NETWORK MANAGEMENT SYSTEM

Software Requirements Specification

Version 1.0

Project Name	Network Management System (NMS)
Document Version	1.0
Date	February 2026
Technology Stack	Go (Golang)

Daftar Isi

1. Pendahuluan

1.1 Tujuan Dokumen

Dokumen ini menjelaskan spesifikasi kebutuhan perangkat lunak untuk Network Management System (NMS) yang dirancang untuk mengelola ribuan perangkat jaringan secara scalable dan efisien.

1.2 Ruang Lingkup Sistem

NMS adalah sistem manajemen jaringan yang dapat:

- Mengumpulkan data dari perangkat jaringan secara periodik
- Mendukung berbagai protokol: Mikrotik API, SSH, Telnet, TR-069
- Mengelola Router, Switch, OLT, ONT, Access Point, dan perangkat wireless
- Melakukan konfigurasi perangkat secara remote
- Menganalisis dan mengolah data monitoring
- Menskalakan hingga ribuan perangkat

1.3 Target Pengguna

- Network Administrator
- NOC (Network Operations Center) Team
- System Administrator
- IT Manager / Supervisor

2. Deskripsi Umum Sistem

2.1 Perspektif Produk

NMS adalah sistem standalone yang dapat diintegrasikan dengan sistem monitoring eksternal. Sistem ini dirancang dengan arsitektur microservices untuk kemudahan scaling dan maintenance.

2.2 Fungsi Utama Produk

Device Discovery & Inventory Management
Sistem dapat secara otomatis menemukan dan mencatat perangkat jaringan dalam database inventory.

Data Collection
Mengumpulkan metrics seperti CPU usage, memory usage, bandwidth, interface status, temperature, dan custom metrics dari berbagai jenis perangkat.

Configuration Management
Melakukan konfigurasi perangkat, backup konfigurasi, rollback, dan version control.

Monitoring & Alerting
Real-time monitoring dengan threshold-based alerting melalui berbagai channel (email, telegram, webhook).

Reporting & Analytics
Dashboard visualisasi data, historical reports, dan trend analysis.

3. Kebutuhan Fungsional

3.1 Device Management

FR-001: Device Registration

- User dapat menambahkan perangkat dengan memasukkan IP, credentials, dan device type
- Sistem melakukan validasi koneksi sebelum menyimpan
- Support bulk import via CSV/Excel

FR-002: Device Discovery

- Sistem dapat melakukan network scanning untuk menemukan perangkat baru
- Auto-detection device type berdasarkan fingerprinting
- Suggest credentials dari template yang tersimpan

FR-003: Device Grouping

- User dapat membuat group/kategori perangkat (by location, by type, custom tags)
- Hierarchical grouping support (parent-child relationship)

3.2 Data Collection

FR-004: Polling Mechanism

- Configurable polling interval per device atau per device type
- Intelligent polling: prioritas tinggi untuk critical devices
- Automatic retry mechanism dengan exponential backoff

FR-005: Protocol Support

- Mikrotik API: RouterOS devices
- SSH: Command execution dan data parsing
- Telnet: Legacy device support
- TR-069/CWMP: ONT management
- SNMP v2c/v3: Universal device support

FR-006: Metrics Collection

Data yang dikumpulkan meliputi:

- System metrics: CPU, Memory, Disk, Uptime
- Interface metrics: Status, Traffic In/Out, Errors, Packet Loss
- Wireless metrics: Signal strength, Connected clients, Channel utilization
- OLT/ONT metrics: Optical power, Distance, Error counters
- Custom metrics: User-defined OID/command output

3.3 Configuration Management

FR-007: Remote Configuration

- Execute commands pada single atau multiple devices
- Configuration templates dengan variable substitution
- Dry-run mode untuk preview changes
- Approval workflow untuk critical changes

FR-008: Config Backup & Restore

- Automatic scheduled backup
- Manual on-demand backup
- Config versioning dengan diff view
- One-click rollback ke previous version

3.4 Monitoring & Alerting

FR-009: Real-time Monitoring

- Live dashboard dengan auto-refresh
- Device status indicator (online/offline/degraded)
- Interactive topology map

