

Manual | EN

TwinCAT/BSD

Operating system

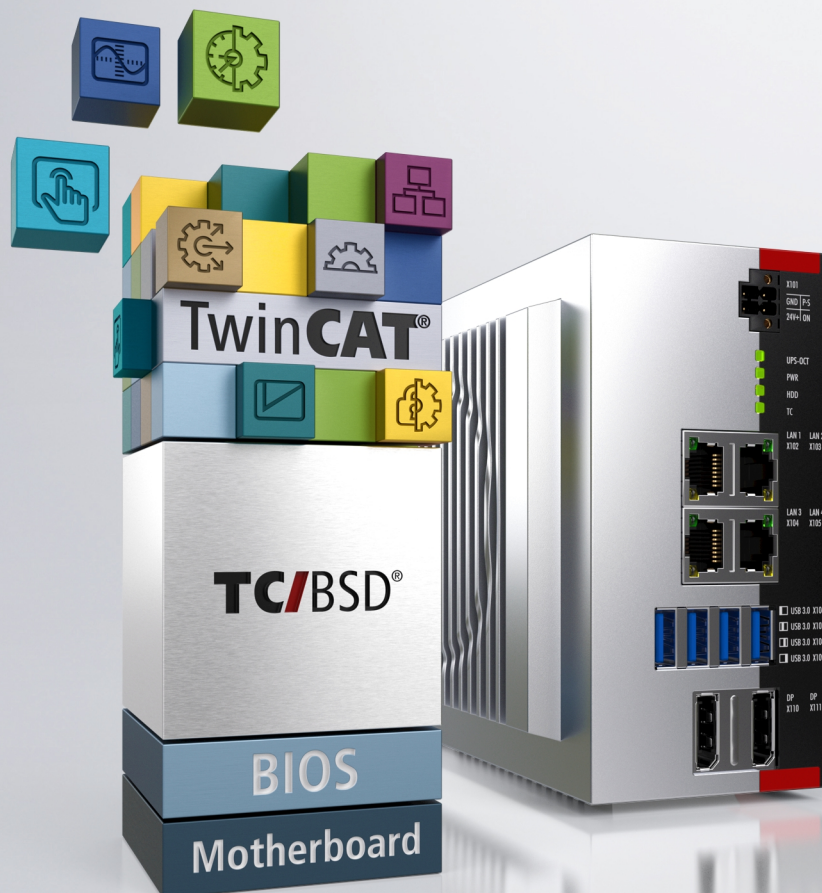


Table of contents

1	Notes on the documentation.....	7
1.1	Symbol explanation	7
1.2	Documentation issue status	8
2	First steps	9
2.1	First system startup	9
2.2	Determining the IP address	9
2.3	Access via SSH, web-based console or web interface	11
2.4	Changing the standard password	13
3	Beckhoff TwinCAT/BSD installer	14
3.1	Partitions and folder structure	15
3.2	Setup and installation	16
3.2.1	Create bootable USB stick	17
3.2.2	Check BIOS settings	18
3.2.3	Installing TwinCAT/BSD	18
3.3	Remote access to the TwinCAT/BSD installer via SSH	19
3.4	Automation of processes - using scripts in practice	19
4	TwinCAT/BSD	21
4.1	Credentials	22
4.2	ZFS properties	23
4.3	Directory structure	25
4.4	Write filter	26
4.4.1	Enabling or disabling the write filter	26
4.4.2	Defining exceptions	26
4.5	Text Editors	26
5	Network settings	28
5.1	Settings for systems with dhcpcd	29
5.1.1	Post-install and activate dhcpcd	29
5.1.2	Set IP address (dhcpcd)	29
5.2	Set IP address for systems with dhclient	31
5.3	Changing the host name	32
5.4	Firewall	32
5.4.1	Enable and disable firewall	33
5.4.2	Enable port	33
5.5	WLAN configuration	34
5.5.1	Connecting to WLAN	34
5.5.2	Configuring as access point	35
5.5.3	Setting up a DHCP server	36
5.6	Establish connection with Beckhoff LTE stick	37
5.7	HTTPS certificates	39
5.7.1	Disabling automatic certificate creation	39
5.7.2	Requesting or creating an HTTPS certificate	39
5.7.3	Importing the certificate	40
6	System update	41

6.1	Update TwinCAT/BSD	43
6.2	Update major version	43
6.3	Update CPU microcode	44
7	Package Server	46
7.1	Switching to a Chinese server	46
7.2	Switching to the FreeBSD repository	47
7.3	Package management	47
7.3.1	Search	48
7.3.2	Install	48
7.3.3	Update	49
7.3.4	Uninstall	49
7.3.5	Lock	50
7.4	Set up a local repository	50
7.4.1	Provide repository on USB stick	50
7.4.2	Setting up your own Package Server	52
8	Configuration	54
8.1	System information	54
8.2	User and rights management	55
8.2.1	Create new user	55
8.2.2	Edit user information	57
8.2.3	Deleting a user	58
8.3	Integrating a USB stick	59
8.4	Enable Automount for external data carriers	60
8.5	Configuring the UPS software	61
8.6	Disable real-time Ethernet	61
8.7	Start services automatically (Autostart)	63
8.8	Changing the shell	63
8.9	Change keyboard language	63
8.10	Synchronize time with NTP	64
9	Remote access	65
9.1	Beckhoff Device Manager: web interface	65
9.2	Remote access with SSH	67
9.3	Managing files with the WinSCP client	69
9.3.1	Starting and using the WinSCP client	69
9.3.2	WinSCP as the root	70
9.3.3	Opening and editing files	71
9.4	Editing the SSH settings	72
10	TwinCAT/BSD Hypervisor	73
10.1	Device and feature support	73
10.2	Start and manage virtual machines	73
10.3	Use shell scripts	76
10.4	Autostart shell scripts	78
10.5	Advanced VM configuration	78
10.5.1	ZFS data sets as storage location for virtual machines	78
10.5.2	UEFI-based virtual machines	79

10.5.3	VNC-based interaction with virtual machines.....	79
10.5.4	Virtual drives	81
10.5.5	Virtual machine network configuration	83
10.5.6	PCI device passthrough	90
10.5.7	Pass-through of input devices	92
10.6	Installing Debian Linux as a guest operating system	93
10.7	Windows as guest operating system.....	98
10.7.1	C9900-S620: Windows 10 pre-installed with Device Passthrough	99
10.7.2	C9900-S621: Windows 10 pre-installed without Device Passthrough	100
10.7.3	Subsequent installation of Windows VM with vm-installer	101
10.7.4	Adapting the Windows VM configuration	102
11	C/C++ projects for TwinCAT/BSD	103
11.1	Compiling under TwinCAT/BSD	103
11.2	Remote debugging of applications with Visual Studio Code (VSC)	103
12	TwinCAT 3.....	107
12.1	Searching for target systems	107
12.2	Scan devices	109
12.3	Changing the AMS NetID	110
12.4	Put TwinCAT into Run or Config mode	110
12.5	Create or delete ADS routes manually	111
12.6	Increase heap memory	112
12.7	Adapting the router memory.....	113
12.8	Assign isolated cores	114
13	Restore options	115
13.1	Restore point.....	115
13.1.1	Resetting to factory settings.....	116
13.1.2	Creating a restore point.....	116
13.1.3	Resetting to the restore point	117
13.1.4	Using the restore boot environment	117
13.2	Backup and restore	119
13.2.1	Creating a backup	119
13.2.2	Restoring a backup	119
13.2.3	Creating and restoring a backup from the live system	120
14	Error handling and diagnostics	121
14.1	Using kernel messages for diagnosis	121
14.2	Log files	121
14.3	Dumps	122
14.3.1	Using automatic process dump	122
14.3.2	Creating a process dump manually.....	122
14.3.3	Provide system information for analysis.....	122
14.4	Using the ADS monitor.....	123
14.5	Analyzing network traffic with Wireshark.....	124
14.6	System repair	125
14.6.1	Bootting from the USB installer stick.....	125
14.6.2	Start single-user mode	126

15 Appendix	127
15.1 Important commands	127
15.1.1 TwinCAT	127
15.1.2 Shell	127
15.1.3 File and directory management.....	129
15.1.4 System administration.....	131
15.1.5 Important files and directories.....	133
15.1.6 Text Editors	134
15.1.7 Documentation	135
15.2 Bibliography	136
15.3 FreeBSD Copyright	136
15.4 Support and Service.....	137
List of tables	138
List of figures.....	139

1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.

The documentation and the following notes and explanations must be complied with when installing and commissioning the components.

The trained specialists must always use the current valid documentation.

The trained specialists must ensure that the application and use of the products described is in line with all safety requirements, including all relevant laws, regulations, guidelines, and standards.

Disclaimer

The documentation has been compiled with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without notice.

Claims to modify products that have already been supplied may not be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar®, and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

If third parties make use of the designations or trademarks contained in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.



EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The distribution and reproduction of this document, as well as the use and communication of its contents without express authorization, are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event that a patent, utility model, or design are registered.

Third-party trademarks

Trademarks of third parties may be used in this documentation. You can find the trademark notices here: <https://www.beckhoff.com/trademarks>.

1.1 Symbol explanation

The following warnings are used in the documentation. Read and follow the warnings.

Warnings relating to damage to property or the environment:

NOTICE

There is a potential hazard to the environment and equipment.

Notes showing further information or tips:



This notice provides important information that will be of assistance in dealing with the product or software. There is no immediate danger to product, people or environment.

1.2 Documentation issue status

Table 1: Change notes for the documentation.

Version	Changes
1.0	<ul style="list-style-type: none"> First version
1.1	<ul style="list-style-type: none"> Chapter TwinCAT/BSD, network settings, Package Server, configuration, TwinCAT 3, recovery options and error handling and diagnostics revised
1.2	<ul style="list-style-type: none"> Chapter System update, System information, Beckhoff Device Manager and Synchronize time with NTP added. Chapter TwinCAT revised.
1.3	<ul style="list-style-type: none"> Chapter TwinCAT/BSD Hypervisor added.
1.4	<ul style="list-style-type: none"> Chapter "Remote debugging of applications with Visual Studio Code (VSC)" added.
1.5	<ul style="list-style-type: none"> Chapter "System update" revised. Chapter "Disable IPv6" removed Chapter "Network settings" extended by information about the new dhcpcd client. Chapter "Starting services automatically" adapted. Chapter "Remote debugging of applications with Visual Studio Code (VSC)" revised
1.6	<ul style="list-style-type: none"> Chapters "WLAN configuration" and "TwinCAT/BSD Hypervisor" revised. Chapters "First steps", "Beckhoff TwinCAT/BSD Installer" and "Update CPU microcode" added.
1.7	<ul style="list-style-type: none"> Chapter "Setting up your own Package Server" added.
1.8	<ul style="list-style-type: none"> Chapter "Enable Automount for external data carriers" and "Configuring the UPS software" added. Change of the home directory from TwinCAT/BSD version 13 to 14. All data adapted from <code>/usr/home</code> on <code>/home</code> Chapters "Backup and restore", "Remote access with SSH", and "Set up local repository" adapted.
1.9	<ul style="list-style-type: none"> Chapter "TwinCAT/BSD Hypervisor" revised.
2.0	<ul style="list-style-type: none"> Chapter "HTTPS certificates" added.

Table 2: Change notes for the operating system.

Repository version	TwinCAT/BSD version	Changes
89449	13.2.0.6	<ul style="list-style-type: none"> dhcpcd(8) is delivered as the standard DHCP client and replaces the older client
126815	14.0.4.1	<ul style="list-style-type: none"> Home directory is now located directly under <code>/home</code> (no more symlink) and no longer under <code>/usr/home</code> Automount is supported UPS software is available Wi-Fi 6 is supported

2 First steps

The following chapter describes the basic first steps with TwinCAT/BSD. It is assumed that an industrial PC with pre-installed TwinCAT/BSD operating system has been purchased from Beckhoff. The presence of such an installation can be identified by the TwinCAT/BSD license sticker on the device.

2.1 First system startup

Connect all network cables to the industrial PC and switch on the power supply. If you have connected a monitor to the device, you will see the BIOS boot screen and then the TwinCAT/BSD bootloader.

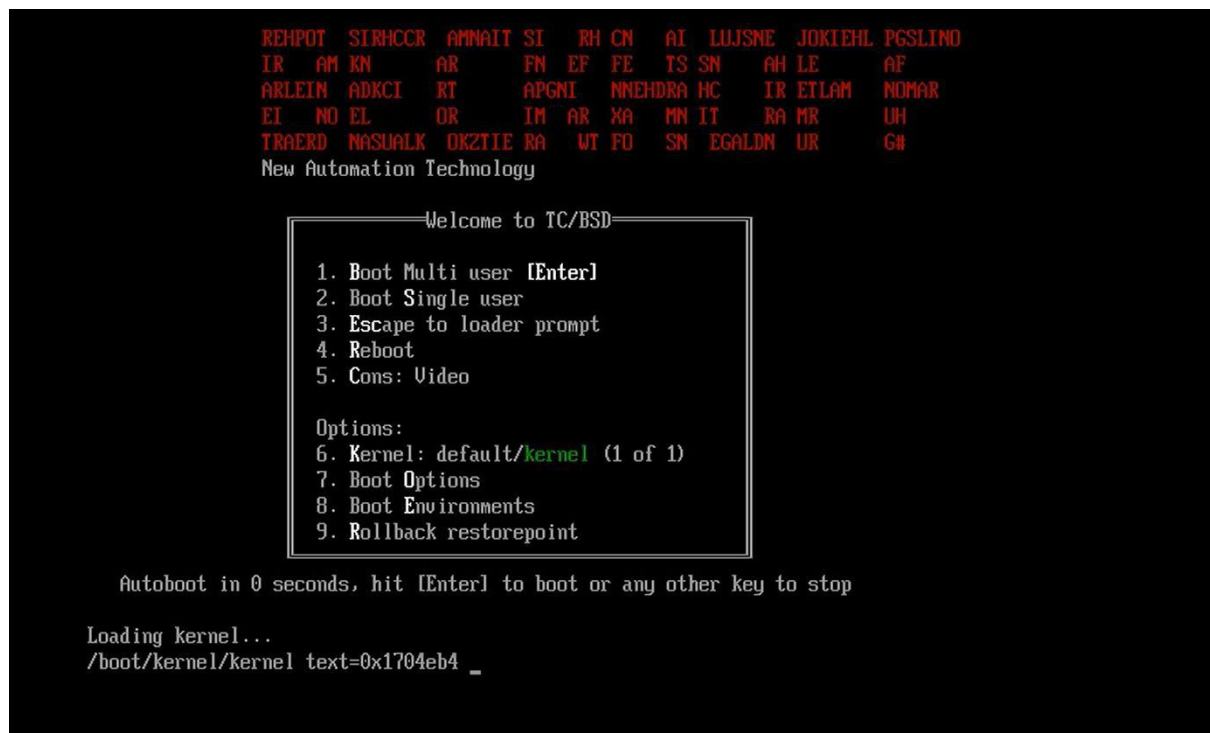


Fig. 1: TwinCAT/BSD bootloader during startup.

Then, during the boot phase of TwinCAT/BSD, various messages appear, ending with the login prompt. Log in to the console with user "Administrator" and password "1". The password should be changed immediately after logging in (see: [Changing the standard password \[► 131\]](#)).

```

login: Administrator
Password:
Last login: Thu Sep 14 13:53:41 on ttyv0
FreeBSD 13.2-RELEASE-p1 n254639-8330e4d7c03- BHF

The software licenses can be found in this folder: /usr/local/etc/TwinCAT/3.1/System/Legal/
TcOsSys.dll:      4026.1.20
TwinCAT Build:    3.1.4026.0
AMS Net Id:       5.61.105.18.1.1
TC/BSD:           13.2.1.1,2

```

If you do not have a monitor connected, the easiest way to see if the device has booted properly is to see a TwinCAT LED that lights up blue.

2.2 Determining the IP address

For all subsequent work with the industrial PC and the system, the IP address of the system is required. Using the IP address, remote access via SSH, the Beckhoff Device Manager (web interface) or working with TwinCAT is possible, for example.

Determine IP address with monitor

There are several ways to determine the IP address to access the device via the network. If you have connected a local monitor, you can log in with the user "Administrator" and the password "1".

Enter the command `ifconfig` in the console to output all available Ethernet interfaces in the system:

```
Administrator@CX-3D6912:~ $ ifconfig
igb0: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    options=4a024a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,WOL_MAGIC,RXCSUM_IPV6,NOMAP>
    ether 00:01:05:3d:69:12
    inet6 fe80::78ff:f9f9:ef31:454d%igb0 prefixlen 64 scopeid 0x1
    inet 169.254.228.5 netmask 0xffff0000 broadcast 169.254.255.255
    media: Ethernet autoselect
    status: no carrier
    nd6 options=1<PERFORMNUD>
igb1: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    options=4a024a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,WOL_MAGIC,RXCSUM_IPV6,NOMAP>
    ether 00:01:05:3d:69:13
    inet6 fe80::d2c1:31fd:a18e:elc3%igb1 prefixlen 64 scopeid 0x2
    inet 172.17.40.26 netmask 0xfffffc00 broadcast 172.17.43.255
    media: Ethernet autoselect (100baseTX <full-duplex>)
    status: active
    nd6 options=1<PERFORMNUD>
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
    options=680003<RXCSUM,TXCSUM,LINKSTATE,RXCSUM_IPV6,TXCSUM_IPV6>
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
    inet 127.0.0.1 netmask 0xff000000
    groups: lo
    nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
Administrator@CX-3D6912:~ $
```

By default, the Ethernet interfaces are configured to receive an IP address from a local DHCP server. In this sample, this is the IP address: 172.17.40.26 which was assigned for the interface `igb1` via which the industrial PC was connected to the network.

