COIT13146 System and Network Administration Project

**Term 1 2025**

**Authors**

Ben Hopkins, ID 12196328

Tim Ho, ID 12241028

**Unit Coordinator**

Zhenglin Wang

09/06/2025 

Contents

[Introduction 2](#_Toc1935039033)

[Network and Server Design 2](#_Toc762230097)

[Network Diagram 3](#_Toc745561493)

[Table of Servers, MAC addresses, Allocated IP Addresses, Client IP Address Ranges. 4](#_Toc1724238915)

[Server Installation and Configuration Details 4](#_Toc1732459990)

[Rocky 4](#_Toc1706251163)

[Sydney 5](#_Toc312537728)

[Adelaide 6](#_Toc1822415694)

[Bundaberg 7](#_Toc1160244461)

[Gladstone 8](#_Toc1091679959)

[Darwin 12](#_Toc1348337271)

[Justification of the network design and assumptions 12](#_Toc325752665)

[1. Lack of experienced System / Network Administrators 12](#_Toc384363557)

[2. Ubuntu 24.04 LTS (Desktop) 12](#_Toc294638284)

[3. No enterprise-grade firewall appliance or VLAN segmentation yet 13](#_Toc1198871384)

[4. Internal-only Dokuwiki 13](#_Toc263292261)

[5. Sydney as a purpose-built SSH Jump Box 13](#_Toc1819591628)

[6. Local system accounts instead of central identity (LDAP/Azure Entra) 13](#_Toc267003229)

[Backup and Recovery Design 13](#_Toc905379636)

[Instructions for a staff member to perform a recovery of the web server adelaide. 15](#_Toc279690318)

[Security Design 15](#_Toc61491091)

[Details of a password policy and its implementation. 16](#_Toc76926057)

[Implementation 16](#_Toc958127560)

[Instructions for staff to systems. 17](#_Toc2024357043)

[Digital Certificate Renewal Instructions 17](#_Toc1381214523)

# Introduction

This report presents the design, implementation, and testing of a small business network for the software company cqunix. The company is transitioning from cloud-based services to on-premises infrastructure. This new infrastructure needs to be configured in a secure and reliable way.

The on-premises network is delivered using six key systems:

1. adelaide (web server)
2. sydney (SSH server)
3. gladstone (Git server)
4. bundaberg (backup server)
5. darwin (DHCP server)
6. rocky (gateway/firewall

Each server fulfills a distinct role and is integrated into the cohesive network, supported by secure access controls, automated backups, and documentation hosted on an internal docuwiki server.

This report is divided into three main sections:

* Network and Server Design, which outlines the architecture and technical setup of the internal systems
* Backup and Recovery Design, which details the strategy for preserving and restoring critical data
* Security Design, which explains the access policies, firewall rules, and account management practices.

Each section includes system details, implementation steps, and relevant testing to demonstrate that the design meets the requirements.