FR-010: Alert Rules

- Threshold-based alerts (CPU > 80%, Memory > 90%, etc)
- Device down/up alerts
- Interface state change alerts
- Custom expression-based alerts
- Alert severity levels (critical, warning, info)

FR-011: Notification Channels

- Email notifications
- Telegram bot integration

- Webhook untuk integrasi dengan sistem lain
- SMS gateway integration (optional)

3.5 Reporting

FR-012: Dashboard & Visualization

- Customizable dashboard layout
- Multiple chart types (line, bar, pie, heatmap)
- Time range selection
- Data export (CSV, PDF, Excel)

FR-013: Scheduled Reports

- Daily/weekly/monthly automated reports
- Email delivery dengan attachment
- Custom report templates

4. Kebutuhan Non-Fungsional

4.1 Performance Requirements

- System harus dapat mengelola minimal 5,000 devices secara concurrent
- Polling latency maksimal 5 detik per device pada load normal
- API response time < 200ms untuk 95% requests
- Dashboard load time < 2 detik
- Support 1,000 concurrent web users

4.2 Scalability

- Horizontal scaling capability (add more worker nodes)
- Database sharding support untuk data partitioning
- Message queue untuk async processing
- Microservices dapat di-deploy independently

4.3 Reliability & Availability

- System uptime target: 99.5% (excluding planned maintenance)
- Automatic failover untuk critical services
- Data replication untuk disaster recovery
- Graceful degradation saat partial system failure

4.4 Security

- Role-Based Access Control (RBAC)
- Encrypted storage untuk credentials (AES-256)
- TLS/SSL untuk semua network communication
- Audit logging untuk semua configuration changes
- API authentication dengan JWT tokens
- Rate limiting untuk API endpoints

4.5 Maintainability

- Comprehensive logging dengan structured format
- Health check endpoints untuk monitoring
- Metrics export untuk observability
- Configuration via environment variables dan config files

5. Arsitektur Sistem

5.1 Arsitektur Overview

Sistem menggunakan microservices architecture dengan komponen-komponen berikut:

Component	Deskripsi
API Gateway	Entry point untuk semua HTTP requests. Handle authentication, rate limiting, dan routing.
Device Service	Manage device inventory, credentials, grouping.
Collector Service	Orchestrate data collection dari devices. Manage polling schedules.
Worker Pool	Execute actual polling tasks. Multiple workers untuk parallel processing.
Config Service	Handle device configuration, backup, restore, versioning.
Alert Service	Evaluate alert rules dan trigger notifications.
Notification Service	Send notifications via email, telegram, webhook.
Analytics Service	Process data untuk reporting dan analytics.
Web UI	Frontend dashboard dan user interface.

5.2 Communication Pattern

- Synchronous: REST API untuk user-facing operations
- Asynchronous: Message Queue (NATS/RabbitMQ) untuk background jobs
- Event-driven: Pub/Sub pattern untuk real-time updates

5.3 Data Flow

Polling Flow:

1. Collector Service membaca schedule dari database
2. Mengirim polling task ke Message Queue
3. Worker mengambil task dan connect ke device
4. Worker menyimpan hasil ke Time-Series Database
5. Alert Service evaluate rules dan trigger notifications jika perlu

6. Spesifikasi Teknologi

6.1 Backend Technology Stack

Komponen	Teknologi
Language	Go 1.21+
Web Framework	Gin / Echo / Fiber (pilih salah satu)
Message Queue	NATS / RabbitMQ
Time-Series DB	InfluxDB 2.x / TimescaleDB
Relational DB	PostgreSQL 14+
Cache	Redis 7+
Container	Docker + Docker Compose
Orchestration	Kubernetes (optional untuk production)

