APPLICATION NOTE

Communications Device

Communications Protocols and Security Options for Campbell Scientific Data Loggers

03/2026

RELIABLE MONITORING SINCE 1974

Campbell SCIENTIFIC®

Table of contents

1. Introduction

Campbell Scientific data loggers support a wide range of communications options—from direct USB connections during setup to IP-based services used for day-to-day data retrieval and remote management. Selecting the right protocol (and configuring it securely) is essential for reliability, interoperability with third-party systems, and protection of data and device access.

This application note provides a practical, public-facing overview of common protocols available on Campbell Scientific data loggers, what they are typically used for, and the key security considerations for each. It also summarizes built-in security controls and configuration practices that help reduce risk, such as disabling unused services, enforcing strong authentication, and using encryption (for example TLS, VPN tunnels, or PakBus® encryption) where applicable.

Use this note as a starting point when designing a telemetry architecture, preparing devices for deployment, or performing a security review of an existing installation.

How to use this note:

- Identify the protocols you need (based on required features and network constraints).
- Disable any services you do not need.
- Prefer encrypted and authenticated variants (HTTPS over HTTP, FTPS/SFTP over FTP, secure MQTT, etc.).
- Restrict access with firewalls/VLANs/VPNs and strong credentials.
- Validate the configuration before deployment using the ***Device Configuration Utility*** (***DevConfig***) Security Check feature. You can download ***Device Configuration Utility*** for free here: DevConfig: Device Configuration Utility. For details on how the Security Check works, see the ***DevConfig*** Help or the relevant section in your data logger manual: Device Configuration Utility Security Check.

> NOTE:
> Links to online data logger manuals in this document point to the online CR1000X series manual. In most cases, the Help content for other data loggers is similar; however, consult the manual for your specific data logger for device-specific details.

2. Protocols

The following sections detail communications protocols available in Campbell Scientific data loggers.

2.1 PakBus

PakBus is Campbell Scientific's proprietary communications protocol designed for reliable data logger communications. The protocol enables communications between data loggers and support software (such as ***LoggerNet*** and ***RTDAQ***), supports data logger-to-data logger networking, and manages data collection, program uploads, settings changes, and system monitoring.

PakBus operates over multiple physical layers including serial connections (RS-232, RS-485), TCP/IP networks, cellular modems, and radio links. The protocol is packet-based and optimized for reliable data transmission over low-bandwidth, high-latency connections common in remote monitoring applications.

Security considerations:

PakBus incorporates multiple layers of security:

- PakBus TCP Password

  Provides authentication for TCP connections using a proprietary CRAM-MD5–like method. While it authenticates the connection, it does not encrypt packet contents. The PakBus TCP Password can be configured in the ***Device Configuration Utility*** (**Settings Editor** tab).

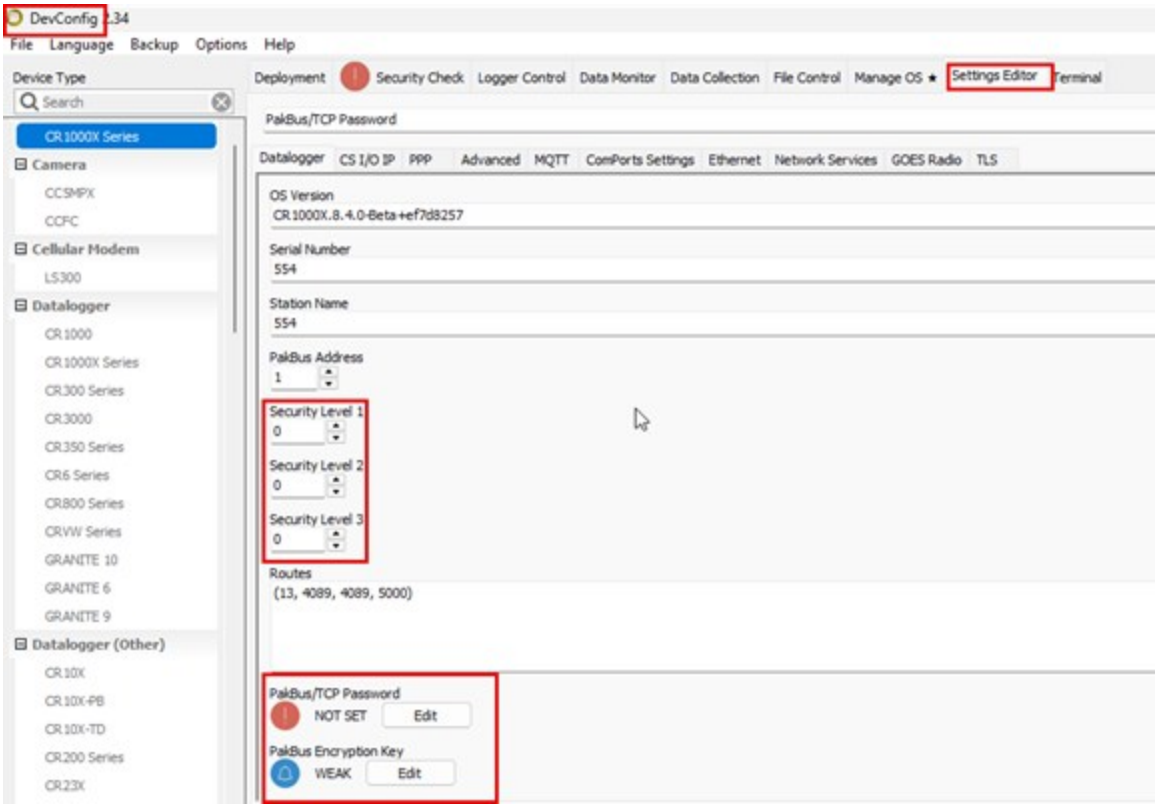
- PakBus Encryption (AES-128)

