prototype taking up a rack space. It can be secured to the side of the rack using an adhesive or for a more temporary deployment Velcro can be used.
There is also a potential for the prototype to be installed on top of the racks themselves either inside or outside. There are holes in the top of racks for cabling to exit to traverse to other racks. The prototype can be located near the entrance of one of these holes to allow the prototype to access the rack and utilise the unused space above the racks.

Sometimes server rooms are installed on top of a false floor. This allows cabling to run underneath the racks below the floor. For cabling to enter a rack it would come up through a trapdoor in the floor directly below a rack. The prototype can be installed below this trapdoor.

Another location for installation consideration would be behind cable management systems. Administrators install cable management systems to easily track which devices cables are connected. These cable management systems take up one or more rack units, even though the system only uses the space at the front of the rack. The space behind the cable management system can then be used to house the prototype, utilising space which would otherwise remain empty.

### 4.1.4. Dual Power System
The point was made that the prototype only has a single power point, this means the device has a single point of failure. The device can be powered from a battery but this only powers the device for a short period of time acting as a UPS.

Usually servers and networking equipment have two or more sources or power. The devices can then be plugged into two different sources of power providing redundant power in the case one should fail.

The Raspberry Pi only has one power point natively. For this feature to be added a means of supplying power from two different sources is required. One way this can be done is having it attached to two power adapters but only one is used. If one is to fail a mechanism is triggered to draw power from the secondary. The battery power can be used to support the transition between the two.

### 4.1.5. Integration with LDAP
The interviewees suggested that they would like to login to the management interface with their own credentials which is stored in the LDAP server. The current method used by the prototype
is to verify submitted login credentials against the local SQL database for the prototype to authenticate the credentials stored in the LDAP server.

A mechanism needs to be implemented to retrieve this information. This can be accomplished by either forwarding the credentials to the LDAP server, letting the LDAP server verify the credentials, listening to the reply sent by the LDAP server and then allowing or denying access depending on what the response is. Another way this can be accomplished is to sync the repository of the LDAP database into the SQL local database. This will require the prototype to request this information at regular intervals to keep the information current.

4.1.6. Centralization
The topic of centralisation was raised. The Raritan currently allows the engineers to manage all the connected devices from one place. The prototype however would not support this on a large scale deployment. If only one prototype was deployed in a rack that could support all required devices then centralised administration would be possible. However if several devices where needed then each device would work independently.

This would require an IP address to access device connected to one prototype and another IP address to access device attached to another prototype. While it would be theoretically possible to connect all devices to one prototype by using inline couplers to extend the length of the console cables to cable them between racks. Depending on the distance required, this can increase the cost of deployment due to excessive Ethernet costs. It would also increase the cable management of the deployment.

A solution to this is to stack two or more prototypes into one logical machine. This can be done over Ethernet. By stacking the prototypes in such a way, two physical prototypes can be joined into one logical device but be located in different racks as long as they are connected to the same local network. This means that devices located in different racks can be accessed by one logical prototype.

For security purposes, careful consideration needs to be taken as communication between the devices should not be sent over the same local network that is used for the day to day communication. Communication between devices should be segregated on a private VLAN that
cannot be accessed by other VLANs. If there is a special VLAN already created and secured for the management traffic then this VLAN should suffice.

4.1.7. Wireless Interface
One comment about the devices is the wireless capability. In an integrated environment it would not be required and may be a security concern. The wireless can be easily turned off or uninstalled as it is a USB device. This can be an option that potential users of the device can choose whether to have the wireless capability or not.

4.2. Case Study 2: University of Derby Students

4.2.1. Introduction
The Students who participated in the case studies were second and third year students on the Networks and security course. This case study would highlight if the prototype would help facilitate students with gaining access to lab equipment.

The prototype was demonstrated using the lab equipment located in the networking room of B210. The demonstration showed how students could connect to the Raspberry Pi over Ethernet or Wireless using a wireless dongle. The demonstration compared this to how the students currently connected to the lab equipment. Students currently have to plug a console cable which needed to be connected to a desktop computer and into a patch panel port near the computer. This then connects to a corresponding port located within the rack containing the lab equipment.

