

SCADAPack 47xi Application

Note - Docker App

Connectivity and Redis

Overview

CONTENTS

1. SCADAPACK 47XI APPLICATION NOTE - DOCKER APP CONNECTIVITY AND REDIS OVERVIEW 4

1.1. Legal Information.....	4
1.2. Trademarks	4
1.3. Address	5
1.4. Technical Support	5
1.5. Safety Information	5
1.5.1. Important information.....	5
1.5.2. Please note	6
1.5.3. Before you begin.....	6
1.5.4. Operation and adjustments.....	7
1.5.5. Acceptable use	7
1.6. About the Book.....	8
1.6.1. Audience.....	8
1.6.2. Document scope	8
1.6.3. Validity note.....	8
1.6.4. Related documents.....	9
1.7. Cybersecurity.....	11
1.7.1. Troubleshooting USB Communication.....	11
1.7.2. Contact us.....	13
1.7.3. Additional Resources	13

2. INTRODUCTION.....14

3. SCADAPACK 47XI CONNECTIVITY OVERVIEW15

3.1. SCADAPack 47xi Internet Access	16
3.2. Docker Container Redis Access.....	17

4. REDIS OVERVIEW	18
4.1. Redis Data Types.....	18
4.2. Redis Login.....	19
4.3. Understanding the Redis-CLI format for Binary Values	19
4.4. Available Redis Keys.....	21
4.4.1. SYSTEM:CONFIGURATION_STATE	21
4.4.2. SYSTEM:OBJID_DIRECTORY	22
4.4.3. SYSTEM: OBJTYPE_TBL	23
4.4.4. SYSTEM: SYSTEM:OBJID:XXXXXX	24
4.4.5. Other Redis Keys.....	25
4.5. Example Operation – Read from Object Database	25
4.5.1. Available Fields	26
4.6. Example Operation – Write to Object Database	27
4.7. Bulk Operation LUA Scripts.....	28
4.7.1. Bulk Operation Overview.....	28
4.7.2. Redis LUA Script Overview.....	29
4.7.3. SCADAPack 47xi Multiread Redis Lua Script.....	30
4.7.4. SCADAPack 47xi Multi-Write Redis Lua Script	31
5. APPENDIX A – DOCKER REFERENCE.....	32
5.1. Introduction.....	32
5.2. Simple Test of Docker Installation	33
5.3. Example Commands	33
5.4. Saving and Loading Containers.....	36
5.5. Editing/Inspecting Existing Containers	37
5.6. Enabling Linux Commands from an Ubuntu Container	37
5.7. Dockerfile.....	38
5.7.1. Executing a Dockerfile Locally on the SCADAPack 47xi	38
5.7.2. Executing a Dockerfile from the SCADAPack 47xi SDK	39

1. SCADAPack 47xi Application Note - Docker App Connectivity and Redis Overview

1.1. Legal Information

The information provided in this document contains general descriptions, technical characteristics and/or recommendations related to products/solutions.

This document is not intended as a substitute for a detailed study or operational and site-specific development or schematic plan. It is not to be used for determining suitability or reliability of the products/solutions for specific user applications. It is the duty of any such user to perform or have any professional expert of its choice (integrator, specifier or the like) perform the appropriate and comprehensive risk analysis, evaluation and testing of the products/solutions with respect to the relevant specific application or use thereof.

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this document are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owner.

This document and its content are protected under applicable copyright laws and provided for informative use only. No part of this document may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the document or its content, except for a non-exclusive and personal license to consult it on an "as is" basis.

Schneider Electric reserves the right to make changes or updates with respect to or in the content of this document or the format thereof, at any time without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this document, as well as any non-intended use or misuse of the content thereof.

1.2. Trademarks

Schneider Electric, EcoStruxure Geo SCADA Expert (ClearSCADA), Modbus, SCADAPack, Telepace, and Trio are trademarks and the property of Schneider Electric SE, its subsidiaries and affiliated companies. All other trademarks are the property of their respective owners.

1.3. Address

Schneider Electric
35 rue Joseph Monier
92500 Rueil-Malmaison, France

Email: RemoteOperations@se.com
www.se.com

© 2023 Schneider Electric
All rights reserved.

1.4. Technical Support

Questions and requests related to any part of this documentation can be directed to one of the following support centers.

Technical support: Americas, Europe, Middle East, Asia

Available Monday to Friday 8:00 am – 6:30 pm Eastern Time

	Check our FAQs	Explore our extensive knowledge database and FAQ videos to find answers quickly: https://se.com/faq
	Email us	Save time by emailing us your inquiry and an expert will contact you: supportRO@se.com Send us an email anytime.
	Call us	Need someone to provide some technical support? Toll free within North America: 1-888-226-6876 Direct Worldwide: +1-613-591-1943

1.5. Safety Information

1.5.1. Important information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a Danger or Warning safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

 DANGER
DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.

 WARNING
WARNING indicates a hazardous situation which, if not avoided, can result in death or serious injury.

 CAUTION
CAUTION indicates a hazardous situation which, if not avoided, can result in minor or moderate injury.

NOTICE
NOTICE is used to address practices not related to physical injury.

1.5.2. Please note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction, installation, and operation of electrical equipment and has received safety training to recognize and avoid the hazards involved.

1.5.3. Before you begin

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

 WARNING
EQUIPMENT OPERATION HAZARD
Verify that all installation and setup procedures have been completed.

Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.

Remove tools, meters, and debris from equipment.

Failure to follow these instructions can result in death or serious injury.

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future reference.

Test all software in both simulated and real environments.

Verify that the completed system is free from all short circuits and grounds, except those grounds installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to help prevent accidental equipment damage.

1.5.4. Operation and adjustments

The following precautions prevail:

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

1.5.5. Acceptable use

WARNING

UNACCEPTABLE USE

Do not use SCADAPacks or I/O modules as an integral part of a safety system. These devices are not safety products.

Failure to follow these instructions can result in death or serious injury.

CAUTION

EQUIPMENT OPERATION HAZARD

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Use only Schneider Electric software or approved software with Schneider Electric hardware products.

Failure to follow these instructions can result in minor or moderate injury.

1.6. About the Book

1.6.1. Audience

WARNING

UNINTENDED EQUIPMENT OPERATION

The application of this product requires expertise in the design and programming of control systems and application programming. Only persons with such expertise are allowed to program, install, alter, and apply this product.

Follow all local and national safety codes and standards.

Failure to follow these instructions can result in death or serious injury.

This manual is written for people who are typically:

- Systems Engineers
- Commissioning Engineers
- Maintenance Technicians

1.6.2. Document scope

This manual describes:

- Installing and using a Ubuntu demonstration on the SCADAPack 47xi Edge platform

1.6.3. Validity note

This document is valid for:

- SCADAPack x70 firmware version 9.8.4
- SCADAPack RemoteConnect configuration software version 4.1.2
- SCADAPack 47xi Edge platform firmware version 9.8.4
- SCADAPack 47xi Edge application SDK version 3.0.2

1.6.4. Related documents

Use this manual with the other manuals included in your SCADAPack 47xi documentation set. The table below describes the manuals available in the documentation set.