# Network and Server Design

## Network Diagram

Figure 1: Network Diagram

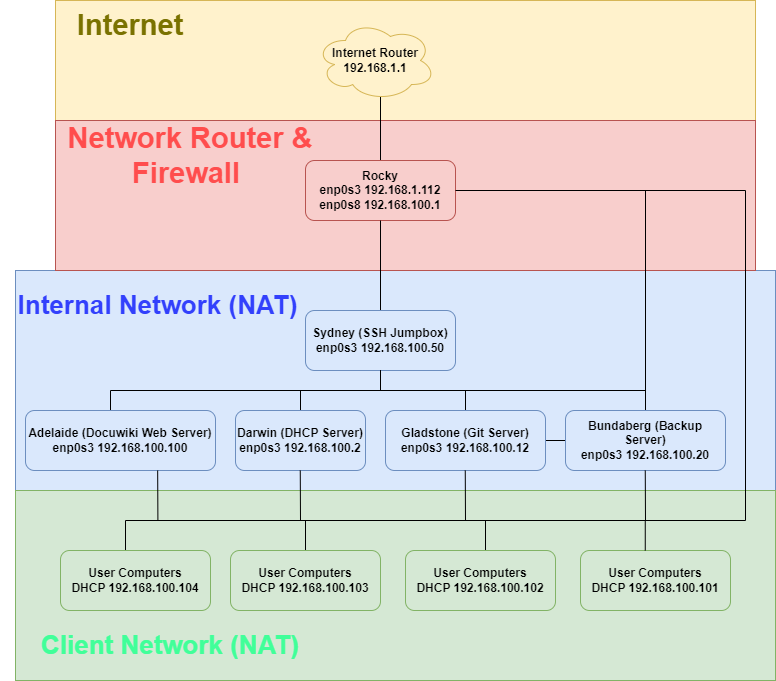


Figure 1 demonstrates the full network layout. We can use it to see a visual representation of how devices connect to each other and the addresses they use.

End user computers such as developer laptops can connect to all servers and are assigned an IP address in the range of 192.168.100.101 to 192.168.100.251 by the DHCP server Darwin. The diagram shows an example of the addresses four different user's computers might obtain. All systems are on the 192.168.100.0/24 network for ease of administration for a small company such as cqunix.

All systems except for Rocky and Darwin (The DHCP server itself) use addresses obtained from the DHCP server. All other Server IP addresses are reserved based on MAC address.

The only gateway for the network is the server Rocky. This ensures Rocky can monitor and filter network traffic as required. This is the only server with direct access to the internet, all other servers use Rocky’s internet connection via Network Address Translation (NAT).

## Table of Servers, MAC addresses, Allocated IP Addresses, Client IP Address Ranges.

The following tables detail the network configuration. Figure 2 shows how the network addresses are assigned and the reserved IP addresses. Reservations are managed through MAC addresses.

Figure 2: Network Address Assignment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **System Name** | **Reserved IP** | **Purpose** | **MAC** | **DHCP Address** |
| adelaide | 192.168.100.100 | Web server (Dokuwiki) | 08:00:27:53:35:2a | Y |
| sydney | 192.168.100.50 | SSH server | 08:00:27:41:68:d1 | Y |
| bundaberg | 192.168.100.20 | Backup server | 08:00:27:2a:3e:d0 | Y |
| gladstone | 192.168.100.12 | Git server | 08:00:27:28:b9:7c | Y |
| darwin | 192.168.100.2 | DHCP server | 08:00:27:e5:c2:4e | N |
| rocky | 192.168.100.1 | Router/firewall/gateway | 08:00:27:26:db:93 | N |
| User Computers | None | Company Laptops and PCs | Variable | Y |

Figure 3 shows the specific network details that the DHCP server is sending out when a client requests an IP. This includes the gateway which ensures that all internet connections go through the Router / Firewall server rocky.

Figure 3: DHCP Range and settings

|  |  |
| --- | --- |
| **DHCP Range** | 192.168.100.101-192.168.100.251 |
| **Gateway** | 192.168.100.1 |
| **DNS 1** | 8.8.8.8 |
| **DNS 2** | 8.8.4.4 |
| **Broadcast Address** | 192.168.100.255 |
| **Subnet Mask** | 255.255.255.0 |

## Server Installation and Configuration Details

### Rocky

Rocky functions as the network's gateway router and firewall. It uses iptables to implement NAT for outbound traffic and to enforce security rules that restrict or permit incoming connections as required. Rocky also forwards DNS and internet traffic, ensuring only authorised services such as SSH, are reachable from external networks.

The firewall is configured by a firewall script. This script removes all other rules and applies the correct rules. This ensures that the firewall is always operating in the expected configuration and can be recovered quickly if the system needs to be rebuilt.

### Sydney

The server “Sydney” provides secure shell access (SSH) for all staff, both locally and remotely. The intention of this server is to function as a “jumpbox” where all connections must go through this server. Once a user is connected to “Sydney”, other servers can then be accessed using SSH from the Sydney server. This enables improved auditing and access control.

Fail2ban is configured to stop brute force login attempts. There are 5 failed login attempts allowed. If the login is failed 5 times then the user will have to wait 900 seconds until they can attempt a login again.