4.2.2. Background
Students currently have access to two server racks that contain various Cisco and Netgear switches and routers. This includes models such as Cisco 2600, 1941, 1841, 3560, 2960, Netgear FS516 and DS516. These server racks are used to give students hands on experience with configuring simulated network configurations. Students are encouraged to work through Cisco Packet tracer exercises to gain basic understanding of configuration commands. Students can then physically set up the packet tracer tutorial on the lab equipment within the racks.

4.2.3. Feedback
Students gave positive feedback with regards to the prototype giving the students the ability to be connected to multiple devices at once. Before students would only be capable of connecting
to one device at a time. To connect to a different device a student would have to physically move the console port to a different rack.

The students felt that the prototype would be a far more preferable way to connect to lab equipment if the prototype had enough console ports to connect to every device. It would also allow students to use personal devices to connect to the equipment such as laptops and tablets. If students wanted to currently use a personal device, that device would have to support an RS-232 port or a USB to console lead would be required.

4.3. Case Study 3: Ashfield School Networks Team

4.3.1. Introduction
The participants in the case study work within the networking department. The school deploys HP routers and switches. There is no current implementation for accessing the console connections of network equipment within the server rooms. If console access is required then engineers would walk to the server room with a laptop and USB-console to access networking equipment.

4.3.2. Background
Ashfield School was found 1964 in Kirkby-in-Ashfield and is currently the largest secondary school in the country with over 2500 students. The school has three areas of specialism: Technology, Applied Learning and Languages and is classed as a high performing secondary school with 52% of pupils achieving five or more A* - C GCSEs and 98% of pupils achieving A* - E for A levels in 2014.

4.3.3. Feedback
The prototype was demonstrated in a meeting room with several members of the network team. After the features were shown and explained there was positive feedback in the application of the device. While the network technicians do not use console connections and prefer to access devices over the IP network the participants said the prototype would be a useful device if console access is required.

4.3.4. KVM support.
The server environment is supported using KVM servers. A suggestion was put forward that KVM support would be a desirable feature. If KVM support was implemented alongside
console ports, it would make the prototype a more flexible device and more likely to be implemented because it would have multiple roles.

4.3.5. Centralisation
To support multiple racks, multiple prototypes would have to be deployed. The participants highlighted that this would create multiple management interfaces for each prototype. It would be desirable to have one management interface to configure all the console ports. Providing this feature would also allow administrators to gain access to console ports connected to different prototypes behind one IP address/Hostname rather than having to remember multiple addresses for different console connections.

4.3.6. LDAP integration
The topic of LDAP integration was raised. Current users are authenticated against credentials stored in the local SQL database. The participants suggested that integrating the database with LDAP would be an advantageous feature as creating and maintaining a separate database for accessing the prototype would be unnecessary and require administrators to remember multiple login credentials. Having the prototype pull the credentials of administrators from LDAP would be more desirable.

4.3.7. Out-of-band access
The participants suggested implementing the ability to access the device out-of-band such as over a 3G network. This would provide a means of accessing the device in a network down situation. If the network was down and an engineer was off site, the engineer would have to attend site to access the devices. If a cellular module was added to the device it would allow the engineer to access the device remotely to troubleshoot the issue.

4.4. Case Study 4: Nowcomm Network Engineers
4.4.1. Introduction
Nowcomm are a third Line networks company based in Derby that specialise in selling and supporting Cisco equipment. Nowcomm have multiple Cisco accreditations and specialisations including being a Cisco silver partner which is awarded to companies that hold two advanced Specializations certificates.

4.4.2. Background
Nowcomm support a wide range products from networking equipment such as routers and switches, VOIP voice systems such as Cisco Call Managers, firewalls including Cisco ASA and Juniper and VMWare environments for companies such as Claire's, Hawksford, AES Seal, The Wrekin Housing Trust, Derby City Council, API, The Coal Authority and BMW to name a few. These companies are based throughout the UK.