Folder	Manual	Content
Getting Started	Getting Started	• The SCADAPack 47xi family of products available in this release • The basic steps to get your SCADAPack 47xi device operational • Where to get more information about configuring, monitoring, and managing your SCADAPack 47xi device
SCADAPack Software Installation	SCADAPack Software Installation	• Hardware and software requirements • Installation procedures • Accessing help • Troubleshooting guidance
Hardware Manuals	The hardware manual for your SCADAPack 47xi device	• Installation, wiring and addressing information • Diagnostics capabilities • Maintenance recommendations • Hardware specifications
Configuration Manuals	SCADAPack RemoteConnect Configuration Software	• Setting up and managing projects for your SCADAPack 47xi device
	PC Communication Settings -SCADAPack CommDTM	• Setting up communications between SCADAPack RemoteConnect and your SCADAPack 47xi device
	SCADAPack 47xi Configuration	• Configuring SCADAPack 47xi device operation
	Porting Guide for SCADAPack E to SCADAPack RemoteConnect	• Moving from SCADAPack E to SCADAPack RemoteConnect • Locating SCADAPack E Configurator features in SCADAPack RemoteConnect • Locating SCADAPack Workbench features in SCADAPack RemoteConnect • Compatibility chart

	Porting Guide for Telepace to SCADAPack RemoteConnect	• Moving from Telepace to SCADAPack RemoteConnect • Tutorial for creating a project • Compatibility chart
Technical Reference Manuals	SCADAPack Communication Interfaces Technical Reference	• USB, serial, and IP communications • Mobile communications • Dialup modem communications
	SCADAPack Operations Technical Reference	• The SCADAPack 47xi device file system • Command line operations • Diagnostics operations • Telnet server operations • FTP server operations
	SCADAPack SCADA Protocols Technical Reference	• DNP3 protocol support • Modbus protocol support • IEC 60870-5-104 protocol support
	SCADAPack 47xi Edge Technical Reference	• The SCADAPack 47xi Edge applications platform • Edge platform services • Edge platform software development kit (SDK) workflow, installation, and usage • Edge platform technical description
Logic Programming Manuals	SCADAPack Logic Programming Overview	• The differences between EcoStruxure Control Expert (Unity Pro) and the SCADAPack 47xi Logic Editor environment • Key programming concepts • Basic procedures needed to use the SCADAPack 47xi Logic Editor
	SCADAPack Function Blocks Technical Reference	• The custom SCADAPack 47xi function blocks that are available for developing IEC 61131-3 applications
	Using EFB Toolkit with SCADAPack 47xi	• Using the Schneider Electric EFB Toolkit with SCADAPack 47xi devices and SCADAPack RemoteConnect configuration software
IoT View Operations	IoT View Operations	• The SCADAPack 47xi Edge platform web server • How to enable IoT View

Security Administrator Manuals	SCADAPack DNP3 Security Administrator	Configuring security on your SCADAPack 47xi device
	SCADAPack DNP3 Security Technical Reference	- Security standards - Security overview - DNP3 Secure Authentication - Diagnostics - Attack vectors and requirements

1.7. Cybersecurity

Cybersecurity is a branch of network administration that addresses attacks on or by computer systems and through computer networks that can result in accidental or intentional disruptions. The objective of cybersecurity is to help provide increased levels of protection for information and physical assets from theft, corruption, misuse, or accidents while maintaining access for intended users.

No single cybersecurity approach is adequate. Schneider Electric recommends a defense-in-depth approach. This approach layers the network with security features, appliances, and processes. The basic components of this approach are:

- Risk assessment: A systematic security analysis of the environment and related systems.
- A security plan built on the results of the risk assessment
- A multi-phase training campaign
- Network separation and segmentation: Physical separation of the control network from other networks, and the division of the control network itself into segments and security zones.
- System Access Control: Controlling access to the system with firewalls, authentication, authorization, and other software means, and traditional physical security measures such as video surveillance, fences, locked doors and gates, and locked equipment cabinets.
- Device hardening: The process of configuring a device against communication-based threats. Device hardening measures include disabling unused network ports, password management, access control, disabling of unnecessary protocols and services, and preventing usage of USB devices. The SCADAPack presents itself as a USB device, which may be blocked by IT teams. See section **Troubleshooting USB Communication** for details on how IT teams can identify SCADAPack devices.
- Network monitoring and maintenance: An effective defense-in-depth campaign requires continual monitoring and system maintenance to meet the challenge of new threats as they develop.
- See **Security Considerations** in the Security Technical Reference manual

1.7.1. Troubleshooting USB Communication

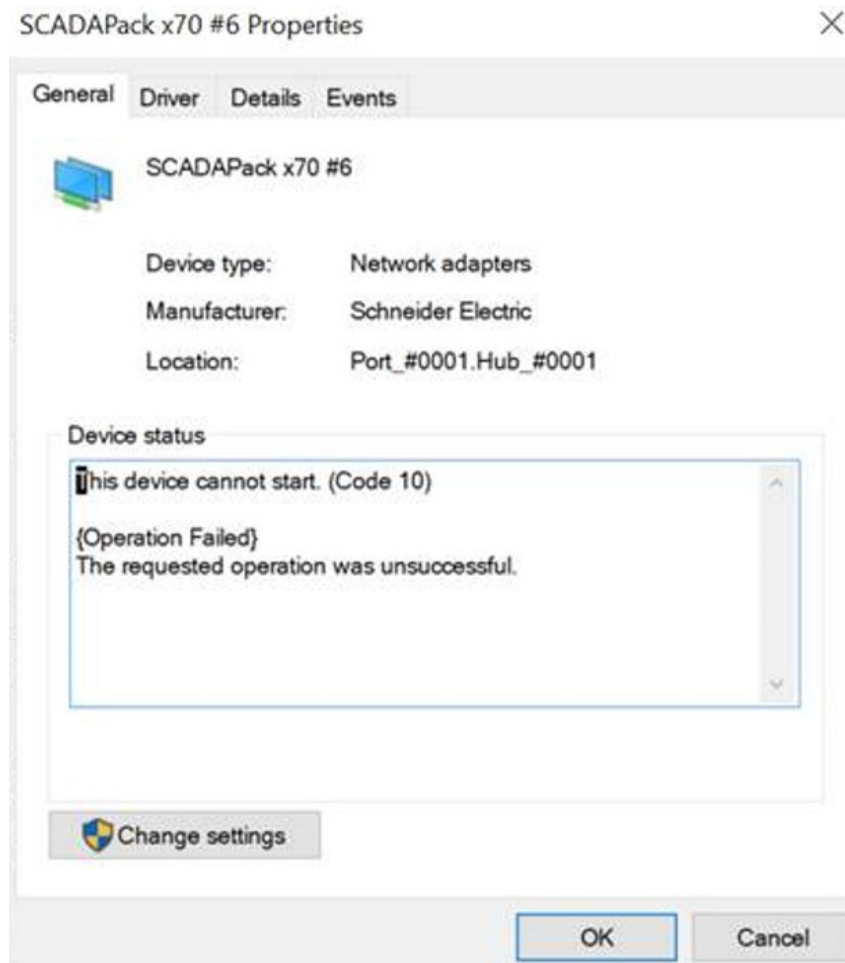
SCADAPack RTUs can connect to SCADAPack RemoteConnect via the USB interface. Endpoint protection software, such as SentinelOne, may be used by IT teams to limit USB connectivity. These limitations can interfere with the communication between your SCADAPack and RemoteConnect.

1.1.1.1. Indicators that an IT control may be affecting communication can include

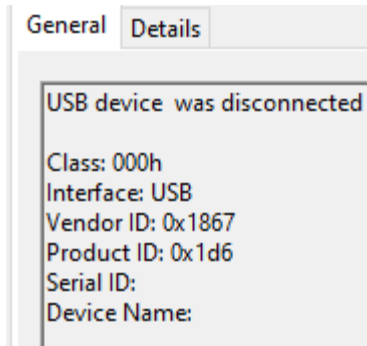
- RemoteConnect has identified the serial number of the connected SCADAPack yet is unable to read or write to it via USB.