#### Configuration specifics

Below is a script which will create the failed login report and the corresponding cron job to run it every day.

Figure 4: SSH Failed Logins Report configuration

|  |
| --- |
| #!/bin/bash  #create script to build failed login report  echo "build ssh login report script"  cat > /usr/local/bin/ssh-login-summary.sh <<'EOF'  #!/bin/bash  TIMESTAMP=$(date "+%Y-%m-%d %H:%M:%S")  #save report here  LOG\_FILE="/var/log/ssh-login-summary.log"  #ensure the file exists  touch /var/log/ssh-login-summary.log  {  echo "====== SSH Login Summary - $TIMESTAMP ======"  echo "Failed SSH login attempts (past 24 hours):"  journalctl \_COMM=sshd --since "1 day ago" | grep "Failed password" | awk '{print $1, $2, $3, $11, $13}' | sort | uniq -c | sort -nr  echo ""  } >> "$LOG\_FILE"  EOF  #fix permissions on script  chmod +x /usr/local/bin/ssh-login-summary.sh  #auto run report every 24 hours  echo "setting up daily cron job"  (crontab -l 2>/dev/null; echo "0 2 \* \* \* /usr/local/bin/ssh-login-summary.sh") | crontab -  /var/log/ssh-login-summary.log  echo "configuration complete" |

Figure 5 is an example of a report when there has a been a failed login that day

Figure 5: Failed SSH login report

|  |
| --- |
| ====== SSH Login Summary - 2025-06-09 05:43:14 ======  Failed SSH login attempts (past 24 hours):  1 Jun 09 05:41:00 127.0.0.1 47502 |

Below is the configuration file of fail2ban located at “/etc/fail2ban/jail.d/sshd.conf”

Figure 6: fail2ban configuration

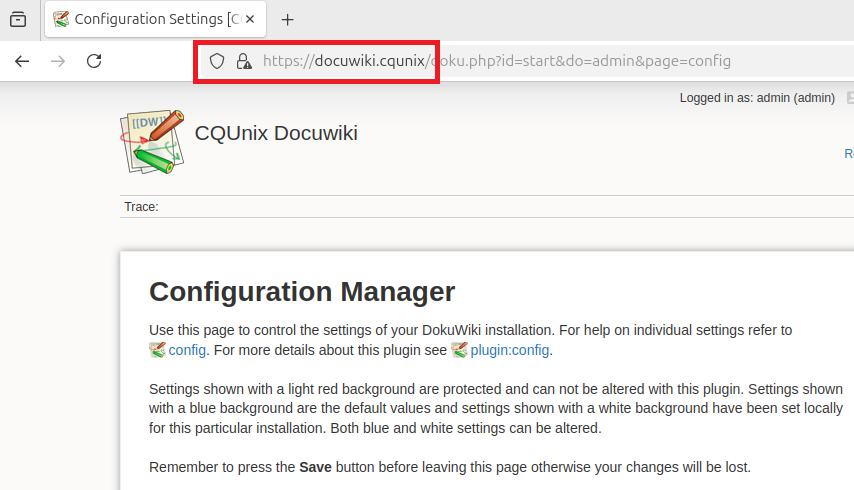
|  |
| --- |
| [sshd]  enabled = true  port = ssh  logpath = %(sshd\_log)s  backend = systemd  maxretry = 5  bantime = 900  findtime = 600 |

### Adelaide

Adelaide is configured as a secure PHP web server which is designed to host the Docuwiki site. The site is hosted using Apache 2. Since the company does not have its own certificate authority or domain name, the site has been configured with a self-signed certificate. This will allow network traffic to remain encrypted.

The pages on the docuwiki are synced to the git server. Only authorised users can edit content, with access control lists (ACLs) applied to all pages.