4.4.3. Feedback
After presenting the prototype to several network engineers there was a lot of positive feedback. Access to networks is provided to Nowcomm by each Client. Depending on the client's environment, there is no uniformed method for accessing client networks. Some clients require engineers to establish a remote VPN tunnel to a client's site and from there Telnet/SSH to network devices. Other clients require remote sessions to boxes within the network using RDP or VNC software. Different clients implement different firewalls and VPNs from different vendors which requires Nowcomm to support and document the different software and access methods for each client. The prototype can be used to as a way for engineers to gain access to client's networks in a uniform and cost effective way.

4.4.4. VPN advantages
Allowing access to device over a VPN would require the clients to maintain a list of allowed devices Nowcomm would be permitted to access for security purposes. This can lead to long and complex lists which the client will have to maintain. Using the prototype, only a short list would need to be implemented as access to devices would only be allowed over console ports. Access to any other IP address would not be necessary and would be denied. The client could easily remove the device from the network, stopping any remote access. This can then be done with ease without the risk of affecting rules implemented on a firewall.

4.4.5. Mobility
The engineers also gave positive feedback in regards to the mobility of the device. When installation of new devices is required the only possible means of initially configuring the
device is via a console connection. Having a mobile terminal server at site would save a lot of time. It means engineers would be able to configure equipment from one location and not need to move from isle to isle.

An idea was put forward to install a GSM module to the prototype to provide a means of out of band access. When client networks suffer a network outage accessing devices over VPN, Telnet or SSH is usually not possible. The engineers then have to find a way of accessing the device another way. This can mean the engineers needing to remotely control a client's laptop while the client provides a console connection. This is a complicated and time consuming method of accessing the network. Providing a different means of access to the device over a cellular network will provide engineers a less intrusive method of access to the client's network in network down situations.

4.4.6. Device logging
Another suggestion was to provide a facility for keeping track of users who have logged in and the commands that have been entered. It should be possible to keep track of connections through the prototype. Using the SQL database it would be easy to keep track of the source and destination of traffic as well as the time. Keeping track of what commands where sent would require opening the packets and determining what was sent. This would require a lot of programming. An alternative would be to set up the prototype to receive SNMP traffic from the devices that are connected. Using this protocol would reduce the need to program bespoke software that would provide the same functionality as SNMP.

4.4.7. GSM/3G Module
The use of a GSM module to provide out-of-band access would be a great addition from a support prospectus. In network down situations, it is likely that engineers would not be able to access the customer's network to troubleshoot the issue. This is when an engineer would require the customer to connect to the device physically with a laptop and connect the laptop to an alternate internet connection for example by tethering the laptop to a mobile phone. The engineer can then connect to and control the customer's laptop via a remote session such as WebEx. While this is a clever way to get around the problem it can be a lengthy process that requires the engineer to instruct and efficiently communicate with the customer.

The engineer could gain access to the device without the assistance of the customer by using the GSM/3G module to gain direct access to the prototype and thus the networking equipment.
4.5. Case Study 5: Courtaulds (UK) Limited Network team

4.5.1. Introduction
The Courtaulds network team manage over one hundred and thirty devices including seventy servers and sixty networking devices. This network supports over four hundred end users which includes remote home users. The network is spread over four sites based in the UK located at Nottingham, Belper, Loscoe and London which are all connected to a managed WAN.

4.5.2. Feedback
The prototype was demonstrated in a meeting room. After demonstrating the prototype a lot of positive feedback was given in regards to the potential of the prototype. Having a small device that could be easily sent to a customer’s site to install could be a key selling point.

4.5.3. Physical power control
One idea was raised concerning the physical control on power to networking equipment. In some situations there has been problems accessing a device over a console/IP network connection due to resource issues. There was an issue once involving a memory leak on a switch due to a bug. Once the memory leak used all of the available RAM packets were not able to be processed and hosts lost connection to the network. When an engineer tried to connect to the device via console, the engineer was unable and was only able to reboot the device by removing power.

In situations like this the prototype would also be unable to access the device over console connection. An alternative method of removing power would be advantageous. An idea would be to place a physical switch inline between a power source and the network device. The device would break the connection if a signal was received. This can be implemented using the GPIO pins on the Raspberry Pi.

