



Birmingham City University
School of Computing and Digital Technology
Bachelor of Science with Honours Computer Networks

**LoRaWAN Gateways as Remote Access Servers:
Designing a secure remote access solution for
branch office network devices**

Final Year Project Dissertation

Luke Dominic Tainton

16114711

A report submitted as part of the requirements for the degree of BSc (Hons)
Computer Networks at the School of Computing and Digital Technology,
Birmingham City University.

May 2020

Supervisor: Dalia El-Banna

Word Count: 10,168

Abstract

Multi-Tech Systems, an IT hardware company specialising in the Internet of Things and niche networking hardware, required a replacement solution for their ageing and problematic out of band management offering using analogue modems. The networking industry has changed drastically since modems were commonplace and many companies expect companies such as Multi-Tech to provide modern solutions to their problems. In preparation for the design stage, research into the area of network management was conducted to investigate the different approaches that are available to remotely manage networks, the systems and protocols that can be used to accomplish this, and the potential issues with managing networks in each of the approaches. A Kanban-style agile development methodology was used to develop a cost-effective solution that used Multi-Tech's Conduit LoRaWAN gateway product as a remote access server to gain access to a network device such as a switch or router. The proposed solution allows network engineers to gain easy remote access to networking equipment via SSH in the event that primary management systems become non-operational. Testing proved the security of the new system alongside the improved reliability it exhibits, whether the administrator is connecting from a local network or another country. The solution offered by this project is suitable for use in a production environment with some minor improvements, which are discussed at the end of this report.

Acknowledgements

I would like to take this opportunity to pay thanks to my project supervisor, Dalia El-Banna, for her support and guidance throughout the project. I would also like to thank Campbell Elder from Multi-Tech Systems for providing equipment and technical assistance during the testing and build phases of the project, and also to the Multi-Tech Systems support team for their timely responses to support requests that were opened during the course of the project. Thank you to Shaheed Parvez, not only for the weekly support sessions throughout the past year, but also for providing inspiration and support throughout the entire course as course leader. Finally, I would like to thank my family and friends for their support, which has helped me keep positive throughout my time at University.

Contents

1. Introduction	8
1.1. Problem definition	8
1.2. Scope	9
1.3. Rationale	9
1.4. Project aims and objectives	10
1.4.1. Aim	10
1.4.2. Objectives	10
1.5. Background information	10
1.5.1. In-band and out-of-band management	10
1.5.2. Multi-Tech Systems	11
2. Literature Review	12
2.1. Definition of out of band management	13
2.2. Types of out-of band management	13
2.3. Functionality issues	14
2.3.1. Access Control Lists	14
2.3.2. Simple Network Management Protocol	14
2.3.3. Analogue modems	15
2.4. Security issues	15
2.4.1. Telnet	15
2.4.2. Access Control Lists	16
2.4.3. Simple Network Management Protocol	16
3. Design	17
3.1. Methodology	17
3.1.1. Possible approaches	18
3.1.1.1. Write the remote access software from scratch	18
3.1.1.2. Develop a web portal to invoke terminal emulation software	18

3.1.1.3. Invoke terminal emulation software via a standard SSH connection	19
3.1.2. Chosen approach	19
3.2. Conceptual system diagram	20
3.2.1. Network device	21
3.2.2. Conduit	21
3.2.3. Router	22
3.2.4. Internet	23
4. Development	24
4.1. System preparation	24
4.1.1. Installing the hardware	24
4.1.2. Updating the Conduit	25
4.2. Compiling GNU Screen for ARM	26
4.3. Running a second SSH daemon	27
4.3.1. Creating the second daemon	27
4.3.2. Configuration	28
4.3.2.1. Sudoers file	29
4.3.2.2. Service definition file	29
4.3.2.3. Connection banners	30
4.3.2.4. Server configuration	30
5. Testing	32
5.1. Functionality	32
5.1.1. SSH daemon	32
5.1.2. Managing the network device	34
5.2. Performance	36
5.2.1. Testing methodology	36
5.2.2. Network layouts	36
5.2.3. Packet loss	37
5.2.3.1. Local network	37
5.2.3.2. Remote network	39
5.2.4. Delay & jitter	40
5.2.4.1. Local network	41

5.2.4.2. Remote network	42
6. Conclusion	44
6.1. Research summary	44
6.2. Development summary	45
6.3. Further work	45
6.3.1. SSH daemon	45
6.3.2. System security	46
6.3.3. Deployment script	47
6.3.4. Research into branch office network management	47
References	48
Bibliography	51
A. Appendix	56
A.1. Multi-Tech project proposal	56
A.1.1. Scope	56
A.1.2. Proposed solution	56
A.2. Configuration: /etc/sudoers	58
A.3. Configuration: /etc/init.d/sshd-oobm	61
A.4. Configuration: /etc/ssh/sshd_banner	64
A.5. Configuration: /etc/ssh/sshd_oobm_banner	64
A.6. Configuration: /etc/ssh/sshd_oobm_cmd	64
A.7. Configuration: /etc/ssh/sshd_config	65
A.8. Configuration: /etc/ssh/sshd_oobm_config	69

List of Figures

3.1. Physical solution connectivity diagram	20
4.1. Multi-Tech MTAC-MFSEr mCards (Multi-Tech Systems, 2019a)	24
4.2. Post-compilation binary information	27
4.3. Filesystem tree of changed files	28
4.4. Banner: standard server	30
4.5. Banner: OOBm server	30
5.1. Connecting to Conduit OOBm server	32
5.2. Switch prompt through Conduit OOBm server	33
5.3. Connecting to Conduit main SSH server	34
5.4. Conduit CLI prompt	34
5.5. SSH packets from local network	35
5.6. SSH packets from remote network	35

Listings

5.1. iPerf results (local) - packet loss - host machine	37
5.2. iPerf results (local) - packet loss - Conduit	38
5.3. iPerf results (remote) - packet loss - host machine	39
5.4. iPerf results (remote) - packet loss - Conduit	39
5.5. iPerf results (local) - delay - host machine	41
5.6. iPerf results (local) - delay - Conduit	41
5.7. iPerf results (remote) - delay - host machine	42
5.8. iPerf results (remote) - delay - Conduit	42

1. Introduction

This report will focus on the area of network management, specifically looking into methods of managing network devices remotely; these methods are commonly known as 'in band' and 'out of band' management. The difference between these is explained in section 1.5.1, but generally the terms are used to describe the different methods and protocols that are used when managing such devices. The report will discuss the development of a remote access gateway for use when managing remote network devices, designed for use in small businesses or in locations such as branch offices.

1.1. Problem definition

Most out of band management solutions are aimed at enterprise networks and involve complex setup and a large array of hardware. For branch offices and small businesses, an easier approach is needed that is easy to install and manage. Multi-Tech Systems meets this requirement with their analog modem offerings, however, these are nearing the end of their life cycle.

The Faculty was approached by Multi-Tech and was tasked with designing and building a replacement for their ageing remote access solution using analog modems, namely the MT9234ZBA. Due to the ageing, the hardware that is currently offered to customers has reached the end of its life and is no longer supported, meaning that Multi-Tech support engineers are severely limited in the support they can give to end users if the hardware develops a fault. Some functionality and security concerns were also raised in the project proposal which needed to be fixed.

It is expected that the end product of this project will be developed further by Multi-Tech and marketed to their customer base as a replacement of, and an alternative to, the current offering.

1.2. Scope

The project focused solely on out of band management systems and the protocols which run on these systems to allow engineers remote access to a network device. Most research around out of band systems was research based around software intended for managing server hardware, such as Cisco Integrated Management Controller (CIMC) and HPE's Integrated Lights Out (iLO). Any relevant parts of this research, such as the way in which the out of band networks these systems connected to were built, was used as background knowledge, however it was not taken into account during the design phase of the end product.

In the subset of research that was based on the area of network out of band management, a large majority of research articles were of an enterprise network background and referred to large, dedicated out of band networks. There was little research based specifically on out of band management for branch offices or small networks, but the research that did exist made a clear distinction between how an enterprise network would be deployed versus a branch office network. This distinction helped to limit the scope of the project to only the types of devices that might be used in a small office to provide out of band management, although the enterprise-focused research was used as background knowledge.

1.3. Rationale

This project was given to the University by Multi-Tech Systems, who directly sponsored and supported the research, development, and build phases. As a result of this, the main motivator for the project is commercialisation - both Multi-Tech Systems and Birmingham City University are seeking to benefit directly from the outcome of the project. Multi-Tech Systems are likely to attempt to market the solution on their product catalogue and make it available for their customers to purchase.

However, there are flaws in the out of band management method that this project is aiming to replace (using modems to 'dial in' to a network device), and this is the secondary motivator of the project. There is a lack of research into the security and functionality issues of using out of band management systems to access network devices, however, Multi-Tech Systems (and likely other vendors also) have received numerous reports from their customers that functionality and security issues do indeed exist within their product lines. The way in which

modems are designed to operate has security implications when paired with a device such as a Cisco router that can cause network reliability issues at best, and downtime in the worst case scenario, and these issues are something which vendors such as Cisco have needed to build protections for.

1.4. Project aims and objectives

1.4.1. Aim

The aim of this project was to design and implement a method of using a Multi-Tech Conduit LoRaWAN Gateway as a Remote Access server, to be used to remotely access the console line of a networking device such as a Cisco router.

1.4.2. Objectives

A list of objectives were identified that needed to be met to ensure that the aim was reached.

These are listed below:

- Identify the functionality and security issues that exist regarding remote access to devices via a console session.
- Evaluate the robustness of the current solution in comparison to the issues identified in the previous objective.
- Create the new RAS solution for the Multi-Tech Conduit gateway.
- Test the functionality and security of the new solution against the existing product.

1.5. Background information

1.5.1. In-band and out-of-band management

The main types of network management solutions can be placed into two groups: 'in band' management or 'out of band' management. Most methods of network management can be grouped into either one or the other, while some other methods can be a member of both groups simultaneously, dependant on the use case within a particular organisation. The main difference between in band management systems and out of band management systems is the network used to manage the devices (Outpost Sentinel, 2004). In an 'in band'

management scenario, management systems send and receive traffic over the production network alongside users' data traffic, and all production network configurations for security and quality of service apply equally to data and management traffic. An administrator attempting to remotely access a network device would connect to said device using an IP address residing on the main corporate network. In an 'out of band' management scenario, network management systems instead run on a dedicated network specifically provisioned to manage the network, allowing separation of data plane traffic and management traffic. Using a completely separate network allows administrators to make different decisions over the security and quality of service requirements that are needed. Administrators would instead connect to the device using an IP address separate to the production network, and their traffic would normally require routing from the corporate network to the out of band network.

1.5.2. Multi-Tech Systems

Multi-Tech Systems manufactures equipment for use cases relevant to the industrial Internet of Things. Founded in 1970, Multi-Tech holds over eighty United States patents in technologies ranging from DSL modems to proxy servers. The company has offices in the United States, the United Kingdom, and Japan, allowing it to reach customers from anywhere in the world. Their fast-selling product ranges are in the areas of 4G and LoRaWAN, as companies start to use IoT devices and sensors in their business processes. The main use case for the Multi-Tech Conduit used in this project is as a LoRaWAN gateway, forming a bridge between LoRaWAN-equipped sensors and an Ethernet network (such as the public internet).

2. Literature Review

To ensure that the design of the new out of band management solution fit the needs not only of Multi-Tech but of the industry in general, research was conducted using literature that was related to the following set of four topics:

- How is out of band management defined and why is it used?
- What types of out-of band management are there?
- What are the functionality issues?
- What are the security issues?

The review of this literature will emphasise the reasons why out of band management systems are used in industry, the different forms in which these systems appear, and the positive and negative effects of using such systems to manage networks ranging from small businesses to large global enterprises.

Many technologies that are used in out of band management systems can also be, and often are, used as part of in band management systems. Therefore this review, and the report as a whole, will take research centred around 'in-band' management into account, but will not go into any depth analysing it as the use of these systems in this way is beyond the scope of the project. While large-scale out of band networks were not originally within the scope of this project, a lack of available research on branch office out of band management systems required that the scope of the research phase be expanded to include this. However, a focus was applied on systems that could also be used within the smaller environments of branch offices.

2.1. Definition of out of band management

All industry experts and academics define out of band management in a similar manner, as a separate, parallel network that is designed specifically to grant IT engineers remote access to managed devices. However, most articles are focused on a specific use for the technology and therefore define it specifically for their use case; Brenkosh et al. (2005) specifically defines the requirements of an out of band network for managing military networks and ensuring they stay online even in the most challenging conditions. Alternatively, Chen et al. (2018) describe out of band management as it could be suited to a data centre, providing emergency remote access to devices in the event of an outage of the main network.

2.2. Types of out-of band management

Most research into out of band management systems was focused on rack-mount servers and virtual machines instead of networking devices such as switches and routers. This is likely due to disaster recovery methods for networks being well defined, even if they are poorly documented.

Where research was focused on network out of band management, it was aimed at large enterprises which have the capability to deploy large, scalable dedicated networks for their out of band facility. There was a lack of research into small business and branch office management using modems, which is the main focus for this project.

Out of band networks, while functionally the same between use cases, will be physically different from each other dependant on the types of system they are connecting to. As an example, Cisco IT (2003) uses console servers in their out of band network to access the console of network devices using a serial connection. In contrast, Pruett et al. (2005) uses IBM's *BladeCenter* software to manage a blade chassis remotely via a web GUI.

Generally, larger out of band management networks are physically separate from the production network and are connected to production by one or more routers. If multiple connection points are used, they are usually geographically separate to ensure that an outage on the production network at one point of connection does not affect access to the out of band network. These routers will be configured to route traffic between the two networks,

likely implementing Network Address Translation, or NAT, to ensure the networks remain physically and logically separate. Access Control Lists, or ACLs, are used at the point of interconnection to filter traffic transitioning between the two networks.

2.3. Functionality issues

Functionality issues with different types of out of band management generally arise when the network has not been installed or configured correctly. For example, it becomes extremely difficult for a large company to manage its fleet of out of band networking devices if they are all standalone access gateways, and do not form an interconnected out of band network. The reverse of this is true for small companies unnecessarily deciding to deploy a large out of band network to manage a small number of devices.