  Using a PakBus Encryption Key is the most secure option for protecting PakBus communications. For data loggers with a unique identifier (UID), encryption is enabled by default, and the default encryption key is the UID. The PakBus Encryption Key (AES-128 encryption key) can be set or cleared in the ***Device Configuration Utility***.

- Three-Level Security Code System

  An additional security mechanism that controls access to different data logger functions based on user authorization levels. See: How to Use Data Logger Security Codes.

All of these security settings can be configured in the *Device Configuration Utility Settings* Editor tab:



For more information: see the Data logger security section of your data logger manual, and the following resources: How to Keep Your Data Safe with PakBus® Encryption and this video: Unique Identification Number (UID) and New Security Features.

2.2 Transmission Control Protocol (TCP)

TCP (Transmission Control Protocol) is a core Internet protocol that provides reliable, connection-oriented communications over IP networks. Campbell Scientific data loggers extensively use TCP as the transport layer for numerous application protocols, including PakBus/TCP (default port 6785), Modbus TCP (port 502), DNP3 (port 20000), HTTP/HTTPS (ports 80/443), FTP/FTPS (ports 20/21), and custom socket communications.

Custom TCP functionality in data loggers can be implemented through the TCPOpen() CRBasic instruction, which can establish the data logger as either a **TCP client** (initiating connections to remote servers) or a **TCP server** (listening for incoming connections on specified ports).

TCP provides:

- **Connection establishment** via the three-way handshake
- **Guaranteed delivery** through acknowledgments and retransmission
- **Ordered delivery** of packets
- **Flow control** mechanisms to manage sender/receiver rates

Common Campbell Scientific applications include initiating callback connections to ***LoggerNet*** for remote data logger management, establishing Modbus TCP communications with SCADA systems, creating custom data exchange protocols with external systems, and routing PakBus communications across TCP/IP networks.

For more information on IP Networking with Campbell Scientific data loggers, see these videos:

IP Networking with Data Loggers Part 1

IP Networking with Data Loggers Part 2

Security considerations:

TCP itself provides no encryption or authentication—it only ensures reliable transport of data between endpoints. Security depends entirely on the application-layer protocols running over TCP, such as TLS for HTTPS and FTPS, or PakBus encryption for PakBus/TCP communications.

2.3 User Datagram Protocol (UDP)

UDP (User Datagram Protocol) is a connectionless network protocol supported in Campbell Scientific data loggers as part of the IPv4/IPv6 protocol stack. Unlike TCP, which establishes and maintains connections with handshaking and acknowledgment mechanisms, UDP transmits data packets (datagrams) without establishing a formal connection or guaranteeing delivery.

Campbell Scientific data loggers implement UDP functionality through the UDPSocketOpen(), UDPSocketClose(), UDPSocketSend(), UDPSocketRecv(), UDPOpen(), and UDPDataGram() CRBasic instructions. These allow data loggers to send and receive custom messages over UDP sockets. In typical implementations, the data logger acts as the initiator of UDP communications by opening UDP sockets to specified IP addresses and port numbers.

UDP is commonly used for applications where speed is more important than guaranteed delivery, such as real-time data streaming, data logger discovery (as implemented in the ***Device Configuration Utility*** application), and simple request-response messaging systems. Because UDP is connectionless, it introduces less overhead than TCP and can be more efficient for applications that can tolerate occasional packet loss.

Security considerations:

UDP provides no inherent encryption or authentication mechanisms—data is transmitted in plain text unless additional security measures are implemented at the application layer. UDP communications should be restricted to trusted networks through firewall rules and IP filtering. For sensitive applications requiring UDP functionality, additional protections such as application-layer encryption or VPN tunneling should be implemented to protect data in transit.

2.4 Remote Network Driver Interface Specification (RNDIS)

RNDIS (Remote Network Driver Interface Specification) is a Microsoft proprietary protocol that provides virtual Ethernet functionality over USB connections. Newer Campbell Scientific data loggers support RNDIS, enabling TCP/IP communications over the USB port without requiring separate Ethernet hardware. When a data logger is connected to a computer via USB with RNDIS enabled, the data logger appears as a virtual Ethernet adapter with a default IP address (typically 192.168.66.1), allowing users to access the data logger through standard network protocols including PakBus/TCP, HTTP, and other IP-based services. This functionality provides convenient direct access for configuration, programming, and data retrieval during setup and field deployment without requiring separate communications cables or network infrastructure.