Determine IP address without monitor

Use the Broadcast Search function under TwinCAT if no monitor is connected to determine the IP address of a TwinCAT/BSD target system.

Add Route Dialog

Enter Host Name / IP: Refresh Status Broadcast Search

Host Name	Connected	Address	AMS NetId	TwinCAT	OS Version	Fingerprint
CX-505912		192.168.6.79	5.80.89.16.1.1	3.1.4025	TwinCAT/BSD (13.1)	AFD3D2826B3EE028E2A71E

Route Name (Target): Route Name (Remote):

AmsNetId:

Virtual AmsNetId (NAT):

Transport Type:

Address Info:

☐ Host Name ☒ IP Address

Connection Timeout (s):

Max Fragment Size (kByte):

Target Route: ☐ Project ☒ Static ☐ Temporary

Remote Route: ☐ None / Server ☒ Static ☐ Temporary

☒ Advanced Settings ☐ Unidirectional

Add Route Close

Fig. 2: Add-Route dialog with a TwinCAT/BSD target system and associated IP address.

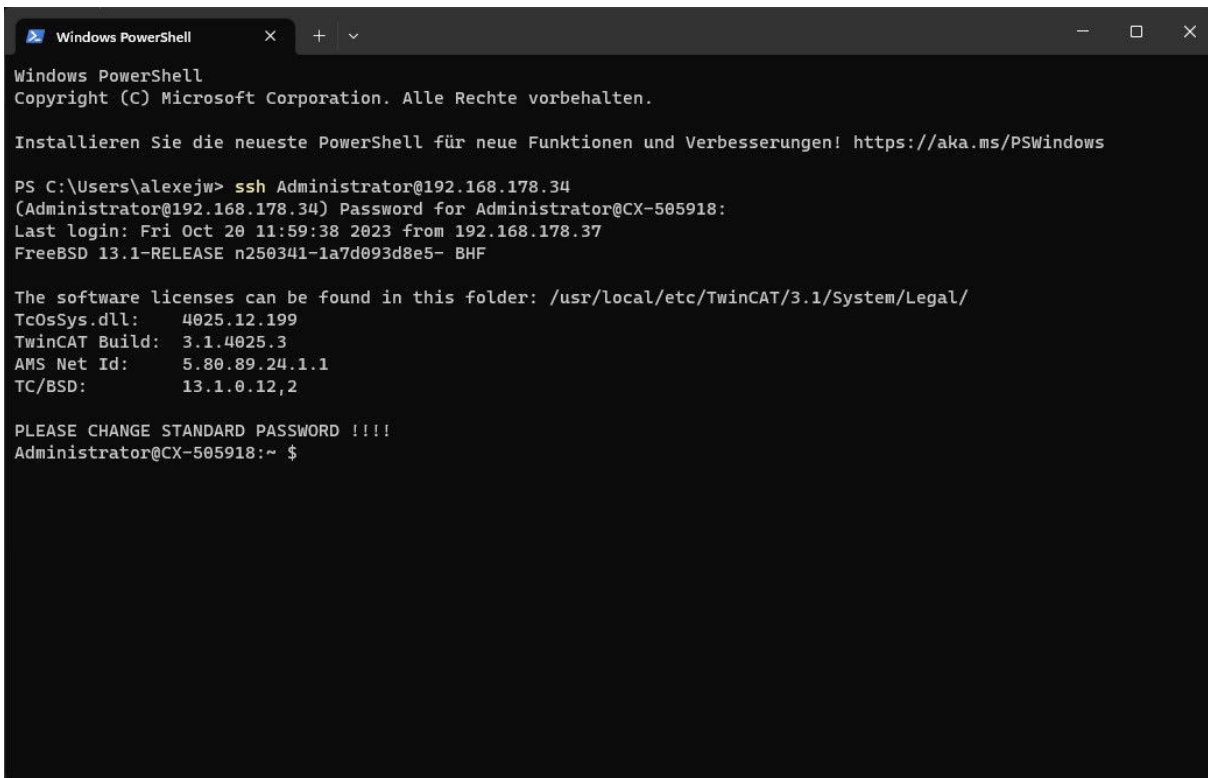
If you have multiple controllers, you can identify them by name or operating system. Note that under TwinCAT/BSD only Secure-ADS is enabled by default. A detailed description of how to use the Broadcast Search function and how to search for a TwinCAT/BSD target system can be found at: [Searching for target systems](#) [107].

If you have a direct connection to your laptop without DHCP, the TwinCAT/BSD device assigns itself an auto-IP address from the range 169.254, and the last two bytes are then derived from the last two bytes of the MAC address (169.254.{MAC5}.{MAC6}) printed on the name plate.

2.3 Access via SSH, web-based console or web interface

SSH access

If you have an SSH client on your laptop, you can use it to connect. For example, in the current Windows 10 versions, this is a Windows function that can be used via PowerShell. To do this, use the command `ssh Administrator@<bsd-ip>`.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Alle Rechte vorbehalten.

Installieren Sie die neueste PowerShell für neue Funktionen und Verbesserungen! https://aka.ms/PSWindows

PS C:\Users\alexejw> ssh Administrator@192.168.178.34
(Administrator@192.168.178.34) Password for Administrator@CX-505918:
Last login: Fri Oct 20 11:59:38 2023 from 192.168.178.37
FreeBSD 13.1-RELEASE n250341-1a7d093d8e5- BHF

The software licenses can be found in this folder: /usr/local/etc/TwinCAT/3.1/System/Legal/
TcOsSys.dll:      4025.12.199
TwinCAT Build:   3.1.4025.3
AMS Net Id:      5.80.89.24.1.1
TC/BSD:          13.1.0.12,2