4.5.4. KVM support
Having access to servers as well as routers and switches would allow the prototype to become a one device supports all. The prototype would then be used not just by the networks team but the server team as well, giving the prototype more value and usability.
4.5.5. Monitoring
The participants mentioned a potential need for a monitoring system as a feature of the prototype. Monitoring status and availability of routers, switches and servers would allow the prototype not only to log but react to alerts. If it detected a switch had gone down it could be configured to email or if a 3G module was attached, send an SMS message to everyone in the IT department. It would be possible to implement a feature for the prototype to log into the device on certain events and request log information from that device to be sent via email or SMS.

5. Results & Analysis

This section of the report will discuss how the web interface interacts with the services running on the prototype and the options available to be configured by an administrator. It will also detail the menu options presented to an administrator via the LCD display module.

It will also evaluate the energy usage of the final version prototype in a quantitative manner to determine the efficiency compared to other terminal servers.

5.1. Web Interface

The web interface allows an administrator to configure parameters of the prototype such as IP addresses, wireless settings, connected nodes, port configurations, DHCP settings and VPN settings. This section describes how the interactions are handled between an administrator and the prototype.

5.1.1. Home Page and logging in
The configuration and administration of the prototype is handled using a web browser. The Management interface is presented to the user by entering the IP address on the prototype. This IP address can be either the IP address of the Ethernet or wireless interface.
OctoPi: a low cost, energy efficient console server alternative.

Initial web page presented when connecting to the prototype:

![Initial web page](image)

*Figure 5: initial web page*

If the IP address is not known, the user can use the LCD display to ascertain an IP address.

![LCD displaying interface configuration information](image)

*Figure 6: LCD displaying Interface configuration information*

Authentication is done using credentials stored in the MySQL database. Users enter login credentials and select the login button. This sends the credentials to a PHP script. This script connects to the database to verify the details. If the credentials match, a HTML Session variable is created.

The Session Variable is checked each Time a page is loaded. If there is no Session variable then the user has not been authenticated and the page is not loaded and the user is redirected back to the home page.
5.1.2. Interface Settings Page
The first page is the interface configuration page. This page loads the information which is contained in a text file located at "/etc/networks/interfaces". If changes are made, a PHP script is used to write the new data back to the file and restart the interfaces.

![Interface Settings Page](image)

**Figure 6: Interface Settings page**

- AP mode instructs the prototype to act as an access point and broadcast an SSID.
- Client mode instructs the prototype to try and connect to a wireless network.
- Wireless IP address and subnet mask is the IP information the prototype will assign to the wireless interface.
- Static configures the Ethernet interfaces as a static IP address and uses the IP information specified in the fields Ethernet IP Address and Ethernet Subnet Mask.
- DHCP configures the Ethernet interface to get IP address information from a DHCP source.

5.1.3. Wireless Settings Page
The wireless page is where an administrator can configure the access point and wireless client parameters. The wireless method used is specified on the Interface Settings page.
under Access Point Mode
  o SSID is the ID which the prototype will broadcast for users to connect to
  o Password is the password which is needed for a wireless client to connect to the prototype
  o 802.11g channel is the channel which the SSID is broadcast on.

Under Client Mode
  o SSID is the ID which the prototype will try and connect with.
  o Password is the password required to authenticate with the wireless network.

5.1.4. Node Settings Page
Nodes page is where the administrator can configure nodes that connect to the prototype. This section is still under construction. The functionality so far is the ability to detect if nodes are able to communicate with the prototype. This is done by the administrator adding the IP address of a node to the MySQL database. A python program is then running in the background that pings each of the nodes. If the prototype gets a reply then that node is up. If the node does not reply to the ping request then for some reason that node has gone down.

A mechanism is then required to forward incoming Telnet connections onto nodes. This will allow two or more prototypes in different physical locations to act as one logical device. The master prototype will then forward Telnet connections onto the correct node. This will allow users to connect to console connections behind one IP address.
Node ID is the name of the node the administrator wants to join to this prototype.

- IP address is the IP address of the node
- The table list the currently added nodes in the database and the correct connection status of each node.

5.1.5. Network to serial Settings Page

The Network to Serial Settings page is where the administrator configures console ports for the prototype. Once the administrator adds the ports a PHP script opens the /etc/ser2net.conf file and writes the ports to the file. It then restarts the ser2net service.