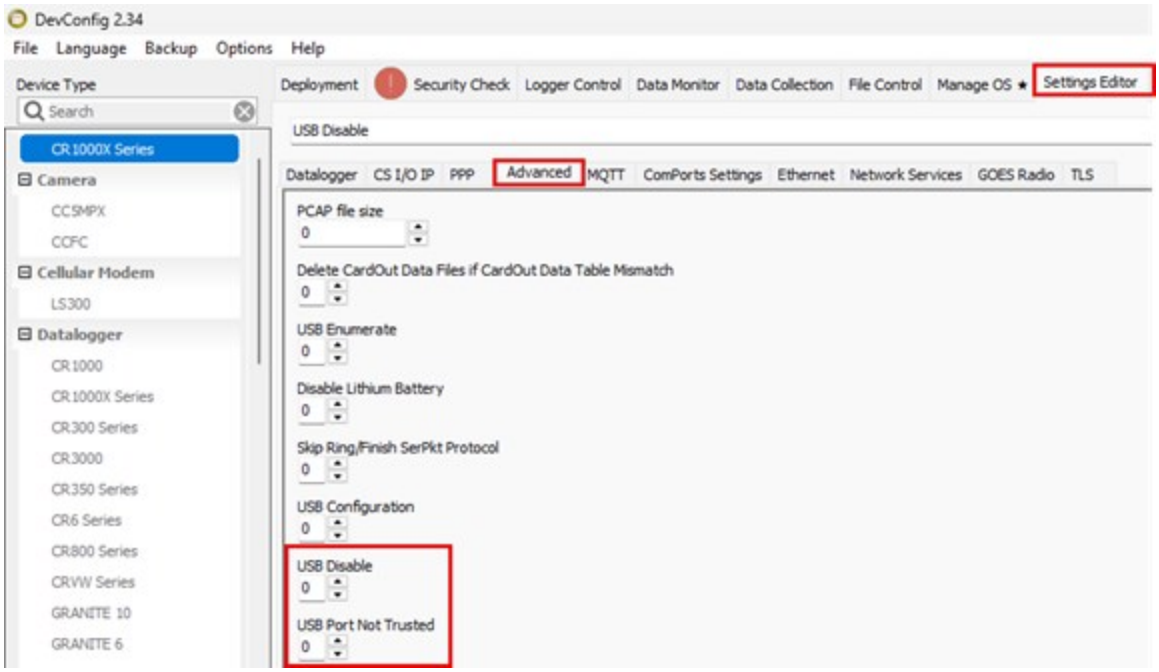
For a video demonstration of RNDIS, see: Ethernet over USB RNDIS.

Security considerations:

New data logger USB security settings: Any device with physical USB access can potentially access the data logger with full privileges. To better control USB-based networking and reduce exposure, newer data loggers provide settings that govern whether USB communications features are available at all and whether an attached host is treated as trusted.

- **USBDisable** — When enabled, the data logger disables its USB interface. This prevents USB-based connectivity—including RNDIS—from being used, which is helpful when USB access is not required or when physical access to the device cannot be controlled.
- **USBNotTrusted** — When enabled, the data logger treats USB-connected hosts as untrusted. This setting can be used to restrict or limit operations initiated over USB (including those enabled by RNDIS), reducing the risk of unauthorized configuration or data access if a laptop or other host device is connected.

These settings are configured in Device Config > Settings Editor > Advanced:



In practice, RNDIS should be enabled only when USB-based IP access is needed, and administrators should consider using **USBDisable** or **USBNotTrusted** (as appropriate for the deployment) to align USB access with site security requirements.

2.5 National Transportation Communication for Intelligent Transportation Systems (NTCIP)

NTCIP (National Transportation Communications for Intelligent Transportation Systems Protocol) is a family of open standards designed to achieve interoperability and interchangeability between computers and electronic traffic control equipment from different manufacturers. Campbell Scientific data loggers support NTCIP primarily for **Road Weather Information Systems (RWIS)** applications, implementing the **NTCIP Environmental Sensor Station (ESS) 1204** and **NTCIP 1201** standards.

When a data logger program invokes the ESSInitialize() instruction, **SNMP services** are enabled on the data logger, allowing it to communicate using the NTCIP-defined **Management Information Base (MIB)** structure. The ESSVariables() instruction declares the variables used by these NTCIP standards, enabling the data logger to report atmospheric weather conditions, road surface conditions, and sensor status to central traffic management systems.

This implementation allows Campbell Scientific data loggers to integrate into existing NTCIP-compliant transportation infrastructure, enabling agencies to deploy environmental monitoring stations that communicate using standardized protocols regardless of manufacturer. However,

NTCIP support in Campbell Scientific data loggers is limited to the specific MIB tables compiled into the operating system for road weather applications and does not provide general-purpose SNMP functionality for arbitrary network management applications.

Security considerations:

NTCIP functionality requires explicit enablement through CRBasic instructions (`ESSInitialize()` and `ESSVariables()`) executed within an active data logger program and is limited to NTCIP ESS applications for road weather reporting.

2.6 File Transfer Protocol (FTP)

FTP (File Transfer Protocol) is a standard network protocol for transferring files between clients and servers over TCP/IP. Campbell Scientific data loggers support FTP in two ways:

- **FTP client** — The data logger can upload files to a remote FTP server. This is typically done using the FTPClient() CRBasic instruction, which allows scheduled, automated transfers of data files.
- **FTP server** — The data logger can host files so a remote system can browse the logger's storage and download (and, if permitted, upload) files.