PLEASE CHANGE STANDARD PASSWORD !!!!
Administrator@CX-505918:~ $
```

Fig. 3: Remote access via SSH using Windows PowerShell.

Web-based console

If you do not have an SSH client, a web-based console is available on the device that you can access with a browser using HTTPS and the IP address of the device. Entering `https://<ip-adress>/console` in the browser will take you to the web-based console.

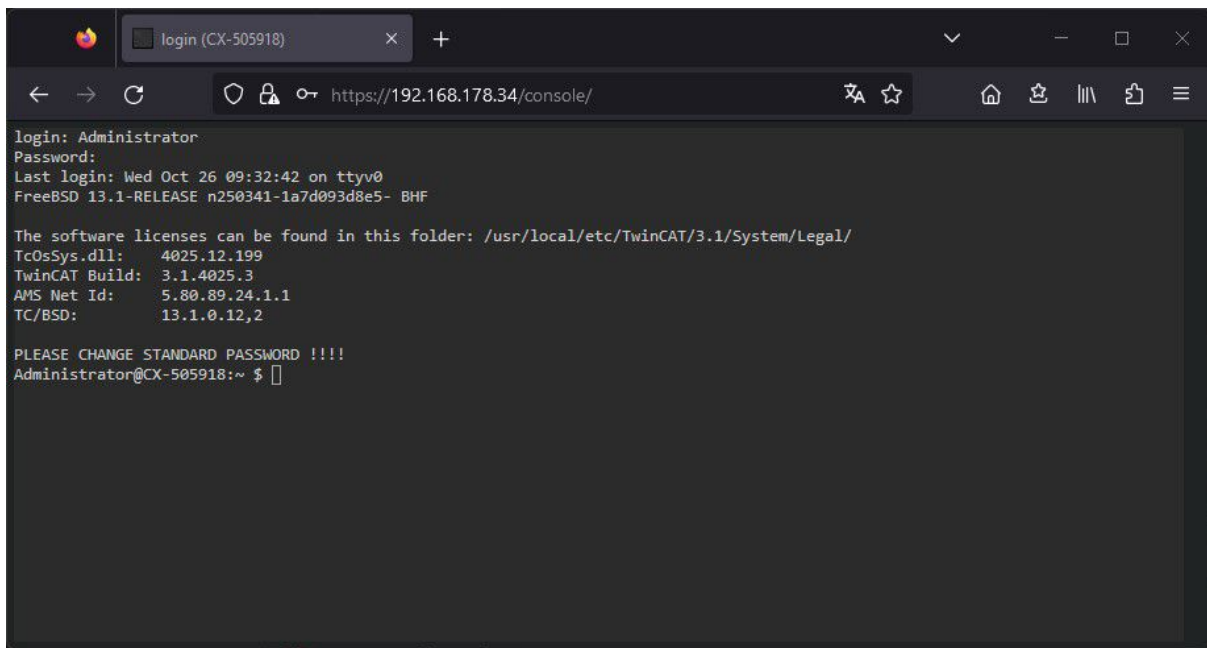


Fig. 4: Access via the web-based console of a TwinCAT/BSD system.

Since the device has a self-signed certificate, the browser displays a warning that it could not verify the certificate. You can simply ignore this and continue. You will now see a console in your web browser

Beckhoff Device Manager: web interface

If you prefer a graphical interface and do not want to use a terminal, you can also access the Beckhoff Device Manager website, which is the easiest way to configure the device. To do this, simply enter the IP address in the browser. For more information, see: [Beckhoff Device Manager: web interface](#) [► 65].

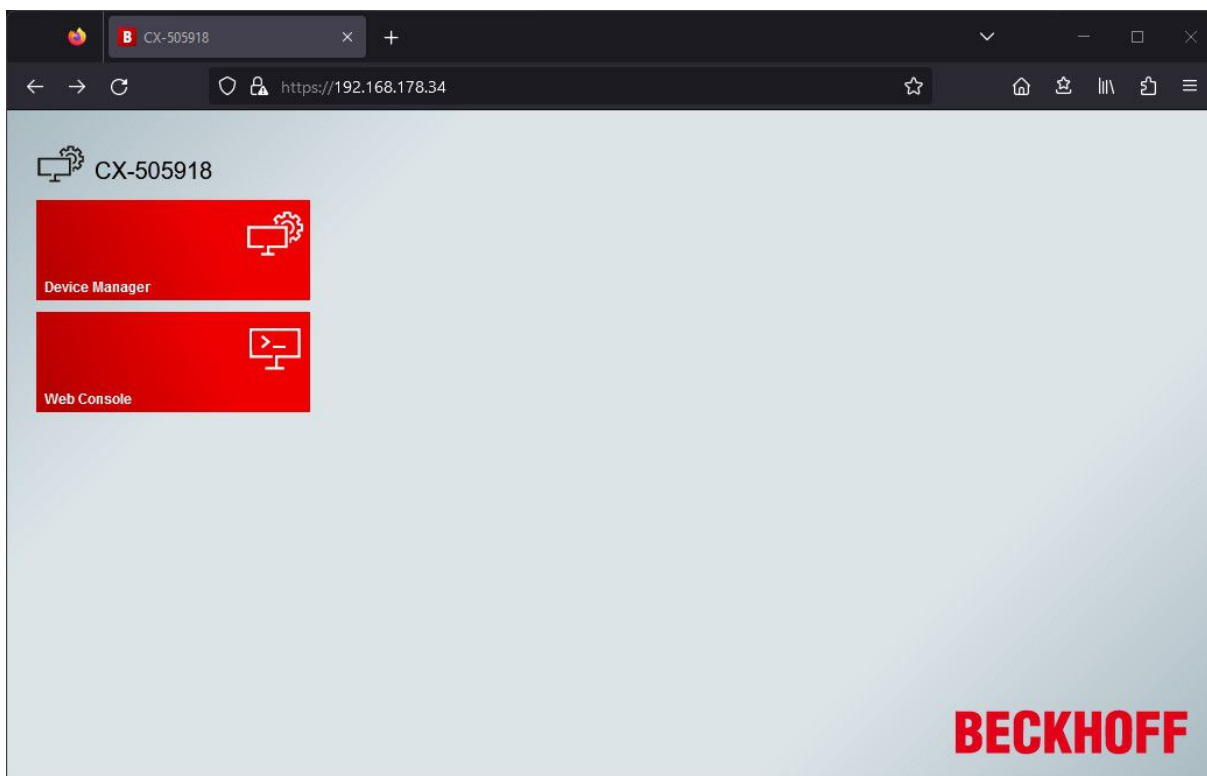


Fig. 5: Access to the device manager of a TwinCAT/BSD system.

2.4 Changing the standard password

Change the password for the administrator account, which should always be one of the first steps when you get a new system. Depending on whether you have connected a monitor, the default password can be changed in the console or in the web interface of the Beckhoff Device Manager.

Change password in the console

You can change the password of the currently logged-in user with the command `passwd`. When TwinCAT/BSD is delivered, a user (Administrator) is available by default, with which you can log in to the console. This user does not have conventional administrator rights like under Windows systems but has the authority to obtain root rights for certain purposes.

1. Start TwinCAT/BSD.
 2. Log in with the user name `Administrator` and the password `1`.
 3. After successful login, the user and the host name of the industrial PC is displayed. For example: `CX-1D7BD4`.
 4. Enter the command `passwd` in order to set a new password for TwinCAT/BSD. Follow the instructions.
- ⇒ You have successfully set a new password for TwinCAT/BSD and the user Administrator.

Change password in the Beckhoff Device Manager

The web interface of the Beckhoff Device Manager can be accessed via a standard web browser. To do this, enter the IP address or the host name of the industrial PC in the search bar. Change the default password at **Security > Login**.

The screenshot shows the Beckhoff Device Manager web interface. On the left is a sidebar with navigation options: Device, Hardware, Software, TwinCAT, and Security (highlighted in red). The main content area is titled 'Login' and 'Change User Passwords'. It contains four input fields: Username, Password, New Password, and New Password (confirm). Below this is the 'User Management' section with an 'Add User' button and three input fields: Username, Password, and Password (confirm). Further down is the 'Add Local Group' section with a 'Groupname' input field. At the bottom is the 'Set Local Group Membership' section with a 'Membership' radio button (set to 'Add'), and 'Username' and 'Groupname' input fields. Each section has a confirmation button (checkmark) and a cancel button (X).

Fig. 6: Change password in the web interface of the Beckhoff Device Manager.

3 Beckhoff TwinCAT/BSD installer

The Beckhoff TwinCAT/BSD installer offers a wide range of functions and use cases that go far beyond the simple installation of the TwinCAT/BSD operating system on an industrial PC. Here are the different options in detail:

1. [Installing TwinCAT/BSD \[► 16\]](#): As already mentioned, the primary function of the TwinCAT/BSD installer is to install the TwinCAT/BSD operating system on an industrial PC.
2. [Backup creation \[► 119\]](#): With the TwinCAT/BSD installer you can create backups to save data and configurations. These backups can be used later to quickly restore the system in the event of a failure or malfunction.
3. [Backup restore \[► 119\]](#): If a problem occurs or a restore is required, the TwinCAT/BSD installer allows restoring previously created backups. This ensures continuity of operation and minimizes downtime.
4. [Starting a shell \[► 125\]](#): The TwinCAT/BSD installer also allows starting a shell. This shell provides access to a full FreeBSD system installed on the USB stick. This is extremely useful for maintenance and diagnostic purposes, as you can access the FreeBSD system directly from the USB stick without affecting the main installation.
5. [Automation of processes with scripts \[► 19\]](#): The TwinCAT/BSD installer enables the saving and execution of scripts for the automation of processes. These scripts provide the flexibility to run custom automation processes to automate recurring tasks and further optimize the installation and configuration process.

In summary, the Beckhoff TwinCAT/BSD installer offers a comprehensive tool for backup, installing, managing and automating TwinCAT/BSD systems on industrial PCs. From initial installation to recovery and the ability to directly manage systems and automate processes.

Download and installation

The TwinCAT/BSD installer, including the current TwinCAT/BSD version, is available for download as a bootable ISO file from the [Beckhoff homepage](#). The ISO file can be used to create a [bootable USB stick \[► 17\]](#) and boot an industrial PC from it to subsequently start a graphical user interface with interactive menu.

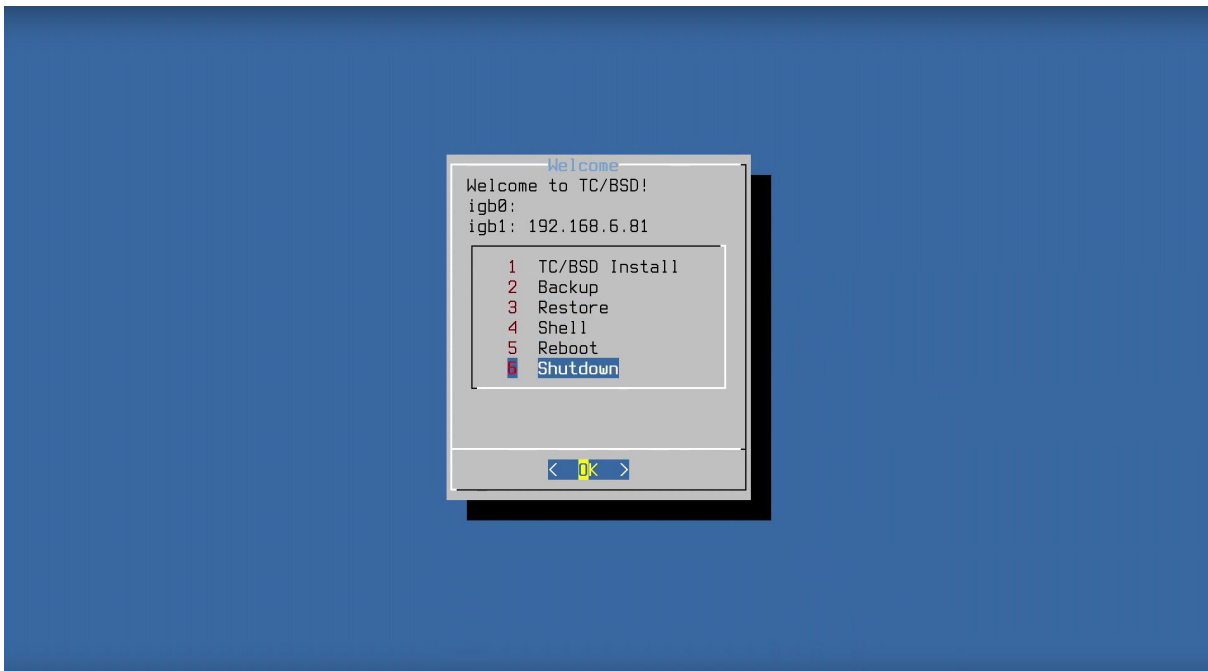


Fig. 7: Beckhoff TwinCAT/BSD installer: Start page with interactive menu.

To do this, use a flash tool such as [Rufus](#) to transfer the bootable ISO file to a USB stick, which first turns the USB stick into a portable TwinCAT/BSD installer.

Software and licensing

The TwinCAT/BSD installation includes a TwinCAT 3 runtime, with the possibility to install further software and TwinCAT 3 functions. This software does not include a separate commercial license and is intended for testing purposes only. Commercial use of the product requires a separate license and license sticker. A TwinCAT/BSD license can be requested from the service for existing devices.

3.1 Partitions and folder structure

There are two partitions on the Beckhoff TwinCAT/BSD installer stick, a FreeBSD partition with a read-only Unix file system (UFS) and a FAT partition for storing backups and scripts that can be used to automate processes. Make sure that the FAT partition is set up only after the USB stick is booted for the first time.

If you insert the USB stick into an industrial PC directly after creation, the FAT partition is initially called BHF-PREOP. Only when you boot from the USB stick for the first time, the FAT partition is renamed to BHF.

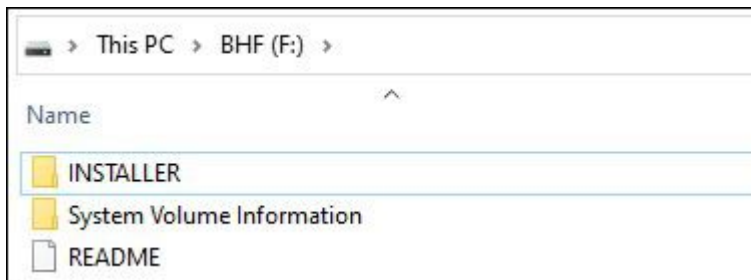


Fig. 8: FAT partition and folder structure of the TwinCAT/BSD installer stick under Windows.

On the FAT partition there are two folders and a readme file with short description of all folders. The installer folder contains folders for SSH keys, logs and everything to automate processes with the TwinCAT/BSD installer stick.

Backups created with the TwinCAT/BSD installer stick are stored in a separate folder with the file extension *.tcbkp00. Since it is a FAT partition, the maximum size of a single file is 4 GB. If the backup exceeds the 4 GB, the backup will be split into several parts of 4 GB.



Fig. 9: Folder structure of the TwinCAT/BSD installer stick with two backup folders.

To copy existing backups, for example from an archive, onto the TwinCAT/BSD installer stick, proceed as follows: Create a folder on the stick with a name of your choice. Save the backup in this folder, using the file extension *.tcbkp00.

Notice Once the TwinCAT/BSD installer stick has been created, it must be booted once from the stick so that the required partitions are created.

The installer folder contains further subfolders, which on the one hand contain sample scripts and on the other hand can be used to save own scripts for automating processes with the TwinCAT/BSD installer stick. For example, scripts can be created to run immediately after installation to customize a freshly installed image to your needs. Or scripts can be used that automatically backup or restore a TwinCAT/BSD system as soon as the TwinCAT/BSD installer stick is plugged in.

The samples listed are not exhaustive and other scenarios for the use of scripts are conceivable. The following table describes in which directory the scripts for each task must be placed in order to function properly.

Table 3: Description of the directories in the installer folder.

Directory	Description
/INSTALLER	Base directory for files that are not backup data.
/INSTALLER/.ssh	Copy your public key here to connect to the TwinCAT/BSD installer via SSH. You must enable the "start_ssh_server" example to connect. The sample is already in the Autorun folder by default.
/INSTALLER/autorun	Scripts stored here are automatically executed in alphabetical order after booting.
/INSTALLER/logs	Here you can find log files of autorun scripts.
/INSTALLER/autorun_samples	Folder with ready samples. All scripts from the sample folder must be copied to the / <i>INSTALLER/autorun/</i> directory to perform a specific task (e.g. start_ssh_server).
/INSTALLER/autorun_samples/start_ssh_server	This sample starts the SSH daemon to enable SSH connections and runs by default.
/INSTALLER/autorun_samples/autoinstall_tcbsd	Sample for the automatic installation of TwinCAT/BSD. The system is installed on the largest available hard disk by default. A specific hard disk can be configured as a parameter in the script. The default password for the installed system is provided as an environment variable and can be customized in the script.
/INSTALLER/autorun_samples/auto_backup	Sample for the automatic backup of a TwinCAT/BSD system. The script searches for a TwinCAT/BSD system by looking for a hard disk that contains a pool "zroot". A backup is generated with a file name from the host name of the system and the current timestamp and stored in the \INSTALLER directory.
/INSTALLER/autorun_samples/auto_restore	Sample for the automatic restore of a TwinCAT/BSD backup. You can select the target hard disk by a parameter in the script. Additionally, you can select a backup file by its name as a pattern (e.g. ".tcbkp" for each backup or "backup1.tcbkp00" for a specific backup). By default, available backups are listed alphabetically by their file extension. The first one is selected for recovery and restored to the largest available hard disk if no disk is specified.
/INSTALLER/post_install.d	This folder contains scripts designed to modify the installed system. The scripts must have one of the following names: <ul style="list-style-type: none"> • *.installer: Scripts ending in ".installer" are executed within the TwinCAT/BSD installer. The path to the folder where the installed system is mounted is provided as argument \$1. This is useful if you want to copy files to the system, for example, since in this case you access the file system of the TwinCAT/BSD installer. • *.chroot: Scripts that end in ".chroot" are executed within the scope of the installed system. This provides a more convenient method for modifying user accounts or installing packages.
/INSTALLER/runonce.d	The contents of this folder are copied to the "runonce" folder in the installed system. Scripts in the Runonce folder are executed once and then disabled by moving them to either "runonce.d/succeeded" or "runonce.d/failed". This is helpful if the industrial PC has to be booted once, for example if certain kernel modules are required that can only be loaded after booting.
/autorun_preop.sh	Executed during the first boot of the TwinCAT/BSD installer to make device-specific adjustments.

3.2 Setup and installation

This chapter provides detailed instructions for installing TwinCAT/BSD on an industrial PC. The first section explains the steps to create a bootable USB stick using the Rufus tool, including the necessary prerequisites. The second section describes the necessary steps to configure the BIOS settings to ensure that the industrial PC can boot from this USB stick. Finally, the third section covers the entire installation process.

3.2.1 Create bootable USB stick

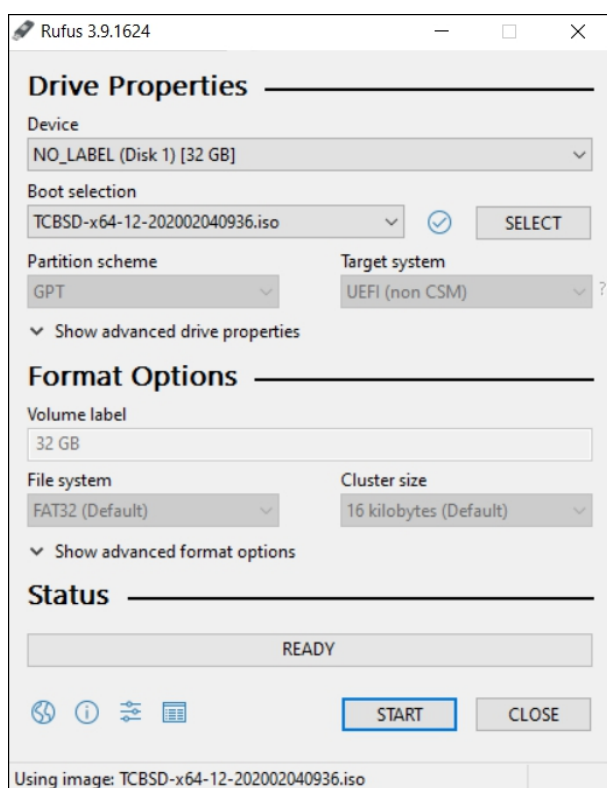
Before you can install TwinCAT/BSD on a Industrial PC, you must create a bootable USB stick and upload the current image to the USB stick. Use a flash tool such as Rufus to do this. You can then start Industrial PC from the USB stick and install TwinCAT/BSD.

Requirements for this step:

- Download the Rufus tool from <https://rufus.ie/>
Notice : Note that newer versions of Rufus may not be compatible with hard disk encryption tools. Rufus 3.13 is therefore recommended.
- [Bootable ISO file](#) for TwinCAT/BSD installation
- USB stick with at least 2 GB storage space.

Proceed as follows:

1. Start the Rufus tool on a PC with Windows operating system.
2. Click **Select** and select the image you want to upload to the USB stick.



3. Under **Device**, select a USB stick as the target drive. If only one external drive is connected to your PC, the USB stick will be selected automatically.
Notice : Data on the USB stick is irrevocably deleted.
4. Click **Start** to upload the image to the USB stick.
⇒ The process may take a few minutes. Do not cancel the process until the **Ready** message appears. You have successfully created a bootable USB stick and can install TwinCAT/BSD on the Industrial PC in the next step.

3.2.2 Check BIOS settings

Check the BIOS settings to be able to start the Industrial PC from the bootable USB stick you created. For TwinCAT/BSD the boot mode in the BIOS must be set to UEFI or Dual Boot. Use Dual Boot if you wish to switch between storage media with different operating systems.

Start the BIOS setup and adjust the boot mode if the settings on your Industrial PC differ.

Proceed as follows:

1. Restart your Industrial PC and press **[Del]** to start the BIOS setup.
The BIOS setup window appears.
 2. Under **Boot > Boot mode select** set the option **UEFI** or **DUAL**.
 3. Press **[F4]** to save the settings and exit the BIOS setup.
The device is restarted.
- ⇒ You have successfully configured the BIOS and can install TwinCAT/BSD in the next step.

3.2.3 Installing TwinCAT/BSD

Connect the bootable USB stick with TwinCAT/BSD image to a Industrial PC and start the device.

Requirements:

- Bootable USB stick with TwinCAT/BSD image.
- Min. 4 GB free storage space.

Proceed as follows:

1. Connect the USB stick with TwinCAT/BSD image to the Industrial PC.
2. Start the Industrial PC and press **[F7]** to enter the boot menu.
3. Select the UEFI entry for the USB stick and confirm with **[Enter]**.
The Industrial PC boots from the USB stick and the Beckhoff TwinCAT/BSD installer is executed.
4. Select the option **TC/BSD Install** to install TwinCAT/BSD.



5. Issue a password and follow the further installation instructions.
- ⇒ Restart the Industrial PC. TwinCAT/BSD is loaded.

3.3 Remote access to the TwinCAT/BSD installer via SSH

SSH stands for Secure Shell and is a method used to establish a secure connection between two computers. SSH works by authentication based on a key pair, where the private key is located on a remote server or industrial PC and the corresponding public key is located on a local computer. If the keys match, the user is granted access.

This chapter shows how to establish an SSH connection from a local PC to an industrial PC with a TwinCAT/BSD installer and how to access its shell. This gives you access to a TwinCAT/BSD system that is fully functional, which you can use for repair or data recovery if, for example, a faulty process prevents a system startup or a faulty TwinCAT project causes a boot loop.

Requirements:

- Before you can establish a remote connection via SSH, you must ensure that the TwinCAT/BSD installer is set up on the USB stick.
- The SSH server must be active. The script "start_ssh_server" is already on the USB stick in the folder `/INSTALLER/autorun`.

Proceed as follows:

1. Before you can establish an SSH connection, you must generate an SSH key pair on your local PC. This key pair consists of a private and a public key.

Under Windows 10, the SSH key pair can be generated with OpenSSH. Open the Command Prompt as an administrator on your Windows 10 PC and execute the following command:

```
ssh-keygen -t ed25519
```

2. You will be prompted to specify a location and to set a password (optional). The generated key pair is stored by default in the directory `C:\Users\<your username>\.ssh`.

```
Generating public/private ed25519 key pair.
```

```
Enter file in which to save the key (C:\Users\username\.ssh\id_ed25519):
```

3. Now you have a public and private keys in the specified location. The `.pub` files are public keys, and files without extension are private keys.

4. Use the command `ssh-add` to load the private key into the SSH agent. This makes SSH authentication easier and more secure, as the private key is stored encrypted in the SSH agent. To do this, open the command prompt on your local PC and enter the following command:

```
ssh-add $env:USERPROFILE\.ssh\id_ed25519
```

5. The content of the public key (`.ssh\id_ed25519.pub`) must be stored on the TwinCAT/BSD installer in a file named `authorized_keys` at `\INSTALLER\.ssh`.

6. From now on you can connect to the TwinCAT/BSD installer from your local PC or from any client that has the private key. Open the command prompt on your local PC and use the following command:

```
ssh root@<IP of TwinCAT/BSD installer>
```

7. The IP address of the TwinCAT/BSD installer is displayed in the graphical user interface after booting. In addition, a fixed IP address for the USB stick can be configured in advance using the shell.

⇒ After successful authentication via SSH you have access to the TwinCAT/BSD installer and thus to the shell of the TwinCAT/BSD system installed on it. You can now perform the necessary tasks, such as installing or maintaining the operating system.

3.4 Automation of processes - using scripts in practice

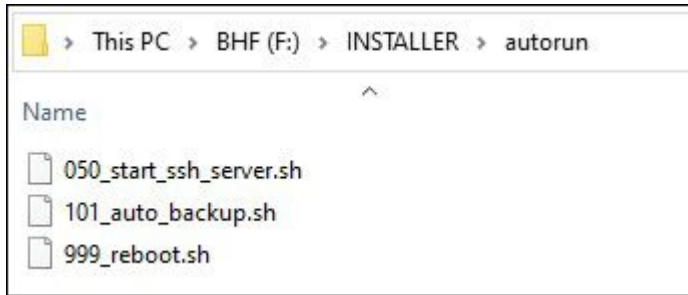
This chapter uses a sample to describe how scripts can be used in conjunction with the TwinCAT/BSD installer. Scripts are extremely useful for automating various tasks related to TwinCAT/BSD systems and the TwinCAT/BSD installer. They can be used to create backups, restore backups, configure the system, perform automatic installations and much more.

This chapter is limited to a scene in which a backup is created automatically from a TwinCAT/BSD system with the aid of a TwinCAT/BSD installer.

Other scenarios for the use of scripts are conceivable. As a reminder, the following [table \[► 16\]](#) describes in which directory the scripts for each task must be placed in order to function properly.

Proceed as follows:

1. Copy the scripts `auto_backup.sh` and `reboot.sh` from the directory `\\INSTALLER\\autorun_samples\\auto_backup` to the directory `\\INSTALLER\\autorun`.



2. Insert the USB stick prepared in this way into an industrial PC.
 3. Restart the industrial PC and press **[F7]** to enter the boot menu. In the boot menu, select the USB stick you want to boot from.
 4. The TwinCAT/BSD installer is started and then a backup is created automatically. A backup is generated with a file name from the host name of the system and the current timestamp and stored in the `\\INSTALLER` directory. Interaction with the graphical user interface is not required.
 5. In the last step the industrial PC is restarted after the backup with the help of the script `reboot.sh`.
- ⇒ The backup is saved on the USB stick in the directory `\\INSTALLER`. If a Beckhoff stick is used for this task, it is not necessary to call up the boot menu with **[F7]**. In the BIOS, Beckhoff USB sticks are set up as the first boot medium by default and are recognized automatically. When a Beckhoff USB stick is inserted, the USB stick is therefore booted directly.

4 TwinCAT/BSD

TwinCAT/BSD combines the TwinCAT runtime with FreeBSD, an industrially tested and reliable open source operating system. In addition to multi-core support and a small footprint, TwinCAT/BSD with the Beckhoff Package Server offers a simple way to install TwinCAT Functions and FreeBSD applications or to update the complete system.

What is FreeBSD

FreeBSD is a Unix-compatible open source operating system directly originating from Berkeley Software Distribution (BSD). As an open source project, FreeBSD is continually being developed further, improved and optimized by a large group of developers. On account of the generous BSD license, Beckhoff has opted for FreeBSD, which enables the integration of TwinCAT without licensing problems.

FreeBSD is very popular and is used worldwide by renowned companies. A detailed list of users can be found here:

<https://www.freebsd.foundation.org/freebsd/#whois>

FreeBSD supports both x86 and X64 platforms and makes scalable systems possible with ARM CPUs extending up to powerful Xeon CPUs.

Further information on FreeBSD can be found on the homepage of the FreeBSD Foundation or that of the FreeBSD project:

<https://www.freebsd.foundation.org/>

<https://www.freebsd.org/>

TwinCAT

TwinCAT/BSD supports all TwinCAT 3 runtime functions. The programming is still carried out with the familiar Microsoft Visual Studio®-based TwinCAT XAE from a Windows development computer. TwinCAT/BSD offers multi-core support, allowing individual cores to also be reserved for the exclusive use of TwinCAT.

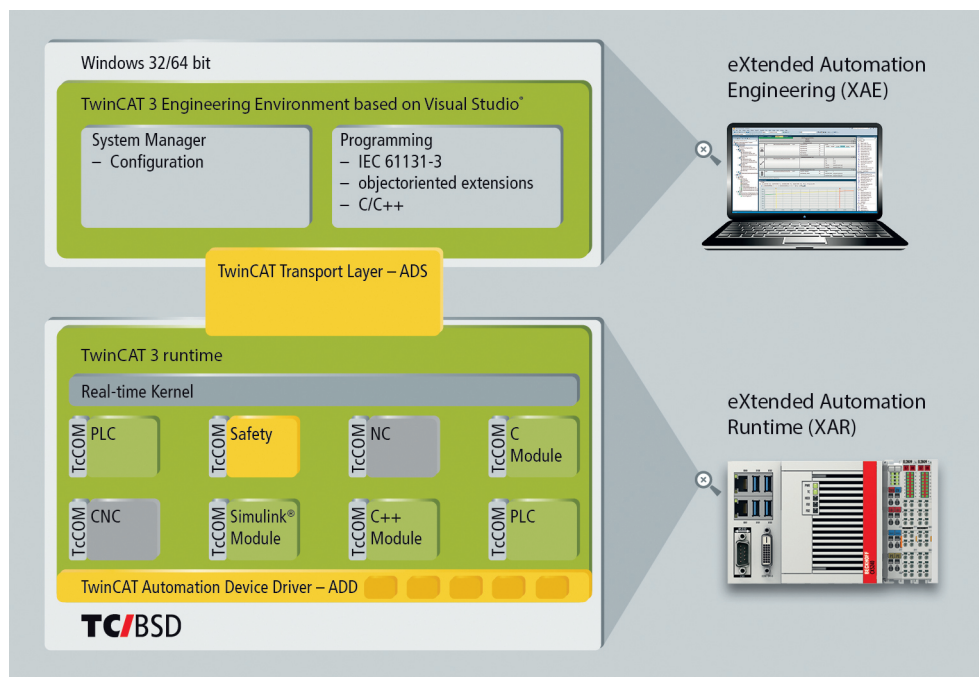


Fig. 10: Structure of the TwinCAT 3 Runtime under TwinCAT/BSD.

In addition to the TwinCAT HMI Server, an HTML5 web browser can be used as a client for TwinCAT HMI. The configuration takes place as usual via the graphic editor of the TwinCAT XAE development environment.

Software and updates

In addition to a large number of FreeBSD programs, TwinCAT Functions can also be installed via the Beckhoff Package Server. Moreover, the uncomplicated updating of the operating system as well as the TwinCAT runtime is possible in this way via the network. Software packages can also be installed offline. The software packages are first loaded to a development computer with a network connection and later installed directly on the Beckhoff Industrial PC. The hosting of the customer's own package server on their side is also possible. Apart from FreeBSD programs that can be offered in this way, many well-known programs from Linux are also available:

<https://www.freebsd.org/ports/>

Write filter

As is familiar from the Windows operating system, TwinCAT/BSD provides a write filter that protects the system from persistent changes. With the write filter activated, the system is in a previously defined state following a restart.

Backup and restore

A TwinCAT/BSD system can be backed up and restored using a USB stick that offers similar functions as the Beckhoff Service Stick for Windows operating systems. A backup can also be created from the live system, which is backed up locally or via the network to a remote system.

4.1 Credentials



Changing the standard password

For security reasons, change the standard password after logging in for the first time.

When TwinCAT/BSD is delivered, a user (`Administrator`) is available by default, with which you can log on to the console. This user does not have conventional administrator rights like under Windows systems but has the authority to obtain root rights for certain purposes. Use the `doas` command to obtain root rights. `doas` corresponds to the command `sudo`, a command known from other Unix-like operating systems.

Login data:

- Login: `Administrator`
- Password: `1`

Proceed as follows:

1. Start the Industrial PC.
2. Log in with the user name `Administrator` and the password `1`.
3. After successful login, the user and the host name of the Industrial PC is displayed. For example:
`CX-1D7BD4.`
`Administrator@CX-1D7BD4$`
4. Enter the command `passwd` in order to set a new password for TwinCAT/BSD. Follow the instructions.
⇒ You have successfully logged in and set a new password for TwinCAT/BSD.

4.2 ZFS properties

ZFS is a file system that combines the roles of volume manager and file system. What is special here is that ZFS knows the structure of the storage media and a contiguous memory pool (zpool) is thus available. The memory pool is divided between the available file systems. As soon as more storage media are added to the pool, the existing file systems automatically grow with them and the new storage space is made available to all file systems.

Conventional file systems such as NTFS, ext3 or UFS behave differently. This separates hard disks, RAID controllers, volume managers, and file systems from one another. File systems can only be created on one hard disk at the same time. As soon as a second hard disk is added, two separate file systems have to be created.

Other advantages of ZFS are:

- RAID functionality is available by default.
- Switch-off-proof thanks to copy-on-write
- Automatic data error detection through checksums
- Convenient backup options through snapshots

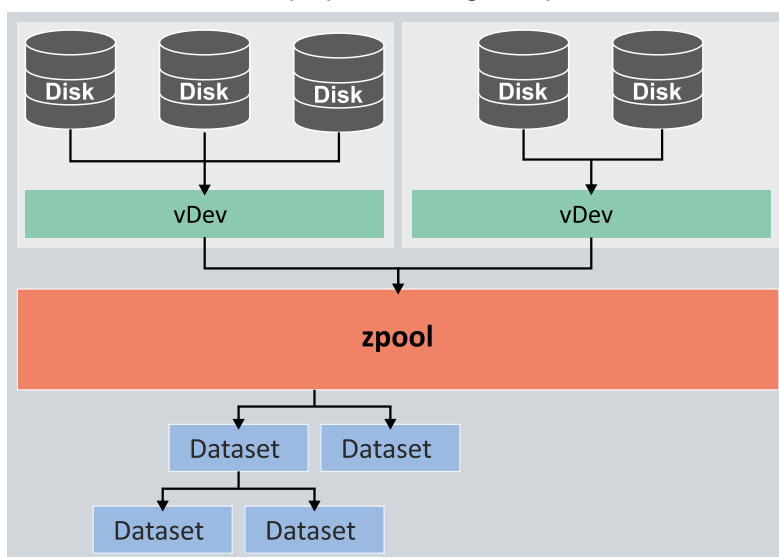


Fig. 11: Overview of the structure of the memory pool, including storage media and datasets.

vDev

The vDevs represent the basic hardware, such as HDDs, SSDs or CFast cards. There are various types of vDev. A vDev can consist of a hard disk, a group of hard disks, a file, a mirror of two or more hard disks or various RAID-Z configurations.

If several vDevs are used, then the data will be divided among the available vDevs in order to increase the speed and to utilize the storage space to the optimum. If one vdev fails, the data of the entire pool is lost. Suitable redundancy (e.g. RAID1) is therefore useful for a vdev.

Memory pool (zpool)

A memory pool (zpool) consists in turn of one or more vDevs. ZFS is based on a memory pool (zpool), which is essentially a collection of vDevs. The vDevs for their part represent the basic hardware, such as HDDs, SSDs or CFast cards, which store the data.

The vDevs are combined into a memory pool. A memory pool is used when one or more file systems (datasets) or block devices (volumes) are to be created. These datasets and volumes share the storage space available in the pool.

Datasets

Dataset is the general term for a ZFS file system, volume, snapshot or clone. Any number of datasets can be created, which are based on a memory pool and contain directories and files. Datasets are hierarchically based on one another. There is a root dataset with following parent datasets, child datasets and further graduations.

The datasets inherit all properties from the parents and grandparents. However, it is also possible to change and overwrite the default values inherited from the parents and grandparents. For each dataset properties such as compression, write and read access, storage space quotas or network shares can be defined.

Example of a dataset:

`zroot/tmp`

In this example, `zroot/` is the root dataset and also the name of the memory pool (zpool) under TwinCAT/BSD. The command `zfs list` can be used to display all available datasets.

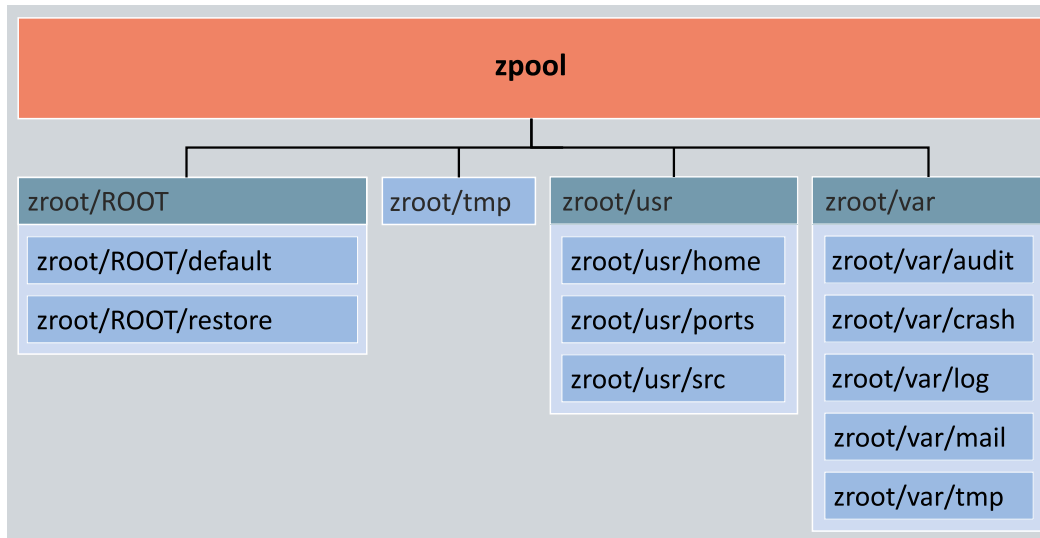


Fig. 12: Datasets of the TwinCAT/BSD operating system.

The dataset `zroot/ROOT/default` contains the basic system, all programs and TwinCAT. The dataset `zroot/ROOT/restore` is a boot environment that can be used for restoring recovery points and resetting to factory settings (see [Restore options](#) [► 115]). The other datasets are mounted at their respective mount points and can be accessed through the file system hierarchy (see [Directory structure](#) [► 25]). Datasets facilitate customization of options such as read and write permissions for entire memory areas or limitation of storage space for log files or the home directory, for example. Furthermore, individual datasets can be backed up using snapshots.

In addition, `zfs list` specifies the default mount point for each dataset, i.e. the point in the operating system's file system hierarchy through which the dataset can be accessed when it is mounted. Most datasets are mounted automatically directly after system startup. The command `zfs mount` displays the datasets that are currently mounted. A file system, directory or device is only made accessible to the user when a dataset is mounted. The memory pool (zpool) and the associated datasets are mounted in TwinCAT/BSD directly after booting.

Volumes

A volume is a special type of dataset. It is not inserted as a file system and is instead a block device under `/dev/zvol/poolname/dataset`. This allows the volume to be used for other file systems, to provide the hard disks to a virtual machine or to be exported via protocols such as iSCSI or HAST (Highly Available Storage). A volume can be formatted with any file system or it can function as a pure data memory. A volume appears to the user to be a normal disk with a fixed size.

4.3 Directory structure

The directory structure of TwinCAT/BSD is based on the Filesystem Hierarchy Standard (FHS).

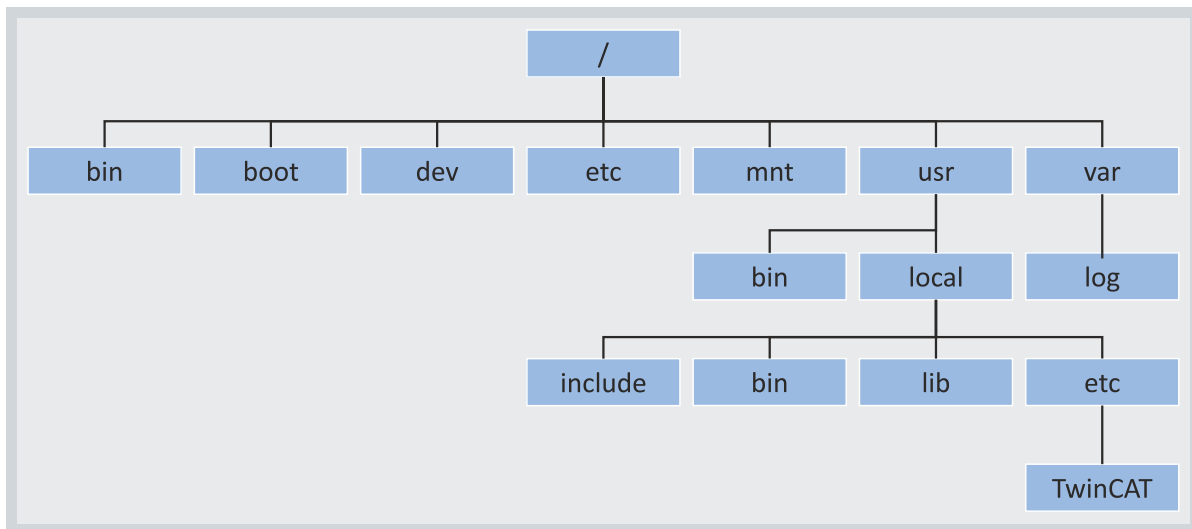


Fig. 13: TwinCAT/BSD directory structure.

The command `cd` can be used to navigate to a different directory. `cd . .` moves up one level in the directory hierarchy, `cd /bin` moves to the directory `/bin`. The command `ls` lists the files in the current directory.

Table 4: Overview of important TwinCAT/BSD directories.

Directory	Description
/	Root directory and top-level directory hierarchy.
/bin/	Basic user applications for single-user and multi-user environments.
/boot/	Kernel, drivers, programs and configuration files for the boot process.
/dev/	Device nodes, which can be used to access hardware directly, for example.
/etc/	System-relevant scripts and configuration files.
/home/	The users' home directories are located here.
/mnt/	Empty directory; usually serves as a mount point for USB sticks, for example.
/root/	Home directory of the superuser <code>root</code> .
/sbin/	Basic system applications for single-user and multi-user environments.
/usr/	Unix system resources, contains most of the user applications.
/usr/bin	General applications.
/usr/include/	Contains header files for C compilers.
/usr/local/	Local programs and libraries, i.e., software installed by a user, such as software unrelated to the basic FreeBSD system itself.
/usr/local/bin/	Mainly Beckhoff applications
/usr/local/etc/	Configuration files, TwinCAT directory with TwinCAT Functions and PLC project .
/usr/local/include/	Including ADS header files TcAdsDef.h and TcAdsAPI.h
/usr/sbin/	System applications that are executed by the user.
/var/	Variable files, i.e. temporary files with changing content such as log files.
/var/log/	Contains system log files.

Programs that are located in one of the `bin` or `sbin` directories can usually be called from the command line without specifying the path. They are defined in the shells as environment variables.

4.4 Write filter

TwinCAT/BSD has a write filter that protects certain data sets against write access. The advantage of a write filter is that the user can secure a system in a preconfigured state. Following a restart, the system is automatically reset to the originally defined state.

The dataset `zroot/ROOT/default`, which contains most of the system and TwinCAT, is protected against write accesses when the Write filter is active. No other data sets are covered by the Write filter. For example, user files can still be persistently stored at `/home` or log files at `/var/log`, even if the rest of the system is reset after a restart.

4.4.1 Enabling or disabling the write filter

This step shows how to enable or disable a write filter under TwinCAT/BSD. Note that the changes to the write filter only take effect after a restart.

Proceed as follows:

1. Enter the command `doas service bwf enable` in the console to enable the write filter.
2. Confirm the command with the administrator password.

```
Administrator@CX-3D6912:~ $ doas service bwf enable
Password:
bwf_enable: NO -> YES
writefilter enabled, please reboot to make your changes take effect.
```

3. Restart the Industrial PC with `shutdown -r now` to apply the settings.

⇒ The write filter is active after the restart. The write filter is deactivated again with the command `doas service bwf disable`.

4.4.2 Defining exceptions

Exceptions for the write filter can be defined by creating new datasets, since only the dataset `zroot/ROOT/default` is protected from write accesses; all other system datasets, including newly created datasets, are excluded from the protection.

This chapter shows an example of how a separate dataset can be created for the TwinCAT boot directory, thereby excluding this directory from the write filter protection.

Requirements:

- Save the TwinCAT boot directory in advance if you follow this example.
- Disable the write filter (see [Enabling or disabling the write filter](#) ► 26)).

Proceed as follows:

1. Enter the command `doas rm -rf /usr/local/etc/TwinCAT/3.1/Boot/*`.
 2. The directory `usr/local/etc/TwinCAT/3.1/Boot` is detached from the file hierarchy.
 3. Enter the command `doas zfs create -o mountpoint=/usr/local/etc/TwinCAT/3.1/Boot zroot/usr/TwinCAT-Boot` to mount the new dataset `zroot/usr/TwinCAT-Boot`.
- ⇒ You have successfully created a new dataset for the TwinCAT boot directory. Use `zfs mount` to display all mounted datasets, including the new dataset `zroot/usr/TwinCAT-Boot`. From now on, all directories below this directory are no longer protected from write access by an active write filter.

4.5 Text Editors

The use of text editors is the simplest method of configuring TwinCAT/BSD without additional programs. Text files can be opened and edited in the console using a text editor.

Easy Editor (ee)

With TwinCAT/BSD the Easy Editor (ee) can be used for this task. Enter the command `ee` in the console to start the Easy Editor. Use `ee filename` to open the file to be edited with the name `filename`.

System files are protected for security reasons. Under TwinCAT/BSD they can only be opened by users with extended rights (root rights). Use (doas) `doas ee filename` to open system files with root privileges.

After opening the editor, the most important functions are listed at the top in the display.