The administrator can also create a configuration file for a node. This feature has not been fully implemented yet and is a part of the road map at the end of the paper. A way this may be implemented is once a configuration file has been configured for a node it is then sent to be a part of configuration file for that node. The master prototype will then assign port forwarding using SOCAT to forward incoming Telnet connections to that node.
Figure 9: Network to Serial Settings page

- Banner can be specified to display a message when a user connects to any port.
- The ports specify which Telnet port ser2net service should listen to and forward. port 1 connects to USB1 on the prototype.
- Node shows all nodes that are showing in the database as up. This feature is not fully implemented.

5.1.6. DHCP Settings page

The DHCP page allows the administrator to configure the IP address range for clients connecting to the wireless interface.

Figure 10: DHCP Settings Page

- Start IP specifies the IP address the DHCP daemon will lease from.
- End IP specifies the IP address the DHCP daemon will finish leasing at.
5.1.7. VPN Settings Page

The VPN page allows the administrator to configure the prototype as a VPN server. This will allow clients to connect to the prototype securely over an unsecure network such as the internet.

**Figure 11: VPN Settings page**

- Country, Province, City, Organisation, Email are used when generating the VPN server public and private key.
- Server IP address is the VPN tunnel termination address and should either be the Ethernet or wireless interface.
- Port is the port which the prototype will listen for incoming VPN connections. The OpenVPN default port is 1194.
- Reset server runs a batch script which generates a new public, private key pair for the server.
- Add client takes the administrator to a new page where public and private keys for a clients can be configured.

5.1.8. Client VPN Certificate Page

Once a client certificate is generated a client can use the certificate and the OpenVPN client program to connect to the prototype. This will create a secure VPN tunnel.
• New UserName is the name of the user for whom a certificate is being generated for
• Password is a password used to encrypted the client certificate with
• Reachable Address is either an IP address or hostname used to reach the prototype.
  This can either be the private hostname or IP address of the prototype or if NAT is
  configured a FQDN or public IP address.

5.2. LCD Menu Display
The PiFace Control and Display module allows the prototype to give basic configuration
information to a user without the need to connect to the prototype via HDMI, SSH or web
interface. This can help facilitate users to connect to the prototype if interface IP addresses are
forgotten or not known.

The menus can be traversed by using the first and second buttons and entered using the far right
button. Figure 13 shows the hierarchical menu displayed by the LCD screen.

Figure 12: Add VPN Client page
Figure 13: Menu options available through the LCD screen.

5.3. Power consumption of OctoPi

This section will talk about the power utilisation of the prototype and how it compares to current terminal servers. Comparing the Watt usage of current terminal servers to the prototype will illustrate the potential energy saving. Implementing a terminal server with lower energy requirements will not only reduce the annual cost of running a terminal server and cost of ownership, but will also decrease the carbon footprint of that company.

To determine the energy consumption of the prototype the following formula is used to determine the current Watts:

\[ Watts(w) = Amps(a) \times Volts(v) \]

A Device called the Charger Doctor measures the Voltage and Amps of a plugged in USB device.
Using a Charger Doctor, the voltage going into and the amps used by the prototype can be recorded. With this information Watts can be calculated and then used to compare against other terminal servers.

To gauge the power consumption a range of tests will be conducted to ascertain how much power each module draws from the prototype.

5.3.1. Test 1 basic setup
For this test only the Raspberry Pi and all of the services running on the prototype will be powered without the Console ports, Wireless interface, or LCD display connected. This will provide a baseline of the amount the prototype will draw as a minimum.

5.3.2. Test 2 Ethernet
This test involves connecting the Ethernet to determine the increase of energy consumption using the Ethernet Interface. This test will also include connecting to the prototype using a web browser and changing configuration options.

5.3.3. Test 3 Wireless Interface
This test involves the setup in test 1 with the addition of the wireless interface to determine the energy increase of using the wireless interface. A wireless client will connect and change configuration information through the web browser.
5.3.4. Test 4 LCD display
The next test will involve the basic setup with the addition of the PiFace control and display module. This test will include selecting different options within the menus and will also record the value of *sleep* mode that has been programmed.

5.3.5. Test 5 Console module
This test will involve the basic setup with the console module connected. Opening four putty sessions to each of the console ports and connecting them with lab equipment will determine if communications over the connections draw additional energy.