Security considerations:

Standard FTP sends both login credentials and file data in **plain text**, which makes it inappropriate for use on untrusted networks. When encryption is required, Campbell Scientific recommends using **FTPS (FTP over TLS)** whenever possible, which protects both authentication and data transfers with TLS. If FTP server access is not needed, disable it. If it is needed, limit exposure by using **strong usernames/passwords** and restricting access to **trusted networks**, for example through **IP filtering**. You can also use **SFTP (SSH File Transfer Protocol)** as a secure alternative. SFTP runs over SSH (Secure Shell). To use SFTP with the datalogger, configure an SFTP public key and SFTP private key in the datalogger's advanced settings.

2.7 Hypertext Transfer Protocol (HTTP)

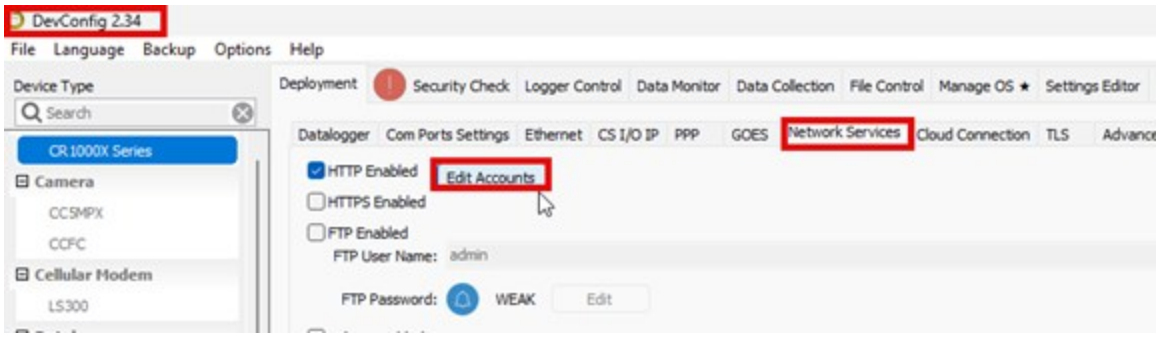
HTTP (Hypertext Transfer Protocol) and **HTTPS (HTTP Secure)** provide web-based access to a data logger and a way for the data logger to interact programmatically with web services. Campbell Scientific data loggers support HTTP/HTTPS in two roles:

- **Server mode** — the data logger hosts web pages and services for browser-based access.
- **Client mode** — the data logger initiates outbound HTTP requests to external web services.

HTTP Server Mode: To use a standard IP connection (not RNDIS) to access the web interface, HTTP or HTTPS access must be enabled. By default, HTTP is enabled and HTTPS is disabled. Additionally, **anonymous** HTTP access is disabled by default. The default HTTP login credentials are **admin** for the username and the data logger UID as the admin password (if the data logger has a UID).

Data loggers can host embedded web servers that provide real-time data viewing, file management, and configuration through standard web browsers. On newer data loggers, user authentication and permissions are managed in ***Device Configuration Utility***.

- Go to **Device Configuration Utility** > **Network Services** > **Edit Accounts** (in older versions, this button is labeled **Edit .csipasswd File**).



- To add permissions, click **Add User**.
- Multiple user accounts can be defined on a single data logger, each with different access levels (for example, read-only access versus administrative access).

Security considerations:

- **HTTP** sends all traffic—including credentials—in **plain text**, which is vulnerable to packet sniffing and man-in-the-middle attacks.
- **HTTPS** uses **TLS (Transport Layer Security)** to encrypt the connection, protecting both authentication and data in transit.

> **NOTE:**
> The currently supported version of TLS is 1.2. TLS 1.0 and 1.1 are obsolete and have been deprecated.

HTTP Client Mode:

Data loggers can also act as HTTP clients using the following CRBasic instructions:

- HTTPGet() — retrieves content from a web server (for example, downloading images from an IP camera, fetching data from a web API, or retrieving a configuration file).

- HTTPPost() — sends data to a web server (for example, posting measurements to a cloud platform or updating a web service).
- HTTPPut() — similar to POST, but with different HTTP semantics used by some APIs.

These instructions support both HTTP and HTTPS. When an **HTTPS URL** is used, **TLS is automatically enabled**.

Common uses include sending measurements to cloud and web platforms (for example, Azure IoT Hub, Weather Underground, or a custom API), retrieving external data via REST APIs, downloading files from network devices, and implementing streaming or integration workflows without relying on FTP or email. HTTP client instructions also support custom headers (for example, Basic authentication, bearer tokens/API keys), efficient connection handling (for example, keep-alive), and direct streaming from data tables (reducing or eliminating intermediate file creation). HTTP client features can be combined with IPRoute() to control which network interface is used and TCPOpen() for more advanced socket-based workflows.