2.3.1. Access Control Lists

The use of Access Control Lists to filter traffic, while excellent for securing both the production and out of band network, can quickly become an administrative nightmare for the network team responsible for managing them. This is especially true if the production and out of band networks are interconnected at multiple points as the ACLs at each site will need to be identical. ACLs usually require manual configuration from an administrator, which is a manual process and takes time to complete. Alternatively, a Network Management System, or NMS, may be used to automate this configuration, but this will require extension of the NMS into the out of band network, which may not only be expensive to implement, but may expose security issues if the same NMS is used to manage both the out of band and production network.

2.3.2. Simple Network Management Protocol

On the topic of NMS, the Simple Network Management Protocol, or SNMP, is a widely used industry standard protocol for monitoring and managing network devices. In the case of an out of band management network, SNMP can be used when the devices being managed via the network are assigned IP addresses and have a dedicated interface connecting them to the network for the purpose of out of band management. SNMP cannot be used if the out of band practice is to remotely access devices via a console or auxiliary port. As stated by Amley (1994), however, SNMP can be responsible for network-wide latency issues if

it is allowed to generate an uncontrolled volume of management traffic. While it would make sense to place this burden on the out of band network and free up bandwidth on the production network, this can cause access problems in an emergency, rendering the out of band network useless.

2.3.3. Analogue modems

As stated by Multi-Tech in section A.1, the main functionality issue that is present when using analogue modems for out of band device management is the ability for an administrator to accidentally place the modem into command mode, rendering the connection unusable until someone can physically power cycle the modem. There is a lack of research into this area, both in academic journals and online. While there is no clear reason as to why this is, it is possible that this issue is specific to Multi-Tech's solution and there are no wider functional issues that would warrant investigation.

2.4. Security issues

2.4.1. Telnet

Most journals mentioned the failings of Telnet in terms of keeping management traffic secure, with some suggesting workarounds and alternative methods of accessing the intended device. Ivanović & Saitović (2011) provide an example of this in their best practice document published by the pan-European research network GÉANT, where they detail Telnet's non-encrypted nature and suggest that secure shell (SSH) is used instead; Mahmood (2003) suggests using TLS as an encryption layer to Telnet instead of SSH, which would essentially achieve the same result.

Many network administrators mitigate these issues by segmenting management traffic from the main production network (Dooley, 2014). This network can be physically separate, or run on the same physical hardware but be separated logically. Plixer (2009) suggests a method of logical separation involving VLANs at Layer 2 and VRFs at Layer 3, and explains the differences between them.

2.4.2. Access Control Lists

Access Control Lists are designed to improve network security by filtering unwanted traffic from the network. However, if an ACL is configured incorrectly, it can cause major security problems. Administrators must take care to ensure that ACLs are configured correctly at all times, otherwise they risk blocking necessary network traffic. In Cisco devices all traffic that is not explicitly permitted will be blocked, meaning that it is possible to block all incoming or outgoing traffic. This causes a problem not only for the users who can no longer work, but also for the administrators who can no longer remotely access the device.

2.4.3. Simple Network Management Protocol

The Simple Network Management Protocol, or SNMP, is used extensively by company IT departments to monitor computer networks and make configuration changes where required, as detailed in section 2.3.2. While SNMP can be an invaluable tool for network administrators, it can introduce security issues depending on the version of SNMP that is in use. The Department of Homeland Security (2017) issued an advisory detailing the security vulnerabilities with using SNMP version 2 over version 3. As part of the security advisory, they warned that "SNMPv3 should be the only version of SNMP employed because SNMPv3 has the ability to authenticate and encrypt payloads". In contrast, SNMP version 2 transmits information in clear text and relies on unencrypted community strings to authenticate requests. This makes it easy for a person with malicious intent to 'packet sniff' the network and retrieve these community strings.

Some level of security can be achieved with SNMP, especially when using SNMP version 2, by using traps instead of polling. Polling is the default behaviour with most NMS deployments, which queries devices under the remit of the NMS server regularly and gathers information on system health and traffic statistics. This is more insecure than traps as a malicious user can sniff for these packets and deduce a pattern based on the time interval between packets and the protocol used inside it. Sniffing tools such as Wireshark can perform this well and allow the attacker to find out where the server is located on the network, who can use this information to narrow the scope of a potential attack. SNMP traps are sent by the device to alert the NMS that something has happened. This not only saves network bandwidth by reducing the amount of traffic generated by SNMP, but improves security by reducing the attack footprint available for an attacker to exploit.

3. Design

The design stage of the project had many possible approaches that could be taken to achieve an acceptable outcome that met the aim set out in section 1.4.1. This allowed for flexibility and agility when designing the solution, which directly influenced the choice of development methodology.

3.1. Methodology

As chapter 2 proves, little academic research exists around out-of-band management of network devices, and specifically the security risks associated with it. Therefore, an inductive approach was taken during the research phase to create a base theory that was used as part of the evaluation of the finished product.

The Kanban development and task management methodology was used during this project. Kanban was designed originally for software development, where a list of tasks is kept in a Backlog, and developers pull tasks from this list into their 'in progress' queue. Once a task is complete, the developer will move the task into either a 'in review' queue, or a 'complete' queue. This allows the entire development team to easily visualise the status of tasks in the project.

While Kanban is designed for software development, in the case of this project, no software was developed so this system was used to easily see what tasks were completed and what still needed to be done. It was mainly used during the design and development phases. When designing the final product, functionality and security ideas were added to an 'Ideas' column, similar to the 'Backlog' in standard Kanban. If an idea was selected it was moved to a 'Confirmed' column, while other ideas were moved to a 'Rejected' column. This helped to prevent repetition of ideas during the design phase, and allowed previously rejected ideas to be approved if it was found that they were actually needed.

3.1.1. Possible approaches

Three potential approaches to the problem were identified and are described below. The project proposal issued by Multi-Tech Systems in appendix A.1 had some influence over the decision to choose one method over the others, however, user experience was the key driving factor behind the method of choice. Given that the main reason for using out of band management is due to loss of in-band access to a network device, the decision to use one method to develop the solution over the others was primarily based on which method would provide the easiest overall user experience for a network engineer.

3.1.1.1. Write the remote access software from scratch

This approach would require the administrator to interface with the remote device via a web portal. This web portal would allow the administrator to enter commands to be sent to the device via a serial connection, and would then return the output of that command to the administrator. A new connection between the Conduit and the device would be opened each time a command is sent, and the connection closed when the output is returned. The alternative to this is a button allowing the administrator to manually establish and terminate the connection.

All software that is available for the Conduit gateways is written in either the *C* or *C++* programming languages. Therefore, this approach requires an understanding of these programming languages, source code compilation, and software development workflows. The Multi-Tech Conduit is based on an ARM processor which also requires some prior knowledge, as compiling software for this architecture follows a different process to that of x86 architecture software compilation and requires specialist libraries.

3.1.1.2. Develop a web portal to invoke terminal emulation software

This approach is similar to the approach above and would also require the network administrator to remotely access the Conduit gateway via a web-based interface. The interface would contain an input field and a section for returned output to be displayed. The main difference between this approach and the previous approach is that the administrator would be required to click a button to start a pre-installed terminal emulator in the background (likely to be either GNU Screen or Minicom). They would then would enter commands they wish to

execute, one at a time, into a form field and click another button to send them. The application would send the command to the terminal emulator in the background, and would return the output of the executed command to the administrator. This approach would consume less system resources overall as it would not need to constantly establish and terminate terminal emulation sessions, however it may be harder to capture only the returned output for the most recent command if only one session is used.

3.1.1.3. Invoke terminal emulation software via a standard SSH connection

This approach would require the network administrator to remotely access the Conduit gateway via a standard SSH connection, either via the main SSH server or through an additional server created specifically for the purpose of out of band management. Once they are connected, and assuming that they have the rights to do so on the gateway, the administrator would then execute the terminal emulation software on the attached serial card, opening a console session to the attached device.

While it is possible to manually execute the terminal emulation software each time an administrator connects, this is cumbersome and would likely require training or a prompt to be displayed on the command line upon connection. An administrator could overcome this by writing a script that could be executed each time they login, but this would require either each administrator to do this on their first login, or major modifications to the authentication system on the Conduit to automatically copy a master script to a new user's home directory when the user is created. Both of these rely on the administrator not moving, renaming, or deleting this script. The more elegant solution is to run a second SSH server alongside the pre-existing server, which would automatically execute the terminal emulator upon connection by an authorised user. A standard Linux group could be used to manage authorised users, and the SSH server configured to check that a connecting user is a member of the designated group before executing the terminal emulation software.

3.1.2. Chosen approach

The most feasible approach to the problem was the solution described in section 3.1.1.3: to use a standard SSH connection to the Conduit, then execute terminal emulation software inside the SSH session. This allows the administrator full access to the console of the attached device. It was decided that the terminal emulation software should be invoked au-

tomatically by the SSH server when a session is established by an authorised user, requiring an additional SSH server to run alongside the primary instance.

While it was impractical to complete tests to prove this, approaching the problem using a web application would have likely overwhelmed the available system resources on the Conduit. As the Conduit is intended for use cases in the Internet of Things, it does not generally have a large amount of resources built-in and upgrading the unit is both difficult and not recommended by the manufacturer. This system would require a web server to be installed which would take up system resources, in addition to the back-end application itself and the terminal emulation software it would execute in the background on the administrator's behalf.

3.2. Conceptual system diagram

The physical design of the product is similar to that of the current solution and can be shown in a diagram in a similar manner to a network topology. Figure 3.1 shows the differences in hardware between the current and new solution.

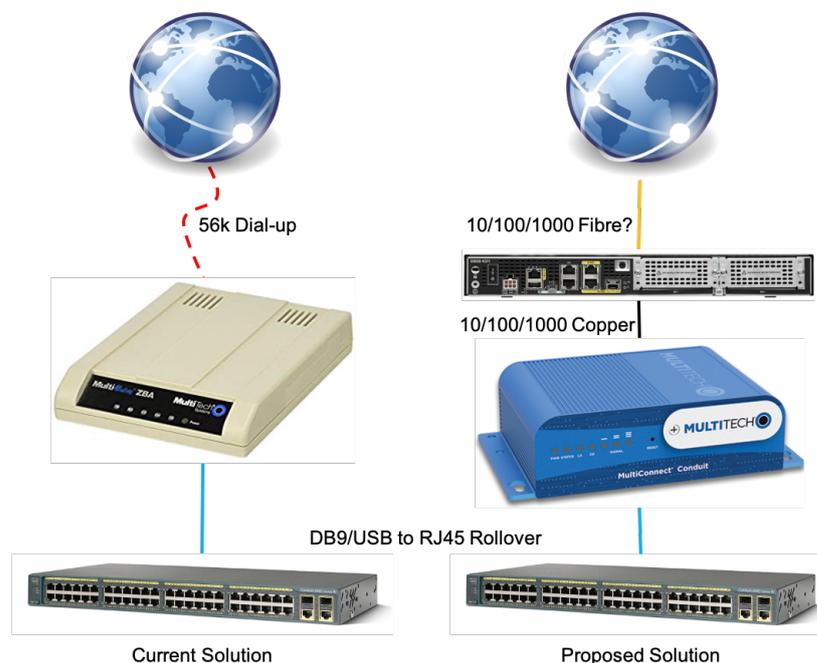


Figure 3.1.: Physical solution connectivity diagram

The main difference is that the modem that forms the main section of the original architecture has been replaced by the Multi-Tech Conduit and a router. While this does require one additional device in the new solution to achieve the same result, the additional hardware

replaces slower, outdated connectivity via a phone line with a faster and more reliable connection based on the Ethernet standard. If required, the Conduit can instead be connected to a much larger network as part of a wider out of band management strategy, unlike the previous solution which required a dedicated phone line for each modem.

3.2.1. Network device

The network device can be any device that has the capability to be managed via a console connection; the example in Figure 3.1 is a Cisco Catalyst 2960. Consumer grade or 'unmanaged' networking devices that cannot be configured via a console cable are not suitable for use with this solution. No special configuration was required on the network device; the Conduit was configured to connect to a console line using the Cisco default serial line settings (Cisco Systems, 2008):

- **Baud rate:** 9600
- **Data bits:** 8
- **Parity:** None
- **Stop bits:** 1
- **Flow control:** None

An administrator may configure the network device itself to add security or other features to console line connectivity, but this was not required for the prototype. Such features may include authentication of users on the switch itself, or a banner that may appear upon connection to warn users about logging or monitoring systems.

3.2.2. Conduit

The Conduit was connected to the console port of the network device being managed using a standard rollover cable via the MTAC-MFSER serial card installed into the back of the Conduit's chassis. This cable can either be an RJ45-DB9 cable, or an RJ45-RJ45 cable with an RJ45-DB9 adaptor. The latter was used with the prototype with no issues. It is possible that a USB console cable may also work, however this was not tested as part of the project.

The Conduit uses GNU Screen to interface with the serial card and through it, the device that is being managed. This is a CLI application requiring an administrator to open an interactive

shell session with the Conduit - running remote commands one at a time via SSH does not work with Screen and will be rejected by the SSH server.

A second SSH server was created on the Conduit to segment the SSH traffic into traffic managing the Conduit itself and traffic managing the connected device. This server ran separately from the main server, allowing different security settings to be applied to incoming connections. This allows certain users to access the Conduit via SSH to manage the network device, while still restricting their access to the Conduit itself via the main SSH server.

The Conduit was connected to the internet via the RJ45 Ethernet port at the back of the Chassis. For development and testing purposes this cable was connected to a laptop running a VyOS virtual machine acting as a NAT router. In a production scenario, the Conduit would be connected to the internet via a router or firewall. Alternatively, the Conduit is capable of connecting to a wireless network if a wired connection is not available, although there may be reliability issues when using wireless networking to connect to the internet.

