3: Infrastructure Prerequisites

This chapter covers the essential infrastructure components that must be in place before deploying applications with Infinity Tools. The system includes automated readiness checks, but understanding the underlying architecture is crucial for troubleshooting and optimization.

Prerequisites Overview

Infinity Tools requires the following infrastructure components, which it handles automatically:

- **Docker Engine** - Container runtime and orchestration
- **Docker Compose** - Multi-container application definitions
- **Docker Network** - Service discovery and communication
- **System Dependencies** - GUM, Dialog, curl/wget
- **Resource Requirements** - Disk space, memory, network

Automated Readiness Check

Infinity Tools includes an automated readiness checker that validates and installs prerequisites:

```
sudo infinity-tools
```

The readiness check performs the following operations in sequence:

1. System Requirements Validation

```
# Check root privileges
[ "$EUID" -eq 0 ]

# Verify disk space (minimum 1GB)
df / | awk 'NR==2 {print $4}'

# Check for download tools
command -v curl || command -v wget

# WSL2 detection and configuration
```

```
grep -qi microsoft /proc/version
```

2. GUM Installation

GUM provides the modern terminal UI. Installation process:

```
# Architecture detection
arch=$(uname -m)
case $arch in
    x86_64) arch="x86_64" ;;
    aarch64|arm64) arch="arm64" ;;
    armv7l) arch="armv7" ;;
esac

# Download and install binary
gum_version="v0.13.0"
download_url="https://github.com/charmbracelet/gum/releases/download/${gum_version}/gum_${gum_
version#v}_Linux_${arch}.tar.gz"
```

3. Docker Installation and Configuration

Docker installation is handled by the `install-docker.sh` script:

```
# Check existing installation
command -v docker

# Verify Docker service status
systemctl is-active docker

# Check Docker Compose availability
docker compose version || docker-compose --version
```

4. Docker Network Setup

Creates the default "proxy" network for service communication:

```
# Check existing networks
docker network ls --format '{{.Name}}' | grep -v -E '^(bridge|host|none)$'
```

```
# Create network if needed
docker network create proxy

# Store network configuration
echo "DOCKER_NETWORK=proxy" > /tmp/infinity-tools-network.conf
```

Docker Architecture

Container Runtime

Infinity Tools uses Docker Engine with the following configuration:

- **Storage Driver:** Overlay2 (default)
- **Network Driver:** Bridge networks
- **Logging Driver:** JSON-file
- **Restart Policy:** unless-stopped

Docker Compose Integration

All services are defined using Docker Compose v2 (plugin) format:

```
version: '3.8'

services:
  service-name:
    image: image:tag
    container_name: service-name
    restart: unless-stopped
    networks:
      - proxy
    volumes:
      - /opt/speedbits/service:/data
    environment:
      - KEY=value
```

Network Architecture

Default Network Configuration

The "proxy" network provides:

- **Service Discovery:** Containers can reach each other by name
- **Isolation:** Services are isolated from host network
- **Traefik Integration:** Enables automatic reverse proxy configuration
- **DNS Resolution:** Built-in container name resolution

Network Topology

```
Internet → Traefik (proxy network) → Application Containers
                                  ↓
                      Database Containers (borgmatic-db network)
```

System Dependencies

Required Packages

The readiness check installs the following packages:

```
# Core utilities
curl wget git jq

# UI components
gum dialog

# Optional tools
rclone
```

Package Manager Support

Infinity Tools supports multiple package managers:

- **APT** (Debian/Ubuntu)
- **DNF** (Fedora/RHEL)
- **YUM** (CentOS/RHEL)
- **PACMAN** (Arch Linux)

- **APK** (Alpine Linux)

Resource Requirements

Minimum Specifications

- **CPU:** 1 core (2+ recommended)
- **RAM:** 2GB (4GB+ recommended)
- **Storage:** 20GB SSD (50GB+ recommended)
- **Network:** Stable internet connection

Storage Considerations

Infinity Tools uses the following storage structure:

```
/opt/speedbits/
├── _configuration/          # Global configuration
├── traefik/                 # Traefik data and certificates
├── wordpress/               # WordPress instances
├── vaultwarden/             # Vaultwarden data
└── ...                      # Other service data

/var/lib/docker/             # Docker system data
├── volumes/                 # Named volumes
├── networks/                # Network configurations
└── containers/              # Container data
```

Security Considerations

Docker Security

- **Non-root containers:** Services run with PUID/PGID 1000
- **Read-only filesystems:** Where applicable
- **No new privileges:** Security opt applied
- **Network isolation:** Services on isolated networks

File Permissions

```
# Check script permissions
find /opt/InfinityTools -name "*.sh" -exec ls -la {} \;

# Verify ownership
ls -la /opt/InfinityTools/

# Check for world-writable files
find /opt/InfinityTools -perm -002 -type f
```

Monitoring and Troubleshooting

System Status Commands

```
# Docker status
systemctl status docker
docker info

# Network status
docker network ls
docker network inspect proxy

# Container status
docker ps -a
docker stats

# Logs
journalctl -u docker
docker logs container-name
```

Common Issues

Docker service not starting:

```
sudo systemctl start docker
sudo systemctl enable docker
sudo systemctl status docker
```

Network creation fails:

```
# Check if network already exists
docker network ls | grep proxy

# Remove and recreate if needed
docker network rm proxy
docker network create proxy
```

Permission issues:

```
# Add user to docker group
sudo usermod -aG docker $USER
newgrp docker

# Check group membership
groups $USER
```

Performance Optimization

Docker Configuration

For production environments, consider these optimizations:

```
# /etc/docker/daemon.json
{
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "10m",
    "max-file": "3"
  },
  "storage-driver": "overlay2",
  "storage-opts": [
    "overlay2.override_kernel_check=true"
  ]
}
```

Resource Limits

Set appropriate resource limits for containers:

```
services:
  service-name:
    deploy:
      resources:
        limits:
          memory: 512M
          cpus: '0.5'
        reservations:
          memory: 256M
          cpus: '0.25'
```

Backup Considerations

Docker Data Backup

Important data locations for backup:

- **Application Data:** `/opt/speedbits/`
- **Docker Volumes:** `/var/lib/docker/volumes/`
- **Configuration:** `/opt/InfinityTools/`
- **SSL Certificates:** `/opt/speedbits/traefik/letsencrypt/`

Next Steps

With infrastructure prerequisites satisfied, you're ready to deploy Traefik - the reverse proxy that provides SSL termination and routing for all services.

Verification Checklist

- □ Docker Engine running and accessible
- □ Docker Compose available
- □ Proxy network created
- □ GUM installed for UI
- □ System resources adequate
- □ Network connectivity verified

Next: Installing Traefik - Reverse Proxy Configuration (Chapter 4)

Revision #5
Created 29 October 2025 16:57:34 by bjoern

Updated 31 October 2025 13:08:32 by bjoern