Security considerations:

- **For server mode:** Disable the HTTP/HTTPS server unless it is operationally required. If it is required, use HTTPS only, enforce strong account passwords in DevConfig (Network Services > Edit Accounts), and apply least-privilege access by creating separate user accounts with only the permissions required for each role. Also restrict access to trusted networks using IP filtering and/or network segmentation.
- **For client mode (HTTPGet/HTTPPost/HTTPPut):** Prefer HTTPS URLs for any external service to ensure encryption in transit. Use appropriate authentication (typically via headers rather than URL-embedded credentials), ensure TLS connections use proper certificate validation, and consider application-layer encryption for sensitive payloads. Monitor outbound connections and limit egress to known endpoints using firewall rules or segmentation where feasible.

Default HTTP service port: 80 (configurable)

Default HTTPS service port: 443 (configurable)

2.8 Telnet

Telnet is a network protocol that provides command-line interface access to remote devices over TCP/IP networks. Campbell Scientific data loggers support Telnet connections, allowing users to access the terminal menu for system configuration, diagnostics, and troubleshooting without requiring direct serial connections or specialized software. Through the Telnet interface, users can view system status, modify settings, perform file operations, and execute diagnostic commands.

Security considerations:

Telnet is inherently insecure—it transmits all data, including authentication credentials and commands, in unencrypted plain text over the network. This makes Telnet sessions vulnerable to packet sniffing and man-in-the-middle attacks, where malicious actors can intercept sensitive information or hijack active sessions. While data logger security codes provide some access control for Telnet sessions, they do not encrypt the communication channel itself.

Telnet is **disabled entirely by default**. Only enable Telnet, if terminal access is specifically required for maintenance operations.

Telnet port number: 23 (not changeable in the data logger settings)

2.9 Ping Internet Control Message Protocol (ICMP)

ICMP (Internet Control Message Protocol) is a network-layer protocol used for diagnostic and control purposes in IP networks, most commonly implemented through the ping utility, which tests reachability by sending echo request packets and awaiting echo reply responses.

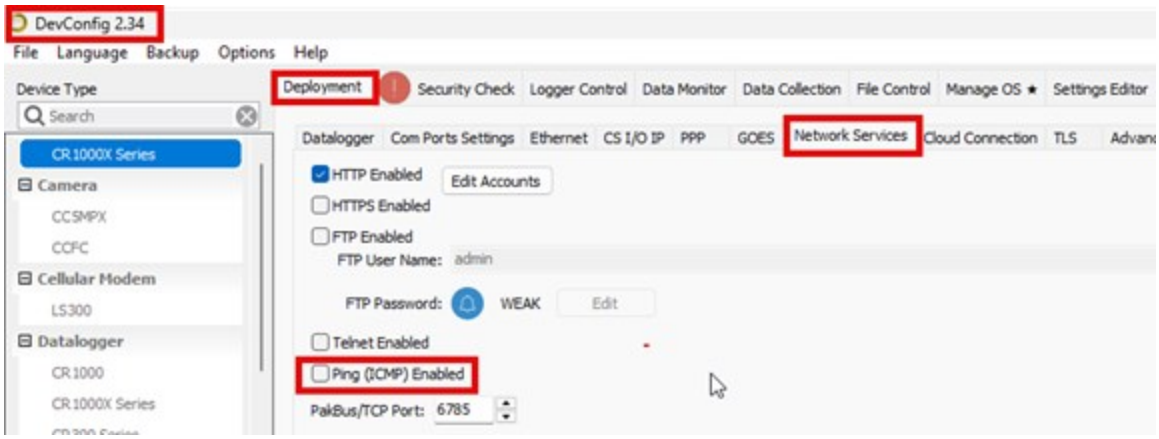
Campbell Scientific data loggers support ICMP in two modes:

- **Client mode** via the PingIP() CRBasic instruction, which sends ICMP echo requests to remote hosts and returns round-trip response times in milliseconds (or **0** for timeouts). This is commonly used in data logger programs to verify network connectivity, monitor link availability, and implement automatic connection-recovery logic (for example, triggering a modem reset when ping responses fail).
- **Server mode** by **responding to incoming ICMP echo requests** from external devices.

The CR6, CR1000X-Series and CR300-Series data loggers additionally provide a **ping** command in terminal mode for interactive troubleshooting.

Security considerations:

Server-side ICMP response capability can be enabled or disabled through ***Device Configuration Utility*** under the **Network Services** tab.



ICMP provides **no authentication or encryption** and, because it operates at the network layer, it has **no concept of ports**—it uses message **types** and **codes** instead.

The primary security concern with ICMP is **network reconnaissance**: responding to ping requests makes data loggers more discoverable, which can help attackers map network topology, identify active devices, and gather information for targeted attacks. Publicly accessible data loggers that respond to pings may also experience increased data costs on metered cellular connections due to unsolicited ping traffic or automated scanning.

Disabling ICMP responses, however, can significantly complicate legitimate network troubleshooting and monitoring, since ping is a fundamental tool for verifying reachability and basic network health.

Recommended practice:

Disable ICMP responses on data loggers exposed to untrusted networks (public internet, shared networks). Enable ICMP only when required for troubleshooting or when the data logger resides on trusted, access-controlled networks protected by firewalls. When ICMP must be enabled, use IP filtering and/or firewall rules to restrict ping access to authorized management networks or specific IP addresses, and monitor for unusual ping traffic patterns that may indicate reconnaissance activity.