3.2.3. Router

As mentioned above, a standard Ethernet connection should be used to connect the Conduit to a router or firewall capable of providing access to the internet. In the prototyping stage, a crossover cable was used to connect the Conduit to a laptop's Ethernet jack. The laptop, through the use of a virtual machine, acted as a NAT router for the Conduit, connecting it to the internet via the University's wireless network.

The Multi-Tech Systems Conduit does have wireless capability built in so it is possible to use a wireless network to connect the Conduit to a network with access to the internet, however this should not be preferred over a wired connection if at all possible. Wireless connections can be less reliable which has the potential to cause issues when connecting to the attached device. Using a wireless connection over a point to point wired connection also allows other users to connect to the same network, allowing them to sniff for packets and plan for attacks.

The method used to connect the router to the internet is not important, but it should be of a speed equal to or greater than the speed of the Ethernet connection provided to the Conduit (likely 100Mbps or 1Gbps). In a production environment, the firewall rules or router

access control lists should be configured so only the minimum access required is exposed to the internet. By default, this should be TCP port 22 for the standard SSH server and TCP port 1024 for the out of band management server, however it is more likely that the standard SSH server will not be exposed to the internet. For greater security, the SSH servers should not be directly exposed to the internet, and network administrators should use a Virtual Private Network to establish a secure tunnel between them and the router.

3.2.4. Internet

Any standard internet connection from a reputable Internet Service Provider will be suitable for connecting the Conduit to the internet, however a static IP address is required to ensure that the network administrator does not lose access to the device in the event that the IP address of the Conduit changes. Most ISPs offer this service for a small fee. If it is not possible to acquire a static IP address, it may be possible to use a dynamic DNS service that can monitor the public IP address of the Conduit for changes and automatically update the DNS system. Dynamic DNS is not covered by this project.

Once the Conduit is connected to the internet and the specified SSH ports are reachable from remote machines, the administrator will establish a connection to the Conduit using SSH on a non-standard port for security. The Conduit will prompt the administrator for their credentials and once the user passes authentication and authorisation, the SSH server automatically executes the terminal emulator in the user's shell. Authentication was handled locally in development, but in production could be handled by a remote server.

In a production environment, redundancy is paramount to ensure that engineers always have access to network devices. While redundancy the out of band network is not covered in this project, basic redundancy can be achieved by operating at least two separate internet connections, ideally from multiple providers. More than one router should be used to terminate these connections into the same out of band network, providing multiple exit routes for traffic leaving the network.

4. Development

Almost all of the work completed to develop the Conduit solution was done in software. However, the Conduit did not come equipped with serial cards by default, so some hardware changes were required before the software changes were completed. No new software was developed; instead, pre-existing software was used, their configuration files edited substantially to allow them to operate in the necessary ways for the Conduit to function as a remote access gateway.

4.1. System preparation

The Conduit required preparatory work before the main development stage could begin. The serial card that had been loaned from Multi-Tech Systems needed to be installed, and the Linux operating system was severely out of date. This work is described in more detail in sections 4.1.1 and 4.1.2 respectively.

4.1.1. Installing the hardware

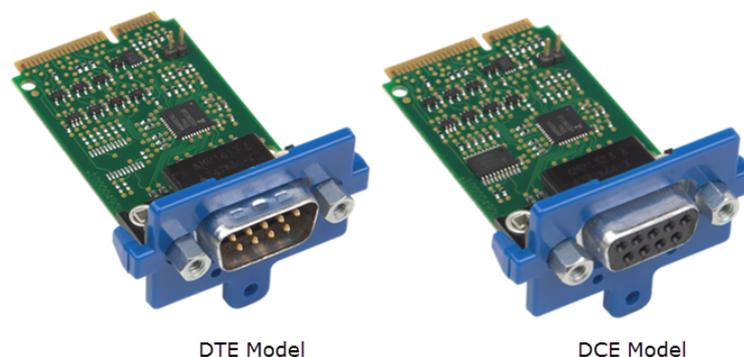


Figure 4.1.: Multi-Tech MTAC-MF5ER mCards (Multi-Tech Systems, 2019a)

The Conduit gateway is shipped by default with the MTAC-LORA accessory card installed, providing the capability for the Conduit to act as a LoRaWAN gateway for IoT applications. For this project the LoRa capability was not required, so this mCard was removed from

the Conduit and replaced with an MTAC-MF5ER-DCE card (supplied on loan by Multi-Tech Systems), which provides a female DB9 connector allowing the console cable to be attached. These cards are easy to install - a single screw holds them in place, and once the screw is removed the card can be removed or installed by sliding it in or out of the Conduit. For the purposes of the project only one mCard was installed and the other slot was covered by a blanking panel, however for production use a second mCard can be installed, allowing the Conduit to provide access to two devices with some minor reconfiguration.

4.1.2. Updating the Conduit

The next stage of development was to ensure that the Conduit gateway was in a good state to work with. Writing software and editing configuration files while the operating system and its packages are out of date may cause issues later on in the project or when used in production, where any updates that are installed can break configurations in a way that is not easily recoverable.

The Multi-Tech Conduit uses a custom distribution of Linux, although there are similarities between it and Debian. The installed package manager was not one of these similarities, however. Most Debian-based distributions use APT package manager which is managed with the `apt-get` command, but this distribution of Linux uses Opkg package manager, which is managed with the `opkg` command. APT can update the operating system and installed packages by running commands such as `apt-get upgrade`, whereas Opkg can only install and update packages, the install files of which (`.ipk` files) must be passed to Opkg as parameters upon command execution.

According to Multi-Tech Systems (2019c), upgrading the operating system on the Conduit requires that the upgrade files are placed in a specific directory (`/var/volatile` at the time of writing), then executing the upgrade executable with root privileges. It is recommended that this is done via a console connection to the gateway rather than a remote connection as the upgrade causes the gateway to reboot, which terminates the SSH server and therefore any remote connections are immediately closed.

4.2. Compiling GNU Screen for ARM

GNU Screen was chosen as the terminal emulation software that would be used for this project. Installing Screen using the Opkg package manager was impossible as there was no available install package, meaning that the software had to be installed by compiling the binary from the source code. This proved more difficult than normal, as while most desktops and laptops use an x86 architecture, the Conduit is based on an ARM architecture. The Conduit did not have the software or resources necessary to compile Screen locally, therefore this required cross-compilation of the source code on another machine - that is, a x86-based computer was used to produce an ARM-compatible executable. For this project, a virtual machine was used as this was simpler than acquiring a physical machine.

Multi-Tech provides a C/C++ toolchain for Linux-based computers, which is a specialist script that aids developers with compiling software for ARM-based processors by automatically setting environment variables, installing dependencies, and installing libraries that are required for the compilation to complete. This toolchain requires that compilation takes place on a Linux PC, therefore the compilation was completed on a virtual machine running Ubuntu 19.10. The latest version of the toolchain (5.1.8 at the time of writing) was used to ensure that any potential bugs that had been identified in previous versions were patched, and it was installed by downloading and executing the installation script from the Multi-Tech Developer portal. The command `source /opt/mlinux/5.1.8/environment-setup-arm926ejste-mlinux-linux-gnueabi` was added to the `.zshrc` file that executes each time a command line is launched, meaning that the toolchain did not need to be manually activated each time a command line session was started.

The source code for GNU Screen was downloaded from GNU's Git server, Savannah. By default the user is shown the source code in an active development state, which has the potential to cause stability issues when running the software if issues are accidentally introduced during development activities. To ensure that the software worked correctly once compiled, a stable version was used instead of the latest development version. The latest stable version at the time of writing was 4.8.0, so the version of the code tagged v4.8.0 was downloaded. Once downloaded, the `./autogen.sh` script included in the source code was executed, which automatically configures the build environment on the local machine. This script executed `autoreconf`, the utility responsible for generating the build environment

needed to compile the software. Then, `./configure` was executed, which configured the `Makefile` file. This file is designed to provide instruction to the `make` command when it is executed. In this case, `Makefile` was configured to compile the software in a way that was tailored to the computer that the software was being compiled on. This command was executed as `./configure --host arm` to indicate that the software should be compiled for ARM-based systems. Finally, the `make` command was executed, which took instruction from the `Makefile` and compiled the software into an executable binary.

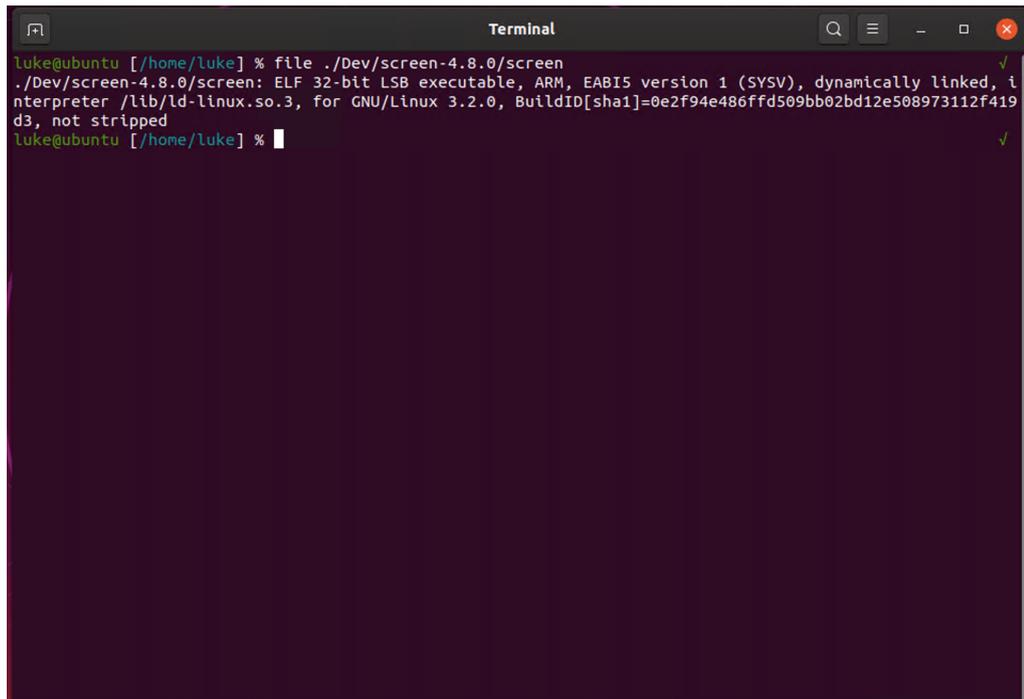
A terminal window titled "Terminal" showing the output of the `file` command. The prompt is `luke@ubuntu [/home/luke] %`. The command entered is `file ./Dev/screen-4.8.0/screen`. The output is: `./Dev/screen-4.8.0/screen: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.3, for GNU/Linux 3.2.0, BuildID[sha1]=0e2f94e486ffd509bb02bd12e508973112f419d3, not stripped`. The prompt is now `luke@ubuntu [/home/luke] %`. There are green checkmarks at the end of the command and output lines.

Figure 4.2.: Post-compilation binary information

4.3. Running a second SSH daemon

The decision was taken to run a second SSH daemon alongside the pre-existing one specifically for out of band access. Although a second SSH server wasn't required for the system to function, implementing it makes connecting to the managed device easier for the network engineers that need to do so. The alternative to running a secondary daemon is that an engineer must connect normally to the gateway via SSH and manually execute the terminal emulation software.

4.3.1. Creating the second daemon

In this distribution of Linux, in keeping with older versions of Debian, system services (daemons) are defined within the `/etc/init.d` directory. A default OpenSSH service is installed

when the operating system is installed and is defined in the `/etc/init.d/sshd` file. This definition file was duplicated to `/etc/init.d/sshd-oobm` to create a second OpenSSH service specifically for out of band device access, ensuring that as much of the configuration as possible was kept the same to ensure consistency. The service definition file for the out of band service is included in Appendix A.3.

To ensure the two SSH daemons acted independently of each other, separate PID files were used. PID files store the process identifier for a given process, which is a unique integer for each running process. Using different PID files should force the default and custom SSH daemons to start under different process IDs and therefore force the custom service to act without reliance on the default service.

Once the service definition files were created, the configuration files for the custom SSH server were copied from the original files used by the default daemon:

```
/
  etc/
    init.d/
      sshd
      sshd-oobm
    ssh/
      sshd_banner
      sshd_config
      sshd_oobm_banner
      sshd_oobm_cmd
      sshd_oobm_config
```

Figure 4.3.: Filesystem tree of changed files

The files in `/etc/init.d` define the services themselves, whereas the files in `/etc/ssh` define the configuration parameters for the service. The location of these files is specified in the service definition file.

4.3.2. Configuration

The configuration parameters for the second SSH daemon will be kept the same as the original server as much as possible to ensure consistency in their configurations and to

prevent crashes.

4.3.2.1. Sudoers file

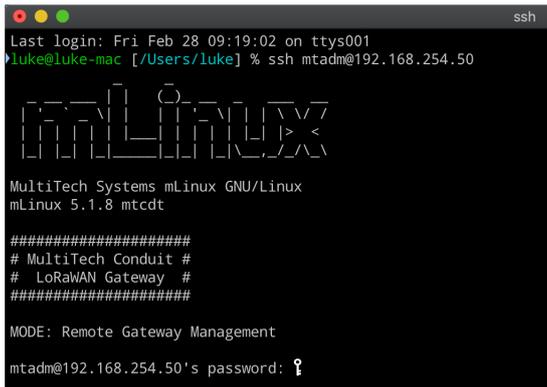
The `/etc/sudoers` file, which controls who can execute commands with root privileges, required editing to allow network administrators to reconfigure the serial card to work in RS-232 mode (Multi-Tech Systems, 2019b). The `mts-io-sysfs` command must be executed with root privileges and therefore requires the use of `sudo`. However, it was important that non-administrator users on the gateway were only able to execute that specific command with root privileges, and also that it did not prompt the user for their password. This was accomplished with `%oobm ALL=(ALL:ALL) NOPASSWD: /usr/sbin/mts-io-sysfs`, which specifies that all users in the `oobm` group can become any user, not be asked for their password, and execute only that command.