- In Device Manager the SCADAPack x70 Network Adaptor is displaying the caution symbol such as  SCADAPack x70

- The SCADAPack x70 driver details is reporting that the device cannot start. For example,



- Windows Event Viewer logs is indicating that the device was disconnected. These logs may be under the “Application and Services Logs” in a branch named for the endpoint protection tool in use. Launching Event Viewer with Administrator permissions may be necessary to see these logs. For example,



Endpoint protection software does not always log blocked connections as an alert in the administration console used by IT teams. This can cause additional diagnostics steps to be needed for IT to confirm that the endpoint protection has prevented the SCADAPack from being fully recognized by Windows. The impact of the endpoint protection software can also be tested by temporarily disabling the endpoint protection.

Local IT teams should investigate the cause of the USB connection not being established as expected. Remediation may include entering an exception rule in endpoint protection software to create a permit (or allow) rule for specific devices. Where a rule is added, full read write access is necessary. Schneider Electric SCADAPacks present themselves with the following information that can be used to create rules permitting their connection.

	SCADAPack 470 SCADAPack 470i SCADAPack 474 SCADAPack 474i	SCADAPack 570 SCADAPack 574 SCADAPack 575
Class	000h	000h
Interface	USB	USB
Vendor ID	0x1867	0x1867
Product ID	0x1d6	0x23A

1.7.2. Contact us

For more information, refer to the Schneider Electric Cybersecurity Support Portal at <http://www.se.com/b2b/en/support/cybersecurity/overview.jsp>.

1.7.3. Additional Resources

Schneider Electric Recommended Cybersecurity Best Practices

<https://www.se.com/ww/en/download/document/CS-Best-Practices-2019-340/>

Cybersecurity and Infrastructure Security Agency

<https://www.cisa.gov/>

ICS-CERT Recommended Practices

<https://ics-cert.us-cert.gov/Recommended-Practices>

Center for Internet Security (CIS) Top 20 Critical Security Controls

<https://www.cisecurity.org/cybersecurity-best-practices>

FBI Cyber Crime

<https://www.fbi.gov/investigate/cyber>

Guide to Industrial Control Systems (ICS) Security

<https://www.nist.gov/publications/guide-industrial-control-systems-ics-security>

WaterISAC Water Security Network

<https://www.waterisac.org>

2. Introduction

The SCADAPack 47xi object database is accessible through a provided Redis server. API's have been implemented for accessing the database via C, C++, and in JavaScript using a Node-RED container.

This document is intended for intermediate and advanced users who want to access the Redis database in other ways and using other programming languages either from the SCADAPack natively or from within a container. This document will also give you a deeper understanding of how the SCADAPack 47xi works and give guidance on how to configure it for web access.

This application note does the following:

- Provides an application on the SCADAPack 47xi, (including applications running in a docker container) with access to the SCADAPack 47xi's object database
- Bring internet access to a SCADAPack 47xi
- Provides a general overview of Docker

The SCADAPack 47xi filesystem image is read-only but can accept a read/write filesystem overlay that resides on the Edge platform's SD Card. Installing the overlay causes the read-only filesystem to become read-write. To make any changes to files on your Edge platform, you need to have a filesystem overlay installed.

Any changes described to files on the Edge platform described in this doc can be made by either of the following techniques:

- By making the change in the SDK, packaging to overlay, then applying it to your SCADAPack 47xi(s)
- By making them directly on a SCADAPack 47xi after an overlay (including a blank overlay) has already been installed.

See also:

- Installing and Using the SCADAPack 47xi Edge Applications SDK
- SCADAPack 47xi Application Note – Docker Container with Ubuntu for Python Solutions
- SCADAPack 47xi Application Note – Building a Node-RED Installation

3. SCADAPack 47xi Connectivity Overview

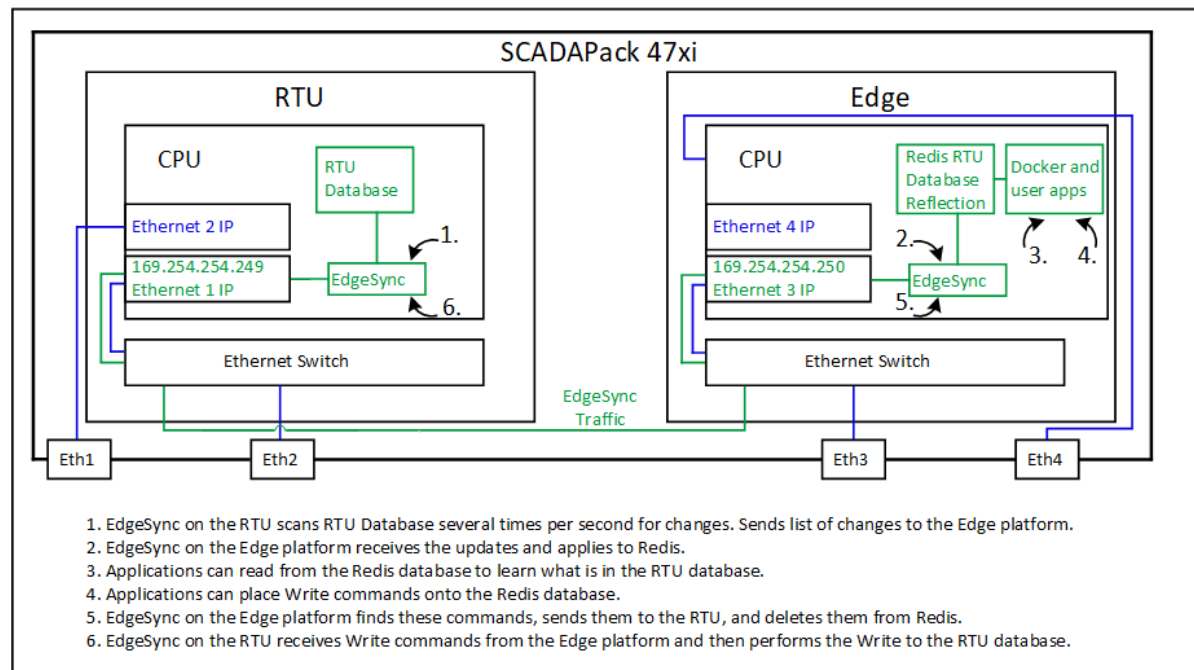
The SCADAPack 47xi is composed of two interconnected platforms:

- An RTU platform running a real-time logic application
- A Linux Edge platform

The RTU and its real-time logic application contain and have direct access to the principal copy of the object database. The RTU and Edge platforms both contain an EdgeSync service. This service is responsible for maintaining the Redis database on the Edge platform with an updated reflection of the status of the values found in the principal database on the RTU. The update frequency depends on several factors.

When object writes are performed from the Edge platform, these write requests are queued, sent to the RTU and performed. The updated values will then be reflected in the Redis database and will be visible to Edge applications.

This information workflow is described below.



3.1. SCADAPack 47xi Internet Access

The application notes listed in the Introduction describe how to use the SDK to create an overlay with containers from your PC and how to deploy this overlay to your SCADAPack 47xi.

If you instead want to create containers directly on your SCADAPack 47xi, or if you want to install packages into containers that already exist on your SCADAPack 47xi, your SCADAPack 47xi will need internet access.

Once your SCADAPack 47xi has internet access, docker and your docker containers will also have internet access.

To configure your SCADAPack 47xi to have internet access:

1. Configure Ethernet 3 or Ethernet 4 to be on an IP address on your LAN subnet, but outside the DHCP range of your router.
2. Configure the Default Gateway IP Address of the Edge platform to point to the LAN IP of your router.

For example:

If your router is configured like this.