For outbound ping functionality using PingIP(), no special security considerations apply because the data logger is initiating the diagnostic test rather than exposing a service.

2.10 ModBus TCP (Modbus over TCP/IP)

Modbus TCP is an industrial communications protocol that adapts the traditional Modbus RTU serial protocol for operation over TCP/IP networks, providing data exchange capabilities between SCADA (Supervisory Control and Data Acquisition) systems and field devices such as programmable logic controllers (PLCs), remote terminal units (RTUs), sensors, and actuators.

Campbell Scientific data loggers implement Modbus TCP through the ModbusServer() and ModbusClient() CRBasic instructions, enabling data loggers to function as either **Modbus servers** (responding to polls from SCADA systems) or **Modbus clients** (initiating requests to other Modbus devices).

When configured as a **Modbus server**, data loggers set the **ComPort** parameter to **502** (or any port **≥ 502**) to listen for incoming TCP connections on the specified IP port, mapping data logger variables to **Modbus holding registers** (address offset **40000**) and **input registers** (address offset **30000**) for access by SCADA systems.

When configured as a **Modbus client**, data loggers use the TCPOpen() function to establish connections to remote Modbus devices on port **502**, with the socket handle passed to the ModbusClient() instruction.

Supported Modbus function codes include **01** (Read Coils), **02** (Read Discrete Inputs), **03** (Read Holding Registers), **04** (Read Input Registers), **05** (Write Single Coil), **06** (Write Single Register), **15** (Write Multiple Coils), and **16** (Write Multiple Registers), providing comprehensive read/write access to device data.

Modbus TCP is widely deployed in water and wastewater systems, electric utility SCADA networks, building automation, manufacturing process control, and remote monitoring applications where integration with existing SCADA infrastructure is required. The protocol operates in a client-server architecture with short-lived request-response transactions, providing simplicity and broad vendor interoperability at the cost of limited advanced features.

Security considerations:

Modbus TCP was designed for reliability and speed in isolated industrial environments, not security, and consequently provides no inherent encryption, authentication, or access control mechanisms. All Modbus communications—including control commands, sensor readings, and device configuration—are transmitted in plain text, making them vulnerable to passive eavesdropping via packet-capture tools (for example, Wireshark). The lack of authentication allows any device with network access to issue commands to Modbus servers without verification, enabling unauthorized control, command injection, man-in-the-middle attacks (intercepting and modifying commands in transit), and denial-of-service attacks through excessive requests or malformed packets.

Port **502** is well known and routinely scanned by attackers seeking exposed industrial control systems, and vulnerabilities in Modbus implementations have been documented (including parsing errors, memory-handling flaws, and session-handling weaknesses). Many embedded devices also lack robust session protections, increasing the risk of **TCP session hijacking** in poorly secured networks.

A "**Modbus Secure**" specification (**Modbus/TCP Security**) released in **2018** adds **TLS encryption** and **certificate-based authentication** and commonly uses port **802**, but adoption remains limited due to legacy device incompatibility, certificate management complexity, and processing overhead.

Recommended practice:

Modbus TCP services should be disabled unless specifically required for SCADA integration. When Modbus TCP is operationally necessary:

- Use strict **network segmentation** (dedicated VLANs or physically separate networks) to isolate Modbus traffic from corporate and public networks.
- Configure **firewalls** to permit port **502** traffic only between authorized SCADA masters and field devices, ideally with explicit **IP allowlists**.
- Use **VPN tunneling** or dedicated secure links for any remote Modbus access.
- Deploy industrial-protocol-aware **intrusion detection/monitoring** to flag anomalous Modbus activity.
- Keep device **firmware up to date** to address known vulnerabilities.
- Consider proxy/gateway solutions that add authentication and logging to legacy Modbus communications.
- Maintain strong **physical security** (locked enclosures and controlled access), since direct network/serial access can bypass network-layer controls.

Modbus functionality requires explicit enablement through CRBasic instructions (`ModbusServer()` or `ModbusClient()`) executed within an active data logger program.

For more information on using Modbus with Campbell Scientific data loggers see the following articles:

Why Modbus Matters: An Introduction

How to Access Live Measurement Data Using Modbus

Using Campbell Scientific Data Loggers as Modbus Servers in a SCADA Network

2.11 Distributed Network Protocol 3 (DNP3)

DNP3 (Distributed Network Protocol 3) is an industrial communications protocol designed for SCADA (Supervisory Control and Data Acquisition) systems, providing robust, efficient data exchange between master stations and remote terminal units (RTUs) or intelligent electronic devices (IEDs). Widely deployed in electric utilities, water and wastewater management, oil and gas operations, and transportation infrastructure, DNP3 emphasizes data integrity over minimal-bandwidth connections, making it well suited for remote monitoring and control applications.

Campbell Scientific data loggers implement DNP3 as **Level 2 outstation–compliant** devices with selected **Level 3** operations, supporting both **serial communications** (RS-232, COM1–COM4) and **TCP/IP networks**. DNP3 operates over **TCP or UDP** on the standard port **20000** (assigned by IANA), though **TCP** is more commonly used due to its reliable, connection-oriented transmission.