```
^[ (escape) menu    ^y search prompt    ^k delete line      ^p prev li          ^g prev page
^o ascii code       ^x search            ^l undelete line    ^n next li          ^v next page
^u end of file      ^a begin of line     ^w delete word       ^b back 1 char
^t top of text      ^e end of line       ^r restore word      ^f forward 1 char
^c command          ^d delete char       ^j undelete char     ^z next word
=====line 1 col 0 lines from top 1 =====
```

The (^) character represents the **[Ctrl]** key. Therefore, if you wish to use the function ^c you must press the key combination **[Ctrl] + [c]**.

Further information about the Easy Editor and its functions can be found in:

<https://www.freebsd.org/cgi/man.cgi?query=ee&sektion=1&manpath=freebsd-release-ports>

vi editor

TwinCAT/BSD also has more powerful text editors such as the vi editor, which can be used by experienced users. This text editor offers more functionality than the Easy Editor (ee), but is less intuitive.

For more information and functions about the vi editor, see:

<https://www.freebsd.org/cgi/man.cgi?query=vi&sektion=1>

5 Network settings

This chapter describes the network settings under TwinCAT/BSD and guides you through the necessary steps to configure your system. Topics such as IP address assignment, host name customization, and firewall activation with port sharing are covered here.

- **Set IP address:** This section tells you how to set a fixed IP address or obtain one dynamically via DHCP.
- **Change host name:** The host name identifies your system on the network. Here we explain how to customize the host name to enable unique identification.
- **Firewall:** In this section, you will learn how to enable and configure the firewall on your system. In addition, it shows how you can release specific ports to allow access to specific applications.
- **WLAN:** This section shows you how to connect to a WLAN network and configure your system as an access point. Furthermore, we will show you how to set up a DHCP server to enable automatic IP address assignment in your WLAN network.

DHCP client

Under TwinCAT/BSD the `dhclient(8)` was used as DHCP client so far. As of version 13.2.0.6 / 89449, `dhcpcd(8)` is shipped as the default DHCP client and replaces the older client. For the sake of completeness, we will continue to describe how to set a fixed IP address with the old DHCP client (see: [Set IP address for systems with dhclient \[► 31\]](#)) and how to set a fixed IP address with `dhcpcd` (see: [Set IP address \(dhcpcd\) \[► 29\]](#)) in comparison.

The new DHCP client offers the following advantages:

- Faster auto-IP assignment (169,254.x.x).
- Shortened boot times.
- Modern and fast daemon.
- Other features, including the management of multiple IP addresses on one network interface.

If you want to use the `dhcpcd` also under older TwinCAT/BSD versions, the package `dhcpcd` has to be post-installed manually and then configured under `/etc/rc.conf` (see: [Post-install and activate dhcpcd \[► 29\]](#)).

Notice Note that the older `dhclient` and `dhcpcd` must not be configured at the same time, as the two services interfere with each other.

DHCP is still preset for each network interface and the IP address is obtained automatically. To set a static IP address, `ifconfig` can still be used.

5.1 Settings for systems with dhcpcd

As of version 13.2.0.6 / 89449, `dhcpcd (8)` is shipped as the default DHCP client and replaces the older client. This chapter describes how a fixed IP address can be set under TwinCAT/BSD with `dhcpcd`.

Additionally, it is shown how to post-install the `dhcpcd` package for older TwinCAT/BSD versions and how to deactivate the previous DHCP client.

5.1.1 Post-install and activate dhcpcd

● Creating a restore point



Create a restore point before making a major system change or installing programs (see: [Restore options](#) [► 115]).

This chapter describes how `dhcpcd` can also be post-installed and activated under older TwinCAT/BSD versions. For this purpose, the file `/etc/rc.conf` must be extended by some entries or certain entries must be removed. With each installation, the `pkg` program checks whether the local data stock matches that on the Package Server; it is updated if necessary.

Proceed as follows:

1. Enter the `doas pkg install dhcpcd` command in the console.
2. Confirm the installation with **[y]**, so that the package is retrieved from the repository and installed on the system.
3. Enter `doas ee /etc/rc.conf` in the console.
The file `rc.conf` opens in the editor.
4. Remove the line `ifconfig_DEFAULT="DHCP"` to disable the older DHCP client. **Notice** The older `dhclient` and `dhcpcd` must not be configured at the same time, because the two services interfere with each other.
5. Remove the entry `background_dhclient="YES"` and continue to clean the configuration file.
6. Add the following lines to the configuration file.

```
dhcpcd_enable="YES"
dhcpcd_flags="--waitip"
```

7. Press **[Esc]** and select the option a) `leave editor` and subsequently a) `save changes`.
⇒ Restart the system with `shutdown -r now` for the changes to take effect. Then `dhcpcd` is ready for use. If the IP address is not to be obtained via DHCP, then a fixed IP address can be set in the next step (see: [Set IP address \(dhcpcd\)](#) [► 29]).

5.1.2 Set IP address (dhcpcd)

DHCP is enabled by default in delivery state. If there is no DHCP server in the network, TwinCAT/BSD automatically assigns an IP address (169.254.x.x) after a timeout of five seconds. The alternative is a fixed IP address. In this work step you will be shown how to set a fixed IP address in the console in a system with `dhcpcd`.

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface](#) [► 65]).

Proceed as follows:

1. Enter `ifconfig` in the console in order to query the network configuration. The Ethernet interfaces `igb0` and `igb1` of an industrial PC with two interfaces are listed in this sample. The interface `igb1` is active and connected to a network.

```
igb0: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=4a024a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,WOL_MAGIC,RXCSUM_IPV6,NOMAP>
ether 00:01:05:3d:69:12
inet6 fe80::25b2:4227:1a65:b77a%igb0 prefixlen 64 scopeid 0x1
inet 169.254.228.5 netmask 0xffff0000 broadcast 169.254.255.255
media: Ethernet autoselect
status: no carrier
nd6 options=1<PERFORMNUD>
```

```
igb1: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=4a024a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,WOL_MAGIC,RXCSUM_IPV6,NOMAP>
ether 00:01:05:3d:69:13
inet6 fe80::4207:801c:e08a:9ede%igb1 prefixlen 64 scopeid 0x2
inet 172.17.42.57 netmask 0xfffffc00 broadcast 172.17.43.255
media: Ethernet autoselect (100baseTX <full-duplex>)
status: active
nd6 options=1<PERFORMNUD>
```

2. Enter `doas ee /etc/rc.conf` in the console.

The file `rc.conf` opens in the editor.

3. Use the arrow keys to navigate below the entry `dhcpcd_enable="YES"` and add the following line:

```
ifconfig_igb1="inet 192.168.25.25 netmask 255.255.255.0"
```

4. Note the order of the entries in the configuration file. The system reads configuration files from top to bottom. Configuring a static IP address after the DHCP configuration overwrites the previous DHCP configuration.

5. Define the IP address with `inet` and the subnet mask for the Ethernet interface `igb1` with `netmask`.

6. Add the parameter `--denyinterfaces igb1` to the entry `dhcpcd_flags` to disable DHCP for this interface. Otherwise, the interface receives two IP addresses. A fixed IP address that you have defined and additionally an IP address from the DHCP server. You can skip this step if this behavior is desired.

```
dhcpcd_flags="--waitip --denyinterfaces igb1"
```

7. If DHCP is to be disabled for multiple interfaces, they can be listed separated by commas.

```
dhcpcd_flags="--waitip --denyinterfaces igb1,igb0"
```

8. Press **[Esc]** and select the option a) leave editor and subsequently a) save changes.

⇒ You have successfully set `192.168.25.25` as the fixed IP address. Enter the command `doas service netif restart && doas service dhcpcd restart` in the console to have the settings applied. Use the command `doas sh -c "service netif restart && service dhcpcd restart"` when accessing the system via SSH. Then check the network settings with the command `ifconfig`.

5.2 Set IP address for systems with dhclient

DHCP is enabled by default in delivery state. If there is no DHCP server in the network, TwinCAT/BSD automatically assigns an IP address (169.254.x.x) after a timeout of five seconds. The alternative is a fixed IP address. This step shows how to set a fixed IP address in the console.

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface](#) [► 65]).

Proceed as follows:

1. Enter `ifconfig` in the console in order to query the network configuration. The Ethernet interfaces `igb0` and `igb1` of an industrial PC with two interfaces are listed in this sample. The interface `igb1` is active and connected to a network.

```
igb0: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=4a024a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,WOL_MAGIC,RXCSUM_IPV6,NOMAP>
ether 00:01:05:3d:69:12
inet6 fe80::25b2:4227:1a65:b77a%igb0 prefixlen 64 scopeid 0x1
inet 169.254.228.5 netmask 0xffff0000 broadcast 169.254.255.255
media: Ethernet autoselect
status: no carrier
nd6 options=1<PERFORMNUD>
igb1: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=4a024a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,WOL_MAGIC,RXCSUM_IPV6,NOMAP>
ether 00:01:05:3d:69:13
inet6 fe80::4207:801c:e08a:9ede%igb1 prefixlen 64 scopeid 0x2
inet 172.17.42.57 netmask 0xfffffc00 broadcast 172.17.43.255
media: Ethernet autoselect (100baseTX <full-duplex>)
status: active
nd6 options=1<PERFORMNUD>
```

2. Enter `doas ee /etc/rc.conf` in the console.
The file `rc.conf` opens in the editor.
3. Use the arrow keys to navigate below the entry `ifconfig_default="DHCP"` and add the following line:

```
ifconfig_igb1="inet 172.17.40.30 netmask 255.255.255.0"
```

4. Note the order of the entries in the configuration file. The system reads configuration files from top to bottom. Configuring a static IP address after the DHCP configuration overwrites the previous DHCP configuration. The default in `ifconfig_default` means that this configuration applies to all interfaces. The following entries can be used to partially or completely overwrite this configuration.
 5. Define the IP address with `inet` and the subnet mask for the Ethernet interface `igb1` with `netmask`.
 6. Press **[Esc]** and select the option a) leave editor and subsequently a) save changes.
- ⇒ You have successfully set 172.17.40.30 as the fixed IP address. Enter the command `doas service netif restart` in the console to have the settings applied. Then check the network settings with the command `ifconfig`.

5.3 Changing the host name

This step shows how to change the host name of the industrial PC. Note that in doing so you will also change the unique name of the industrial PC in a network.

Host names for Beckhoff Industrial PCs on delivery

With older industrial PCs, the host name is automatically generated from the prefix CX, CP or BK_IPC and the last 3 bytes of the MAC address. The MAC address is 6 bytes long, where the first 3 bytes are the Beckhoff manufacturer ID 00 01 05.

In contrast, the host name of current devices is formed from the Beckhoff Traceability Number (BTN), which is used on all new devices and printed on the name plate as the serial number. The BTN is a unique, eight-character serial number that will replace all other serial number systems at Beckhoff in the long term and will be introduced gradually for the purpose of standardization.

The information on the name plate of the industrial PC is decisive. If no BTN number is available, the host name is formed from the MAC address. If a BTN number exists, the BTN number is used to form the host name instead. Examples of host names for industrial PCs that are formed from the MAC address or BTN number:

- Embedded PC with MAC address "00-01-05-12-24-A3" receives the host name CX-1224A3.
- Industrial PC with BTN number "000fgyeg" receives the host name BTN-000fgyeg.

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface \[► 65\]](#)).

Proceed as follows:

1. Enter the `doas ee /etc/rc.conf` command in the console.
The `rc.conf` file opens.

```
zfs_enable="YES"
# network services and TwinCAT settings
pf_enable="YES"
sshd_enable="YES"
TcSystemService_enable=YES
# custom settings
hostname="CX-1D7BD4"
ifconfig_igb0="DHCP"
ifconfig_igb1="DHCP"
allscreens_kbdflags="-b quiet.off"
# Debugging settings
syslogd_flags="-ss"
#keymap="de.noacc.kbd"
```

2. Change the host name under the entry `hostname="CX-112233"`.
Press **[Esc]** and save the changes.

⇒ The new host name will only be adopted after a restart by the command `shutdown -r now`.

5.4 Firewall

TwinCAT/BSD provides a complete and fully-featured firewall within the package filter (PF). The firewall is factory-set to be restrictive and allows only a few incoming and outgoing connections. The rules for the firewall are stored in a configuration file. You can open the configuration file using the command `doas ee /etc/pf.conf`.

The rules for ports used by Beckhoff services are included through "anchor bhf" in the file `pf.conf` and are created dynamically for TwinCAT Functions. Custom rules for the firewall should still be added to the `pf.conf` file.

Note that the unencrypted ADS port 48898 is disabled by default. Use Secure ADS instead or enable ADS port 48898 with the following entry in the firewall:

Table 5: Firewall rule for unencrypted ADS communication.

Rule	Description
pass in quick proto tcp to port 48898 synproxy state	TCP connections on ADS port 48898 (ADS/TCP), disabled by default.

5.4.1 Enable and disable firewall

The firewall is enabled by default. Disabling the firewall can be useful or even necessary in many cases, e.g. in a test environment. This step shows how to disable the firewall. Note that without a firewall, incoming and outgoing connections will no longer be scanned. Never disable the firewall permanently.

Proceed as follows:

1. Enter the `doas service pf stop` command in the console.
The firewall is disabled.
2. Enter the `doas service pf start` command in the console to re-enable the firewall.
⇒ The firewall is automatically re-enabled after each restart. This behavior is ensured by the entry `pf_enable="YES"` in the `rc.conf` file.

5.4.2 Enable port

● Automatic port enabling for TwinCAT Functions



Ports that are required for TwinCAT Functions are automatically enabled once the TwinCAT Functions have been installed.

This step shows how to enable a TCP port. As an example, an incoming connection is created for TCP port 502, which is required for Modbus/TCP communication.

Enable a port as follows:

1. Enter the command `doas ee /etc/pf.conf` on the console.
The `pf.conf` configuration file is opened.
2. Create the `pass in quick proto tcp to port 502 keep state` rule to enable TCP port 502.
3. Press **[Esc]** and save the changes.
4. Enter the command `doas pfctl -f /etc/pf.conf` to reload the rules. The firewall must be activated for this.
⇒ You have successfully enabled a port. Use the command `doas pfctl -f /etc/pf.conf` to enable the rules immediately. Otherwise, the rule takes effect after the next firewall restart.

5.5 WLAN configuration

This chapter describes various aspects of WLAN configuration under TwinCAT/BSD and includes step-by-step instructions on how to connect an industrial PC to a WLAN, configure it as an access point or set up a DHCP server.

The first section shows how to establish a WLAN connection, including searching for networks and assigning an IP address via a DHCP server. The second section explains the configuration of the industrial PC as an access point with the packet `hostapd`. The third section is dedicated to the installation and configuration of a DHCP server, which may be required depending on the network infrastructure.

5.5.1 Connecting to WLAN

This step shows how to establish a WLAN connection with an access point under TwinCAT/BSD. In addition, you will learn how to search for WLAN networks and determine the SSID.

WLAN is encrypted with WPA2, and the IP address is automatically assigned by a DHCP server.

Requirements:

- Beckhoff WLAN sticks: CU8210-D001-0101 or CU8210-D001-0102
- SSID and password of an existing WLAN.

Proceed as follows:

1. Enter the `sysctl net.wlan.devices` command in the console to obtain the device name. With a Beckhoff WLAN stick, for example, `rtwn0` is given as the device name.
2. Open the `rc.conf` file with `doas ee /etc/rc.conf` and add the following lines:

```
# wireless
wlans_rtwn0="wlan0"      #wlan0 is now your network interface
create_args_wlan0="country DE"
ifconfig_wlan0="up scan WPA DHCP"
```

3. The following lines must be added if `dhcpcd` is used (standard from version 13.2.0.6 / 89449).

```
# wireless
wlans_rtwn0="wlan0"      #wlan0 is now your network interface
create_args_wlan0="country DE"
ifconfig_wlan0="up scan WPA"
```

4. If DHCP is not active or desired, a fixed IP address is defined with the following entry.

```
# wireless
wlans_rtwn0="wlan0"      #wlan0 is now your network interface
create_args_wlan0="country DE"
ifconfig_wlan0="WPA inet 192.168.0.100 netmask 255.255.255.0 up scan"
```

5. Restart the network service with `doas service netif restart` to apply the settings in the `rc.conf` file.
6. Search for new WLAN networks using `doas ifconfig wlan0 up scan`. The `doas ifconfig wlan0 list scan` command displays networks that are already known.
7. Save the access data for a WLAN network with the `doas ee /etc/wpa_supplicant.conf` command by adding the following lines to the `wpa_supplicant.conf` file.