4.3.2.2. Service definition file

The `/etc/init.d/sshd-oobm` is the service definition file for the `sshd-oobm` service. This file was initially copied from `/etc/init.d/sshd` (the definition file for the standard SSH server), but required some editing to allow the OOBM service to function independently. The PID file that should be used by the service is defined by the variable `PIDFILE` in the service definition. In this instance, the PID file was changed to `/var/run/sshd-oobm.pid`, instead of `/var/run/sshd.pid` which is used by the standard server. The `CONFFILE` variable, which defines where the service should look for its configuration parameters, was also changed to `/etc/ssh/sshd_oobm_config` instead of `/etc/ssh/sshd_config`. The `check_for_no_start` function was modified to check for the existence of `/etc/ssh/sshd_oobm_not_to_be_run` instead of `/etc/ssh/sshd_not_to_be_run`. This function will prevent the server from being started if this file exists. The `check_privsep_dir` function, which checks for the existence of the Privilege Separation home directory, was changed to look for the `/var/run/sshd-oobm` directory as opposed to `/var/run/sshd`. The start, restart, and stop functions were updated to point to OOBM-specific configuration files or PIDs, and the console output (where present) was edited to remind the user that they were changing the behaviour of the OOBM SSH server instead of the standard server.

4.3.2.3. Connection banners

Two files, `/etc/ssh/sshd_banner` and `/etc/ssh/sshd_oobm_banner`, were created to define connection banners. The SSH daemons were configured to show the banners to the user by adding the line `Banner /etc/ssh/sshd[_oobm]_banner` to `/etc/ssh/sshd[_oobm]_config`. These banners are presented to the user upon connection, before a password is entered, and are intended to remind the user which SSH server they have connected to. Figures 4.4 and 4.5 show the banners in operation.



```
ssh
Last login: Fri Feb 28 09:19:02 on ttys001
>luke@luke-mac [/Users/luke] % ssh mtadm@192.168.254.50

mLinux

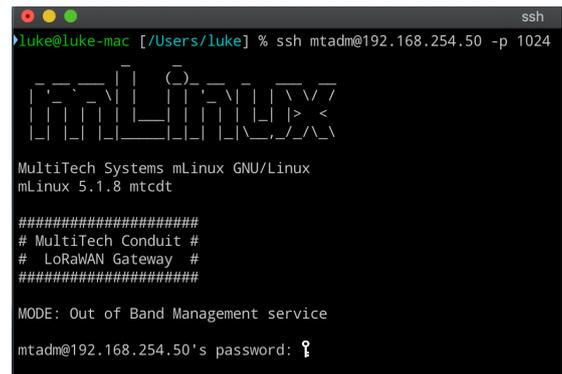
MultiTech Systems mLinux GNU/Linux
mLinux 5.1.8 mtcddt

#####
# MultiTech Conduit #
# LoRaWAN Gateway #
#####

MODE: Remote Gateway Management

mtadm@192.168.254.50's password: ?
```

Figure 4.4.: Banner: standard server



```
ssh
>luke@luke-mac [/Users/luke] % ssh mtadm@192.168.254.50 -p 1024

mLinux

MultiTech Systems mLinux GNU/Linux
mLinux 5.1.8 mtcddt

#####
# MultiTech Conduit #
# LoRaWAN Gateway #
#####

MODE: Out of Band Management service

mtadm@192.168.254.50's password: ?
```

Figure 4.5.: Banner: OOBM server

4.3.2.4. Server configuration

The configuration parameters for the servers are stored in `/etc/ssh/sshd_config` and `/etc/ssh/sshd_oobm_config`. The `/etc/ssh/sshd_oobm_config` file was created by copying the original file. This helped to save time as all required parameters were already present and only required modification. If starting from scratch, the OpenSSH documentation would have been required to identify the parameters that were required. Many of the parameters in the OOBM configuration were removed as OpenSSH mostly works on the basis that parameters only need to be present if their default value needs to be overridden.

The `Port` parameter, which controls which TCP port the server listens on, was changed from 22 to 1024, as port 22 is used by the standard SSH server. In a production scenario the chosen port would most likely be changed to a larger number, at least above 10,000. This would help to defend against port scanning attacks which usually associate SSH servers with port 22.

The Authentication parameters were changed to ensure that authorised users are able to access the connected networking device. A group named `oobm` was created where

authorised user accounts can be added. The `AllowGroups` parameter was set to `oobm`, which will ensure that only users in the `oobm` are allowed access to the device. This includes any accounts that have administrative rights on the gateway but are not authorised to access the network device. The `ForceCommand` parameter configures the SSH server to run a command or script upon connection and close the remote connection when the command exits. For the purposes of the OOBM server, this parameter was set to `/etc/ssh/sshd_oobm_cmd` (included in Appendix A.6). This script configures the serial card to operate in RS232 mode, then executes the GNU Screen terminal emulation software. When the user exists GNU Screen, their SSH session is automatically closed.

5. Testing

Testing formed an important part of the development phase. Due to the nature of the artefact that was developed, components were tested as soon as possible after development was completed on that component, and any issues that were identified during this testing were fixed. A complete testing programme was completed once the development of the product was completed.

No third parties were involved in the testing of the artefact as it was not deemed necessary at this stage. If it is decided that the product should be placed on sale, third parties will be consulted to conduct User Acceptance Testing (UAT), where it will be determined if the product is sufficiently easy to install and use.

5.1. Functionality

5.1.1. SSH daemon

The OOBM SSH daemon generally worked as expected, allowing the test user to remotely access the test device (a Cisco Catalyst 2950 switch), as shown by Figures 5.1 and 5.2.



```
luke@luke-mac [/Users/luke] $ ssh mtadm@192.168.0.10 -p 1024
MultiTech Systems mLinux GNU/Linux
mLinux 5.1.8 mtcddt
#####
# MultiTech Conduit #
# LoRaWAN Gateway #
#####
MODE: Out of Band Management service
mtadm@192.168.0.10's password: 
```

Figure 5.1.: Connecting to Conduit OOBM server

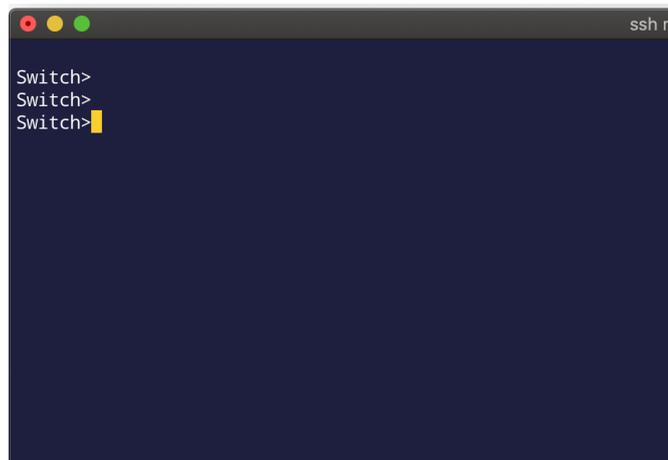


Figure 5.2.: Switch prompt through Conduit OOBM server

Unfortunately, there were issues with decoupling it from the standard SSH server for managing the gateway itself, and these issues have caused minor administrative problems. Even after manually editing the service configuration file in an attempt to force the OOBM SSH server to use a separate PID file, the two daemons continued to use the same PID. Therefore, if either SSH server requires restarting due to a software error or configuration change, both SSH servers must be restarted. This causes an issue where a mistake in the configuration file for either SSH server may cause a failure for the SSH servers to successfully restart, locking administrators out of not only the switch, but the Conduit itself. The command to restart the SSH servers must be executed against the pre-existing SSH server as `/etc/init.d/sshd restart`. The OOBM server configuration file does accept these arguments, therefore `/etc/init.d/sshd-oobm restart` can be executed, however the success of these commands upon execution was not consistent.

The pre-existing SSH server continued to function without issue and is only bound to the OOBM server via the Process ID shared between them. Even so, a software crash on the OOBM server did not cause the standard SSH server to terminate, although if this was reversed it is likely that an issue with the main SSH server could adversely affect the OOBM server. Figures 5.3 and 5.4 show a successful connection to the pre-existing SSH server.

```
luke@luke-mac [/Users/luke] $ ssh mtadm@192.168.2.2

MultiTech Systems mLinux GNU/Linux
mLinux 5.1.8 mtcdt

#####
# MultiTech Conduit #
# LoRaWAN Gateway #
#####

MODE: Remote Gateway Management

mtadm@192.168.2.2's password: █
```

Figure 5.3.: Connecting to Conduit main SSH server

```
MultiTech Systems mLinux GNU/Linux
mLinux 5.1.8 mtcdt

#####
# MultiTech Conduit #
# LoRaWAN Gateway #
#####

MODE: Remote Gateway Management

mtadm@192.168.2.2's password:
Last login: Fri May 1 15:24:35 2020 from 192.168.2.1
mtcdt:~$
mtcdt:~$
mtcdt:~$
mtcdt:~$ █
```

Figure 5.4.: Conduit CLI prompt

The banners configured to display when an administrator establishes a connection to the Conduit were presented consistently to the user at every connection.

5.1.2. Managing the network device

As discussed in section 5.1.1, the out of band SSH server worked as expected and allowed access to the command line of the connected switch. Figures 5.5 and 5.6 present packet captures that were taken by capturing SSH packets received by the Conduit using `tcpdump`. The 'Protocol' field proved that SSH version 2 was used to communicate with the Conduit, and highlighting a packet and exploring the contents of it showed that the payload data was encrypted.

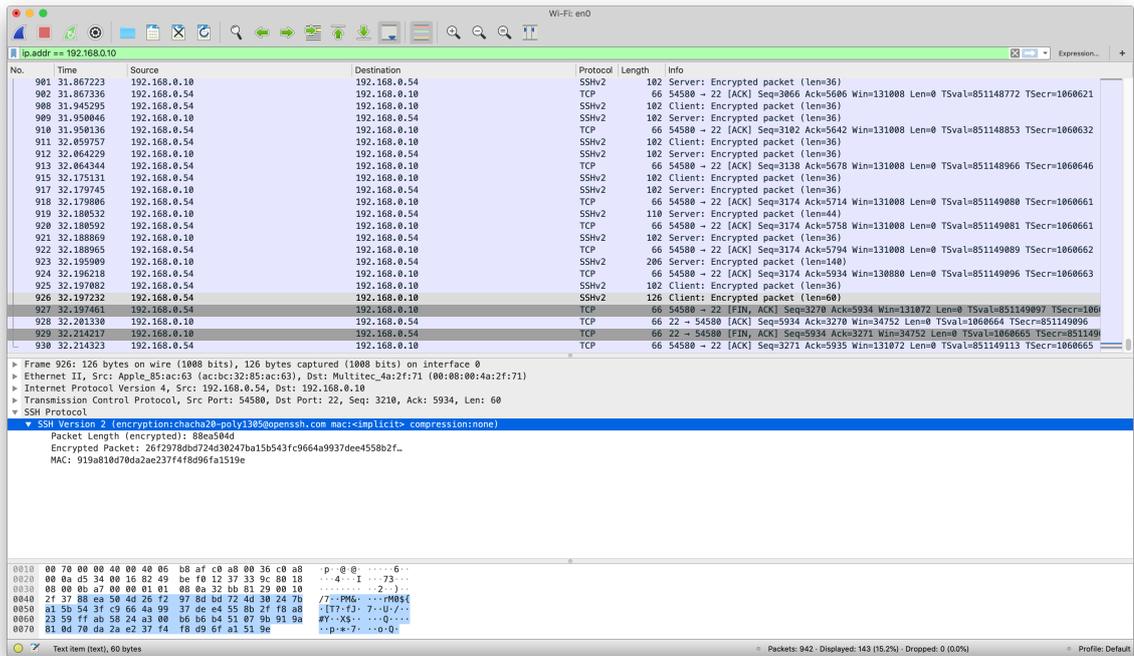


Figure 5.5.: SSH packets from local network

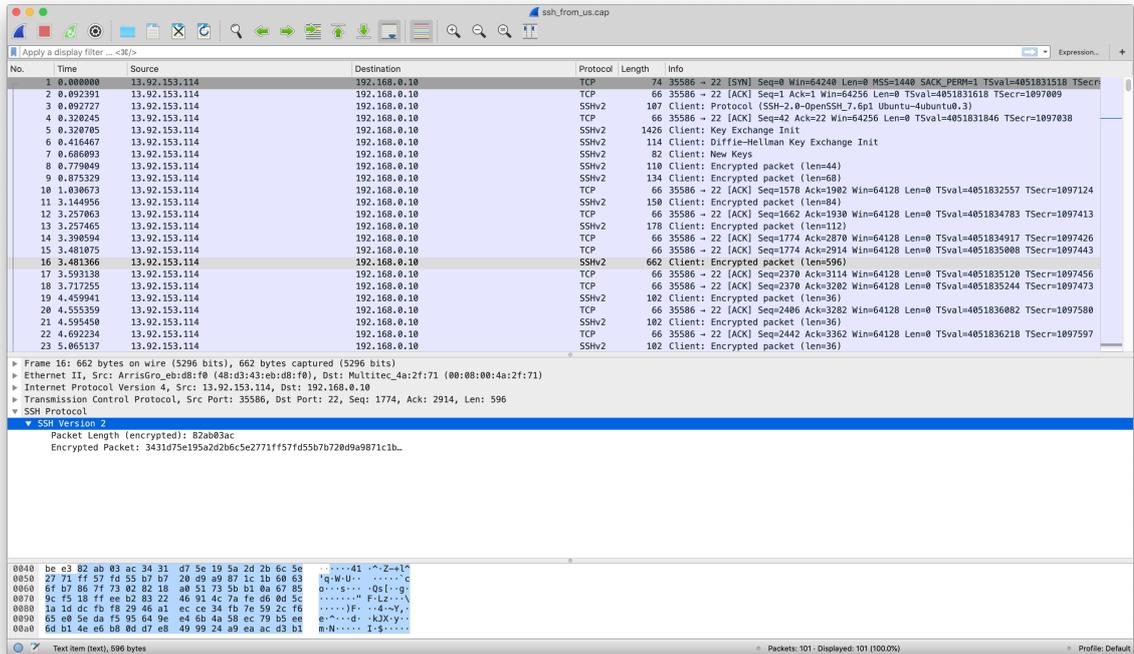


Figure 5.6.: SSH packets from remote network

5.2. Performance

5.2.1. Testing methodology

Most of the performance tests that were completed were in relation to network conditions that exist when attempting to connect to the remote access server. Connections over a phone line, as is required by analogue modems, can be unreliable at times and lead to problems such as packet loss. It was hoped that using an Ethernet connection over a phone line would alleviate some of the issues related to latency and packet loss.