The protocol organizes data into **classes**: **Class 0** contains static (current point-in-time) data analogous to real-time measurements, while **Classes 1, 2, and 3** hold event history with configurable priorities and thresholds for change detection. This event-driven architecture significantly improves bandwidth efficiency compared to continuous polling, because only significant data changes are transmitted after an initial **integrity poll** retrieves the complete system state.

Campbell Scientific data loggers implement DNP3 through three primary CRBasic instructions:

- DNP() establishes the communications port and protocol parameters.
- DNPVariable() maps data logger arrays to DNP3 objects with specified variations and classes.
- DNPUpdate() refreshes DNP3 data points with current measurements.

Supported data logger models include **CR1000**, **CR3000**, **CR800 series**, **CR1000X**, **CR6**, and **CR350**, with DNP3 capability added via operating system updates.

For more information, see: DNP3 with Campbell Scientific Data Loggers.

Security considerations:

Base DNP3 provides no inherent encryption or authentication, transmitting all data—including control commands—in plain text. This makes DNP3 communications vulnerable to eavesdropping, man-in-the-middle attacks, command injection, and denial-of-service attacks. Port **20000** is well known and commonly scanned during network reconnaissance for exposed SCADA infrastructure, and vulnerabilities in DNP3 implementations have been documented in industry advisories.

Campbell Scientific addresses these concerns through optional **TLS (Transport Layer Security)** support on newer data loggers (for example, **CR1000X series** and **CR6**), providing encryption and

authentication for DNP3-over-TCP communications via the optional **DNPTLS** parameter in the `DNP()` instruction. Older data loggers (CR800, CR1000, CR3000) typically do not support TLS due to processor limitations. DNP3 also supports **Secure Authentication** mechanisms (commonly referenced with IEC 62351-5–aligned security approaches), but this requires compatible SCADA master systems.

Recommended practice:

Disable DNP3 services unless specifically required for SCADA integration. When DNP3 is necessary:

- Enable **TLS** on supported data loggers.
- Use strict **network segmentation** (VLANs/DMZs) to isolate SCADA networks from corporate and public networks.
- Configure **firewalls** to permit port **20000** traffic only between trusted control centers and field devices, with explicit source/destination IP restrictions.
- Use **VPN tunneling** for any remote access to DNP3-enabled devices.
- Monitor port **20000** traffic with intrusion detection/monitoring tools for anomalous patterns.
- Consider **DNP3 Secure Authentication** for critical control functions when supported end-to-end.
- Maintain strong **physical security**, since direct serial access can bypass network-layer protections.

DNP3 functionality requires explicit enablement through CRBasic instructions (`DNP()`, `DNPVariable()`, and `DNPUpdate()`) executed within an active data logger program.

2.12 Message Queuing Telemetry Transport (MQTT)

MQTT (Message Queuing Telemetry Transport) is an open, lightweight communication protocol designed for Internet of Things (IoT) applications using a publish/subscribe architecture. Campbell Scientific data loggers—including the CR1000X series, CR6, CR300/CR310, and GRANITE 9/10—support MQTT as clients, enabling them to publish data to cloud platforms such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud IoT without requiring specialized Campbell Scientific software.

The protocol uses a broker to facilitate communications between publishers and subscribers, with data loggers initiating all connections, which minimizes firewall-related complications

common in other protocols. MQTT functionality is implemented through the MQTTPublishTable() CRBasic instruction, which publishes data table contents in CSIJSON (Campbell Scientific JSON), GeoJSON, or BASICJSON formats. The data logger supports Quality of Service (QoS) levels 0 and 1 per MQTT v3.1.1 specification, providing options for message delivery guarantees.