Figure 7: HTTPS functional on docuwiki site



### Bundaberg

The gladstone server was configured with a permanent static IP address of 192.168.100.20 using Netplan to make it able to perform backup administration. It automates backups of the Adelaide, gladstone, and Sydney server using a schedule cron job and rysnc-based script to copy files and configurations. Separate folders for each server were made in the ‘backups’ directory. The backups are saved with the timestamp in their own designated directory and within, the log files and ownership meta data. Keeping each server's own backup separate helps with backup management and restoration. To securely connect to the respective servers for backup, user ‘a’ SSH key was added to those servers to allow Bundaberg to pull backups without needing to store any passwords. This helps with password management and security. A cron tab was created to schedule the automation of the backup of the servers at different time daily.

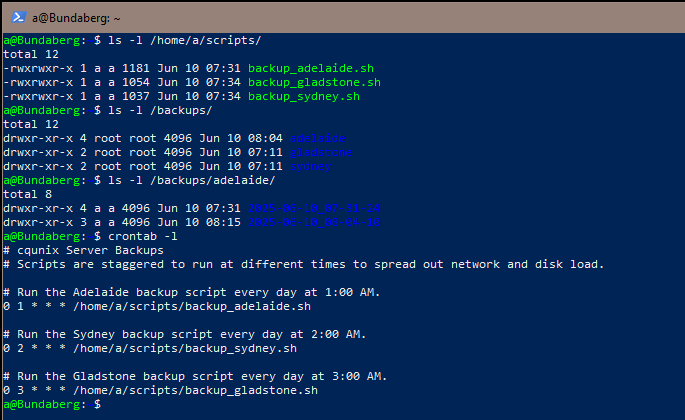
Figure 8: Bundaberg relevant backup configuration  


Figure 9: Enabling cron  
A computer screen with text on it

AI-generated content may be incorrect.

### Gladstone

The gladstone server was configured with a permanent static IP address of 192.168.100.12 using Netplan to make it easy for Git upload and sync. An internal web interface Git server was self-hosted on this server as an internal backup location for the ‘[**dokuwiki-pages**](http://localhost:3000/tim/dokuwiki-pages)’ hosted on the Adelaide server. This is the primary backup that syncs every change on the Adelaide server (Dokuwiki), and commits the changes to the gitea repository. This method provides a granular backup mechanism that protects against accidental modification made to the Dokuwiki server without having to restore the entire server from the Bundaberg backup server.

Internal dokuwiki backup (private github repo called ‘[cqunix-dokuwiki-backup](https://github.com.mcas.ms/TimPortFoliHo/cqunix-dokuwiki-backup?McasTsid=11760&McasCtx=4)’ ) configuration

|  |
| --- |
| Step 1: install git packages and the Gitea application and make it executable, moved it to the system path  ```bash sudo apt install git -y wget -O gitea <https://dl.gitea.io/gitea/1.22.1/gitea-1.22.1-linux-amd64> chmod +x gitea  sudo mv gitea /usr/local/bin/gitea sudo mkdir -p /var/lib/gitea/{custom,data,log} sudo mkdir /etc/gitea sudo chmod 770 /etc/gitea ```  step 2: Allow ‘systemd’ service file to manage Gitea application and restart to enable Gitea to start on boot  ```bash sudo nano /etc/systemd/system/gitea.service sudo systemctl daemon-reload  sudo systemctl start gitea  sudo systemctl enable gitea sudo systemctl restart gitea ```  step 3: use SSH tunnel to access web interfact and create the private repository ‘[**dokuwiki-pages**](http://localhost:3000/tim/dokuwiki-pages)’  open <http://localhost:3000> on local host and create gitea ‘[**dokuwiki-pages**](http://localhost:3000/tim/dokuwiki-pages)’ and add users |

#### Figure 10: Gitea example of users created A screenshot of a computer AI-generated content may be incorrect.

An external backup was also created off-site into a private repository on GitHub called ‘[**cqunix-dokuwiki-backup**](https://github.com.mcas.ms/TimPortFoliHo/cqunix-dokuwiki-backup?McasTsid=11760&McasCtx=4)’. This is our off-site disaster recovery strategy that incorporates a cron script on the gladstone server to push changes every hour when commits are made to the private GitHub repository. This method is to have a safety protocol encase the Bundaberg, gladstone, and Adelaide servers ever fail or are destroyed. By creating an external backup strategy that backup every hour, it prevents and limits the damages to our data to within the last hour of it being backed up. Also, the script could be run to manually backup the data giving us more granular control over data recovery.