Tests were completed from both a local site and a remote site. At the local site, the Conduit was directly connected to another computer or was on the same local network subnet as the computer acting as the test client. The remote site was created in Microsoft's Azure Cloud, consisting of a virtual machine residing in the East US region. Running each test from two geographically separate locations allowed for a comparison of reliability and speed between a local administrator and a remote administrator.

The main tools that were used to complete performance testing were `ping`, `tcpdump`, and `iPerf` (the `iPerf3` variant). The `ping` tool was used to measure latency between the two devices, while the `tcpdump` and `iPerf` tools were primarily used for monitoring bandwidth and packet loss between two devices.

5.2.2. Network layouts

The layout of a network and the medium with which data is transferred between connected devices can have a substantial effect on the quality of the performance of the connection between two devices; wired networks are usually more reliable than wireless networks. If packets are lost during transmission, extra resources must be used by the sender to re-transmit those packets. Delay and jitter, as explained further in section 5.2.4, can cause a negative user experience and can lead to packet loss or re-transmission if the connection times out before all the data is received.

The networks used during the tests were very different, and this difference was taken into account when the results were analysed. The 'local' network was a home network, mainly using wireless radio as the transmission medium. The 'remote' network used a virtual ma-

chine running in Microsoft Azure, and therefore was mainly formed using the infrastructure of the public internet.

5.2.3. Packet loss

Packet loss was detected by monitoring the size of the TCP congestion window between the host and the Conduit. The amount of data the sender can transmit to the receiver before requiring an acknowledgement (ACK) is known as the 'congestion window'. During normal TCP operation this window increases in size as data is transferred, allowing an increasing volume of data to be transmitted to the receiving device. However, if a problem with the connection is detected, the window size is scaled down in an attempt to fix or mitigate the effects of the problem. Packet loss can be indicative of an issue with the network and is one reason why the congestion window may be narrowed.

On the Conduit, the command `./iperf3 -f m -s -i 5` was used to run the iPerf server. On the test device, the `iperf3 -c 192.168.2.2 -f m -i 5 -t 30 -R` command was used to run the iPerf client. The various configuration flags have the following effects:

- **-s:** Run in server mode
- **-c 192.168.2.2:** Run in client mode, and connect to 192.168.2.2
- **-i 5:** Set a time interval of 5 seconds
- **-f m:** Set the unit of measurement to Megabits per second (Mbps)
- **-t 30:** Set the length of the test to thirty seconds
- **-R:** Run in reverse mode, forcing the server to send traffic to the client

5.2.3.1. Local network

In this test, the host machine that simulated an administrator's client machine was a MacBook Pro running MacOS 10.15.4 Catalina. While iPerf was able to run on both MacOS and Linux, it is possible that the differences in the operating systems caused slight differences in the execution of the tests and the output of their results.

```
1 Connecting to host 192.168.2.2, port 5201
2 Reverse mode, remote host 192.168.2.2 is sending
3 [ 5] local 192.168.2.1 port 61102 connected to 192.168.2.2 port 5201
```

```

4 [ID] Interval          Transfer Bitrate
5 [ 5] 00.00-05.00 sec  54.9 MB  92.0 Mbps
6 [ 5] 05.00-10.00 sec  55.5 MB  93.2 Mbps
7 [ 5] 10.00-15.00 sec  55.5 MB  93.1 Mbps
8 [ 5] 15.00-20.00 sec  55.5 MB  93.1 Mbps
9 [ 5] 20.00-25.00 sec  55.5 MB  93.1 Mbps
10 [ 5] 25.00-30.00 sec  55.5 MB  93.1 Mbps
11 - - - - -
12 [ID] Interval          Transfer Bitrate   Retr
13 [ 5] 00.00-30.00 sec  333 MB  93.2 Mbps    0  sender
14 [ 5] 00.00-30.00 sec  332 MB  92.9 Mbps           receiver

```

Listing 5.1: iPerf results (local) - packet loss - host machine

```

1 -----
2 Server listening on 5201
3 -----
4 Accepted connection from 192.168.2.1, port 61101
5 [ 5] local 192.168.2.2 port 5201 connected to 192.168.2.1 port 61102
6 [ID] Interval          Transfer Bandwidth   Retr  Cwnd
7 [ 5] 00.00-05.02 sec  55.7 MB  93.2 Mbps    0  223 KB
8 [ 5] 05.02-10.01 sec  55.6 MB  93.4 Mbps    0  274 KB
9 [ 5] 10.01-15.00 sec  55.7 MB  93.7 Mbps    0  419 KB
10 [ 5] 15.00-20.00 sec  55.5 MB  93.1 Mbps    0  419 KB
11 [ 5] 20.00-25.00 sec  55.4 MB  93.0 Mbps    0  419 KB
12 [ 5] 25.00-30.00 sec  55.4 MB  92.9 Mbps    0  419 KB
13 [ 5] 30.00-30.01 sec   0.00 B   0.00 Mbps    0  419 KB
14 - - - - -
15 [ID] Interval          Transfer Bandwidth   Retr
16 [ 5] 00.00-30.01 sec  333 MB  93.2 Mbps    0  sender
17 [ 5] 00.00-30.01 sec   0.00 B   0.00 Mbps           receiver

```

Listing 5.2: iPerf results (local) - packet loss - Conduit

The test results presented in Listing 5.1 show the results from the perspective of the test machine, which acted in this instance as the receiver. The bandwidth (called Bitrate on MacOS) was consistent throughout the test, averaging around 93Mbps. Strangely, the bandwidth displayed on the server in Figure 5.2 did show some slight variation, but still averaged around 93Mbps. The main metric required to measure packet loss is `Cwnd`, which reports the size of the TCP congestion window during the specified time interval. As also shown in Figure 5.2, the congestion window increased to 419KB and stabilised, indicating no issues with the

network. It is likely that the window could not increase any more in size due to memory limitations with the Conduit.

5.2.3.2. Remote network

For this test, the remote host machine used to simulate a network administrator was an Ubuntu 18.04 virtual machine running in Microsoft's Azure Cloud. The same command was used to start the iPerf server on the Conduit, however the client command changed slightly to allow the virtual machine to connect to the Conduit from the Eastern US. The **-c** flag was changed to use the public IP address of the Conduit. Establishing this connection required port forwarding on the router at the edge of the network, as Network Address Translation was being used to masquerade the Conduit's private IP address. All other flags were kept the same.

```
1 Connecting to host 86.6.167.110, port 5201
2 Reverse mode, remote host 86.6.167.110 is sending
3 [ 5] local 10.0.0.4 port 42034 connected to 86.6.167.110 port 5201
4 [ID] Interval          Transfer Bitrate
5 [ 5] 00.00-05.00 sec    5.06 MB  8.50 Mbps
6 [ 5] 05.00-10.00 sec    5.76 MB  9.66 Mbps
7 [ 5] 10.00-15.00 sec    5.88 MB  9.87 Mbps
8 [ 5] 15.00-20.00 sec    5.92 MB  9.94 Mbps
9 [ 5] 20.00-25.00 sec    5.93 MB  9.95 Mbps
10 [ 5] 25.00-30.00 sec    5.94 MB  9.97 Mbps
11 - - - - -
12 [ID] Interval          Transfer Bitrate    Retr
13 [ 5] 00.00-30.00 sec    34.9 MB  9.76 Mbps    30  sender
14 [ 5] 00.00-30.00 sec    34.5 MB  9.65 Mbps           receiver
```

Listing 5.3: iPerf results (remote) - packet loss - host machine

```
1 -----
2 Server listening on 5201
3 -----
4 Accepted connection from 13.92.153.114, port 42032
5 [ 5] local 192.168.0.10 port 5201 connected to 13.92.153.114 port 42034
6 [ID] Interval          Transfer Bandwidth  Retr  Cwnd
7 [ 5] 00.00-05.00 sec    5.30 MB  8.90 Mbps    12   108 KB
8 [ 5] 05.00-10.00 sec    5.85 MB  9.81 Mbps     0   140 KB
9 [ 5] 10.00-15.00 sec    5.79 MB  9.71 Mbps     7   133 KB
10 [ 5] 15.00-20.00 sec    5.91 MB  9.91 Mbps     0   158 KB
```

```

11 [ 5] 20.00-25.00 sec 5.97 MB 10.0 Mbps 11 140 KB
12 [ 5] 25.00-30.00 sec 6.09 MB 10.2 Mbps 0 159 KB
13 [ 5] 30.00-30.09 sec 0.00 B 0.00 Mbps 0 161 KB
14 - - - - -
15 [ID] Interval Transfer Bandwidth Retr
16 [ 5] 00.00-30.09 sec 34.9 MB 9.73 Mbps 30 sender
17 [ 5] 00.00-30.09 sec 0.00 B 0.00 Mbps receiver

```

Listing 5.4: iPerf results (remote) - packet loss - Conduit

The test results presented in Listing 5.3 show the results from the perspective of the test machine, which also acted as the receiver. Listing 5.4 shows the test results from the perspective of the Conduit. As expected, the bandwidth was lower than over the local network but remained relatively stable nonetheless. The TCP congestion window size seemed to decrease at the same time as when retries were required, indicating packet loss between the two devices. A total number of thirty re-transmissions were required over the course of the test, which, while it is desirable to have no re-transmissions, is an acceptable number considering the distance the traffic was required to travel while traversing the internet. Due to the policies in place on the Azure network, it was impossible to retrieve the list of hops the traffic passed through using `traceroute`, therefore the exact journey the test traffic took was unclear.

5.2.4. Delay & jitter

According to Indiana University NOC (2018), delay is defined as "the time that elapses between a request for information and its arrival". The delay between two endpoints often increases as the number of 'hops' or devices between two endpoints increases. Jitter is related to delay and is defined by Kamp (n.d.) as "the variation in latency of packets carrying voice or video data over a communications channel". Jitter between two devices can be caused by a number of factors, a major contributor of which is network congestion.

The same `iPerf` tool was used to measure the delay and jitter between a test machine and the Conduit, on both local and remote networks. For both tests, the command executed on the Conduit was `./iperf3 -f m -s -i 5`. This command had the same effect as in section 5.2.3.1. The command issued on the test device was different to the packet loss test. For these tests, `iperf3 -c 192.168.0.10 -f m -i 5 -t 30 -u` was issued. The main differences between the commands are that these tests were run using UDP traffic instead of

TCP traffic (as was the case in the last test), and that reverse mode was not used, meaning the client sent data to the server (in the last test, the Conduit sent data to the client).

5.2.4.1. Local network

```

1 Connecting to host 192.168.0.10, port 5201
2 [ 5] local 192.168.0.54 port 51817 connected to 192.168.0.10 port 5201
3 [ID] Interval          Transfer  Bitrate    Total Datagrams
4 [ 5] 00.00-05.00 sec    641 KB    1.05 Mbps  453
5 [ 5] 05.00-10.00 sec    641 KB    1.05 Mbps  453
6 [ 5] 10.00-15.00 sec    639 KB    1.05 Mbps  452
7 [ 5] 15.00-20.00 sec    641 KB    1.05 Mbps  453
8 [ 5] 20.00-25.00 sec    639 KB    1.05 Mbps  452
9 [ 5] 25.00-30.00 sec    641 KB    1.05 Mbps  453
10 -----
11 [ID] Interval          Transfer  Bitrate    Jitter      Lost/Total Datagrams
12 [ 5] 00.00-30.00 sec    3.75 MB   1.05 Mbps  0.000 ms    0/2716 (0%) sender
13 [ 5] 00.00-30.00 sec    3.75 MB   1.05 Mbps  5.261 ms    0/2716 (0%) receiver

```

Listing 5.5: iPerf results (local) - delay - host machine

```

1 -----
2 Server listening on 5201
3 -----
4 Accepted connection from 192.168.0.54, port 50370
5 [ 5] local 192.168.0.10 port 5201 connected to 192.168.0.54 port 51817
6 [ID] Interval          Transfer Bandwidth  Jitter      Lost/Total Datagrams
7 [ 5] 00.00-05.00 sec    638 KB    1.04 Mbps  7.556 ms    0/451 (0%)
8 [ 5] 05.00-10.00 sec    641 KB    1.05 Mbps  5.955 ms    0/453 (0%)
9 [ 5] 10.00-15.00 sec    639 KB    1.05 Mbps  4.855 ms    0/452 (0%)
10 [ 5] 15.00-20.00 sec    642 KB    1.05 Mbps  4.211 ms    0/454 (0%)
11 [ 5] 20.00-25.00 sec    639 KB    1.05 Mbps  4.716 ms    0/452 (0%)
12 [ 5] 25.00-30.00 sec    639 KB    1.05 Mbps  5.384 ms    0/452 (0%)
13 [ 5] 30.00-30.01 sec    2.83 KB    2.14 Mbps  5.261 ms    0/2 (0%)
14 -----
15 [ID] Interval          Transfer  Bandwidth  Jitter      Lost/Total Datagrams
16 [ 5] 00.00-30.01 sec    0.00 B     0.00 Mbps  5.261 ms    0/2716 (0%)

```

Listing 5.6: iPerf results (local) - delay - Conduit

The results shown in Listings 5.5 and 5.6 show that the average jitter between the two devices while connected to the same network was around five milliseconds, while the jitter of

individual time intervals increased to around seven milliseconds. Acceptable levels of jitter should not exceed thirty milliseconds (Szigeti & Hattingh, 2004), therefore both the average volume of jitter experienced in this test, and also all jitter measurements across all time intervals, are acceptable and are unlikely to cause issues when connecting to either the Conduit itself or a network device through the out of band SSH daemon.