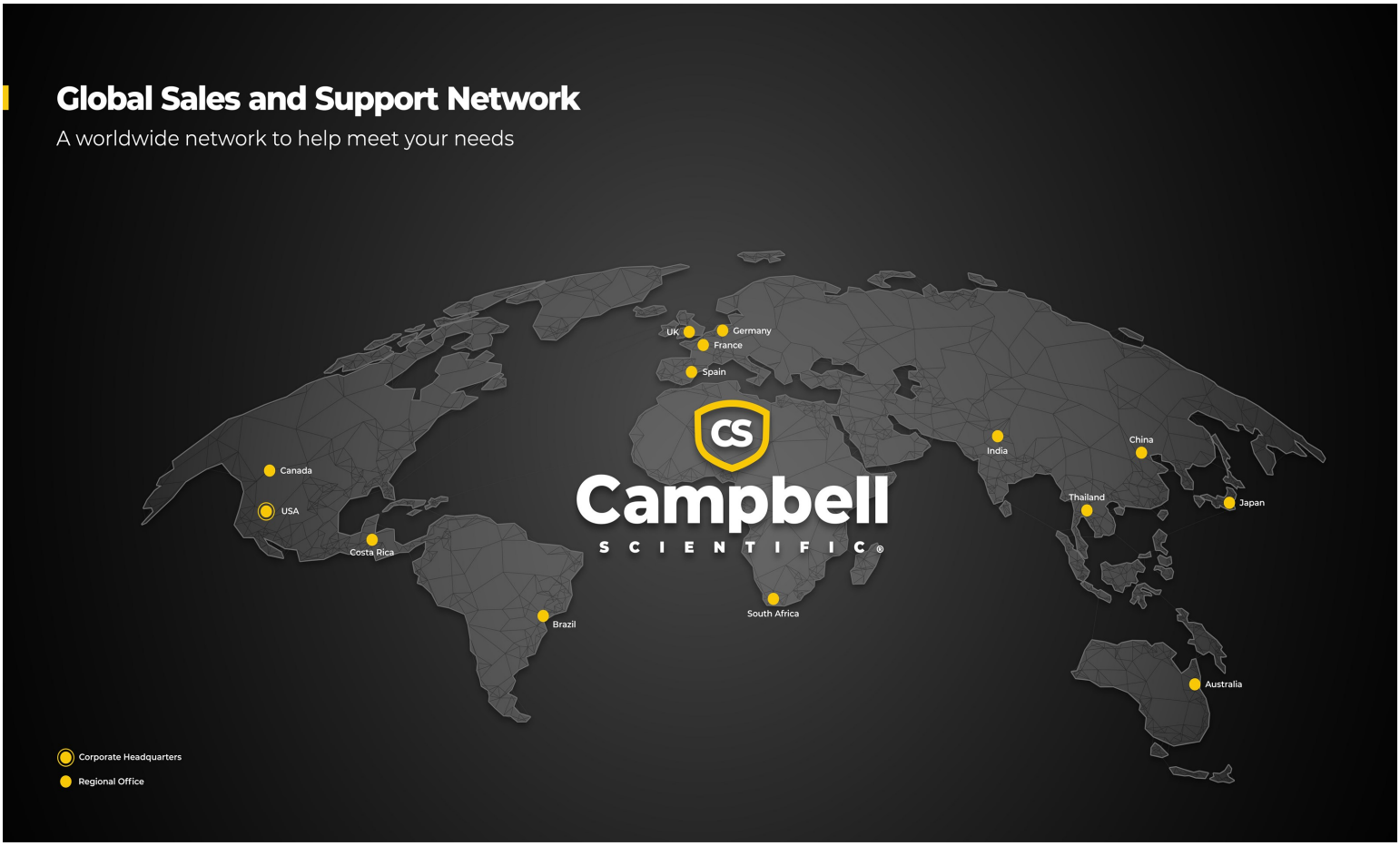
For more information see: Using MQTT with Campbell Scientific Data Loggers.

Security considerations:

From a security perspective, Campbell Scientific data loggers support MQTT with TLS encryption and TLS-Mutual Authentication, typically communicating over port 8883 for secure connections. When MQTT is enabled in the data logger settings, TLS or TLS-Mutual Authentication should be selected rather than unencrypted connections, and proper certificate configuration is required for broker authentication. This ensures that both data transmission and broker authentication are cryptographically protected.

Global Sales and Support Network

A worldwide network to help meet your needs



Campbell Scientific Regional Offices

Australia

Location:	Garbutt, QLD Australia
Phone:	61.7.4401.7700
Email:	info@campbellsci.com.au
Website:	www.campbellsci.com.au

Brazil

Location:	São Paulo, SP Brazil
Phone:	11.3732.3399
Email:	vendas@campbellsci.com.br
Website:	www.campbellsci.com.br

Canada

Location:	Edmonton, AB Canada
Phone:	780.454.2505
Email:	dataloggers@campbellsci.ca
Website:	www.campbellsci.ca

China

Location:	Beijing, P. R. China
Phone:	86.10.6561.0080
Email:	info@campbellsci.com.cn
Website:	www.campbellsci.com.cn

Costa Rica

Location:	San Pedro, Costa Rica
Phone:	506.2280.1564
Email:	info@campbellsci.cc
Website:	www.campbellsci.cc

France

Location:	Montrouge, France
Phone:	0033.0.1.56.45.15.20
Email:	info@campbellsci.fr
Website:	www.campbellsci.fr

Germany

Location:	Bremen, Germany
Phone:	49.0.421.460974.0
Email:	info@campbellsci.de
Website:	www.campbellsci.de

India

Location:	New Delhi, DL India
Phone:	91.11.46500481.482
Email:	info@campbellsci.in
Website:	www.campbellsci.in

Japan

Location:	Kawagishi, Toda City, Japan
Phone:	048.400.5001
Email:	jp-info@campbellsci.com
Website:	www.campbellsci.co.jp

South Africa

Location:	Stellenbosch, South Africa
Phone:	27.21.8809960
Email:	sales@campbellsci.co.za
Website:	www.campbellsci.co.za

Spain

Location:	Barcelona, Spain
Phone:	34.93.2323938
Email:	info@campbellsci.es
Website:	www.campbellsci.es

Thailand

Location:	Bangkok, Thailand
Phone:	66.2.719.3399
Email:	info@campbellsci.asia
Website:	www.campbellsci.asia

UK

Location:	Shepshed, Loughborough, UK
Phone:	44.0.1509.601141
Email:	sales@campbellsci.co.uk
Website:	www.campbellsci.co.uk

USA

Location:	Logan, UT USA
Phone:	435.227.9120
Email:	info@campbellsci.com
Website:	www.campbellsci.com