5.3.6. Test 6 every module connected
This test will determine the total energy consumption of every module the prototype utilises. This can then be compared to each individual test to verify accuracy of the results.

5.3.7. Results
Over all tests the voltage input remained at the same levels between 5.09 and 5.10 volts. The highest recorded reading of 5.10 volts was used in the calculations. The following table shows the minimum and maximum amps recorded for each test as well as the calculated Watts using the highest amp recorded. It will also display the difference for each module compared to the basic setup.
OctoPi: a low cost, energy efficient console server alternative.

Figure 15: Charge Doctor displaying Amp usage for Prototype using a Raspberry Pi model A

<table>
<thead>
<tr>
<th>Raspberry Pi model A</th>
<th>Amps Lowest Record</th>
<th>Amp Maximum Recorded</th>
<th>Watts</th>
<th>Difference in Watts from Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Setup</td>
<td>0.34</td>
<td>0.36</td>
<td>1.84</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet</td>
<td>0.39</td>
<td>0.42</td>
<td>2.14</td>
<td>0.3</td>
</tr>
<tr>
<td>Wireless</td>
<td>0.42</td>
<td>0.43</td>
<td>2.19</td>
<td>0.35</td>
</tr>
<tr>
<td>PiFace</td>
<td>0.43</td>
<td>0.45</td>
<td>2.3</td>
<td>0.46</td>
</tr>
<tr>
<td>PiFace sleep mode</td>
<td>0.34</td>
<td>0.36</td>
<td>1.84</td>
<td>0</td>
</tr>
<tr>
<td>Console</td>
<td>0.41</td>
<td>0.44</td>
<td>2.24</td>
<td>0.4</td>
</tr>
<tr>
<td>Everything</td>
<td>0.66</td>
<td>0.67</td>
<td>3.41</td>
<td>1.57</td>
</tr>
</tbody>
</table>

Figure 16: Raspberry Pi model A Watt results
<table>
<thead>
<tr>
<th>Raspberry Pi model B+ v1.2</th>
<th>Amps Lowest Recorded</th>
<th>Amp Maximum Recorded</th>
<th>Watts</th>
<th>Difference in Watts from Basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Setup</td>
<td>0.22</td>
<td>0.23</td>
<td>1.17</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet</td>
<td>0.26</td>
<td>0.28</td>
<td>1.42</td>
<td>0.25</td>
</tr>
<tr>
<td>Wireless</td>
<td>0.32</td>
<td>0.33</td>
<td>1.68</td>
<td>0.51</td>
</tr>
<tr>
<td>PiFace</td>
<td>0.3</td>
<td>0.31</td>
<td>1.58</td>
<td>0.41</td>
</tr>
<tr>
<td>PiFace sleep mode</td>
<td>0.22</td>
<td>0.23</td>
<td>1.17</td>
<td>0</td>
</tr>
<tr>
<td>Console</td>
<td>0.28</td>
<td>0.31</td>
<td>1.58</td>
<td>0.41</td>
</tr>
<tr>
<td>Everything</td>
<td>0.51</td>
<td>0.52</td>
<td>2.65</td>
<td>1.48</td>
</tr>
</tbody>
</table>

*Figure 17: Raspberry Pi model B+ Watt results*

The following chart shows the difference in power consumption between the Raspberry Pi model A and model B+ compared to the models currently offered by the current terminal server providers.

*Figure 18: Chart comparing power usage of the prototypes compared to terminal servers*
To extrapolate these results to show how many Watts would be consumed over a period of a year, the following formula is used:

\[ Total(t) = Watts(w) \times 24 \times 365 \]

This formula firstly calculates the Watts used in one 24 hour period and then calculates the usage over 365 days. Figure 19 shows the results for the consumption of Watts over a year period for devices listed in figure 18.