LAN IP: 192.168.8.1

Start IP Address: 192.168.8.100

End IP Address: 192.168.8.249

This is one way to configure your SCADAPack

Basic Configuration

General Setup

IP Address: 192.168.8.41

Subnet Mask: 255.255.255.0

Default Gateway IP Address (Edge Platform): 192.168.8.1

Enable RTU Forwarding: ☒

3. Configure a DNS server for your SCADAPack 47xi by making your `resolv.conf` file look like this:

```
|/etc/resolv.conf|  
-----  
|nameserver 8.8.8.8|
```

```
|nameserver 8.8.4.4|
-----
```

You can use the Google or OpenDNS server or any other DNS server and run the following commands on your SCADAPack 47xi:

```
rm /etc/resolv.conf
echo "nameserver 8.8.8.8" > /etc/resolv.conf
echo "nameserver 8.8.4.4" >> /etc/resolv.conf
sudo systemctl mask systemd-resolved
```

4. Confirm connectivity:

```
root@scadapack-47xi:~# ping www.google.com
PING www.google.com (142.251.15.103): 56 data bytes
64 bytes from 142.251.15.103: seq=0 ttl=109 time=38.084 ms
```

3.2. Docker Container Redis Access

This section describes the steps needed to provide internet to your SCADAPack 47xi (including connecting to applications running in a docker container on your SCADAPack 47xi). If you follow the instructions in either the Ubuntu Python or Node-Red Application Note referenced above, the scripts provided will do most of these steps automatically.

There are several steps that need to be done to allow an application in a docker container to access the Redis database.

1. The Redis configuration needs to be modified so that Redis listens on the Docker Interface.

```
-----
|/etc/redis/redis.conf|
-----
|          BEFORE          |          AFTER          |
-----
|bind 127.0.0.1|bind 127.0.0.1 10.10.0.1|
-----
```

To listen on the docker interface, the docker interface needs to exist. The Redis service file needs to be modified so that it is a dependent of and does not start until after docker is running.

```
-----
|/lib/systemd/system/redis.service|
-----
|          BEFORE          |          AFTER          |
-----
|After=network.target|After=network.target, docker.service|
-----
```

2. The password for the Redis database is changed each time the SCADAPack 47xi boots. It is available on a cryptographic drive with path `/mnt/sec_data/redis` and can be read locally by applications with root access. When creating a container that needs access to Redis, you need to share this drive (or the password it contains) with that container. This allows the container to read the Redis password. One way to do this is to add the following argument to your `docker run` command.

```
-v /mnt/sec_data/redis:/mnt
```

3. From inside the container, use this command to see the Redis credentials:

```
cat /mnt/userredispass
```

4. In addition to sharing the crypt drive with the container, the container should be instantiated with:

```
"--privileged"
```

For example, the complete command to create the docker container may look like this:

```
docker run --privileged -it -p 1880:1880 -v  
/home/root/Ubuntu:/data -v /mnt/sec_data/redis:/mnt --name  
Ubuntu ubuntu
```

See Appendix A – Docker Reference.

The docker IP address for the host SCADAPack 47xi is 10.10.0.1. Redis listens on port 6379.

5. Point your Redis library inside the container to this location.

4. Redis Overview

This section is a general overview of Redis and is intended to provide the foundational knowledge necessary to access the SCADAPack 47xi's object database.

Redis is an in-memory non-relational database. It provides a way to store keys that have values.

4.1. Redis Data Types

The [data type](#) of a Redis key can be seen with the [TYPE](#) command.

These are the Redis types used on the SCADAPack 47xi.

Data Type	Comments and Commands Used
Strings	Redis strings store sequences of bytes and are the most basic Redis data type. Commands used to access Strings: • SET key value • GET key

	DEL key
Hashes	A hash is a collection of field-value pairs. The values are a sequence of bytes. Commands used to access hashes: - HSET: Maps a value to a key within the hash - HGET: Retrieves individual values associated with a key within the hash - HGETALL: Displays the entire hash content - HDEL: Removes an existing key-value pair from a hash
Lists	A list is an ordered sequence of strings. Commands used to access Lists: - LPUSH: Pushes the value to the left end of the list - RPUSH: Pushes the value to the tail end of the list - LRANGE: Retrieves a range of items - LPOP/RPOP: Used to display and remove items from both ends - LINDEX: Obtains a value from a specific position within the list

4.2. Redis Login

This section describes how to log in to Redis to perform reads and writes.

1. The Redis database on each SCADAPack 47xi is locked with a unique password. Redis passwords are stored here.

```
/mnt/sec_data/redis/
```

For example, from the Linux Edge platform command line, use this command to learn the Redis password for the “user” account.

```
cat /mnt/sec_data/redis/userredispass
```

2. Once you know the password, log in with the “[AUTH](#)” command.

This is the syntax from the manual “redis-cli” utility.

```
AUTH USER
0000111122223333444455556666777788889999aaaabbbbccccdddeeeeffff
```

4.3. Understanding the Redis-CLI format for Binary Values

Every major programming language includes libraries for accessing a Redis database. The Redis database can also be accessed manually via the “redis-cli” utility.

When using a programming language with a Redis library to access the Redis database, handling binary values is typically straightforward. Depending on the software used, the values appear as a character array, byte array, ascii string, or JavaScript buffer. This document refers to them as “byte arrays”. Libraries can be used to convert between a “byte array” and a numeric data type.

Manually handling “byte arrays” from the redis-cli is unique to Redis.

Alphanumeric strings are clearly printed

Consider the “Value” of SYS_DEVICE_Edge_RootFSversion. It appears as follows: **R3.1.1 Build 600**

Numeric values may be clear

If no character is printable, then the hexadecimal representation of each byte in the hash is displayed.

Consider the “ONLINE_QUAL” of this point appears as: “\x01\x00\x00\x00”

It can be interpreted as follows

$$1*1 + 0*256 + 0*65536 + 0*16777216 = 1$$

But in some cases, are not obvious.

If some characters are printable but others are not, the result may be surprising.

ASCII characters appear plainly. Even certain ones such as “\t” or “\n” print, but do so as escape characters. Other characters do not print at all but instead appear as raw hexadecimal prefixed by “\x”

This table shows how every possible byte would be printed.

This representation should also be used when providing binary input data to the redis-cli. When multiple bytes exists, their representations are concatenated together.

Redis-CLI Output Format																
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	\x00	\x10		0	@	P	`	p	\x80	\x90	\xa0	\xb0	\xc0	\xd0	\xe0	\xf0
1	\x01	\x11	!	1	A	Q	a	q	\x81	\x91	\xa1	\xb1	\xc1	\xd1	\xe1	\xf1
2	\x02	\x12	"	2	B	R	b	r	\x82	\x92	\xa2	\xb2	\xc2	\xd2	\xe2	\xf2
3	\x03	\x13	#	3	C	S	c	s	\x83	\x93	\xa3	\xb3	\xc3	\xd3	\xe3	\xf3
4	\x04	\x14	\$	4	D	T	d	t	\x84	\x94	\xa4	\xb4	\xc4	\xd4	\xe4	\xf4
5	\x05	\x15	%	5	E	U	e	u	\x85	\x95	\xa5	\xb5	\xc5	\xd5	\xe5	\xf5
6	\x06	\x16	&	6	F	V	f	v	\x86	\x96	\xa6	\xb6	\xc6	\xd6	\xe6	\xf6
7	\a	\x17	'	7	G	W	g	w	\x87	\x97	\xa7	\xb7	\xc7	\xd7	\xe7	\xf7
8	\b	\x18	(	8	H	X	h	x	\x88	\x98	\xa8	\xb8	\xc8	\xd8	\xe8	\xf8
9	\t	\x19	)	9	I	Y	i	y	\x89	\x99	\xa9	\xb9	\xc9	\xd9	\xe9	\xf9
A	\n	\x1a	*	:	J	Z	j	z	\x8a	\x9a	\xaa	\xba	\xca	\xda	\xea	\xfa
b	\x0b	\x1b	+	;	K	[	k	{	\x8b	\x9b	\xab	\xbb	\xcb	\xdb	\xeb	\xfb
c	\x0c	\x1c	,	<	L	\	l		\x8c	\x9c	\xac	\xbc	\xcc	\xdc	\xec	\xfc

d	\r	\x1d	-	=	M	]	h	}	\x8d	\x9d	\xad	\xb8	\xcd	\xdd	\xed	\xfd
e	\x0e	\x1e	.	>	N	^	n	~	\x8e	\x9e	\xae	\xbe	\xce	\xde	\xee	\xfe
f	\x0f	\x1f	/	?	0	_	o		\x7f	\x8f	\x9f	\xaf	\xbf	\xcf	\xdf	\xff