```
network={
    ssid="myssid"      #for myssid specify the name of the network
    psk="mypsk"        #for mypsk enter password of network
}
```

8. Enter `doas service netif restart` to restart the network service.

⇒ The WLAN connection is established. Use `ifconfig` to display the network status of the WLAN interface.

For further information, see: <https://www.freebsd.org/doc/handbook/network-wireless.html>

5.5.2 Configuring as access point

You can configure an Industrial PC as an access point under TwinCAT/BSD. This feature requires the `hostapd` packet to be installed. Install the `hostapd` packet with: `doas pkg install hostapd`

The `hostapd` daemon takes care of client authentication and key management on the WPA2-enabled access point.

Make sure the correct regulatory domain is used for the respective country. This includes, for example, the permitted channels, permitted transmission power and DFS activation for certain 5 GHz channels.

Requirements:

- Installing the `hostapd` packet.
- Internet connection.
- Reinsert the WLAN stick if it was plugged in at startup.

Proceed as follows:

1. Open the file `rc.conf` with `doas ee /etc/rc.conf` and add the following lines:

Example:

```
hostapd_enable="YES"      #starts the hostapd daemon automatically after boot
wlans_rtwn0="wlan0"       #wireless interface used for access point
create_args_wlan0="wlanmode hostap ssid yourSSIDname authmode WPA2"
ifconfig_wlan0="inet 192.168.0.1 netmask 255.255.255.0 country DE"
```

2. Use the command `doas ee /etc/hostapd.conf` to open the file `hostapd.conf` and add the following lines:

Example:

```
interface=wlan0
debug=1
ctrl_interface=/var/run/hostapd
ctrl_interface_group=wheel
ssid=yourSSIDname
wpa=2
wpa_passphrase=freebsdmail #password for wlan network
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
channel=6 #Channel for the desired radio band (default: 0 stands for ACS, automatic Channel Selection)
hw_mode=g #Operation mode, in this case g=IEEE802.11g (2.4 GHz)
country_code=DE #used to set the right regulatory domain for your country
ieee80211d=1 #advertises the country_code an the set of allowed channels and transmit power levels based on the regulatory limits (default=0)
```

3. Enter the command `doas service hostapd forrestart` to start the access point.

⇒ You have successfully configured the Industrial PC as an access point. WLAN devices can now connect to the network using the SSID and the password from the `hostapd.conf` file.

For further information see: <https://www.freebsd.org/doc/handbook/network-wireless.html>

5.5.3 Setting up a DHCP server

Depending on the network infrastructure, you may need a DHCP server. This step shows how to install and configure a DHCP server.

Requirements:

- Internet connection.
- Adjust the firewall and apply the rules to the WLAN interface (see: [Firewall \[► 32\]](#))

Proceed as follows:

1. Enter the `doas pkg install dhcpd` command in the console to install the DHCP server.
2. Open the `dhcpd.conf` file with `doas ee /usr/local/etc/dhcpd.conf` and edit the configuration according to your requirements..

Sample:

```
subnet 192.168.0.0 netmask 255.255.255.0 {  
  range 192.168.0.10 192.168.0.20;  
  default-lease-time 600;  
  max-lease-time 72400;  
  option subnet-mask 255.255.255.0;  
}
```

3. Open the `rc.conf` file with `doas ee /etc/rc.conf` and add the following lines:

```
dhcpd_enable="YES"  
dhcpd_flags="wlan0"  
dhcpd_ifaces="wlan0"
```

4. Enter the `doas service dhcpd start` command to start the DHCP service. After a restart, the DHCP service is started automatically because of the corresponding entry in `rc.conf`.
 5. The `ee /var/db/dhcpd.leases` command can be used to view current and expired leases.
- ⇒ The DHCP server is thus active and only listens to requests on the interface `wlan0`.
For more information, see: https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/network-dhcp.html or <https://man.openbsd.org/dhcpd.8>

5.6 Establish connection with Beckhoff LTE stick

This chapter shows you how to establish a network connection with the LTE stick (CU8210-D004-0200) and how to use the serial interface of the LTE stick.

Prerequisites:

- LTE stick CU8210-D004-0200
- SIM card and corresponding access data of a network provider

Proceed as follows:

1. Connect the LTE stick to the industrial PC.
2. A new network interface (by default ue0) is recognized. The available interfaces are displayed with `ifconfig`.

```
Administrator@CX-3D6912:~ $ ifconfig
---snipped---
ue0: flags=1008843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST,LOWER_UP> metric 0 mtu 1500
    options=80000<LINKSTATE>
    ether 76:ae:02:ec:c9:45
    inet 192.168.225.40 netmask 0xfffff00 broadcast 192.168.225.255
    inet6 fe80::8f92:e7ea:ed:1a66%ue0 prefixlen 64 scopeid 0x4
    inet6 2a01:599:640:6862:c05:263b:b3a4:385f prefixlen 64 autoconf
    media: Ethernet autoselect
    status: active
    nd6 options=1<PERFORMNUD>
---snipped---
```

3. If the `dhclient` is used, an entry must be made in the configuration file `rc.conf` so that the LTE stick receives an IP address. Open the configuration file with `doas ee /etc/rc.conf` and add the line `ifconfig_ue0="SYNCDHCP"`.

This entry ensures that the LTE stick either receives an IP address from an existing DHCP server or assigns one itself. Here, the LTE stick receives an IP address from the provider.

4. If `dhcpcd` (default as of version 13.2.0.6 / 89449) is used instead, this configuration is not required and is carried out using the default settings.
5. Use `ls /dev` to check which virtual COM interface has been recognized by TwinCAT/BSD. Normally the following has been added: `tttyU0`
6. With the command `doas cu -l /dev/tttyU0` we establish a serial connection to the LTE stick in order to enter the SIM pin and the access data of the network provider (APN, Access Point Name). After a successful connection, `Connected` is displayed in the console.

```
Administrator@CX-505918:~ $ doas cu -l /dev/tttyU0
Password:
Connected
```

7. In the next step, AT commands can be exchanged with the LTE stick in order to enter the SIM PIN or to inform the LTE stick of its APN. The APN must be entered the first time the stick is set up and is stored on the stick.
8. Enter the SIM PIN if the SIM card has a PIN. The SIM PIN must of course be re-entered each time the device is rebooted.

```
AT+CPIN=<your pin>
```

9. If your policies allow it, you can disable the SIM PIN by using the following command:

```
AT+CLCK="SC",0,"<your pin>"
```

10. Use the following command to enter the APN:

```
AT+CGDCONT=APN#,"IPV4V6","your_new_apn"
```

11. For example, the command `AT+CGDCONT=1,"IPV4V6","internet.t-mobile"` results in using "internet.t-mobile" as the first prioritized access point (APN1).

12. The created APNs can be displayed with `AT+CGDCONT?`.

```
AT+CGDCONT?
+CGDCONT: 1,"IPV4V6","internet.t-mobile","0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0",0,0,0,0
```

13. Incorrect or unused APNs can be deleted with `AT+CGDCONT=<select Index>`.

14. To apply the above changes (when using the dhclient) to the rc.conf, TwinCAT/BSD must be restarted once.

⇒ After the restart, the interface `ue0` receives an IP address and also has a connection to the Internet if the data plan is active. The LED on the LTE stick flashes "blue" when the SIM card is unlocked and the mobile communication network is accessible with an active data plan. If the SIM card is not unlocked or there are problems reaching the mobile communication network, the LED flashes "red".

External communication is possible by default so that, for example, a ping can be sent to Google (ping: 8.8.8.8) for testing. If you have more restrictive firewall rules, you may need to adjust your firewall settings to allow `ue0` to communicate with the outside world.

Side note: Sending SMS via the LTE stick

If you want to use the SMS function block (SendSMS) from TwinCAT, you can use the serial interface `tttyU0` mentioned above. The TF6340's ADS server (Serial Communication) allows USB devices to be addressed via a virtual serial interface.

Prerequisites:

- TF6340: TwinCAT 3 Serial Communication
- TF6350: TwinCAT 3 SMS/SMTP

In this case, the virtual serial interface can be connected via the TwinCAT 3 function TF6340 (Serial Communication). The TwinCAT 3 function TF6340 must also be installed on the TwinCAT/BSD system. The TwinCAT 3 function TF6350 (SMS/SMTP) then enables SMS to be sent to a recipient using function blocks.

A detailed sample of how an SMS can be sent via the LTE stick can be found here: https://infosys.beckhoff.com/content/1033/tf6350_tc3_sms_smtp/373331083.html?id=3466585722444011586

If, for example, the TwinCAT SMS function block is to be used, port "0" must be entered for the interface `tttyU0` in the function block.

Ausdruck	Datentyp	Wert
fbSendSMS	SendSMS	
sText	STRING	'Please check machine #5, threshold is reached'
sSend	BOOL	F FALSE
sNumber	STRING	F '123456'
sBusy	BOOL	F FALSE
sError	INT	0
RxBuffer	ComBuffer	
TxBuffer	ComBuffer	
fbLineCtrlAds	SerialLineControlADS	
bAdsError	BOOL	F FALSE
nAdsErrorID	UDINT	0
bConnect	BOOL	F TRUE
sNetId	T_AmsNetId	"
stSerialCfg	ComSerialConfig	
ComPort	UDINT	F 0
Baudrate	UDINT	9600
Parity	COMPARITY_T	PARITY_NONE
DataBits	INT	8

Fig. 14: Example of the use of the SMS function block in TwinCAT 3.

5.7 HTTPS certificates

This chapter describes how you can create and import your own HTTPS certificates if the certificates provided by Beckhoff by default for the web interface (Device Manager) are not suitable for your application.

Certificates are used in information technology to securely prove identities. This allows messages or documents to be encrypted so that only the intended recipient can decrypt the content again. Among other things, this technique is used by every web browser when retrieving a page via the HTTPS protocol.

A network subscriber requests the certificate of another subscriber when establishing a communication connection. The certificate and whether the other party authenticates itself using the associated key are checked. Once the identity has been proven, the subsequent message exchange over the connection can be protected against unauthorized manipulation and, optionally, against unauthorized viewing.

In order to use specially generated HTTPS certificates:

- the automatic generation of a certificate for industrial PCs must be disabled,
- a certificate must be requested from a certificate authority (CA)
- and then the HTTPS certificate must be imported.

The exact procedure and the necessary steps are described in this chapter.

5.7.1 Disabling automatic certificate creation

In some use cases it does not make sense for the user to reuse the certificates generated by TwinCAT/BSD® for the web server. In this use case, automatic synchronization can be disabled and own certificates for the TwinCAT/BSD® web server (nginx) can be used.

To do this, you must first disable automatic certificate creation so that your own certificates are installed and not overwritten by the default Beckhoff certificate.

Proceed as follows:

1. Disable the service `IPCDiagnostics` so that the certificate is not overwritten by the default Beckhoff certificate.
 2. To do this, enter the command `doas service IPCDiagnostics disable` in the console.
- ⇒ Automatic generation of certificates has been disabled. In the next step, a certificate can be requested from a certificate authority (CA) (see: [Requesting or creating an HTTPS certificate](#) [► 39]).

5.7.2 Requesting or creating an HTTPS certificate

A certificate authority (CA) typically provides installation instructions on how to install a certificate it has issued. The certificate authority even provides instructions on how to apply for the certificate. Please primarily follow the instructions of the certificate authority.

First, you must create a certificate signing request (CSR) and submit the certificate request to the certificate authority in accordance with its instructions. The certificate authority will then provide you with the server certificate and the intermediate certificates to create a certificate signing request

If you do not have a certificate from an official certificate authority (CA), you can create a self-signed certificate for test purposes.

Proceed as follows:

1. Generate a self-signed certificate for test purposes with the following command:

```
doas openssl req -x509 -newkey rsa:4096 -nodes -sha256 -days 3650 \  
-keyout IPCDiagnostics.key \  
-out IPCDiagnostics.crt \  
-subj '/CN=<hostname>' \  
-addext 'subjectAltName=DNS:<hostname>,IP:<ipaddress>'
```

2. The command creates a private key `IPCDiagnostics.key` and a self-signed certificate `IPCDiagnostics.crt`.

`openssl req`: The command part `req` creates and processes certificate signing requests (CSR). With `-x509`, a self-signed certificate is created directly instead.

`-newkey rsa:4096`: Creates a new key pair with the RSA algorithm and a key length of 4096 bits.

`-nodes`: Means "no DES". The private key is not stored in encrypted form, i.e. no password is required to use the key.

`-sha256`: Uses the secure hash algorithm SHA-256 to create the certificate signature.

`-keyout IPCDiagnostics.key`: Stores the private key in the file `IPCDiagnostics.key`.

`-out IPCDiagnostics.crt`: Stores the created certificate in the file `IPCDiagnostics.crt`.

`-subj '/CN=<hostname>'`: Specifies the data for the certificate without an interactive request. `/CN=<hostname>` specifies the common name (CN), which is usually the host name or domain that is protected by the certificate.

`-addext 'subjectAltName=DNS:<hostname>,IP:<ipaddress>'`: Adds an extension to the certificate. `subjectAltName` (SAN) allows additional names and IP addresses to be covered by the certificate.

3. Replace `<hostname>` with the host name and `<ipaddress>` with the IP address of your TwinCAT/BSD device.

⇒ In the next step, the certificate can be imported (see: [Importing the certificate](#) [► 40]).

5.7.3 Importing the certificate

As soon as you have received a certificate from an official certificate authority (CA), you can import it into your TwinCAT/BSD system. Alternatively, you can also use the self-signed certificate that you have created for test purposes.

Proceed as follows:

1. Replace the existing private key for nginx with your private key:

```
doas cp IPCDiagnostics.key /usr/local/etc/TwinCAT/3.1/Target/PrivateKeys/IPCDiagnostics.key
```

2. Replace the existing certificate for nginx with your certificate:

```
doas cp IPCDiagnostics.crt /usr/local/etc/TwinCAT/3.1/Target/Certificates/IPCDiagnostics.crt
```

3. Restart the nginx web server:

```
doas service nginx restart
```

4. The certificate is ready for use after restarting the service.

⇒ If you are not using a certificate from an official certificate authority (CA), your browser will display a security warning. You can configure your browser to accept the certificate automatically by importing the server certificate into your browser's certificate store. Further information on this can be found at: [Self-signed certificates for https connection](#)

As some browsers (e.g. Mozilla Firefox) use their own certificate stores, it may be necessary to import the certificate directly in the browser.

6 System update

TwinCAT/BSD release process

The system update is performed by default via the preset Beckhoff Package Server or the Beckhoff repository, which provides all required packages (see: [Package Server](#) [► 46]).

As a rule, a new version of the Beckhoff repository is published on the first Tuesday of each month. Provided that all internal tests are passed successfully. If the internal tests fail, the release will be postponed to the next month.

Only the Beckhoff repository is updated, which does not necessarily mean that a new TwinCAT/BSD or TwinCAT version is automatically published in each new repository version. It could also simply be individual packages, third-party packages, or packages that you don't even have installed on your system. When performing an update, it is recommended to update the entire system and not just individual packages. This is because the packages in the Beckhoff repository are tested as a complete system in each release, thus avoiding incompatibilities.

It is possible to download and save a tested version of the Beckhoff repository as a local repository at any time. This allows you to freeze your system in a tested state and use it for series machine construction, for example. In this case, the local repository and all the packages it contains can be made available via a server in the local network or via a USB stick (see: [Provide repository on USB stick](#) [► 50]).

Call TwinCAT/BSD version

The TwinCAT/BSD version can be called with the command `TcSysExe.exe`. There are different version information that you can get from the system. The current version is listed under the TC/BSD entry:

```
Administrator@CX-0C8432:~ $ TcSysExe.exe

The software licenses can be found in this folder: /usr/local/etc/TwinCAT/3.1/System/Legal/
TcOsSys.dll: TcOsSys_Rel131_4024_20210804.2
TwinCAT Build: 3.1.4024.19
AMS Net Id: 5.12.132.50.1.1
TC/BSD: 13.0.8.2,2
Administrator@CX-0C8432:~ $
```

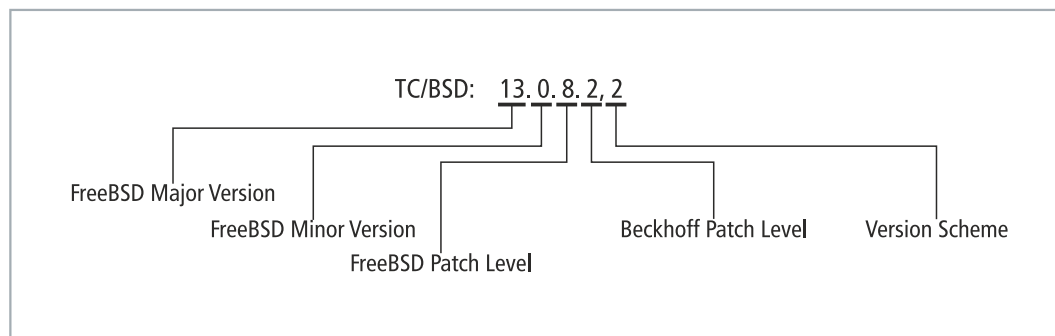


Fig. 15: Breakdown of the TwinCAT/BSD version.

Call Beckhoff repository version

The command `pkg info os-generic-userland-conf | grep Version` displays the pipeline number for the corresponding internal Beckhoff repository. In this sample the pipeline ID 55702 is displayed after the TwinCAT/BSD version:

```
Administrator@CX-3AE2C6:~ $ pkg info os-release-bhf | grep Version
Version : 13.0.11.3_55702
Administrator@CX-3AE2C6:~ $
```

Perform update

There are two ways to update the system. The first way is the regular one for minor versions and TwinCAT updates. The second path is only required for upgrading FreeBSD major releases, which usually bring new FreeBSD features (see: [FreeBSD-Release-Notes](#)).

1. [Update TwinCAT/BSD \[► 43\]](#)
2. [Update major version \[► 43\]](#)

Configuration files such as `rc.conf` and `TcRegistry.xml` are usually not affected by a system update and installation of new packages and remain unchanged. This ensures that any changes the user makes to the configuration files are preserved and not overwritten.

When installing new packages, configuration files may need to be manually adjusted. Since no user configurations are made during the upgrade, a system may differ not only in its configuration but also in its installed packages from a newly installed system, despite having the same version levels.

1. Update TwinCAT/BSD

The `pkg upgrade` command is used to update packages (see: [Update packages \[► 49\]](#)). In doing so, `pkg upgrade` compares the versions of all installed packages with the versions available in the configured package repositories. The Beckhoff Package Server mirrors most of the packages from the normal FreeBSD repository and also contains the following specific Beckhoff packages that can be updated:

- TwinCAT/BSD updates
- TwinCAT 3 updates
- TwinCAT 3 functions

In addition, `pkg upgrade` also tries to update the dependencies of the packages. However, no new packages will be installed except when necessary to satisfy package dependencies.

Individual packages can also be updated, but it is recommended to update the entire system to avoid incompatibilities. This is because the packages in the Beckhoff repository are tested as a complete system in each release.

In addition, the upgrade is carried out in a new boot environment so that the current system remains largely untouched during the upgrade process. If an error occurs during the upgrade, the boot environment is simply removed and the system is left in its original state. The old EFI Loader is also restored.

In addition to the option of updating TwinCAT/BSD with the help of `pkg upgrade`, the `tcbsd-upgrade` tool provides an even simpler option recommended by Beckhoff for updating TwinCAT/BSD (see: [Update TwinCAT/BSD \[► 43\]](#)). The tool automatically creates a restore point and performs the upgrade in a new boot environment so that the current system remains largely untouched during the upgrade process. If an error occurs during the upgrade, the boot environment is simply removed and the system is left in its original state. The old EFI Loader is also restored. Even after a successful update, the old boot environment is available as a clone that can be accessed at any time until it is manually deleted.

2. Update major version

The current major version is required because after a change to the next higher major version, all security updates are provided only for the current version. Older versions are no longer maintained.

In addition to this, new features may be implemented and made available in the future in the latest major version, which will be required for TwinCAT/BSD.

6.1 Update TwinCAT/BSD

In addition to the option of updating TwinCAT/BSD using `pkg upgrade`, the `tcbsd-upgrade` tool provides an even simpler option recommended by Beckhoff for updating TwinCAT/BSD. The goal is to bring the system up to date. The tool automatically creates a restore point with the current timestamp. You can list restore points with `restorepoint status`.

In addition, the upgrade is carried out in a new boot environment so that the current system remains largely untouched during the upgrade process. If an error occurs during the upgrade, the boot environment is simply removed and the system is left in its original state. The old EFI Loader is also restored.

If something goes wrong after the upgrade and restart, you can use the restore point to undo the upgrade. This can be done via SSH or the system terminal if the system is accessible:

```
bectl list
doas bectl activate upgrade-<timestamp>
reboot
```

Update the system as follows:

1. Install the tool with the command `doas pkg install tcbsd-upgrade`
2. Enter the command `doas tcbsd-upgrade minor` in the console and restart the system after a successful update.
 - ⇒ From this point on, there are at least two boot environments on the system, and you can return to the old boot environment at any time with `bectl`, or call it up during the boot process via the boot menu (press the space bar).
3. If everything went well and you are sure that you do not want to undo anything, you can clean the restore point:

```
restorepoint status
doas restorepoint destroy upgrade-backup-<timestamp>
```

4. Replace the old boot environment (default) with the new boot environment (`upgrade-<timestamp>`) and rename the new boot environment back to default:

```
bectl list
doas bectl destroy default
doas bectl rename upgrade-<timestamp> default
```

5. Restart TwinCAT/BSD with `shutdown -r now`.
 - ⇒ It is important to keep the old name (default) for the boot environment to prevent old restore points (e.g. `factoryreset`) from being damaged. The tool `restorepoint` will try to activate the old boot environment (e.g. `"zroot/ROOT/default"`) during the recovery and will fail if the old boot environment no longer exists. The tool `tcbsd-upgrade` creates clones of the current boot environment during the upgrade to preserve the history or snapshots so that the old boot environment can be replaced by the new one to avoid corruption of the restore points.

Also see about this

- 📖 Restore options [► 115]
- 🔒 Lock [► 50]

6.2 Update major version

This chapter describes how to update a major version from TwinCAT/BSD 12 to TwinCAT/BSD 13, for example. The current major version is required for security updates, among other things.

The update is carried out with the aid of a script, because the entire basic system as well as the kernel adapted by Beckhoff is updated. The script also automatically creates a restore point that you can jump back to (see: [Resetting to the restore point \[► 117\]](#)).

Proceed as follows:

1. Update the latest major version with `doas pkg update -f && doas pkg upgrade`.
2. Install the script for the update with the command `doas pkg install tcbsd-upgrade`

3. Execute the script with the command `doas tcbbsd-upgrade major`. At the end of the process, a message about a successful update to the next major version is displayed.

```
Successfully upgraded to TC/BSD 13 and activated the new BE "upgrade-2022-02-17T15:02:01Z"
Reboot is required.
```

4. Restart TwinCAT/BSD with `shutdown -r now` after a successful update.

⇒ After the restart, the new major version is available and displayed after the login.

```
Last login: Thu Feb 17 15:07:47 on pts/0
FreeBSD 13.0-RELEASE-p7 n244930-e4e6bcfbb68- BHF

The software licenses can be found in this folder: /usr/local/etc/TwinCAT/3.1/System/Legal/
TcOsSys.dll: TcOsSys_Rel131_4024_20211129.2
TwinCAT Build: 3.1.4024.22
AMS Net Id: 5.61.105.18.1.1
TC/BSD: 13.0.7.2,2
```

Adaptation of the boot environment

The upgrade is installed in a new boot environment. A boot environment is a ZFS file system that can be booted into. The old ZFS file system is therefore still available and can be used as a backup in case of emergency. You can always select the boot environment you want to boot into from the bootloader menu and the "Boot Environments" menu item (press the space bar while booting).

To remove the old boot environment and free the system from legacy issues, the following procedure is recommended after a successful upgrade:

1. Delete the restore point that is automatically created before the upgrade to the new major version. This also serves as a further fallback. Enter the command `doas restorepoint destroy`.
2. Select the restore point `upgrade-backup-<your timestamp>` from the menu to delete it.
3. Delete old boot environment (default) with `bectl destroy default`
4. Rename the new boot environment to default with `bectl rename upgrade-<timestamp> default`. There must always be a "default" boot environment so that the restore point `factoryreset` can work.

Notice: The boot environment `restore` must not be deleted.

6.3 Update CPU microcode

TwinCAT/BSD offers the possibility to update the microcode of a CPU automatically at startup. This allows, for example, the latest security updates to be installed on a system.

This function is disabled by default. Test each update before installing it, as it may affect the system.

Proceed as follows:

1. Install the `cpu-microcode-amd` or `cpu-microcode-intel` package with the `doas pkg install cpu-microcode-amd` or `doas pkg install cpu-microcode-intel` command, dependent on the CPU.

2. Enter `ee /boot/loader.conf` in the console.

The file `loader.conf` opens in the editor

```
kern.geom.label.disk_ident.enable="0"
kern.geom.label.gptid.enable="0"
cryptodev_load="YES"
zfs_load="YES"
hint.attimer.0.clock="0"
vmm_load="YES"
pptdevs="0/30/3"
```

3. Add the following lines to the `loader.conf` file:

```
cpu_microcode_load="YES"
cpu_microcode_name="/boot/firmware/intel-ucode.bin"
```

4. Restart TwinCAT/BSD with the command `shutdown -r now`.

- ⇒ With these settings, the system checks for packages with updated CPU microcode at every system startup and automatically installs them if necessary. Remove the two lines from the `loader.conf`, if this function is no longer desired.

7 Package Server

The Beckhoff Package Server is a server hosted by Beckhoff that contains a collection of precompiled software – the so-called packages. This is the easiest way to install additional software under TwinCAT/BSD or update existing software.

During installation TwinCAT/BSD accesses the preset Beckhoff repository:

<https://tcbsd.beckhoff.com/TCBSD/14/stable/packages/All/>

The Beckhoff package server mirrors a large part of the normal FreeBSD repository and also includes the following specific Beckhoff packages:

- TwinCAT/BSD updates
- TwinCAT 3 updates
- TwinCAT 3 functions

Supported packages

i Unlike Beckhoff packages, third-party packages are not checked and are not supported.

Repositories

You can fall back on additional repositories if you want to use other packages that are not provided by Beckhoff. In addition to the Beckhoff repository set by default, you can add, enable, disable or remove other repositories. Then switch back to the Beckhoff repository.

Make sure you configure additional repositories in a file of their own under `/etc/pkg`. TwinCAT/BSD has two repositories on delivery:

- `TCBSD.conf` is the standard Beckhoff repository

```
TCBSD: {
  url: "https://tcbsd.beckhoff.com/TCBSD/14/stable/packages"
  enabled: true,
  signature_type: "fingerprints",
  fingerprints: "/usr/share/keys/bhf-pkg"
}
```

- `FreeBSD.conf` is the official FreeBSD repository and is disabled by default

```
FreeBSD: {
  url: "pkg+http://pkg.FreeBSD.org/${ABI}/quarterly",
  mirror_type: "srv",
  signature_type: "fingerprints",
  fingerprints: "/usr/share/keys/pkg",
  enabled: yes
}
```

7.1 Switching to a Chinese server

The Beckhoff repository is not accessible from China and has to be switched over to the Chinese server. Run the `pkgrepo-set` script to switch from the default server to the Chinese server.

Switch to the Chinese server as follows:

1. Enter the `doas sh /usr/local/share/examples/bhf/pkgrepo-set.sh china` command in the console.
 2. The URL in the file `TCBSD.conf` is changed from `https://tcbsd.beckhoff.com/TCBSD/14/stable/packages` to `https://tcbsd.beckhoff.com.cn/TCBSD/14/stable/packages`.
- ⇒ You have successfully set the Chinese server. You can restore the default settings with the command `doas sh /usr/local/share/examples/bhf/pkgrepo-set.sh release`.

7.2 Switching to the FreeBSD repository

i Incompatibilities with already installed packages

The FreeBSD repository can be used for testing purposes to test certain packages that are not provided on the Beckhoff package server. In the meantime, do not update the TwinCAT/BSD system (`doas pkg upgrade`) via the FreeBSD repository, as this can lead to incompatibilities with already installed packages, and switch back to the Beckhoff package server after the tests.

This step shows you how to switch from the standard Beckhoff repository to the official FreeBSD repository in order to test packages that are only provided on the FreeBSD repository. Contact Beckhoff support if certain packages are required so that they can be included in the standard Beckhoff repository.

Note that packages not installed from a Beckhoff repository may not be compatible with TwinCAT/BSD and may not function properly. The reason for this is that the FreeBSD basic system has been modified for TwinCAT. After the tests, switch back to the Beckhoff package server.

Switch to the FreeBSD repository as follows:

1. Enter the `doas ee /usr/local/etc/pkg/repos/FreeBSD.conf` command in the console.
2. Set the `FreeBSD: {enabled: no}` variable to "yes".

You have successfully changed the package server. From now on, all packages will be loaded from the official FreeBSD repository. Set the variable to "no" to use the standard Beckhoff repository again.

7.3 Package management

A TwinCAT/BSD installation includes the software required for the operating system and the TwinCAT 3 Runtime. You can install additional software or TwinCAT Functions.

You have to decide which functions you need and choose software that provides these functions. This chapter shows you how to:

- search for,
- install,
- update,
- uninstall
- and lock packages, i.e. precompiled software.

The command `pkg info` lists all packages installed on the system, together with their respective versions. `pkg info <packagename>` displays information about a specific package.

```
---snipped---
Administrator@CX-3B151A:~ % pkg info IPC-Diagnostics
IPC-Diagnostics-3.1.4024.5_2019110615523410164
Name           : IPC-Diagnostics
Version        : 3.1.4024.5_2019110615523410164
Installed on   : Fri Nov 8 10:55:37 2019 UTC
---snipped---
```

7.3.1 Search

Before you install software, you can determine whether the software is available on the Package Server. Note that the search is not case sensitive. A distinction is made between case and lower case only with the suffix -C.

Proceed as follows:

1. Enter the command `pkg search <packagename>` in the console. Example:

```
Administrator@CX-3B151A:~ % pkg search docbook
docbook-1.5                Meta-port for the different versions of the DocBook DTD
docbook-sgml-4.5_1        DocBook SGML DTD
docbook-xml-5.0_3         DocBook XML DTD
docbook-xsl-1.79.1_1,1    XSL DocBook stylesheets
sdocbook-xml-1.1_2,2      "Simplified" DocBook XML DTD
Administrator@CX-3B151A:~ %
```

2. Several results are displayed.

3. You can view additional information about the package with the command `pkg search -R docbook`.

```
---snippet---
name: "docbook"
origin: "textproc/docbook"
version: "1.5"
comment: "Meta-port for the different versions of the DocBook DTD"
maintainer: doceng@FreeBSD.org
www: http://www.oasis-open.org/docbook/
abi: "FreeBSD:12:*"
arch: "freebsd:12:*"
---snippet---
```

4. Use the additional information to choose the right software.

⇒ In the next step you can install the software.

7.3.2 Install



Creating a restore point

Create a restore point before making a major system change or installing programs (see: [Restore options](#) [► 115]).

This step shows you how to install new software under TwinCAT/BSD. The software must be available on the Package Server (see: [Search](#) [► 48]). You can install additional software or TwinCAT Functions.

With each installation, the `pkg` program checks whether the local data stock matches that on the Package Server; it is updated if necessary.

Proceed as follows:

1. Enter the command `doas pkg install <packagename>` in the console. Sample:

```
---snippet---
Administrator@CX-3B151A:~ % doas pkg install docbook
Password:
Updating FreeBSD12-pkgbase repository catalogue...
FreeBSD12-pkgbase repository is up to date.
All repositories are up to date.
---snippet---
```

2. The `pkg` program automatically searches for additional packages that are required for the installation.

3. Confirm the installation with **[y]**.

⇒ The package is retrieved from the repository and installed on the system. Some packages contain installation messages that contain instructions, warnings and helpful notes.

7.3.3 Update

● Creating a restore point

i Create a restore point before making a major system change or updating programs (see: [Restore options \[► 115\]](#)).

This step illustrates how to update packages under TwinCAT/BSD. Create a backup of your system and start a trial run with the suffix `-n` before you update packages. All the packages that can be updated without performing the installation are listed in the trial run.

Beta packages are no longer updated via `pkg upgrade` once the packages have been released and are no longer listed in the Beckhoff repository. From this point on, the beta packages must be uninstalled and the released packages installed instead.

Update packages as follows:

1. Enter the command `doas pkg upgrade -n` in the console.
The packages to be updated are displayed.

```
Checking for upgrades (27 candidates): 100%
Processing candidates (27 candidates): 100%
The following 1 package(s) will be affected (of 0 checked):

Installed packages to be UPGRADED:
  ca_root_nss: 3.47_1 -> 3.47.1

Number of packages to be upgraded: 1

287 KiB to be downloaded.
```

2. Enter the command `doas pkg upgrade` on the console, in order to update all packages or `doas pkg upgrade <packagename>` in order to update a specific package.
- ⇒ Take a close look at the packages to be updated. If there are any ambiguities, read the release information for the package and only then start the update.
- Define exceptions and lock packages that should not be updated (see: [Lock \[► 50\]](#)). Locked packages are not updated, uninstalled or reinstalled.

7.3.4 Uninstall

This step shows you how to uninstall software under TwinCAT/BSD. The `pkg` program ensures that an uninstall has no negative consequences for the system and pays attention to dependencies among the packages.

If you uninstall software on which another software depends, that software will also be removed automatically.

```
docbook: 1.5
sdocbook-xml: 1.1_2,2
xmlcatmgr: 2.2_2
docbook-xml: 5.0_3
xmlcharent: 0.3_2
docbook-sgml: 4.5_1
iso8879: 1986_3
```

The `docbook` package depends on the `iso8879` package. If you uninstall the `iso8879` package, `docbook` will also be removed. You can suppress this behavior with the suffix `-f` and uninstall only the selected package without the system paying attention to the dependencies.

Proceed as follows:

1. Enter the command `doas pkg delete <packagename>` in the console. Example: `doas pkg delete iso8879`
2. TwinCAT/BSD lists packages that can be deleted and considers the dependencies.

```
---snipped---
Administrator@CX-3B151A:~ % doas pkg delete iso8879
Password:
Checking integrity... done (0 conflicting)
Deinstallation has been requested for the following 3 packages (of 0 packages in the universe):
```



```

Installed packages to be REMOVED:
  iso8879-1986_3
  docbook-sgml-4.5_1
  docbook-1.5
---snipped---

```

3. Confirm with **[y]** to uninstall the listed packages.

⇒ In conclusion, the uninstalled packages are summarized. Note that unwanted packages can accumulate over time, for example, if you uninstall software with the suffix `-f` or if new software versions have different dependencies.

The command `pkg autoremove` identifies, lists and proposes the uninstallation of unnecessary packages and dependencies. Read the list carefully. Important packages can also be locked so that they are not inadvertently uninstalled (see: [Lock \[► 50\]](#)).

7.3.5 Lock

Packages can be locked to prevent them from being inadvertently uninstalled or updated during updates. Locked packages are not updated, uninstalled or reinstalled.

This feature can be especially useful for certain TwinCAT Functions in order to prevent unwanted updates. Locking does not prevent someone with root rights from manipulating the files contained in the package.

Lock packages as follows:

1. Enter the command `doas pkg lock <packagename>` on the console.

```

docbook-1.5: lock this package? [y/N]: y
Locking docbook-1.5

```

2. Confirm the query with **[y]** to lock the package.

⇒ The package remains locked until you unlock it with the command `doas pkg unlock <packagename>`. You can display all locked packages with the command `pkg lock -l`.