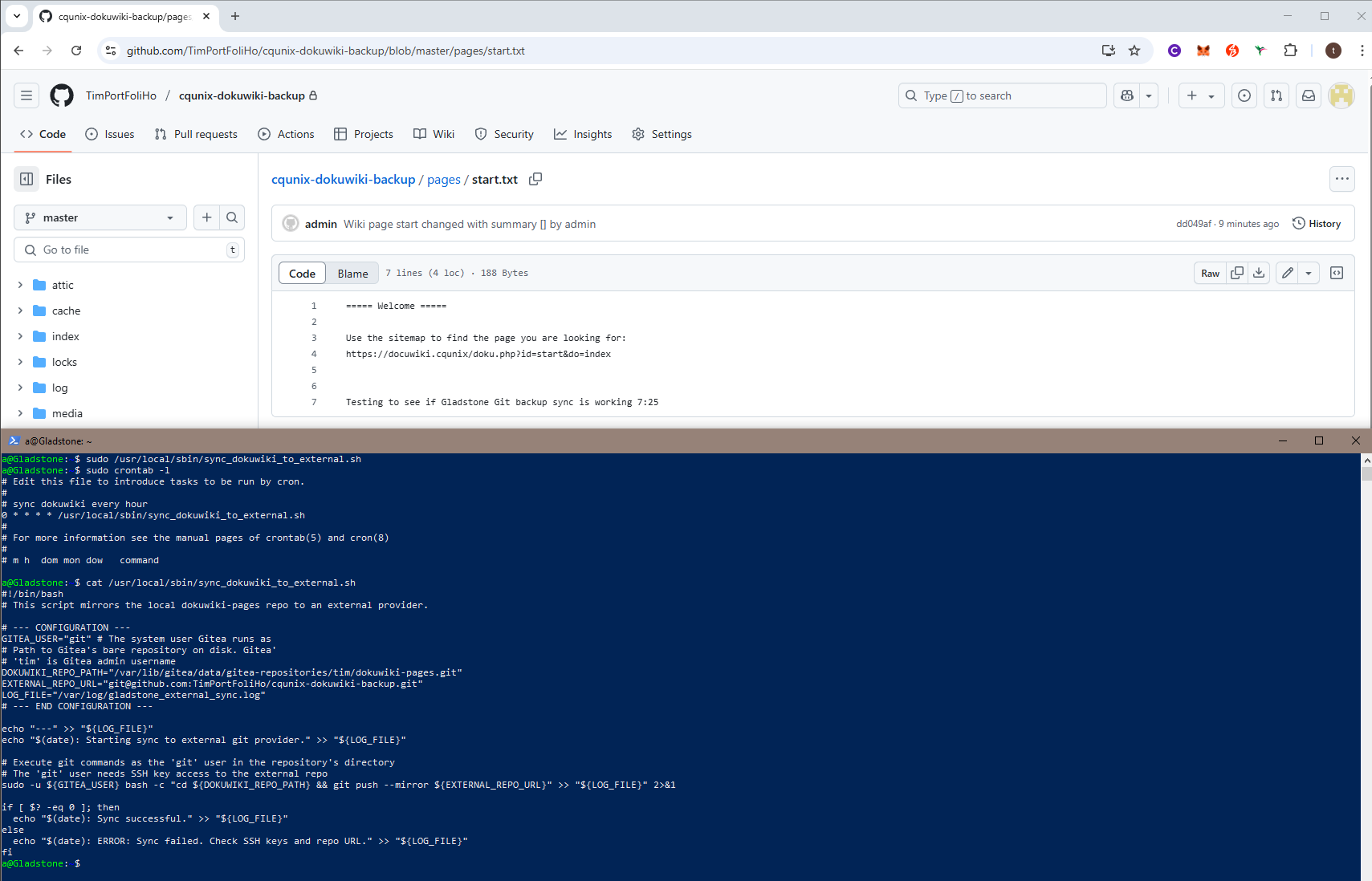
#### External dokuwiki backup configuration and automate backup with cron