For example, consider the "TIMESTAMP_SEC" of a point that might appear in the redis-cli as:
"u\x9f9xd"

It can be interpreted in this way.

Displayed	Printable	Dec	Hex
u	Y	117	75
\x9f	N	249	F9
x	Y	120	78
d	Y	100	64

117*1 + 249*256 + 120*65536 + 100*16777216 = 1685649781
= 2023-06-08 18:16:53 (Unix Time)

4.4. Available Redis Keys

4.4.1. SYSTEM:CONFIGURATION_STATE

Description

When you write a new RTU configuration with RemoteConnect, it might contain different objects than the old one. The Redis database will be refreshed. It is important for Edge applications to be able to detect this.

These are the possible values of this key.

Possible Values	Meaning
CONFIGURATION_WAIT	A new RTU configuration is in-progress. The EdgeSync service on the Linux Board is in the process of receiving the new configuration.
CONFIGURATION_RESET	A new RTU configuration is in-progress. EdgeSync has received the new object database configuration. The Redis database will be wiped and replaced by the new configuration.
CONFIGURATION_COMPLETE	The developed program can now read from or write to RTU objects. This value indicates normal steady-state operation.

Example

This is an example of how to read the key manually from the “redis-cli”

```
127.0.0.1:6379> get SYSTEM:CONFIGURATION_STATE
"CONFIGURATION_COMPLETE"
```

Recommendation

The Edge application should poll this string and whenever it equals anything other than “CONFIGURATION_COMPLETE” it should pause normal operation of reading and writing to RTU objects until the new RTU configuration is complete. At this point, it should re-initialize itself (if necessary) and then resume normal operation.

4.4.2. SYSTEM:OBJID_DIRECTORY

Description

This is a Redis hash that associates each RTU object name with an Object ID. It should be referenced during program startup or on the rising edge of CONFIGURATION_COMPLETE. These Object ID’s can then be used to look up object values.

Example from Redis-CLI

From the “redis-cli” `hgetall` produces a linear list that alternates between object names and Object ID’s.

- Odd numbered values contain RTU object names
- Even numbered values contain RTU object ID’s

```
127.0.0.1:6379> hgetall SYSTEM:OBJID_DIRECTORY
1) "ClockTicks"
2) "1000030"
3) "Load"
4) "1000021"
5) "MotorTorque"
6) "1000031"
```

Example from a Programming Library

From programming environments, the output might be different than this. For example, in JavaScript, `hgetall` outputs a JavaScript object. Any JavaScript objects hves key-value pairs.

In this case:

- The JavaScript object’s keys contain the RTU object names
- The JavaScript object’s values contain the RTU object ID’s

Code	<pre>client.hgetall('SYSTEM:OBJID_DIRECTORY', function(err, objdir){ console.log(objdir)</pre>
Output	<pre>{ ClockTicks: <Buffer 31 30 30 30 30 33 30>, Load: <Buffer 31 30 30 30 30 32 31>, MotorTorque: <Buffer 31 30 30 30 30 33 31>,</pre>

Recommendation

Keep a lookup table or function that can be passed in the object name but that returns the object ID. This object ID will be needed to manipulate the object.

4.4.3. SYSTEM: OBJTYPE_TBL

Description

This is a Redis hash that associates each RTU Object ID with a RemoteConnect Object Editor datatype. It should also be referenced during program startup or on the rising edge of CONFIGURATION_COMPLETE. The datatype needs to be known to read or write to an object's "VALUE" field.

Any field in a Redis hash is a string of bytes. All numeric data types represented in a Redis hash on the SP47xi are in little endian format. This includes the object type.

These are the possible SCADAPack 47xi data types.

Object Type Enum	Meaning
0	RemoteConnect Object Editor "String" datatype The object's "VALUE" contains a UTF-8 string
1	RemoteConnect Object Editor "Binary" datatype The object's "VALUE" is a single byte with value 1 or 0
2	RemoteConnect Object Editor "Analog" datatype The object does not contain a "VALUE" It contains both of the following instead: "VALUE_ENG" with raw 4 bytes of the floating-point value from this field of the RTU object "VALUE_RAW" with raw 4 bytes of the integer value from the RTU object
3	RemoteConnect Object Editor "Counter" datatype

	The object's "VALUE" contains the raw 4 bytes in little endian
--	--

Example

This is an example of how to read the key manually from the "redis-cli"

```
127.0.0.1:6379> hgetall SYSTEM:OBJTYPE_TBL
```

- 1) "1000002"
- 2) "1"
- 3) "1000003"
- 4) "1"

Recommendation

Keep a lookup table or function that can be passed in the object ID but that returns the RemoteConnect Object Editor datatype. This datatype will be needed when interpreting the "VALUE" of the object.

4.4.4. SYSTEM: SYSTEM:OBJID:XXXXXX

Description

This is a Redis hash containing the value of each field of the RTU object with the object ID.

Example

This is an example of how to read the key manually from the "redis-cli"

```
127.0.0.1:6379> hgetall SYSTEM:OBJID:550733
```

- 1) "ALERT_INHIBIT"
- 2) "\x00\x00\x00\x00"
- 3) "FORCED_QUAL"
- 4) "\x00\x00\x00\x00"
- 5) "IO_NONRESPONSE_QUAL"
- 6) "\x00\x00\x00\x00"
- 7) "LOGIC_QUAL"
- 8) "\x00\x00\x00\x00"
- 9) "MODBUS_REGISTER"
- 10) "\x00\x00\x00\x00"
- 11) "OBJ_TYPE"
- 12) "\x00\x00"
- 13) "ONLINE_QUAL"
- 14) "\x01\x00\x00\x00"

- 15) "TIMESTAMP_MS"
- 16) "N\x01\x00\x00"
- 17) "TIMESTAMP_SEC"
- 18) "u\xf9xd"
- 19) "TREND_INHIBIT"
- 20) "\x00\x00\x00\x00"
- 21) "USER_SET_OFFLINE_QUAL"
- 22) "\x00\x00\x00\x00"
- 23) "VALUE"
- 24) ""

4.4.5. Other Redis Keys

These other keys also appear in the Redis database.

Key Name	Description
SYSTEM:OBJVALUE_TBL	This contains the VALUE and/or VALUE_ENG & VALUE_RAW of every object in the object database.
SYSTEM:FUNCTIONS:MULTIREAD SYSTEM:FUNCTIONS:MULTIWRITE SYSTEM:MREADID:<CMDID>:STATUS SYSTEM:MWRITED:<CMDID>:STATUS	These keys are used for bulk operations and are described in more detail later in this document.