<table>
<thead>
<tr>
<th>Model</th>
<th>Consumption (Watts)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi Model A</td>
<td>29,872</td>
</tr>
<tr>
<td>Raspberry Pi Model B+ v1.2</td>
<td>23,214</td>
</tr>
<tr>
<td>Perle IOLAN STS24</td>
<td>105,120</td>
</tr>
<tr>
<td>Perle SCS45C DAC</td>
<td>197,100</td>
</tr>
<tr>
<td>Perle SDS32C LDC</td>
<td>157,680</td>
</tr>
<tr>
<td>Opengear ACM5504-2-P-4</td>
<td>36,792</td>
</tr>
<tr>
<td>Opengear ACM5508-2-M</td>
<td>55,188</td>
</tr>
<tr>
<td>Opengear ACM5508-2-I</td>
<td>56,064</td>
</tr>
<tr>
<td>Opengear ACM5504-5-G-W-I</td>
<td>98,988</td>
</tr>
<tr>
<td>Lantronix SLC 8000</td>
<td>262,800</td>
</tr>
</tbody>
</table>

*Figure 19 Energy consumption of tested devices.*
5.4. Roadmap

This section details plans for short and long-term goals to implement features which will improve the functionality of the prototype. The suggested features within this section derive from feedback given from all of the case studies. Implementing these features will create a console/terminal server with a greater flexibility and diversity which will allow the server to be deployed in a wider range of networks.

5.4.1. KVM support

The majority of the participants interviewed in the case studies either suggested or agreed that KVM support would be a desirable feature in the prototype. This was raised in case studies 1 and 3 as a feature that the participants would require and prefer over console interfaces. Participants in case studies 4 and 5 backed up this statement but mentioned console interfaces would be desirable also and would be more preferable if a product was available that supported both options rather than investing in two different products that only supported one each.

Adding KVM support will allow flexibility for administrators to choose which interfaces are required due to the modular design of the prototype. An administrator would have the ability to easily attach any amount of console and KVM connections as required using USB connections. This will also allow for easy installation and expansion of interfaces due to the plug and play nature of USB.

5.4.2. Centralisation

Centralisation was raised in case study 1 and discussed in the other case studies. To support multiple racks, multiple prototypes will need to be deployed. This will lead to administrators needing to remember different IP Addresses/hostnames to access different devices in different racks. It will also require the administrator to access different management interfaces. Having the prototypes controlled and configured behind one IP address/hostname would be more manageable and scalable. It would require one of the prototypes to take a role of master and all others to become nodes that register and take configuration options from that master.

This has been partially implemented but not fully functional as part of the current version of prototype. At present nodes can be added to the database by entering the IP addresses of the node on the master. The master then uses a heartbeat program to detect whether the node is reachable or not.
5.4.3. LDAP Integration

In Case studies 1, 3, 4 and 5 LDAP integration would be a required feature to reduce the administration of maintaining different login credentials. This feature could be implemented in different ways, either syncing the LDAP database to the local MySQL database or forwarding responses to the LDAP database and listening and responding to the replies accordingly. The advantages of having it copied locally is that less network traffic is needed to authenticate users. If for some reason LDAP server become unresponsive then users will still be able to log into the management interface with their own credentials. However this means the credentials of administrator are stored on the prototype and need to be properly secured. This may be undesirable for some administrators.

The best implementation of this is to configure both methods as features on the prototype and allow the administrator to setup and configure the chosen method. This will allow the administrator to choose the best option for that particular deployment.

5.4.4. Out-Of-Band access

Having a method of accessing the prototype over a different connection would be advantageous in network down situations. Participants in case studies 1, 3, 4 and 5 verified having remote access though an alternate connection would be a key feature. This could be implemented by installing a GSM module to allow the prototype to connect to a 3G network. The prototype would work in much the same way as over an Ethernet or wireless network with little or no extra programming required. As long as devices can connect to the prototype over the 3G network, all functionality and features of the prototype would work in the same way over the IP network.

5.4.5. Monitoring

The participants in case study 5 suggested implementing monitoring software. This is an interesting idea that could be easily implemented using open source monitoring tools native to Linux. Due to the environment and devices the prototype will be connected to enables the prototype to provide monitoring services.

5.4.6. Physical Security

The physical security of the prototype needs consideration when designing the casing for the prototype. The participants in case study 1 raised the issue of security as routers, switches and
servers are normally secured in the place by screwing them to the rack. Access to the equipment is then restricted by locked doors on each rack. With this in mind the prototype will have to follow this convention by allowing an administrator to attach the prototype within the rack in a secure way.