|  |
| --- |
| Step 1: create private repository called ‘cqunix-dokuwiki-backup’  Step 2: create sync script ‘/usr/local/sbin/sync\_dokuwiki\_to\_external.sh’ (can be found in zip) and make it executable with sudo chmod +x  Step 3: Create 4096-bit RSA key and add it to GitHub repository for SSH access  ```bash ## generate key  sudo -u git ssh-keygen -t rsa -b 4096 -f /home/git/.ssh/id\_rsa -N ""  ## copy public key and add to github repo sudo cat /home/git/.ssh/id\_rsa.pub ```  Step 4: Edit cron to run every hour  ```bash sudo crontab -e  ## add line and save to config to run every hour 0 \* \* \* \* /usr/local/sbin/sync\_dokuwiki\_to\_external.sh  ``` |

#### Testing internal and external backup functionality of Gladstone server

The functionality of gladstone server was tested by making a change on the Dokuwiki server by adding the comment ‘Testing to see if Gladstone Git backup sync is working at 7:25’ to the welcome page, which appears as a new commit in the Gitea repository, demonstrating the successful internal sync functionality. The external backup was tested by manually running the ‘sync\_dokuwiki\_to\_external.sh’ script where the commit was verified external on the private github repositroy.

#### Figure 11: Internal sync to internal gitea repository A screenshot of a computer AI-generated content may be incorrect.

Figure 12: External sync to private github repository 

### Darwin

Darwin manages IP address assignments for all devices on the internal network. Since this system is using Ubuntu, the ISC DHCP server was installed. It has been configured with a defined dynamic range for workstations and laptops, and static reservations for servers based on MAC addresses. See attached configuration file and “Table of Servers, MAC addresses, Allocated IP Addresses, Client IP Address Ranges” for specific configurations.

# Justification of the network design and assumptions

We designed this network while trying to balance security, usability, and the practical skill level of the cqunix team. We explain our assumption on cqunix and the project requirements which we used to design the network in the following points.

1. Lack of experienced System / Network Administrators

Since this project is being undertaken by external contractors, we are assuming that this network will be maintained by developers with minimal experience in system and network administration. This results in our design being configured to prioritize ease-of-management over more complex and efficient designs.

1. Ubuntu 24.04 LTS (Desktop)

We standardised on the desktop edition rather than the minimal “Server” build because it ships with a broader set of pre-installed packages and familiar GUI administration tools (GNOME, NetworkManager, etc.). For a small team with limited Linux experience, this reduces setup time, avoids repeated installs, and lets staff troubleshoot services locally via graphical diagnostic tools and editors. We assume that the marginally increased disk and RAM overhead on the new hardware is outweighed by the productivity gain.

1. No enterprise-grade firewall appliance or VLAN segmentation yet

While VLANs would offer improved network isolation, we assume the current staff lack the networking knowledge to maintain complex rule bases combined with VLAN tagging schemes. Instead, Rockyhandles NAT and iptables filtering for the network which is contained to one LAN “192.168.100.0/24”. This keeps firewall management simple, while leaving room to upgrade to enterprise hardware later.

1. Internal-only Dokuwiki

Publishing “Adelaide” externally would expose the network to significant risks since the webserver sits inside the network behind the firewall, with access to all servers and computers. If this server were exposed to the internet, a significantly more complex redesign would be necessary to keep this system secure. This server can still be accessed by remote employees by connecting into their end user devices on the internal network, such as VNC over the internet but this was not specified as a requirement, so it has not been configured. A self-signed TLS certificate is accepted company-wide, avoiding certificate authority and domain name costs.

1. Sydney as a purpose-built SSH Jump Box

All employee and contractor accounts can reach Sydney (SSH jump box) and from there hop to other internal servers. We assume the intention is to dedicate a server solely to SSH. This confines remote SSH attacks (such as Brute-forcing) and auditing to one system which also simplifies fail2ban tuning.