5.2.4.2. Remote network

```

1 Connecting to host 86.6.167.110, port 5201
2 [ 5] local 10.0.0.4 port 49106 connected to 86.6.167.110 port 5201
3 [ID] Interval          Transfer  Bitrate    Total Datagrams
4 [ 5] 00.00-05.00 sec   641 KB    1.05 Mbps   453
5 [ 5] 05.00-10.00 sec   641 KB    1.05 Mbps   453
6 [ 5] 10.00-15.00 sec   639 KB    1.05 Mbps   452
7 [ 5] 15.00-20.00 sec   641 KB    1.05 Mbps   453
8 [ 5] 20.00-25.00 sec   639 KB    1.05 Mbps   452
9 [ 5] 25.00-30.00 sec   641 KB    1.05 Mbps   453
10 - - - - -
11 [ID] Interval          Transfer  Bitrate    Jitter      Lost/Total Datagrams
12 [ 5] 00.00-30.00 sec   3.75 MB   1.05 Mbps   0.000 ms    0/2716 (0%) sender
13 [ 5] 00.00-30.00 sec   3.75 MB   1.05 Mbps   4.388 ms    0/2715 (0%) receiver

```

Listing 5.7: iPerf results (remote) - delay - host machine

```

1 -----
2 Server listening on 5201
3 -----
4 Accepted connection from 13.92.153.114, port 54524
5 [ 5] local 192.168.0.10 port 5201 connected to 13.92.153.114 port 49106
6 [ID] Interval          Transfer  Bandwidth  Jitter      Lost/Total Datagrams
7 [ 5] 00.00-05.00 sec   622 KB    1.02 Mbps   6.033 ms    0/440 (0%)
8 [ 5] 05.00-10.00 sec   641 KB    1.05 Mbps   3.595 ms    0/453 (0%)
9 [ 5] 10.00-15.00 sec   641 KB    1.05 Mbps   5.387 ms    0/453 (0%)
10 [ 5] 15.00-20.00 sec   639 KB    1.05 Mbps   2.981 ms    0/452 (0%)
11 [ 5] 20.00-25.00 sec   639 KB    1.05 Mbps   4.654 ms    0/452 (0%)
12 [ 5] 25.00-30.00 sec   641 KB    1.05 Mbps   5.086 ms    0/453 (0%)
13 [ 5] 30.00-30.13 sec   17.0 KB   1.05 Mbps   4.388 ms    0/12 (0%)
14 - - - - -
15 [ID] Interval          Transfer  Bandwidth  Jitter      Lost/Total Datagrams

```

```
16 [ 5] 00.00-30.13 sec 0.00 B 0.00 Mbps 4.388 ms 0/2715 (0%)
```

Listing 5.8: iPerf results (remote) - delay - Conduit

The results shown in Listings 5.7 and 5.8 show a connection that was surprisingly better than over the local network. The average jitter during this test was a lesser four milliseconds, while the range of jitter times was only as long as six milliseconds. The results of the previous test fell into acceptable levels, and these results are no different. The lower jitter is likely due to a fully wired connection between the virtual machine and the Conduit instead of the local network test, which used a wireless connection.

6. Conclusion

6.1. Research summary

In chapter 2, four questions were set out as a basis for the research segment of the project. Research was focused on the definition and use cases of out of band management, and the functional and security issues that can be associated with it. A general definition for out of band management is an encompassing of any systems that can be used to manage a network device remotely while bypassing the production network.

The general issues that were discovered through research activities included the difficulties in manual configuration management of the dedicated network, including possible configuration discrepancies between devices, and the potential congestion introduced to a network via the addition of management traffic on the network. From a security standpoint, reinforcement was provided for the well-known fact that Telnet is insecure and should be replaced with SSH for remote connections wherever possible. Customer and engineer reports from Multi-Tech Systems detailed that when a modem is used, there is potential for it to be rendered useless through placement into command mode, or for it to inadvertently cause issues with the connected device by sending characters to the device when an administrator connects.

The second part of the research phase of the project, as set out in section 1.4.2, involved the testing of the solution currently offered by Multi-Tech Systems to gather metrics which would be used to evaluate the developed solution. Unfortunately, due to technical difficulties and time constraints while completing the project, this testing could not be completed.

6.2. Development summary

It was clear from the research that a solution was needed that was easy to install and maintain, while providing a high level of security to mitigate the risks of attack. Through the conversion of the Conduit from a LoRaWAN gateway to remote access server, these objectives were achieved. The Conduit itself is a small device and can be easily installed in almost all environments and the additional hardware, including the mCard module and console cable, can be procured at a reasonable cost. The mCard module does not require specialist tools to install. The steps required to set up the second SSH server on the Conduit can be automated via a script (see section 6.3.3), but can be easily performed manually by an administrator. Security is provided through SSH, which encrypts the traffic sent between the administrator and the Conduit, and also by using a non-standard port number for the out of band SSH server. However, the server does not start automatically and requires manual intervention by an administrator.

As mentioned in section 6.1, no metrics were collected that could be used to compare the developed solution with solutions that currently exist. Therefore, the testing performed in chapter 5 was based on standard metrics and practices that measure network latency and connection quality. Packet capturing was also used to prove that the traffic between the engineer and the Conduit was encrypted.

6.3. Further work

6.3.1. SSH daemon

At the end of the project, the out of band device access SSH daemon still did not run as its own entity, and instead was heavily reliant on the main SSH process. As explained in section 4.3.1, the second daemon was given its own PID file which should have allowed the daemon to run as a standalone application alongside the original server. However, even after being assigned a separate process ID, the out of band SSH daemon could not be restarted or stopped in isolation, nor could these actions be executed against the service designated for this daemon. The main SSH server required restarting if changes were made to either server, which would also restart the out of band SSH daemon.

Secondly, the out of band SSH daemon did not start automatically when the Conduit was powered on; this is related to the point made above. An administrator must manually start the SSH daemon each time the Conduit is power cycled by executing the `/etc/init.d/sshd-oobm start` command. This is merely an inconvenience in the event of planned maintenance, but in a situation such as an unplanned power outage, this can be catastrophic and can cause unnecessary work for engineers.

Additional work is required to identify the reason for the inability to decouple the out of band SSH server from the main SSH server. Once the reason is identified, work should be completed to rectify the problem, including ensuring that different process IDs are allocated to the daemons, and that separate log files are used to monitor the performance of each daemon. Configuring the out of band SSH server to start automatically on boot will be easy to complete once it is decoupled from the main server.

6.3.2. System security

In the safety of a private network, especially when working with a prototype, internet-facing security systems are not a major concern. However, security of the Conduit must be taken into account when used in a production environment. Common attacks against various hosts on the internet include SSH brute-forcing, which involves attempting to break into a system on the internet via an SSH server running on the target host. In the case of the Conduit, there are two SSH servers that in a production environment would be exposed directly to the internet, which doubles the attack surface for a malicious user.

Automated security systems should be installed and configured to take action against connections that appear malicious, as configured by the administrator. One such application is `Fail2ban`. It can be configured to monitor both SSH daemons' log files (reliant on section 6.3.1) and dynamically re-configure the firewall to block the IP addresses of systems that breach the failure parameters that are configured by the administrator. All requests made from a blocked IP address would then be dropped by the Conduit as they attempt to establish a connection.

6.3.3. Deployment script

The creation of an automated deployment script would save customers valuable time during the deployment of a Conduit device as a remote access gateway. Such a script would carry the requirement of being written in Bash for maximum compatibility across all models of Conduit gateway. Once created and vetted by Multi-Tech Systems to ensure the safety and functionality of the script, it could be made available to customers either on the Multi-Tech Systems website or via the support portal. The customer should be able to execute the script, likely with root privileges, and watch it automatically create the necessary files and make the required configuration changes to run a second SSH server. For more advanced usage, the script could terminate with an error if it cannot detect an installed serial card or USB to console cable.

6.3.4. Research into branch office network management

As detailed in chapter 2, there is a lack of research into methods of managing branch office network devices remotely. This lack of information made researching the topics related to the project particularly difficult, which forced some assumptions to be made based on information about general network management methods. Despite the lack of research, out of band management systems are commonplace in many enterprises, and network administrators have followed best practices to make decisions about out of band network security and design. This leaves a wealth of information that could be gathered using surveys, in sectors such as design and security, to detail how these dedicated networks are designed and configured.

References

- Amley, C. (1994) *Network management's impact on managed networks*. In: *Proceedings of 19th conference on local computer networks*, pp. 404–410. Available at:
<https://doi.org/10.1109/LCN.1994.386580>.
- Brenkosh, J. P., Witzke, E. L., Kellogg, B. R. and Olsberg, R. R. (2005) *Enhancing network survivability with out of band*. In: *MILCOM 2005 - 2005 IEEE military communications conference*, vol. 4, pp. 2494–2498. Atlantic City, NJ: IEEE, Available at:
<https://doi.org/10.1109/MILCOM.2005.1606042>. ISSN: 2155-7586.
- Chen, L., Xia, J., Yi, B. and Chen, K. (2018) *PowerMan: An Out-of-Band Management Network for Datacenters Using Power Line Communication*. In: *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pp. 561–578. Renton, WA: USENIX Association, Available at:
<https://www.usenix.org/conference/nsdi18/presentation/chen-li>.
- Cisco IT (2003) *How Cisco IT-LAN-SJ achieved high availability*. Tech. rep., Cisco Systems, Available at: https://www.cisco.com/c/dam/en_us/about/ciscoitatwork/downloads/ciscoitatwork/pdf/cisco_it_high_availability.pdf [Accessed 09 November 2019].
- Cisco Systems (2008) *Applying Correct Terminal Emulator Settings for Console Connections*. Available at: <https://www.cisco.com/c/en/us/support/docs/dial-access/asynchronous-connections/9321-terminal-settings.html> [Accessed 30 April 2020].
- Department of Homeland Security (2017) *Reducing the Risk of SNMP Abuse*. Available at:
<https://www.us-cert.gov/ncas/alerts/TA17-156A> [Accessed 30 April 2020].
- Dooley, K. (2014) *Out-of-Band Management*. *Auvik Frankly MSP Blog*. Available at:
<https://www.auvik.com/franklymsp/blog/out-of-band-management> [Accessed 06 April 2020].

- Indiana University NOC (2018) *ARCHIVED: In networking, what are bandwidth, latency, and speed?* Indiana University Knowledge Base. Available at:
<https://kb.iu.edu/d/aeud> [Accessed 06 May 2020].
- Ivanović, I. and Saitović, E. (2011) *Network Monitoring and Management Recommendations*. Tech. rep., GÉANT, Available at:
https://services.geant.net/sites/cbp/Knowledge_Base/Network_Monitoring/Documents/gn3-na3-t4-abpd101.pdf [Accessed 09 November 2019].
- Kamp, P. (n.d.) *Jitter*. Available at:
<https://www.twilio.com/docs/glossary/what-is-jitter> [Accessed 6 May 2020].
- Mahmood, H. B. (2003) *Transport layer security protocol in Telnet*. In: *9th Asia-Pacific Conference on Communications*, vol. 3, pp. 1033–1037. Penang, Malaysia: IEEE, Available at: <https://doi.org/10.1109/APCC.2003.1274255>.
- Multi-Tech Systems (2019a) *MTAC-MFSER*. online, Available at:
<http://www.multitech.net/developer/products/multiconnect-conduit-platform/accessory-cards/mtac-mfser> [Accessed 12 April 2020].
- Multi-Tech Systems (2019b) *MTAC-MFSER Usage*. Multi-Tech Systems, Available at:
<http://www.multitech.net/developer/software/mlinux/using-mlinux/mlinux-using-accessory-cards/mtac-mfser-usage> [Accessed 11 February 2020].
- Multi-Tech Systems (2019c) *Upgrading mLinux*. Multi-Tech Systems, Available at:
<http://www.multitech.net/developer/software/mlinux/upgrading-mlinux> [Accessed 9 February 2020].
- Outpost Sentinel (2004) *In Band / Out of Band Network Access*. Available at:
<http://www.outpostsentinel.com/inband.shtml> [Accessed 30 November 2019].
- Plixer (2009) *What is VRF: Virtual Routing and Forwarding*. Plixer. Available at:
<https://www.plixer.com/blog/what-is-vrf-virtual-routing-and-forwarding/> [Accessed 09 November 2019].
- Pruett, G., Abbondanzio, A., Bielski, J., Fadale, T. D., Merkin, A. E., Rafalovich, Z., Riedle, L. A. and Simpson, J. W. (2005) *BladeCenter systems management software*. *IBM Journal of Research and Development*, vol. 49(6):pp. 963–975. Available at:

<https://search-proquest-com.ezproxy.bcu.ac.uk/docview/220687956>. Tex.isbn:
00188646.

Szigeti, T. and Hattingh, C. (2004) *End-to-End QoS Network Design: Quality of Service in LANs, WANs, and VPNs*. Hoboken, NJ, USA: Cisco Press.

Bibliography

- Amley, C. (1994) *Network management's impact on managed networks*. In: *Proceedings of 19th conference on local computer networks*, pp. 404–410. Available at:
<https://doi.org/10.1109/LCN.1994.386580>.
- Bennett, B. and Transue, S. (2008) *Remote IP SATCOM monitoring and management*. In: *MILCOM 2008 - 2008 IEEE Military Communications Conference*, pp. 1–5. Available at:
<https://doi.org/10.1109/MILCOM.2008.4753503>.
- Brenkosh, J. P., Witzke, E. L., Kellogg, B. R. and Olsberg, R. R. (2005) *Enhancing network survivability with out of band*. In: *MILCOM 2005 - 2005 IEEE military communications conference*, vol. 4, pp. 2494–2498. Atlantic City, NJ: IEEE, Available at:
<https://doi.org/10.1109/MILCOM.2005.1606042>. ISSN: 2155-7586.
- Budgen, D. (2003) *Software Design*. Essex, UK: Pearson Education, 2nd edn.
- Chee, B. (2005) *Managing out-of-band management*. *InfoWorld*, vol. 27(50):p. 14.
Available at: <https://search-proquest-com.ezproxy.bcu.ac.uk/docview/194355400>.
Tex.isbn: 01996649.
- Chen, L., Xia, J., Yi, B. and Chen, K. (2018) *PowerMan: An Out-of-Band Management Network for Datacenters Using Power Line Communication*. In: *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pp. 561–578.
Renton, WA: USENIX Association, Available at:
<https://www.usenix.org/conference/nsdi18/presentation/chen-li>.
- Cisco IT (2003) *How Cisco IT-LAN-SJ achieved high availability*. Tech. rep., Cisco Systems, Available at: https://www.cisco.com/c/dam/en_us/about/ciscoitatwork/downloads/ciscoitatwork/pdf/cisco_it_high_availability.pdf [Accessed 09 November 2019].

