u64-remote-cpp — Change Log & Feature Summary

Version: **v1.1.0**
Author: Dr. Eric O. Flores
Repository: https://github.com/drericflores/c64u-cpp_remote

This document details all significant changes and enhancements introduced since the original port (v1.0.0). It is intended to help developers and users understand the new functionality, how to build and use it, and the reasons behind the updates.

Version History

Version	Summary
v1.0.0	Initial C++ port from AllMeatball/u64-remote (Go)
v1.1.0	Major enhancements: mDNS discovery, caching, improved CLI, verbose mode, device list, and build system improvements

Major Enhancements in v1.1.0

1. mDNS Discovery (via Avahi)

Description:
Devices on the local network advertising HTTP services (e.g., C64U REST API) are now detected using *multicast DNS* (mDNS) via the Avahi library.

Benefits:

- Faster device detection vs. brute subnet scans
- No need to guess or hard-code static IP addresses
- Works automatically on most modern LANs

Usage:

```
./u64-remote --discover myprogram.prg
```

Under the Hood:

- New `DiscoveryService` class (in `discovery.h` / `discovery.cpp`)
- Uses Avahi (pkg-config) to find services of type `_http._tcp`

2. Device Cache

Description:
Last selected/used C64U device is now cached on disk for faster reconnections.

Cache Path:

```
~/.config/u64-remote/cache.json
```

Benefits:

- Avoids repeated scanning on subsequent runs
- Auto-uses valid cached device if reachable
- Falls back to discovery if cache is stale

Cached Format:

```
{
  "address": "http://10.0.0.183",
  "hostname": "C64U.local"
}
```

3. CLI Improvements

New Flags Added:

Flag	Purpose
`--discover`	Force mDNS discovery
`--list`	List discovered devices and exit
`--verbose`	Print detailed debug output
`--creds`	Path to credentials JSON
`--address`	Override device address
`--password`	Override password
`-h`, `--help`	Display usage

Example:

```
./u64-remote --verbose --discover mygame.prg
```

4. Fallback Subnet Scanning

If mDNS discovery fails, the tool automatically falls back to the legacy subnet scan (`util::discoverU64(...)`). This ensures compatibility with networks or devices that do not advertise via mDNS.

5. Cache Validation

Before using a cached device:

- The tool performs a quick `/v1/version` request to confirm accessibility
- If unreachable, cache is ignored

6. Verbose Mode

When invoked with `--verbose`, the program prints:

- Discovery activity
- Cache checks
- HTTP upload details
- PRG size and target address

This assists in debugging or scripting workflows.

Build System Enhancements

🛠️ CMake Modernization

- Unified build setup with `CMakeLists.txt`
- Optional warnings turned on (`-Wall -Wextra -Wpedantic`)
- Includes Avahi discovery support
- Fully portable across Linux systems with libcurl & Avahi dev libs

Dependencies:

- `libcurl4-openssl-dev`
- `libavahi-client-dev`
- `libavahi-common-dev`
- `pkg-config`

Install on Pop!_OS / Ubuntu:

```
sudo apt update
sudo apt install \
  build-essential cmake libcurl4-openssl-dev \
  libavahi-client-dev libavahi-common-dev pkg-config
```

Usage Examples

Upload & Run

```
./u64-remote myprogram.prg
```

📡 Discover & Run

```
./u64-remote --discover myprogram.prg
```

📋 List Devices Only

```
./u64-remote --list
```

🔍 Verbose Mode

```
./u64-remote --verbose myprogram.prg
```

Use credentials file

```
./u64-remote --creds ~/json_examples/creds.json myprogram.prg
```

Runtime Behavior

1. Try cached device if present and valid
2. If `--discover` or no valid address:
    1. mDNS discovery
    2. If empty → subnet scan
3. Prompt user for choice if multiple devices found
4. Run PRG upload over HTTP REST
5. Print progress and exit code

Design Philosophy

This version aims to be:

- **Robust:** Works reliably on real networks
- **Convenient:** mDNS + fallback scan cover most deployments
- **Scriptable:** CLI options for automation
- **Extensible:** Modular classes (DiscoveryService, utility functions)

Technical Notes

mDNS Discovery

Implemented via Avahi client APIs. Uses service type `_http._tcp` to detect REST servers. Important to have Avahi daemon running for full local discovery.

Release v1.1.0

Release Script: `release.sh`

```
#!/usr/bin/env bash
VERSION="1.1.0"
REPO="dreric flores/c64u-cpp_remote"

if [[ -n $(git status --porcelain) ]]; then
  echo "Uncommitted changes found. Commit or stash first."
  exit 1
fi

git tag -a "v$VERSION" -m "Release v$VERSION"
git push origin "v$VERSION"

if command -v gh >/dev/null; then
  gh release create "v$VERSION" \
    --title "v$VERSION" \
    --notes "Release v$VERSION"
else
  echo "gh CLI not found — release draft not created."
fi
```

Test Suite (What It Would Be)

A test suite is a collection of automated tests that verify:

- ✔ Utility functions (file read, cache read/write)
- ✔ mDNS discovery callback behavior
- ✔ REST upload with mock server
- ✔ CLI argument parsing
- ✔ Cache validation logic

Recommended frameworks:

- **Catch2**
- **GoogleTest**
- **CMake test integration**

Summary

This release improves *usability*, *performance*, and *network discovery intelligence*, making your C++ remote tool more reliable and easier to use on modern networks. **v1.1.0 marks a major step forward** in both developer and user experience, while preserving the original design and purpose of the upstream Go project.