1. Local system accounts instead of central identity (LDAP/Azure Entra)

Deploying and maintaining an identity provider such as an LDAP directory or cloud directory like Azure Entra ID requires significant effort and generally, a dedicated administrator. We believe that this has been intentional excluded from the requirements due to the disproportionate level of continuous user management effort required for this small startup.

# Backup and Recovery Design

The Bundaberg backup and recovery design encompasses the critical data (3 servers) and a secure approach involving SSH authentication allows Bundaberg to sever to use the pull model to achieve this. The Bundaberg backup scripts was created to be automated with cron jobs to relieve pressure of sysadmins to allow them to utilise their time performing other functions. It also prioritises data integrity. When a backup job is created, a directory with the timestamp of that server to allow point-in time (PIT) recovery. This ensures data integrity, the log files preserve the ownership metadata which can be verified. The backup run daily at different time at off peak hours with adelaide running at 1:00 AM, sydney at 2:00 AM, and gladstone at 3:00 AM. This helps with network overload issues and improves on backup reliability by spreading the backup at staggered time. By backing up critical data daily, it offers a strong recovery point object (RPO) ensuring that no more thant 24 hours of data could be lost. Additionally, by creating a backup script, it could also be run manually when needed to offer more granular control, saving time and effort backing up each server (if a script wasn’t used).

#### Bundaberg Backup technology:

1. It uses rsync to make copies of changed and new files to provide an incremntal backup that perserves file permissions, ownership, and timestamps.
2. The system uses SSH key-based authentication, allowing `bundaberg` to connect to the other servers without needing to store passwords in scripts.
3. The data is tunneled through SSH to enssure that the backup is encrypted and secure.
4. Cron is used to automate the process however, it could also be performed manually if needed.

#### Bundaberg Backup script:

1. `backup\_adelaide.sh`:

Purpose: Backs up the `adelaide` web server.

Data Backed Up:The entire Dokuwiki directory (`/var/www/html/dokuwiki/`) and the Apache configuration directory (`/etc/apache2/`).

1. `backup\_sydney.sh`:

Purpose: Backs up the `sydney` SSH server.

Data Backed Up: Critical security configurations, including the SSH server config (`/etc/ssh/`) and the Fail2ban config (`/etc/fail2ban/`).

1. `backup\_gladstone.sh`:

Purpose: Backs up the `gladstone` Git server.

Data Backed Up: The Gitea application data (`/var/lib/gitea/`) and its configuration (`/etc/gitea/`).

#### Gladstone Backup design:

A 3-2-1 backup rule was used for as part of the gladstone backup design to enable a strong disaster recovery plan for an application with frequent changes. The dokuwiki server contain the live data, the internal gitea server was backed up on the gladstone server and an off-site backup was created on a private github repository. The two different media used was the virtual hard disk of the adelaide and gladstone server and the cloud storage of the github server. The off-site copy being a GitHub finishes the implementation of this strategy.  
The Gladstone server uses a real-time replication with an internal backup server through gitea and incremental external backup server through approach GitHub for the Dokuwiki backup. As there is more frequent data changes happening on the dokuwiki server, it needs a different strategy than Bundaberg. A dokuwiki plugin was enabled on the gitea server which automatically syncs and commits pushes the changes, providing a real-time backup control without having to do a full server restoration. Doing full server restoration would consume more resources and be more inefficent. The second approach is an automated cron job that runs hourly to push changes to an off-site backup git repository. This protects against server issues where data could be restored within the last hours of changes, making it efficient and reliable.