4.5. Example Operation – Read from Object Database

- Learn the Object ID of the Object with that Name.

```
127.0.0.1:6379> hgetall SYSTEM:OBJID_DIRECTORY
```

- 1) "ClockTicks"
- 2) "1000030"

- Learn its object type.

```
127.0.0.1:6379> hgetall SYSTEM:OBJTYPE_TBL
```

```
..
```

- 57) "1000030"
- 58) "2"

- Since its object type is "2", it is an Analog object and will have both a VALUE_ENG and VALUE_RAW

```
127.0.0.1:6379> hgetall SYSTEM:OBJID:1000030
```

..

95) "VALUE_ENG"

96) "\x14.\xd9B"

97) "VALUE_RAW"

98) "k*\x00\x00"

4.5.1. Available Fields

These are all the types of objects and the fields that are available for each.

The length of each of these fields is as follows:

- VALUE: 4 Bytes if it's a number but if it's a string it can be of variable length
- OBJ_TYPE: 2 Bytes
- ALL OTHERS: 4 Bytes

0 String	1 Binary	2 Analog	3 Counter
ALERT_INHIBIT	ALERT_ACTIVE_STATE	ALERT_INHIBIT	ALERT_INHIBIT
FORCED_QUAL	ALERT_INHIBIT	CHECK_REFERENCE_QUAL	FORCED_QUAL
IO_NONRESPONSE_QUAL	ALERT_STATE_QUAL	ENG_MAXSCALE	HIGH_LIMIT
LOGIC_QUAL	CHATTER_QUAL	ENG_MINSCALE	HIGH_LIMIT_QUAL
MODBUS_REGISTER	FORCED_QUAL	FORCED_QUAL	IO_NONRESPONSE_QUAL
OBJ_TYPE	IO_NONRESPONSE_QUAL	H1_ENABLE	LOGIC_QUAL
ONLINE_QUAL	LOGIC_QUAL	H1_LIMIT	MODBUS_REGISTER
TIMESTAMP_MS	MODBUS_REGISTER	H1_STATE	OBJ_TYPE
TIMESTAMP_SEC	OBJ_TYPE	H2_ENABLE	ONLINE_QUAL
TREND_INHIBIT	ONLINE_QUAL	H2_LIMIT	TIMESTAMP_MS
USER_SET_OFFLINE_QUAL	TIMESTAMP_MS	H2_STATE	TIMESTAMP_SEC
VALUE	TIMESTAMP_SEC	H3_ENABLE	TREND_INHIBIT
	TREND_INHIBIT	H3_LIMIT	USER_SET_OFFLINE_QUAL
	USER_SET_OFFLINE_QUAL	H3_STATE	VALUE
	VALUE	H4_ENABLE	
		H4_LIMIT	
		H4_STATE	
		IO_NONRESPONSE_QUAL	
		L1_ENABLE	
		L1_LIMIT	
		L1_STATE	
		L2_ENABLE	

		L2_LIMIT	
		L2_STATE	
		L3_ENABLE	
		L3_LIMIT	
		L3_STATE	
		L4_ENABLE	
		L4_LIMIT	
		L4_STATE	
		LOGIC_QUAL	
		MODBUS_REGISTER	
		NO_CHANGE_QUAL	
		OBJ_TYPE	
		ONLINE_QUAL	
		OVER_RANGE_LIMIT	
		OVER_RANGE_QUAL	
		RAW_MAXSCALE	
		RAW_MINSCALE	
		ROF_EXCEEDED_QUAL	
		ROR_EXCEEDED_QUAL	
		TIMESTAMP_MS	
		TIMESTAMP_SEC	
		TREND_INHIBIT	
		UNDER_RANGE_LIMIT	
		UNDER_RANGE_QUAL	
		USER_SET_OFFLINE_QUAL	
		VALUE_ENG	
		VALUE_RAW	

4.6. Example Operation – Write to Object Database

Overview

When writing to the object database, do not directly write to the field of interest.

EdgeSync will over-write it and once again the object database will reflect whatever is in the RTU's object database.

Instead, create a particular type of hash on the Redis database which is a request for EdgeSync to perform this write. EdgeSync has a watcher thread that will see the hash, delete it, and perform the write.

When writing to a binary value, enter T or F.

Examples

To write to PIO_SP470_DO2:L

1. Learn its object ID:

```
hgetall SYSTEM:OBJID_DIRECTORY
43) "PIO_SP470_DO2"
44) "1000007"
```

2. Learn its object type:

```
127.0.0.1:6379> hgetall SYSTEM:OBJTYPE_TBL
..
11) "1000007"
12) "1"
```

3. When writing

- a. to a digital object type, write a single byte "T" or "F"

```
127.0.0.1:6379> hset CMDID:1:OBJID:1000007 VALUE T
127.0.0.1:6379> hset CMDID:1:OBJID:1000007 VALUE F
```

- b. When writing to an VALUE_RAW, supply an integer in text format:

```
hset CMDID:1:OBJID:1000025 VALUE_RAW 5
```

- c. When writing to an VALUE_ENG, supply an floating point number in text format:

```
hset CMDID:1:OBJID:1000025 VALUE_ENG 5.5
```

Note: The CMDID is a feature for providing an ID to lookup the result code of the write. This is implemented for multi-write but is tentatively planned for single writes. For now, (and if not looking up the ID) it's recommended to put "1".

4.7. Bulk Operation LUA Scripts

4.7.1. Bulk Operation Overview

Sometimes single reads and single writes will not suffice. If many objects need to be written to or read from it is more efficient to do bulk reads and bulk writes.

Redis can contain and execute LUA scripts. The multi-read and multi-write implementations rely on Redis LUA scripts.

4.7.2. Redis LUA Script Overview

This overview of Redis Lua scripts provides the foundational knowledge necessary for implementing a bulk read or write to the SCADAPack 47xi object database.

You can load and then execute scripts with a single command.

```
127.0.0.1:6379> EVAL 'local val=_VERSION return val' 0
"Lua 5.1"
127.0.0.1:6379>
```

It takes time to load scripts. Scripts can be permanently loaded but are only accessible by referencing the SHA1 hash value of the script's contents. This prevents malicious modification of the script. Use the **"evalsha"** command to run the script.

```
127.0.0.1:6379> script load 'local val=_VERSION return val'
"d90cb26d8550aa9131bd749fcb9a7f82974d26e1"

127.0.0.1:6379> evalsha d90cb26d8550aa9131bd749fcb9a7f82974d26e1 0
"Lua 5.1"
127.0.0.1:6379>
```

The Syntax for [EVALSHA](#) is:

```
EVALSHA sha1 numkeys keys1 keysN arg1 arg2
```

There is a subtle difference between keys and arguments.

The EdgeSync service on the SCADAPack 47xi pushes the contents of two scripts to strings with these names:

- SYSTEM:FUNCTIONS:MULTIREAD
- SYSTEM:FUNCTIONS:MULTIWRITE

It then loads these strings as scripts.

The contents of those strings can be seen with this syntax.

```
127.0.0.1:6379> get SYSTEM:FUNCTIONS:MULTIREAD
```

The goal is not to see the strings but to execute the scripts associated with them. Remember that scripts can only be executed by referencing the SHA1 hash of their contents. EdgeSync pushes into Redis a script for calling the two scripts describe above. The difference is that unlike the scripts described above, updates to the contents of the caller script are not anticipated.