6.2 Key Go Libraries

Library	Purpose
github.com/gin-gonic/gin	HTTP web framework
gorm.io/gorm	ORM untuk PostgreSQL
github.com/go-redis/redis	Redis client
github.com/nats-io/nats.go	NATS messaging client
golang.org/x/crypto/ssh	SSH client
github.com/ziutek/telnet	Telnet client
github.com/go-routeros/routeros	Mikrotik RouterOS API client
github.com/gosnmp/gosnmp	SNMP library
github.com/influxdata/influxdb-client-go	InfluxDB client
github.com/golang-jwt/jwt	JWT authentication
go.uber.org/zap	Structured logging
github.com/prometheus/client_golang	Prometheus metrics
github.com/spf13/viper	Configuration management

6.3 Frontend Technology

- Framework: React 18+ / Vue 3
- Charting: Apache ECharts / Chart.js
- Real-time: WebSocket / Server-Sent Events
- UI Library: Ant Design / Material-UI

7. Interface Requirements

7.1 REST API Endpoints

Device Management API:

- GET /api/v1/devices - List all devices
- POST /api/v1/devices - Register new device
- GET /api/v1/devices/:id - Get device details
- PUT /api/v1/devices/:id - Update device
- DELETE /api/v1/devices/:id - Remove device

Metrics API:

- GET /api/v1/metrics/:deviceId - Get device metrics
- GET /api/v1/metrics/:deviceId/interfaces - Interface metrics
- GET /api/v1/metrics/query - Query metrics with filters

Configuration API:

- POST /api/v1/config/execute - Execute commands
- POST /api/v1/config/backup - Trigger backup
- GET /api/v1/config/backups/:deviceId - List backups
- POST /api/v1/config/restore - Restore configuration

7.2 WebSocket Events

- device.status.changed - Device online/offline status
- metrics.updated - Real-time metrics update
- alert.triggered - Alert notification
- config.job.progress - Configuration job progress

8. Database Schema

8.1 PostgreSQL Tables

devices
Menyimpan informasi device inventory.

- id (UUID, PK)
- name (VARCHAR)
- ip_address (INET)
- device_type (ENUM: router, switch, olt, ont, ap)
- protocol (ENUM: mikrotik_api, ssh, telnet, tr069, snmp)
- credentials_id (UUID, FK)
- status (ENUM: online, offline, unknown)
- polling_interval (INTEGER, seconds)
- created_at, updated_at (TIMESTAMP)

device_credentials
Encrypted credentials storage.

- id (UUID, PK)
- username (VARCHAR)
- password_encrypted (TEXT)
- ssh_key_encrypted (TEXT, nullable)

device_groups

- id (UUID, PK)
- name (VARCHAR)
- parent_id (UUID, FK to device_groups, nullable)
- description (TEXT)

alert_rules

- id (UUID, PK)
- name (VARCHAR)
- metric_name (VARCHAR)
- condition (VARCHAR, e.g., 'value > 80')
- severity (ENUM: critical, warning, info)
- enabled (BOOLEAN)

config_backups

- id (UUID, PK)
- device_id (UUID, FK)
- config_content (TEXT)
- version (INTEGER)
- backup_type (ENUM: manual, scheduled, pre_change)

- created_at (TIMESTAMP)

8.2 InfluxDB Measurements

device_system
Tags: device_id, device_name, device_type
Fields: cpu_usage, memory_usage, disk_usage, uptime, temperature

interface_traffic
Tags: device_id, interface_name
Fields: bytes_in, bytes_out, packets_in, packets_out, errors_in, errors_out, status

wireless_metrics
Tags: device_id, interface_name, ssid
Fields: signal_strength, connected_clients, channel_utilization, tx_rate, rx_rate

9. Kesimpulan

Dokumen SRS ini memberikan spesifikasi lengkap untuk Network Management System yang scalable, reliable, dan maintainable. Dengan menggunakan Go sebagai bahasa pemrograman utama dan arsitektur microservices, sistem ini dirancang untuk mengelola ribuan perangkat jaringan secara efisien.

Implementasi bertahap yang disarankan:

6. Phase 1 (Bulan 1-2): Core infrastructure dan basic device management
7. Phase 2 (Bulan 3-4): Data collection untuk semua protokol
8. Phase 3 (Bulan 5-6): Configuration management dan alerting
9. Phase 4 (Bulan 7-8): Dashboard, reporting, dan optimization

--- End of Document ---