## Instructions for a staff member to perform a recovery of the web server *adelaide*.

1. SSH from ‘Sydney’ into ‘Bundaberg’ and list the content of the backup directory

ls -l /backups/adelaide/

1. Folders have the format of YYYY-MM-DD\_HH-MM-SS. Select the one you want you and view the log

cat /backups/adelaide/2025-06-10\_07-31-24/backup\_log.txt

1. Copy output and save to file as ‘adelaide-backup-output.txt
2. SSH into new adelaide server and use ‘rsync’ to pull backup file to restore dokuwiki file:

sudo rsync -avz -e ssh a@192.168.100.20:/backups/adelaide/2025-06-10\_07-31-24/dokuwiki/ /var/www/html/dokuwiki/

1. Restore Apache configuration:

sudo rsync -avz -e ssh a@192.168.100.20:/backups/adelaide/2025-06-10\_07-31-24/apache2/ /etc/apache2/

1. Stay ‘adelaide’ server and set correct permission

sudo chown -R www-data:www-data /var/www/html/dokuwiki/   
sudo chmod -R g+w /var/www/html/dokuwiki/

1. Enable and restart apache site configuration to get it up and running again

sudo a2ensite dokuwiki.conf   
sudo a2enmod ssl   
sudo a2enmod rewrite  
sudo apache2ctl configtest  
sudo systemctl restart apache2

1. Open web

# Security Design

## Details of a password policy and its implementation.

This policy helps enforce good security hygiene, particularly in a multi-user development environment. It is based on the recommendations in the ACSC Internet Security Manual.

**Secure Password Policy**

* Minimum password length: 12 characters
* Password must not be based on the username or be found in a dictionary.
* Passwords expire every 90 days
* Prevent re-use of the last 20 passwords
* Lock account for 15 minutes after 5 failed login attempts
* Automatically unlock after lockout period

### Implementation

This section is a step by step on how to implement this policy. It can be used to identify where to make changes to the password policies if a change is required later. It is recommended that these changes are pushed out to all systems as a script since there is no central user directory controlling user authentication.

1. Install PAM modules

sudo apt install libpam-pwquality

1. Configure password complexity in /etc/security/pwquality.conf:

minlen = 12

dcredit = -1

ucredit = -1

lcredit = -1

ocredit = -1

reject\_username = 1

1. Configure password aging in /etc/login.defs:

PASS\_MAX\_DAYS 90

PASS\_MIN\_DAYS 1

PASS\_WARN\_AGE 7

1. Apply password history and lockout rules in /etc/pam.d/common-password:

password requisite pam\_pwquality.so retry=3

#enforce password history and store hashes

password required pam\_unix.so use\_authtok remember=20 sha512

1. Configure account lockout in /etc/pam.d/common-auth

auth required pam\_tally2.so deny=5 unlock\_time=900 onerr=fail audit even\_deny\_root

## Instructions for staff to access systems using SSH.

This page contains instructions to access the relevant systems. The server details are as below:  
  
192.168.100.50 - Sydney server

192.168.100.12 - Gladstone server

192.168.100.100 - Adelaide

192.168.100.2 - Darwin

192.168.100.20 - Bundaberg

Step 1: users are to SSH into the Sydney server with their username SSH using [<username>@192.168.100.50](mailto:<username>@192.168.100.50)

Step 2: users are to SSH into the other servers they need access to SSH using [<username>@xxx.xxx.xxx.xx](mailto:<username>@xxx.xxx.xxx.xx)

## Digital Certificate Renewal Instructions

Overview  
 To maintain secure HTTPS access to the internal DokuWiki service, a self-signed SSL certificate is used. This certificate expires every 365 days and must be manually renewed.

It is recommended that the people responsible for this task set calendar reminders.

### Step-by-Step Renewal Instructions

1. SSH into Adelaide  
    Use an SSH client (e.g. PuTTY or terminal) to access the server:  
    ssh <username>@adelaide
2. Generate a New Self-Signed Certificate using this command:  
    sudo openssl req -new -newkey rsa:2048 -days 365 -nodes -x509 -keyout /etc/ssl/private/docuwiki.key -out /etc/ssl/certs/docuwiki.crt

*Note: Follow the prompts to enter details such as Country, State, and Organization.*

1. Restart Apache web server to apply changes:  
    sudo systemctl restart apache2
2. Confirm Certificate Renewal

Go to the docuwiki site and check the certificates expiery to confirm the new one is applied.