This is the content of the caller script. It gets the value of the string and calls it as a function.

```
127.0.0.1:6379> script load "local f=loadstring(redis.call('get','SYSTEM:FUNCTIONS:' .. KEYS[1]));
return f()"
"bb6ae216a17ca2c49533ec0d97467714d3aea0dc"
```

4.7.3. SCADAPack 47xi Multiread Redis Lua Script

Input Arguments

Arg	Type	Desc	Notes
1	String	SHA1 Hash of Caller	bb6ae216a17ca2c49533ec0d97467714d3aea0dc
2	String	KeyCount	3+ <# of Objects to Read> For example, supply "6" if reading 3 objects. The Number of Objects to Read. Note: The 3 is from Direction, CMDID, AnalogField
3	String	Direction	MULTIREAD
4	String	CMDID	This is a number in the form of a string that can be used to read the result code. Example "123"
5	String	AnalogField	RAW or ENG Depending on if you want to Read VALUE_RAW or VALUE_ENG
6..n	String(s)	OBJ ID's	Each Object ID in String format as a separate argument.

Each string described in this table is a purely human readable string.

Output

The output is an array of strings.

Each string (each element of this array) will contain 4 bytes.

- If "RAW" was specified, each element is a little-endian integer
- If "ENG" was selected, each element is a little-endian IEC 741 Float

Result Code

The result code will be 0 for success or -1 for failure.

You can read the result code like this. Keep in mind that BLPOP is a blocking command. The last argument is the timeout for this (in seconds). You **NEED TO** do this to help prevent the result codes from building up in memory.

```
127.0.0.1:6379> BLPOP SYSTEM:MREADID:123:STATUS 0.001
```

```
1) "SYSTEM:MREADID:123:STATUS"
```

```
2) "0"
```

Example

This is an example of performing a multi-read, checking, and deleting the result code.

```
127.0.0.1:6379> evalsha bb6ae216a17ca2c49533ec0d97467714d3aea0dc 6 MULTIREAD 1 RAW 1000025  
1000026 1000027
```

1) ",\x01\x00\x00"

2) "\x90\x01\x00\x00"

3) "\xf4\x01\x00\x00"

```
127.0.0.1:6379> BLPOP SYSTEM:MREADID:1:STATUS 1
```

1) "SYSTEM:MREADID:1:STATUS"

2) "0"

4.7.4. SCADAPack 47xi Multi-Write Redis Lua Script

Input Arguments

Arg	Type	Desc	Notes
1	String	SHA1 Hash of Caller	bb6ae216a17ca2c49533ec0d97467714d3aea0dc
2	String	KeyCount	Put "2" (The script is designed such that Argument 1&2 are Keys but 3..7 are arguments.)
3	String	Direction	MULTIWRITE
4	String	CMDID	This is a number in the form of a string that can be used to read the result code. Example "123"
5	String	AnalogField	RAW or ENG Depending on if you want to Read VALUE_RAW or VALUE_ENG
6	byte array	OBJ ID's	Bytes 0...3 is the OBJ ID of the first Point. Bytes 4...7 is the OBJ ID of the 2 nd point This continues until every object ID has been supplied
7	byte array	New Values	Bytes 0...3 is the New Value of the first Point. Bytes 4...7 is the New Value of the 2 nd point This continues until every value has been supplied If AnalogField is VALUE_RAW these bytes need to be integer format.

			If AnalogField is VALUE_ENG these bytes need to be in Little Endian IEC 741 floating point format.
--	--	--	--

Output

The return value is a normally an empty array and should be discarded.

Result Code

The Multi-Read LUA script will do this to provide a result code:

```
BLPOP SYSTEM:MWRITEID:123:STATUS 1
1) "SYSTEM:MWRITEID:123:STATUS"
```

Example

This is an example of a multi-write followed immediately by a multi-read.

```
127.0.0.1:6379> evalsha bb6ae216a17ca2c49533ec0d97467714d3aea0dc 2 MULTIWRITE 1 RAW
"YB\x0f\x00ZB\x0f\x00" 00020003
```

```
127.0.0.1:6379> evalsha bb6ae216a17ca2c49533ec0d97467714d3aea0dc 5 MULTIREAD 1 RAW 1000025 1000026
```

```
1) "0002"
```

```
2) "0003"
```

```
127.0.0.1:6379>
```

Special Notes

Due to the peculiar binary input and output format, it may be cumbersome to do multi-write operations manually from the redis-cli. See the table in section Understanding the Redis-CLI format for Binary Values.

5. Appendix A – Docker Reference

5.1. Introduction

This section provides general guidance for using docker. Docker is a great way to gain access to the almost limitless world of Ubuntu, PIP, or NPM package repositories. To learn more about Docker, you are encouraged to review other docker resources available on the internet.

There are several ways docker containers can be created in the order of most simple to most complex. Method 1&2 require that the SCADAPack 47xi have internet access. Containers can be created manually from the SCADAPack 47xi command line.

1. You can create a dockerfile to create containers and can execute this dockerfile on a SCADAPack 47xi.

2. You can run the dockerfile in the SDK by using QEMU. This will create a docker image that can be loaded onto the SCADAPack 47xi. This technique does not require that the SCADAPack 47xi ever have internet access.
3. You can follow the example of the Ubuntu Python or Node-RED application note and use QEMU to execute a cross platform dockerfile from the SDK. You can then load the resulting image onto a SCADAPack 47xi even if it does not have internet access.

5.2. Simple Test of Docker Installation

This command should give you a comforting response.

```
sudo docker run hello-world
```

5.3. Example Commands

These are other examples of docker commands. Think of this as a “cheat sheet” that can be used on the SCADAPack 47xi in concert with other [Docker Documentation](#).

Creating Containers	
Command	Purpose
<code>sudo docker run hello-world</code>	Use this command to show that docker is working. https://docs.docker.com/engine/reference/commandline/run run: Creates new containers then starts them. A stopped container can be restarted with all its previous changes intact using docker start
<code>docker run --name h1 hello-world</code> <code>docker run --name h2 hello-world</code>	Run multiple instances of hello-world digests from dockerhub. Give each their own name. If you don't specify a name, docker will make one up for you.
<code>sudo docker run -it ubuntu</code>	Runs Ubuntu interactively and in the terminal (-it) It will not have a persistence through reboots.
Volumes and Devices	
Command	Purpose
<code>-v /path/on/host:/path/inside/container</code>	https://stackoverflow.com/questions/55104551/how-to-access-files-in-host-from-a-docker-container
<code>docker run --privileged -it -v /home/root/Ubuntu:/data --name Ubuntu ubuntu</code>	To make the containers contents persistent through reboots you need to map an external path to the containers “/data”. This is called mapping volumes.

<code>--privileged</code>	Use this flag when instantiating a container if you get complaints that updating from such a repository can't be done securely. May only occur when running <code>apt-get update</code> on a RPI ubuntu container.
<code>docker run -it -v Ubuntu3_data:/data --name Ubuntu3 ubuntu</code>	You can also use docker volumes to provide persistence. This example creates container Ubuntu3 with persistence stored in docker volume "ubuntu3_data" If you docker container rm Ubuntu3 you need to then docker volume rm Ubuntu3_data to delete the volume
<code>-v /path/on/host:/path/inside/container</code>	You can map additional folders into the container with additional -v arguments. https://stackoverflow.com/questions/55104551/how-to-access-files-in-host-from-a-docker-container
<code>docker run -it --device /dev/net/tun:/dev/net/tun -p 4322:4322 -p 80:80 -p 443:443 -v /home/root/Ubuntu3:/data -v /home/root:/media/homeroot --name Ubuntu3 ubuntu</code>	Passing devices into the container is similar to passing volumes. This passes /dev/net/tun device into container
<code>docker volume ls</code>	List all Volumes.
Networking	
Command	Purpose
<code>docker run -it -p 1880:1880 -v node_red_data:/data --name Node-RED-1 nodered/node-red</code>	https://nodered.org/docs/getting-started/docker docker run: Run this container, initially building locally if necessary -it: Attach a terminal session so we can see what is going on -p 1880:1880: Connect local port 1880 to the exposed internal port 1880 -v node_red_data:/data: Mount a docker named volume called node_red_data to the container /data directory so any changes made to flows are persisted --name Node-RED-1: Give this machine a friendly local name nodered/node-red: The image to base it on - currently Node-RED v1.2.0
<code>-p 8080:80</code>	Map TCP port 80 in the container to port 8080 on the Docker host. https://docs.docker.com/config/containers/container-networking/