- Cisco Systems (2008) *Applying Correct Terminal Emulator Settings for Console Connections*. Available at: <https://www.cisco.com/c/en/us/support/docs/dial-access/asynchronous-connections/9321-terminal-settings.html> [Accessed 30 April 2020].
- Department of Homeland Security (2017) *Reducing the Risk of SNMP Abuse*. Available at: <https://www.us-cert.gov/ncas/alerts/TA17-156A> [Accessed 30 April 2020].
- Dooley, K. (2014) *Out-of-Band Management*. *Auvik Frankly MSP Blog*. Available at: <https://www.auvik.com/franklymsp/blog/out-of-band-management> [Accessed 06 April 2020].
- EDUCBA (2018) *C++ vs Java - Know The Top 8 Most Important Differences*. Available at: <https://www.educba.com/c-plus-plus-vs-java/> [Accessed 28 November 2019].
Library Catalog: www.educba.com Section: Top Differences Tutorial.
- Emmert, F. (2015) *Out-of-Band Network Management*. *Network Architectures and Services*, vol. 69:pp. 69–75. Available at: https://doi.org/10.2313/NET-2015-03-1_10.
- eWeek (2005) *Cyclades Extends Out-of-Band Management to Remote Offices*. *eWeek*. Available at: <https://link.gale.com/apps/doc/A132116999/AONE?u=uce&sid=AONE&xid=3ca148f9> [Accessed 06 November 2019].
- Franklin, A. and Perovic, S. (2019) *Experiment in Physics*. In: Zalta, E. N., ed., *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2019 edn.
- Gonçales, L. J., Farias, K., Oliveira, T. C. D. and Scholl, M. (2019) *Comparison of software design models: An extended systematic mapping study*. *ACM Computing Surveys (CSUR)*, vol. 52(3):pp. 1–41. Available at: <https://doi.org/10.1145/3313801>.
Tex.publisher: ACM New York, NY, USA.
- Indiana University NOC (2018) *ARCHIVED: In networking, what are bandwidth, latency, and speed?* *Indiana University Knowledge Base*. Available at: <https://kb.iu.edu/d/aeud> [Accessed 06 May 2020].

- InformationWeek (2005) *Out-Of-Band Systems Management Helps Businesses Expand; Uplogix introduces a new version of its network management software for centralized and remote management that makes use of out-of-band techniques to gain access to servers and other IT systems that have crashed, lost power, or have faulty network connections.* InformationWeek. Available at:
<https://link.gale.com/apps/doc/A137823043/AONE?u=uce&sid=AONE&xid=e8820255>
 [Accessed 06 November 2019].
- Ivanović, I. and Saitović, E. (2011) *Network Monitoring and Management Recommendations*. Tech. rep., GÉANT, Available at:
https://services.geant.net/sites/cbp/Knowledge_Base/Network_Monitoring/Documents/gn3-na3-t4-abpd101.pdf [Accessed 09 November 2019].
- Kamp, P. (n.d.) *Jitter*. Available at:
<https://www.twilio.com/docs/glossary/what-is-jitter> [Accessed 6 May 2020].
- Kawahara, S. and Kourai, K. (2014) *The continuity of out-of-band remote management across virtual machine migration in clouds*. In: *2014 IEEE/ACM 7th international conference on utility and cloud computing*, pp. 176–185. London, UK: IEEE, Available at:
<https://doi.org/10.1109/UCC.2014.26> [Accessed 11 March 2020].
- Kolaks, M. S. (2003) *Securing out-of-band device management*. *SANS Institute*, vol. 21. Available at: <https://www.sans.org/reading-room/whitepapers/networkdevs/securing-out-of-band-device-management-906> [Accessed 11 November 2019].
- Lonergan, K. (2016) *Agile versus Waterfall: pros and cons & difference between them*. Available at: <https://www.pmis-consulting.com/agile-versus-waterfall/>
 [Accessed 22 November 2019].
- Mahmood, H. B. (2003) *Transport layer security protocol in Telnet*. In: *9th Asia-Pacific Conference on Communications*, vol. 3, pp. 1033–1037. Penang, Malaysia: IEEE, Available at: <https://doi.org/10.1109/APCC.2003.1274255>.
- McAfee (2014) *What is Wardriving?* Available at: <https://securingtomorrow.mcafee.com/blogs/consumer/identity-protection/wardriving> [Accessed 28 November 2019].
- MRV Communications (2006) *MRV DEBUTS OUT-OF-BAND MANAGEMENT SYSTEM WITH SECURITY*. *Computer Protocols*, vol. 19(8):p. N/A. Available at:

- <https://search-proquest-com.ezproxy.bcu.ac.uk/docview/202829608>. Tex.isbn: 0899126X.
- Multi-Tech Systems (2019a) *MTAC-MFSER*. online, Available at: <http://www.multitech.net/developer/products/multiconnect-conduit-platform/accessory-cards/mtac-mfser> [Accessed 12 April 2020].
- Multi-Tech Systems (2019b) *MTAC-MFSER Usage*. Multi-Tech Systems, Available at: <http://www.multitech.net/developer/software/mlinux/using-mlinux/mlinux-using-accessory-cards/mtac-mfser-usage> [Accessed 11 February 2020].
- Multi-Tech Systems (2019c) *Upgrading mLinux*. Multi-Tech Systems, Available at: <http://www.multitech.net/developer/software/mlinux/upgrading-mlinux> [Accessed 9 February 2020].
- Outpost Sentinel (2004) *In Band / Out of Band Network Access*. Available at: <http://www.outpostsentinel.com/inband.shtml> [Accessed 30 November 2019].
- Plixer (2009) *What is VRF: Virtual Routing and Forwarding*. Plixer. Available at: <https://www.plixer.com/blog/what-is-vrf-virtual-routing-and-forwarding/> [Accessed 09 November 2019].
- Pontillo, M. (2011) *Can I use http tunnel to ping or traceroute through a proxy with firewall?: comment*. Available at: <https://stackoverflow.com/questions/4903002/can-i-use-http-tunnel-to-ping-or-traceroute-through-a-proxy-with-firewall> [Accessed 4 May 2020].
- Pruett, G., Abbondanzio, A., Bielski, J., Fadale, T. D., Merkin, A. E., Rafalovich, Z., Riedle, L. A. and Simpson, J. W. (2005) *BladeCenter systems management software*. *IBM Journal of Research and Development*, vol. 49(6):pp. 963–975. Available at: <https://search-proquest-com.ezproxy.bcu.ac.uk/docview/220687956>. Tex.isbn: 00188646.
- Stormboard (2018) *Your Guide to Using Stormboard For Kanban*. Available at: <https://stormboard.com/blog-archive/your-guide-to-using-stormboard-for-kanban> [Accessed 08 November 2019].
- Suneja, S., Isci, C., Bala, V., de Lara, E. and Mummert, T. (2014) *Non-Intrusive, out-of-Band and out-of-the-Box Systems Monitoring in the Cloud*. In: *The 2014 ACM*

International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS '14, p. 249–261. New York, NY, USA: Association for Computing Machinery, Available at: <https://doi.org/10.1145/2591971.2592009>.

Szigeti, T. and Hattingh, C. (2004) *End-to-End QoS Network Design: Quality of Service in LANs, WANs, and VPNs*. Hoboken, NJ, USA: Cisco Press.

Zhang, C. and Budgen, D. (2012) *What do we know about the effectiveness of software design patterns? IEEE Transactions on Software Engineering*, vol. 38(5):pp. 1213–1231. Available at: <https://doi.org/10.1109/TSE.2011.79>.

A. Appendix

A.1. Multi-Tech project proposal

A.1.1. Scope

Many companies still use analogue modems for out-of-band management for devices such as Cisco routers and telephone systems so they can have management companies (or themselves) dial into remote locations to make changes. This is then a protected connection as it does not necessitate use of the internal network to access the hardware.

The modems they traditionally use is the Multi-Tech MT9234ZBA analogue modem.

Accessing Cisco routers through a console port is known to have some issues:

- Escape sequences can be passed to the device through data uploads, that can put modems into command mode (breaking the connection to a non-recoverable state without power cycling the modem)
- The modem generates characters when a remote device connects, that can put a Cisco router in to command mode.

The Multi-Tech MT9234ZBA modem is going end of life, and along with it some built in features to protect against the above. The entire solution is now end of life, including the legacy chips inside, with no replacement options from any manufacturer.

A.1.2. Proposed solution

There is a potential 'older' solution that could be re-engineered - the RAS or remote access server. Using Multi-Tech Conduit gateways:

- Build a RAS application into the gateway
- Attach two serial ports to it (Accessory cards) you could attach a still manufactured modem to one port – giving RAS dial in access and an IP connection.

- With the IP connection, you could then enable a terminal to the second serial port and whatever terminal is connected to it (like a Cisco router), thus recovering out of band management capabilities via the requested analogue phone lines with a secured connection (username and password in the RAS) and separation from the terminal.

This could then be offered as an alternative solution to the customer base that will soon have no alternative.

A.2. Configuration: /etc/sudoers

```
1 ## sudoers file.
2
3 ## This file MUST be edited with the 'visudo' command as root.
4 ## Failure to use 'visudo' may result in syntax or file permission errors
5 ## that prevent sudo from running.
6 ##
7 ## See the sudoers man page for the details on how to write a sudoers file.
8
9 ## Host alias specification
10 ##
11 ## Groups of machines. These may include host names (optionally with
12 ## wildcards),
13 ## IP addresses, network numbers or netgroups.
14 # Host_Alias  WEBSERVERS = www1, www2, www3
15
16 ## User alias specification
17 ##
18 ## Groups of users. These may consist of user names, uids, Unix groups,
19 ## or netgroups.
20 # User_Alias  ADMINS = millert, dowdy, mikef
21
22 ## Cmnd alias specification
23 ##
24 ## Groups of commands. Often used to group related commands together.
25 # Cmnd_Alias  PROCESSES = /usr/bin/nice, /bin/kill, /usr/bin/renice, \
26 #             /usr/bin/pkill, /usr/bin/top
27 # Cmnd_Alias  REBOOT = /sbin/halt, /sbin/reboot, /sbin/poweroff
28
29 ## Defaults specification
30 ##
31 ## You may wish to keep some of the following environment variables
32 ## when running commands via sudo.
33 ##
34 ## Locale settings
35 # Defaults env_keep += "LANG LANGUAGE LINGUAS LC_* _XKB_CHARSET"
36 ##
37 ## Run X applications through sudo; HOME is used to find the
38 ## .Xauthority file. Note that other programs use HOME to find
39 ## configuration files and this may lead to privilege escalation!
```

```

39 # Defaults env_keep += "HOME"
40 ##
41 ## X11 resource path settings
42 # Defaults env_keep += "XAPPLRESDIR XFILESEARCHPATH XUSERFILESEARCHPATH"
43 ##
44 ## Desktop path settings
45 # Defaults env_keep += "QTDIR KDEDIR"
46 ##
47 ## Allow sudo-run commands to inherit the callers' ConsoleKit session
48 # Defaults env_keep += "XDG_SESSION_COOKIE"
49 ##
50 ## Uncomment to enable special input methods. Care should be taken as
51 ## this may allow users to subvert the command being run via sudo.
52 # Defaults env_keep += "XMODIFIERS GTK_IM_MODULE QT_IM_MODULE QT_IM_SWITCHER"
53 ##
54 ## Uncomment to use a hard-coded PATH instead of the user's to find commands
55 Defaults secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin
    :/bin"
56 ##
57 ## Uncomment to send mail if the user does not enter the correct password.
58 # Defaults mail_badpass
59 ##
60 ## Uncomment to enable logging of a command's output, except for
61 ## sudoreplay and reboot. Use sudoreplay to play back logged sessions.
62 # Defaults log_output
63 # Defaults!/usr/bin/sudoreplay !log_output
64 # Defaults!/usr/local/bin/sudoreplay !log_output
65 # Defaults!REBOOT !log_output
66
67 ## Runas alias specification
68
69 ## User privilege specification
70 root ALL=(ALL) ALL
71
72 ## Uncomment to allow members of group wheel to execute any command
73 # %wheel ALL=(ALL) ALL
74
75 ## Same thing without a password
76 # %wheel ALL=(ALL) NOPASSWD: ALL
77
78 ## Uncomment to allow members of group sudo to execute any command

```

```
79 %sudo ALL=(ALL) ALL
80
81 ## OOBM
82 %sudo ALL=(ALL:ALL) NOPASSWD: /usr/sbin/mts-io-sysfs
83 %oobm ALL=(ALL:ALL) NOPASSWD: /usr/sbin/mts-io-sysfs
84
85 ## Uncomment to allow any user to run sudo if they know the password
86 ## of the user they are running the command as (root by default).
87 # Defaults targetpw # Ask for the password of the target user
88 # ALL ALL=(ALL) ALL # WARNING: only use this together with 'Defaults
    targetpw'
89
90 ## Read drop-in files from /etc/sudoers.d
91 ## (the '#' here does not indicate a comment)
92 #includedir /etc/sudoers.d
```

