Phase 2: Config Validation - Implementation Summary

Overview

Added comprehensive configuration validation with user-friendly error reporting to catch issues before the service attempts to connect.

Files Created

New Source Files

1. **include/ConfigValidator.hpp**
   - Validation error and result structures
   - Static validation methods for MQTT and D-Bus fields
   - Format validation helpers
   - Error message formatters
2. **src/ConfigValidator.cpp**
   - Implementation of all validation logic
   - MQTT broker/port/topic validation
   - D-Bus service/path/interface/member validation
   - Hostname and IP address validation
   - User-friendly error formatting

Modified Source Files

3. **include/Config.hpp**
    - Added `validate()` method
    - Added private validation helper methods
    - Includes ConfigValidator
4. **src/Config.cpp**
    - Implemented comprehensive validation
    - Validates MQTT configuration
    - Validates all mapping entries
    - Checks for duplicate topics
    - Returns structured validation results
5. **src/main.cpp**
    - Calls validation after loading config
    - Prints validation errors with helpful messages
    - Exits with error code if validation fails
    - Shows warnings but continues if valid
6. **CMakeLists.txt**
    - Added ConfigValidator.cpp to build

Test Files

7. **test-configs/README.md**
    - Documents all test cases
    - Shows expected errors for each invalid config
8. **test-validation.sh**
    - Automated test script
    - Creates test configs dynamically
    - Verifies validation catches all error types
    - Color-coded output

Validation Features

MQTT Validation

- ✅ Broker hostname/IP format validation
- ✅ Port range (1-65535)
- ✅ Topic format validation
- ✅ Wildcard restrictions (publish vs subscribe)
- ✅ Authentication completeness (both or neither)

D-Bus Validation

- ✅ Service name format (reverse-DNS: `org.example.Service`)
- ✅ Object path format (starts with `/`, valid characters)
- ✅ Interface name format (reverse-DNS)
- ✅ Signal/method name format (valid D-Bus member names)
- ✅ Bus type (must be `system` or `session`)

Logical Validation

- ✅ Duplicate topic detection in subscriptions
- ✅ Warning for empty mappings
- ✅ Warning for wildcard usage
- ✅ Comprehensive error messages with field names

Error Message Format

```
Configuration validation failed:

  Field 'mqtt.broker': Invalid MQTT broker 'not-valid!'. Must be a valid hostname or IP address
  Field 'mappings.dbus_to_mqtt[0].service': Invalid D-Bus service name 'bad-name'. Must follow reverse-DNS format (e.g., org.example.Service)
  Field 'mappings.dbus_to_mqtt[0].topic': Invalid MQTT topic 'test/#'. Wildcards (+, #) are not allowed in publish topics

Please fix these errors and try again.
See 'man dbus-mqtt-bridge' for configuration examples.
```

Building and Testing

Build

```bash
cd build
cmake ..
make
```

Run Validation Tests

```bash
chmod +x test-validation.sh
./test-validation.sh
```

Expected output shows all tests passing with green checkmarks.

Manual Testing

Test with an invalid config:

```bash
# Create invalid config
cat > test-bad.yaml <<EOF
mqtt:
  broker: "invalid hostname!"
  port: 99999
bus_type: invalid
mappings:
  dbus_to_mqtt:
    - service: "bad-name"
      path: "no-slash"
      interface: "bad.interface!"
      signal: "123invalid"
      topic: "test/#"
  mqtt_to_dbus: []
EOF

# Run with validation
./build/dbus-mqtt-bridge test-bad.yaml
```

Should output multiple clear error messages and exit with code 1.

Integration with Package

The validation runs automatically when the service starts:

1. Service loads config file
2. Validation runs immediately
3. If errors found:
    - Errors printed to stderr
    - Service exits with code 1
    - systemd shows "Failed with result 'exit-code'"
    - journalctl shows the validation errors
4. If warnings only:
    - Warnings printed to stdout
    - Service continues normally

Validation Rules Reference

MQTT Broker Rules

- Must be valid hostname or IP address
- Hostname: alphanumeric, dots, hyphens
- IPv4: 0-255 for each octet
- "localhost" explicitly allowed

MQTT Port Rules

- Must be 1-65535
- Default: 1883

MQTT Topic Rules

- Can't start with `$`
- Valid characters: `a-z A-Z 0-9 / - _ + #`
- **Publish topics**: No wildcards (`+` or `#`)
- **Subscribe topics**: Wildcards allowed
    - `+` matches single level
    - `#` must be at end, alone after `/`

D-Bus Service/Interface Name Rules

- Must contain at least one dot
- Each element: starts with letter or underscore
- Each element: contains only `[a-zA-Z0-9_]`
- No consecutive dots
- Format: `org.example.Service`

D-Bus Object Path Rules

- Must start with `/`
- Root `/` is valid
- Must not end with `/` (except root)
- No consecutive slashes (`//`)
- Each element: only `[a-zA-Z0-9_]`
- Format: `/org/example/Object`

D-Bus Signal/Method Name Rules

- Must start with letter or underscore
- Contains only `[a-zA-Z0-9_]`
- Format: `SignalName` or `MethodName`

Bus Type Rules

- Must be exactly `system` or `session`
- Case-sensitive

Next Steps

Phase 2 is complete! Ready to proceed with:

- **Phase 3**: CLI improvements (-h, --help, -v, --version)
- **Phase 4**: Man page creation
- **Phase 5**: License headers

Or continue with additional config features:

- Config file search path
- Environment variable support
- Default values documentation