Several ideas were put forward by participants in case study 1 which included securing the device under the floor of the rack if there was a trap door for cabling to run under the floor, behind rack cable tidy systems, attaching the prototype to the sides of the rack and even onto on the rack if the rack environment has a secure roof. When physical security was raised in case studies 3, 4 and 5, these ideas were put forward as possible solutions which the participants agreed with as possibilities.

The prototype must then be able to be installed in these locations in a semi-permanent way. An idea would be to use Velcro to attach the prototype to surfaces such as the rack wall. In case study 5 a participant suggested installing the prototype within casing that would take up one rack even if it would mean a lot of empty space inside the casing. This would be acceptable if rack space was not an issue for an administrator.

6. Conclusion

This paper has presented the prototype OctoPi as a low cost, energy efficient alternative to current console/terminal servers. It has provided examples of console/terminal servers that are currently available on the market. The paper has benchmarked the power consumption of the OctoPi and has shown to be a more energy efficient than comparable models available on the market. Due to this energy efficiency and the low cost of the Raspberry Pi, the total cost of ownership is lower than conventional console/terminal servers. The paper introduces several case studies which involved professionals working in the networking and IT sector as well as students studying networks. After demonstrating the features of the OctoPi, the majority or participants gave positive feedback as well as constructive and insightful recommendations on improving the functionality. The general consensus from participants was the OctoPi prototype could be deployed easily into small or medium sized environments. The Student participants also gave positive feedback on the potential use within a lab environment. The paper then finishes with a road map of features recommended as part of the case study process which would allow the OctoPi to be more flexible and be integrated to support larger network environments. This paper concludes implementing features such as KVM support,
centralisation, out-of-band access and monitoring will allow the OctoPi to support a wider range of devices and meet the demand of larger networks.

6.1. Further Development

Section 5.4 highlights and discusses features which would allow the prototype to be developed further. The addition of KVM support will allow for the prototype to become a more flexible device that can be deployed to support a wider range of devices. Centralisation will enable the prototype to support multiple devices over multiple racks allowing it to support larger datacentres and server environments. Analysing the feedback from the case studies, this paper recommends these two features for the next stage of development due to the majority of participants agreeing these are desirable features.

LDAP integration and Out-of-Band access are the next features this paper recommends. Out-of-Band access would allow administrators an alternate connection method to use to access devices in network down situations. While this would provide a powerful feature, the feedback from participants in the case studies suggested that this would not be required by every administrator and should be an option that is available but the level of importance placed below that of KVM support and Centralisation. While LDAP integration was a desirable feature which would facilitate easier credential management for administrators. This feature alone would not expand the usability of the prototype and therefor the importance is also placed below that of KVM support and centralization.

The features for the next stage recommended by this paper is implementing monitoring tools and terminal access through a web browser. Enabling monitoring tools on the prototype would be advantageous given the locations the prototype may be deployed. This would potentially enable the prototype to be programmed not only to detect and inform administrators of certain events but also respond to alerts. Certain automated actions can be configured to access affected devices to retrieve extra data that ordinary monitoring systems would not be able to achieve. Access to the console ports connected to the prototype was tested using Putty. This would require an administrator to rely on a third party program to access console connections. An alternative method would be to supply a terminal session through the web interface. This would then cut the dependency of using a third party application.
6.2. Limitations

This section discusses the limitations within this report. The prototype was designed and developed using a Raspberry Pi and the Raspbian operating system. While the paper finds that the prototype met the requirements and features needed to produce a low cost energy efficient console/terminal server alternative. Other potential hardware and Operating systems were not tested. Further testing can be conducted to compare different hardware solutions such as the BeagleBone, Banana Pi or Arduino. Also while this paper was being written, Raspberry Pi 2 model B was released. Due to time constraints and this model not supporting Raspbian testing was not conducted on the Raspberry Pi 2 model B. Also testing was done with one operating system, Raspbian. To further expand the data, research using different operating systems can be used with different hardware to ascertain which operating system provides for a more energy efficient environment to develop the prototype with.
7. References


James Brunt  

OctoPi: a low cost, energy efficient console server alternative.