A.3. Configuration: /etc/init.d/sshd-oobm

```
1 #! /bin/sh
2 set -e
3
4 PIDFILE=/var/run/sshd-oobm.pid
5 CONFFILE=/etc/ssh/sshd_oobm_config
6
7 # source function library
8 . /etc/init.d/functions
9
10 # /etc/init.d/sshd-oobm: start and stop the OpenBSD "secure shell" daemon for
    OOBM
11
12 test -x /usr/sbin/sshd || exit 0
13 ( /usr/sbin/sshd -\? 2>&1 | grep -q OpenSSH ) 2>/dev/null || exit 0
14
15 # /etc/default/ssh may set SYSCONFDIR and SSHD_OPTS
16 if test -f /etc/default/ssh; then
17     . /etc/default/ssh
18 fi
19
20 [ -z "$SYSCONFDIR" ] && SYSCONFDIR=/etc/ssh
21 mkdir -p $SYSCONFDIR
22
23 HOST_KEY_RSA=$SYSCONFDIR/ssh_host_rsa_key
24 HOST_KEY_DSA=$SYSCONFDIR/ssh_host_dsa_key
25 HOST_KEY_ECDSA=$SYSCONFDIR/ssh_host_ecdsa_key
26 HOST_KEY_ED25519=$SYSCONFDIR/ssh_host_ed25519_key
27
28 check_for_no_start() {
29     # forget it if we're trying to start, and /etc/ssh/
    sshd_oobm_not_to_be_run exists
30     if [ -e $SYSCONFDIR/sshd_oobm_not_to_be_run ]; then
31         echo "OpenBSD Secure Shell server for OOBM not in use"
32         exit 0
33     fi
34 }
35
36 check_privsep_dir() {
37     # Create the PrivSep empty dir if necessary
```

```

38     if [ ! -d /var/run/sshd-oobm ]; then
39         mkdir /var/run/sshd-oobm
40         chmod 0755 /var/run/sshd-oobm
41     fi
42 }
43
44 check_config() {
45     /usr/sbin/sshd -t $SSHD_OPTS || exit 1
46 }
47
48 check_keys() {
49     # create keys if necessary
50     if [ ! -f $HOST_KEY_RSA ]; then
51         echo " generating ssh RSA key..."
52         ssh-keygen -q -f $HOST_KEY_RSA -N '' -t rsa
53     fi
54     if [ ! -f $HOST_KEY_ECDSA ]; then
55         echo " generating ssh ECDSA key..."
56         ssh-keygen -q -f $HOST_KEY_ECDSA -N '' -t ecdsa
57     fi
58     if [ ! -f $HOST_KEY_DSA ]; then
59         echo " generating ssh DSA key..."
60         ssh-keygen -q -f $HOST_KEY_DSA -N '' -t dsa
61     fi
62     if [ ! -f $HOST_KEY_ED25519 ]; then
63         echo " generating ssh ED25519 key..."
64         ssh-keygen -q -f $HOST_KEY_ED25519 -N '' -t ed25519
65     fi
66 }
67
68 export PATH="${PATH:+$PATH:}/usr/sbin:/sbin"
69
70 case "$1" in
71     start)
72         check_for_no_start
73         echo "Starting OOBM server"
74         check_keys
75         check_privsep_dir
76         start-stop-daemon -S -p $PIDFILE --make-pidfile --startas /usr/sbin/
sshd \
77         -- -f $CONFFILE $SSHD_OPTS

```

```

78     echo "done."
79     ;;
80     stop)
81         echo -n "Stopping OOBM server"
82         pidno1=$(<${PIDFILE})
83         pidno=$((pidno1 + 1))
84         kill $pidno
85         echo "."
86     ;;
87     reload|force-reload|restart)
88         check_for_no_start
89         check_keys
90         check_config
91         echo -n "Reloading OOBM server configuration"
92         /etc/init.d/sshd-oobm stop
93         /etc/init.d/sshd-oobm start
94         echo "."
95     ;;
96     status)
97         pidno1=$(<${PIDFILE})
98         pidno=$((pidno1 + 1))
99         if ps -p $pidno > /dev/null
100        then
101            echo "OOBM server ($pidno) is running"
102        else
103            echo "OOBM server is not running"
104        fi
105        exit $?
106     ;;
107     *)
108         echo "Usage: /etc/init.d/sshd-oobm {start|stop|status|reload \
109         |force-reload|restart}"
110        exit 1
111 esac
112 exit 0

```

A.4. Configuration: /etc/ssh/sshd_banner

```
1 #####
2 # MultiTech Conduit #
3 # LoRaWAN Gateway #
4 #####
5
6 MODE: Remote Gateway Management
```

A.5. Configuration: /etc/ssh/sshd_oobm_banner

```
1 #####
2 # MultiTech Conduit #
3 # LoRaWAN Gateway #
4 #####
5
6 MODE: Out of Band Management service
```

A.6. Configuration: /etc/ssh/sshd_oobm_cmd

```
1 #!/usr/bin/env bash
2 sudo mts-io-sysfs store ap1/serial-mode rs232
3 /usr/bin/screen /dev/ttyAP1 9600
```

A.7. Configuration: /etc/ssh/sshd_config

```
1 # $OpenBSD: sshd_config,v 1.80 2008/07/02 02:24:18 djm Exp $
2
3 # This is the sshd server system-wide configuration file. See
4 # sshd_config(5) for more information.
5
6 # This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin
7
8 # The strategy used for options in the default sshd_config shipped with
9 # OpenSSH is to specify options with their default value where
10 # possible, but leave them commented. Uncommented options change a
11 # default value.
12
13 #Port 22
14 #AddressFamily any
15 #ListenAddress 0.0.0.0
16 #ListenAddress ::
17
18 # The default requires explicit activation of protocol 1
19 Protocol 2
20
21 # HostKey for protocol version 1
22 #HostKey /etc/ssh/ssh_host_key
23 # HostKeys for protocol version 2
24 #HostKey /etc/ssh/ssh_host_rsa_key
25 #HostKey /etc/ssh/ssh_host_dsa_key
26 #HostKey /etc/ssh/ssh_host_ecdsa_key
27 #HostKey /etc/ssh/ssh_host_ed25519_key
28
29 # Lifetime and size of ephemeral version 1 server key
30 #KeyRegenerationInterval 1h
31 #ServerKeyBits 1024
32
33 # Ciphers and keying
34 #RekeyLimit default none
35
36 # Logging
37 # obsoletes QuietMode and FascistLogging
38 #SyslogFacility AUTH
39 #LogLevel INFO
```

```
40
41 # Authentication:
42 #LoginGraceTime 2m
43 #PermitRootLogin yes
44 #StrictModes yes
45 #MaxAuthTries 6
46 #MaxSessions 10
47
48 #RSAAuthentication yes
49 #PubkeyAuthentication yes
50
51 # The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
52 # but this is overridden so installations will only check .ssh/
    authorized_keys
53 AuthorizedKeysFile .ssh/authorized_keys
54
55 #AuthorizedPrincipalsFile none
56
57 #AuthorizedKeysCommand none
58 #AuthorizedKeysCommandUser nobody
59
60 # For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
61 #RhostsRSAAuthentication no
62 # similar for protocol version 2
63 #HostbasedAuthentication no
64 # Change to yes if you don't trust ~/.ssh/known_hosts for
65 # RhostsRSAAuthentication and HostbasedAuthentication
66 #IgnoreUserKnownHosts no
67 # Don't read the user's ~/.rhosts and ~/.shosts files
68 #IgnoreRhosts yes
69
70 # To disable tunneled clear text passwords, change to no here!
71 #PasswordAuthentication yes
72 #PermitEmptyPasswords no
73
74 # Change to no to disable s/key passwords
75 ChallengeResponseAuthentication no
76
77 # Kerberos options
78 #KerberosAuthentication no
79 #KerberosOrLocalPasswd yes
```

```
80 #KerberosTicketCleanup yes
81 #KerberosGetAFSToken no
82
83 # GSSAPI options
84 #GSSAPIAuthentication no
85 #GSSAPICleanupCredentials yes
86
87 # Set this to 'yes' to enable PAM authentication, account processing,
88 # and session processing. If this is enabled, PAM authentication will
89 # be allowed through the ChallengeResponseAuthentication and
90 # PasswordAuthentication. Depending on your PAM configuration,
91 # PAM authentication via ChallengeResponseAuthentication may bypass
92 # the setting of "PermitRootLogin without-password".
93 # If you just want the PAM account and session checks to run without
94 # PAM authentication, then enable this but set PasswordAuthentication
95 # and ChallengeResponseAuthentication to 'no'.
96 UsePAM yes
97
98 #AllowAgentForwarding yes
99 #AllowTcpForwarding yes
100 #GatewayPorts no
101 X11Forwarding yes
102 #X11DisplayOffset 10
103 #X11UseLocalhost yes
104 #PermitTTY yes
105 #PrintMotd yes
106 #PrintLastLog yes
107 #TCPKeepAlive yes
108 #UseLogin no
109 UsePrivilegeSeparation sandbox # Default for new installations.
110 #PermitUserEnvironment no
111 Compression no
112 ClientAliveInterval 15
113 ClientAliveCountMax 4
114 #UseDNS yes
115 #PidFile /var/run/sshd.pid
116 #MaxStartups 10:30:100
117 #PermitTunnel no
118 #ChrootDirectory none
119 #VersionAddendum none
120
```

```
121 # no default banner path
122 Banner /etc/ssh/ssh_banner
123
124 # override default of no subsystems
125 Subsystem sftp /usr/libexec/sftp-server
126
127 # Example of overriding settings on a per-user basis
128 #Match User anoncvs
129 # X11Forwarding no
130 # AllowTcpForwarding no
131 # PermitTTY no
132 # ForceCommand cvs server
```

A.8. Configuration: /etc/ssh/sshd_oobm_config

```
1 # $OpenBSD: sshd_config,v 1.80 2008/07/02 02:24:18 djm Exp $
2
3 # This is the sshd server OOBM configuration file.
4
5 #####
6 # CONNECTIVITY #
7 #####
8 Port 1024
9 #AddressFamily any
10 #ListenAddress 0.0.0.0
11 #ListenAddress ::
12
13 # The default requires explicit activation of protocol 1
14 Protocol 2
15
16 # HostKey for protocol version 1
17 #HostKey /etc/ssh/ssh_host_key
18 # HostKeys for protocol version 2
19 #HostKey /etc/ssh/ssh_host_rsa_key
20 #HostKey /etc/ssh/ssh_host_dsa_key
21 #HostKey /etc/ssh/ssh_host_ecdsa_key
22 #HostKey /etc/ssh/ssh_host_ed25519_key
23
24 # Lifetime and size of ephemeral version 1 server key
25 #KeyRegenerationInterval 1h
26 #ServerKeyBits 1024
27
28 # Ciphers and keying
29 #RekeyLimit default none
30
31 #####
32 # LOGGING #
33 #####
34 SyslogFacility AUTH
35 LogLevel INFO
36
37 #####
38 # AUTHENTICATION #
39 #####
```

```
40 AllowGroups oobm
41 ForceCommand /etc/ssh/sshd_oobm_cmd
42
43 #LoginGraceTime 2m
44 #PermitRootLogin yes
45 #StrictModes yes
46 #MaxAuthTries 6
47 #MaxSessions 10
48
49 #RSAAuthentication yes
50 #PubkeyAuthentication yes
51
52 # The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
53 # but this is overridden so installations will only check .ssh/
    authorized_keys
54 AuthorizedKeysFile .ssh/authorized_keys
55
56 #AuthorizedPrincipalsFile none
57
58 #AuthorizedKeysCommand none
59 #AuthorizedKeysCommandUser nobody
60
61 # For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
62 #RhostsRSAAuthentication no
63 # similar for protocol version 2
64 #HostbasedAuthentication no
65 # Change to yes if you don't trust ~/.ssh/known_hosts for
66 # RhostsRSAAuthentication and HostbasedAuthentication
67 #IgnoreUserKnownHosts no
68 # Don't read the user's ~/.rhosts and ~/.shosts files
69 #IgnoreRhosts yes
70
71 # To disable tunneled clear text passwords, change to no here!
72 #PasswordAuthentication yes
73 #PermitEmptyPasswords no
74
75 # Change to no to disable s/key passwords
76 ChallengeResponseAuthentication no
77
78 # Kerberos options
79 #KerberosAuthentication no
```

```
80 #KerberosOrLocalPasswd yes
81 #KerberosTicketCleanup yes
82 #KerberosGetAFSToken no
83
84 # GSSAPI options
85 #GSSAPIAuthentication no
86 #GSSAPICleanupCredentials yes
87
88 # Set this to 'yes' to enable PAM authentication, account processing,
89 # and session processing. If this is enabled, PAM authentication will
90 # be allowed through the ChallengeResponseAuthentication and
91 # PasswordAuthentication. Depending on your PAM configuration,
92 # PAM authentication via ChallengeResponseAuthentication may bypass
93 # the setting of "PermitRootLogin without-password".
94 # If you just want the PAM account and session checks to run without
95 # PAM authentication, then enable this but set PasswordAuthentication
96 # and ChallengeResponseAuthentication to 'no'.
97 UsePAM yes
98
99 #AllowAgentForwarding yes
100 #AllowTcpForwarding yes
101 #GatewayPorts no
102 X11Forwarding yes
103 #X11DisplayOffset 10
104 #X11UseLocalhost yes
105 #PermitTTY yes
106 #PrintMotd yes
107 #PrintLastLog yes
108 #TCPKeepAlive yes
109 #UseLogin no
110 UsePrivilegeSeparation sandbox # Default for new installations.
111 #PermitUserEnvironment no
112 Compression no
113 ClientAliveInterval 15
114 ClientAliveCountMax 4
115 #UseDNS yes
116 #PidFile /var/run/sshd.pid
117 #MaxStartups 10:30:100
118 #PermitTunnel no
119 #ChrootDirectory none
120 #VersionAddendum none
```

```
121
122 # no default banner path
123 Banner /etc/ssh/ssh_oobm_banner
124
125 # override default of no subsystems
126 Subsystem sftp /usr/libexec/sftp-server
127
128 # Example of overriding settings on a per-user basis
129 #Match User anoncvs
130 # X11Forwarding no
131 # AllowTcpForwarding no
132 # PermitTTY no
133 # ForceCommand cvs server
```