<code>docker run -p <host_port1>:<container_port1> -p <host_port2>:<container_port2></code>	To expose multiple ports, provide multiple <code>-p</code> arguments:
<code>--net=host</code>	Enable host networking This causes the container to listen on any host port that a process inside the container binds to.
<code>docker run -it -p 1880:1880 -v /home/root/Node-RED-1:/data --name Node-RED-1 nodered/node-red:1.2.0-10-arm32v6</code>	Each tag in dockerhub is all lowercase and they sometimes have sub versions specified like this. :1.2.0-10-arm32v6 specifies the "tag" which is the particular version of this container that you want. https://docs.docker.com/engine/reference/commandline/pull/
<code>docker run -it --name Debian2 debian:stretch- 20210816@sha256:62d816b3ad9fc08a0883c0789a2d4 bf241722ad989110a6ddf68c6fef9bee6b4</code>	You can also specify the tag by the SHA hash on docker hub
<code>docker pull debian:stretch- 20210816@sha256:62d816b3ad9fc08a0883c0789a2d4 bf241722ad989110a6ddf68c6fef9bee6b4</code>	If you want to pull it but don't want to create a container yet, use <code>docker pull</code> . This will download and cache it.
Management	
Command	Purpose
<code>docker start Node-RED-1</code>	Start a previously created container.
<code>docker stop Node-RED-1</code>	Stop a container. Note: NodeRED.service acts as a watchdog to automatically restart the Node-RED-1 container when it shuts down for any reason. Stop the NodeRED.service before shutting down container Node-RED-1.
<code>docker start -i Ubuntu1</code>	Start the container in the foreground.
<code>docker attach Node-RED-1</code>	Reattach to the terminal (to see logging) run. This cannot be used to attach a second terminal into an existing container.
<code>docker exec -it Ubuntu1 /bin/bash</code>	To start a second terminal into an existing container, use <code>docker exec</code> to start a second <code>/bin/bash</code> terminal.
<code>docker exec -it --user=root Node-RED-1 /bin/bash</code>	Starts a terminal with root privileges.
<code>Ctrl + c to exit process then Ctrl + d (or type "exit")</code>	Stops and Exits container
<code>Ctrl + p then Ctrl + q</code>	Daemonize container Keep it running. Exit interactive mode The same as starting with <code>-d</code> .

Making Changes	
Command	Purpose
<code>docker inspect Debian1</code>	Displays information about container
<code>docker container ls</code>	Lists all running containers
<code>Docker container ls -a</code>	Lists all running or stopped containers.
Deleting	
Command	Purpose
<code>docker rm -v Node-RED-1</code>	Removes container named Node-RED-1 https://docs.docker.com/engine/reference/commandline/rm/
<code>docker system prune --force</code>	prune: Removes • all stopped containers • all networks not used by at least one container • all dangling images • all build cache
<code>docker system prune -a</code>	Same as above but may provide better results.

5.4. Saving and Loading Containers

Note: This does not save any volumes, that needs to be done separately.

If you have mapped a folder into the container, use these commands to save their contents.

```
tar -zcpvf ./Volume.tgz /home/root/Volume
Unzip
tar -xpvf ./Volume.tgz
```

Save

```
docker commit Node-RED-1 nodered1
rm /home/root/NodeRed/nodered1.tar
docker save nodered1 > /home/root/NodeRed/nodered1.tar
```

Load

```
docker load -i /home/root/NodeRed/nodered1.tar
docker run -t -p 1880:1880 -v $HOME/Node-RED-1:/data --name=Node-RED-1 nodered1 &
```

5.5. Editing/Inspecting Existing Containers

Method 1

Certain things can be changed through the `docker update` command. You can search the docker documentation and other resources online to see what can be changed with this technique.

<code>docker update --restart=no Ubuntu3</code>	This changes an existing container
---	------------------------------------

Method 2

Container settings that cannot be changed with Method 1 can typically be changed by editing the container's `hostconfig.json` with a procedure similar to the following.

1. From the output of the following commands, you will learn the ID of the container.

```
docker inspect CONTAINERName  
docker inspect Ubuntu1
```

2. Use the command, `CD`, to access the docker container's directory.

```
cd /media/sdcard/docker/data/containers/
```

3. Then use `CD` again to access the specific directory for the ID of the container that you learned in step 1.

```
cd CONTAINER ID
```

4. Use a text editor such `VI` (or `Notepad++` with the `NppFTP` and `JSON Viewer` plugins) to make the following modification:

```
nano ./hostconfig.json
```

5.6. Enabling Linux Commands from an Ubuntu Container

These containers have almost nothing to start with.

Before the dockerized `apt-get` can work, run the following command:

```
apt-get update
```

Then, use the following commands to add support for these programs.

Command	Prerequisite for running the command on a container.
<code>apt-get update</code>	apt-get will not work until first running this inside the container if you get errors regarding invalid signatures See this site: https://askubuntu.com/questions/1263284/apt-update-throws-signature-error-in-ubuntu-20-04-container-on-arm
<code>ifconfig</code>	<code>apt-get install net-tools</code>

Command	Prerequisite for running the command on a container.
ping	apt-get install iputils-ping
nano	apt-get install nano
cat	apt-get install coreutils
scp	openssh-client
wget	apt-get install wget
ftp	apt-get install ftp apt-get update && apt-get install -y --no-install-recommends ntp

5.7. Dockerfile

A dockerfile can be used to rapidly create containers that contain certain programs or things already installed. They specify where a container comes from and what commands to run inside the container to finish building the container.

There is documentation about what dockerfiles are and how to use them here:

<https://docs.docker.com/engine/reference/builder/>.

This application note provides a working example how to run a dockerfile on the SCADAPack 47xi.

5.7.1. Executing a Dockerfile Locally on the SCADAPack 47xi

To do this, your SCADAPack 47xi needs to have internet access. See SCADAPack 47xi Internet Access.

Use VI or Notepad++ to create a Dockerfile on the SCADAPack 47xi.

/home/root/Dockerfile
<pre>FROM --platform=armhf ubuntu:latest RUN apt-get update RUN apt-get install -y iputils-ping RUN apt-get install -y nano</pre>

Once the dockerfile is created, do the following.

1. Running the following command from the directory with the dockerfile shown above, will build the container and store it with a custom tag name. It does not instantiate the image. You do not need to explicitly specify a filename since the default filename "Dockerfile" was created in the previous step.

```
docker build -t greatauk .
```

2. Use the following command to see your container image:

```
docker images
```

3. Use the following command to create a container:

```
docker run -it --name=Ubuntu3 greatauk
```

5.7.2. Executing a Dockerfile from the SCADAPack 47xi SDK

You may want to create containers that can be deployed on SCADAPacks that do not have internet access. You can use QEMU which requires that the SCADAPack 47xi SDK be installed.

By reviewing or modifying the Node-RED or Ubuntu Python application notes and the scripts that those application notes contain, an advanced user will be able to adopt the same technique for using the SDK to package other types of containers for remote deployment on a SCADAPack 47xi.