```

Currently locked packages:
docbook-1.5

```

7.4 Set up a local repository

We recommend using a local repository to install software and updates offline. The local repository can then be made available via a USB stick or a server on the local network. To do this, first download the TwinCAT/BSD repository. This repository can then either be copied to the FAT partition of a USB stick or made available via an FTP or web server on the network.

- [Provide repository on USB stick \[► 50\]](#)
- [Setting up your own Package Server \[► 52\]](#)

By mirroring the entire content, the package verification also remains the same. For your TwinCAT/BSD system it is as if the packages continue to be downloaded from the Beckhoff repository.

In this description, all configuration steps are performed locally on a TwinCAT/BSD system. If this is not desired, the TwinCAT/BSD repository can alternatively be downloaded to a Windows system. In this case, the official Beckhoff Powershell module is available for download at <https://github.com/Beckhoff/twincatbsd-tools>.

7.4.1 Provide repository on USB stick

This chapter describes how to provide a repository on a USB stick. TwinCAT/BSD is used as the host system for the work steps.

Requirements:

- Program for recursively downloading files from the Internet. An example is the program `wget` which is available for Linux and TwinCAT/BSD.

Proceed as follows:

1. Download the TwinCAT/BSD repository from the Beckhoff server. If you use Linux or TwinCAT/BSD, use the command `wget --recursive --timestamping --level=inf --no-cache --no-parent --no-cookies --no-host-directories --relative --directory-prefix /tmp/mirror https://tcbsd.beckhoff.com/TCBSD/14/stable/packages/`
 2. Insert the USB stick into your TwinCAT/BSD device. Note that the device may boot directly from the USB stick according to your BIOS setting. Then plug in the USB stick only after booting.
 3. Connect the USB stick to your system (see: [Integrating a USB stick \[► 59\]](#)). Automount was deactivated ex works for security reasons.
 4. Copy the repository from the directory `/tmp/mirror` to the FAT partition of a USB stick using the command `cp -r /tmp/mirror /mnt/usb`. The path `/mnt/usb` is the mount point of the USB stick in this example.
 5. Next, the repository path that points to this Beckhoff Package Server by default must be changed: <https://tcbsd.beckhoff.com/TCBSD/14/stable/packages/>
 6. In this example, the repository path must point to the USB stick in order to use the local repository on the USB stick. Enter the following command: `doas sh /usr/local/share/examples/bhf/pkgrepo-set.sh file:///mnt/mirror/TCBSD/14/stable/packages`
- ⇒ Use the pkg tool as usual. The command `doas pkg upgrade` can now be used to update and install packages from the USB stick. Instead of the Beckhoff Package Server, the local repository on the USB stick is used. Note that after a reboot the USB stick must be integrated manually again.

7.4.2 Setting up your own Package Server

The following steps can be carried out locally on a TwinCAT/BSD system.

1. Creating a key for RSA encryption

First, an RSA key pair must be created. An RSA key pair consists of a private and a public key. The private RSA key is used to generate digital signatures and the public RSA key is used to verify digital signatures.

Create a private RSA key. You have the choice between RSA key sizes of 2048 or 4096 bits:

```
openssl genrsa -out myRSAPrivate.key 2048
```

Create a public RSA key:

```
openssl rsa -in myRSAPrivate.key -out myRSAPublic.key -pubout
```

Restrict access to solely `root` or store the private key in a different location:

```
chmod 0400 myRSAPrivate.key
```

2. Fetching packages from the Beckhoff Package Server

In this step, either all packages can be fetched from the Beckhoff Package Server or only the packages that are installed on the system.

Fetch all packages from the server:

```
doas pkg fetch --yes --output /var/db/myRepository --all
```

Only fetch the installed packages:

```
doas sh -c "pkg info | awk '{print \$1}' | xargs -I {} pkg fetch --yes --output /var/db/myRepository {}"
```

3. Modifying and creating the repository

The downloaded packages can be enriched with user-defined packages to create your own repository.

Move user-defined packages to the repository:

```
mv my-package /var/db/myRepository/All/
```

Create repository and provide it with the previously created private key:

```
doas pkg repo /var/db/myRepository/ myRSAPrivate.key
```

4. Moving the repository directory to the web server

If you want to use HTTPS, you either need a web server with a valid certificate (e.g. obtain via letsencrypt) or you can use HTTP as the packages are already signed.

HTTPS

If you use your own certificate authority (CA), add the certificate of your CA to the certificate list

```
/usr/local/etc/ssl/cert.pem
```

When using the nginx server supplied with TwinCAT/BSD, add the following lines further down in the file `/usr/local/etc/nginx/IPCDiagnostics.conf` under the area `"Server {}"`:

```
location /pkg {
    alias /var/db/myRepository/;
    autoindex on;
}
```

The web server must then be restarted with `doas service nginx restart` for the configuration to become active. Other pages that can be accessed via the web server are also located here.

HTTP

If you only want to make the repository available via HTTP, allow the web browser to listen on port 80 and open the port in the firewall at `/etc/pf.conf`. Add the following lines to the file `pf.conf`:

```
# allow port 80 for pkg repository
pass in quick proto tcp to port 80
```

Add the following lines at the end of the file `/usr/local/etc/nginx/IPCDiagnostics.conf` under the area `"http {"`:

```
server {
    listen 80;
    location /pkg {
        alias /var/db/myRepository/;
        autoindex on;
    }
}
```

The web server must then be restarted with `doas service nginx restart` for the configuration to become active. Make sure that this entry is still enclosed by the outermost bracket of the `http` function. This entry only allows HTTPS access for all services preconfigured by Beckhoff, but allows access to the own repository via HTTP as an exception.

5. Distributing the public key to the target computer

Copy the public key `myRSAPublic.key` to the target computer, for example to the directory `/usr/share/keys` using `scp`.

6. Using the repository on target computers

The user-defined repository or the internal web server can now be added as a target on the target computers and enabled. `TCBSD.conf` is the standard Beckhoff repository and is located in the directory at `/etc/pkg`. The file contains the following entries.

```
TCBSD: {
    url: https://tcbbsd.beckhoff.com/TCBSD/14/stable/packages
    enabled: true,
    signature_type: "fingerprints",
    fingerprints: "/usr/share/keys/bhf-pkg"
}
```

In the first step, copy the file `TCBSD.conf` and rename the file, for example to `TCBSD_original`, so that all changes can be undone quickly and the standard Beckhoff repository can be used again. This step is not mandatory and is only a safety measure.

In the next step, adjust the file `TCBSD.conf` at `/etc/pkg` so that the target computers can access their own web server in future:

```
TCBSD: {
    url: "http://my-webserver/pkg",
    enabled: true,
    signature_type: "pubkey",
    pubkey: "/usr/share/keys/myRSAPublic.key"
}
```

Note the last line, which contains the path to the public key on the target computer that you copied to the target computer in step 5. If necessary, adjust the path if you are using a different storage location.

8 Configuration

8.1 System information

TwinCAT/BSD provides different system information. The most important system information required for daily work with TwinCAT/BSD and TwinCAT can be retrieved with the tool `TcSysExe.exe` and the TwinCAT Registry

TcSysExe.exe

With `TcSysExe.exe` it is for example possible to control the TwinCAT mode from the console and to put TwinCAT into Run or Config mode. Call all available parameters with `TcSysExe.exe -help`. The following system information is particularly important:

- The command `TcSysExe.exe` or `TcSysExe.exe -version` lists information about the used TwinCAT build, the AMS Net Id and the TwinCAT/BSD version:

```
The software licenses can be found in this folder: /usr/local/etc/TwinCAT/3.1/System/Legal/
TcOsSys.dll: TcOsSys_Rel31_4024_20220407.2
TwinCAT Build: 3.1.4024.29
AMS Net Id: 5.66.247.12.1.1
TC/BSD: 13.0.11.1,2
```

- The command `TcSysExe.exe --osImageVersion` shows the TwinCAT/BSD version:

```
Administrator@CX-42F70C:~ $ TcSysExe.exe -osImageVersion
TC/BSD: 13.0.11.1,2
```

- The command `TcSysExe.exe --platformid` shows the TwinCAT 3 platform level of the Industrial PC used:

```
Administrator@CX-42F70C:~ $ TcSysExe.exe -platformid
HW Platform: 70
```

- The command `TcSysExe.exe --netid` displays the AMS Net Id of the Industrial PC:

```
Administrator@CX-42F70C:~ $ TcSysExe.exe -netid
AMS Net Id: 5.66.247.12.1.1
```

TcRegistry.xml

Extensive system settings are possible via the TwinCAT registry. The file is located in the directory `/usr/local/etc/TwinCAT/3.1/TcRegistry.xml` and can be opened and edited with `doas ee /usr/local/etc/TwinCAT/3.1/TcRegistry.xml`.

Before the `TcRegistry.xml` can be edited, the `TcSystemService` must be stopped with the command `doas service TcSystemService stop` and restarted with `doas service TcSystemService start` after editing.

The following system information is particularly important and can be edited in the XML file:

- `AmsNetID`: [Changing the AMS NetID](#) [► 110]
- `HeapMemSize`: [Increase heap memory](#) [► 112]
- `LockedMemSize`: [Adapting the router memory](#) [► 113]

8.2 User and rights management

There are three account types, each of which is subject to different restrictions and is fundamental for account management under TwinCAT/BSD. The following account types are available under TwinCAT/BSD:

- Superuser account
- User accounts
- System user

The **superuser account**, also called `root`, can operate without restrictions. Unlike normal user accounts, `root` has absolute control over TwinCAT/BSD. To ensure system integrity and security, `root` is disabled by default. This means it is not possible to log in directly as `root`.

User accounts are available for normal users who require access to TwinCAT/BSD. They are assigned a unique user name and a home directory and can customize their own user environment. The user `Administrator` is created by default. This user does not have conventional administrator rights like under Windows systems but has the authority to obtain root rights for certain purposes.

The **system users** can start services and programs such as mail or web servers. This makes it possible to restrict programs or services or to enable access rights for certain tasks.

Root rights

Since it is not possible to log in as `root`, users can be assigned root rights in order to operate without restrictions under TwinCAT/BSD. Use the `doas` command to obtain root rights. `doas` corresponds to the command `sudo`, a command known from other Unix-like operating systems.

Groups

TwinCAT/BSD allows user accounts to be grouped together so that their permissions for using individual functions or software can be managed centrally. Instead of assigning the same individual rights to many user accounts, a user role is defined that contains the rights to be assigned. The groups are identified by the group name and the group ID (gid). The TwinCAT/BSD kernel decides on the basis of the user ID (uid) and the group membership of a process whether or not it gives permission to the process.

The file `group` contains all group information, such as group name, group password, group ID and a list of members of the respective group. Call up the file with `cat /etc/group`:

```
wheel:*:0:root,Administrator
```

This excerpt shows the first line of the file `group`. The file is divided into four fields, separated by colons. The first field contains the group name (`wheel`), the second field contains an encrypted password (`*`), the third field contains the group ID (`0`) and the fourth field contains a list with the associated members (`root`, `administrator`).

Each user can determine their group affiliation with `id`. Here is an example for the user `Administrator`, who belongs to the groups `1001 Administrator` and `0 wheel`:

```
uid=1001(Administrator) gid=1001(Administrator) groups=1001(Administrator),0(wheel)
```

Use `doas pw groupadd <groupname>` to create a new group. Use `pw groupshow <groupname>` to display a group. Use the command `doas pw groupmod <groupname> -M <username>` to add a user to a group.

```
Administrator@CX-3B151A:~ % pw groupshow AI
AI:*:1007:Skynet,DeepThought,Ava,HAL
```

8.2.1 Create new user

Use the `adduser` command to create new users under TwinCAT/BSD. You will be guided interactively through the process. Only the superuser (`root`) can create new users. Therefore, you have to run the `adduser` command as administrator and with root rights.

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface](#) ► 65]).

Requirements:

- Access to the administrator account

Create new users as follows:

1. Enter the command `doas adduser` in the console.
Confirm the command with the administrator's password.
2. Specify a user name and enter the full name.

```
Username: NewUser
Full name: John Doe
```

3. In the next step you can set the `Uid`, `Login group`, the secondary `Login group` and the `Login class`. Press **[Enter]** to apply default settings.

```
Uid [Leave empty for defaults]:
Login group [NewUser]:
Login group is NewUser. Invite NewUser into other groups? []:
Login class [default]:
```

4. Under `Shell (sh csh tcsh nologin) [sh]`: select a shell with which the user logs in. By default, the Bourne shell (`sh`) is used for scripts and the command line. However, for direct interaction via the command line, `tcsh` is particularly suitable as a shell thanks to its better clarity and a larger range of functions.

```
Shell (sh csh tcsh nologin) [sh]: tcsh
```

5. For the following parameters press **[Enter]** to apply the default settings.

```
Home directory [/home/NewUser]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
```

6. In this step you can decide whether a password for the user should be generated randomly or whether you want to assign a password yourself.

```
Use a random password? (yes/no) [no]:
```

7. Then press **[Enter]** to apply the default settings under `Lock out the account after creation? [no]:.`

⇒ A summary is then issued. You can check your entries and create another user if required. In addition, the generated user password is displayed.

```
adduser: INFO: Successfully added (NewUser) to the user database.
adduser: INFO: Password for (NewUser) is: Luq39oGIwPhjT
Add another user? (yes/no): no
Goodbye!
```


8.2.2 Edit user information

The program `chpass` can be used to edit further user information. A superuser (root) has extended rights and can edit more fields than other users.

A superuser (root) can change the standard personal settings, the home directory, the Uid, Gid and the login name. In addition, security settings can be implemented by using `Change` to set the time for password changes or `Expire` to set an expiration time for the user account.

```
#Changing user information for NewUser.
Login: NewUser
Password: $6$qlX1/ZB/NGu9ulrF$.JYhoCPsGT6hk0GD34oJYWwOKGxY67ka818lpy/0HY.7XvXK69
JdeYotMkNjNQvqBTTfblYBQ3SZ.MYxChPeQ1
Uid [#]: 1002
Gid [# or name]: 1002
Change [month day year]:
Expire [month day year]:
Class:
Home directory: /home/NewUser
```

A standard user account can edit the first and last name, the shell and the contact information.

```
#Changing user information for NewUser.
Shell: /bin/tcsh
Full Name: John Doe
Office Location:
Office Phone:
Home Phone:
Other information:
```

Edit user information as follows:

1. Enter the command `chpass NewUser` in the console. Or enter `doas chpass NewUser` if you want to edit the information as superuser (root).
 2. Use the arrow keys to navigate to the line you want to edit.
 3. Press **[Esc]** to switch the editor to command mode.
 4. Delete preset values with **[x]**.
 5. Press **[i]** to add new text to the cursor.
 6. Press **[Esc]** and type `:wq` in the console to save the changes and exit the editor. Or type `q!` to exit the editor without saving.
- ⇒ Afterwards, a message is issued to indicate that all changes have been saved successfully. If the entries are incorrect, an error message is displayed with a reference to the location.

```
/etc/pw.6RnflE: 15 lines, 412 characters.
chpass: upper-case letters are dangerous in a login name
chpass: user information updated
Administrator@CX-3B151A:~ %
```

8.2.3 Deleting a user

The program `rmuser` can be used to completely remove user accounts from the system. User accounts can only be deleted by the superuser (root).

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface](#) [► 65]).

Requirements:

- Access to the administrator account

Proceed as follows:

1. Enter the command `doas rmuser NewUser` in the console.
Confirm the command with the administrator's password.
2. The user account is displayed. Check the data before continuing.

```
Administrator@CX-3B151A:~ % doas rmuser NewUser
Password:
Matching password entry:

NewUser:$6$q1X1/ZB/NGu9ulrF$.JYhoCPsGT6hk0GD34oJYWwOKGxY67ka818lpy/0HY.7XvXXK69Jd
eY0tMkNjNQvqBTTfblYBQ3SZ.MYxChPeQ1:1002:1002::0:0:John Doe,,555 433423:/home/Ne wUser:/bin/tcsh

Is this the entry you wish to remove? Yes
```

3. Remove the user account with `yes`.

```
Remove user's home directory (/home/NewUser)? Yes
Removing user (NewUser): mailspool home passwd.
Administrator@CX-3B151A:~ %
```

4. Remove the home directory with `yes`.

⇒ The user account is deleted, along with the home directory. Entries from groups, temporary file storage areas and emails are removed. All processes initiated by the user are terminated.

8.3 Integrating a USB stick

This section shows you how you can integrate a USB stick in TwinCAT/BSD. Check the USB configuration beforehand. To do this, use the command `dmesg` to check whether the USB stick appears in the system messages:

```
umass0: <Generic Mass Storage, class 0/0, rev 2.00/1.01, addr 2> on usb0
umass0: SCSI over Bulk-Only; quirks = 0x4101
umass0:1:0: Attached to scbus1
da0 at umass-sim0 bus 0 scbus1 target 0 lun 0
da0: <Generic Flash Disk 8.07> Removable Direct Access SPC-2 SCSI device
da0: Serial Number 2EFBC899
da0: 40.000MB/s transfers
da0: 3900MB (7987200 512 byte sectors)
da0: quirks=0x2<NO_6_BYTE>
```

The details of the make, device file (da0), speed and capacity may vary depending on the device.

Requirements:

- The USB stick is formatted with FAT32.
- Connect the USB stick to the Industrial PC.

Proceed as follows:

1. Enter the command `gpart show` to search for a FAT partition on the USB stick. The USB stick in this example has a FAT32 partition and a size of 3.8 GB. See entry: 1 fat32 (3.8G)

```
Administrator@CX-3B151A$ gpart show
=>      40  7728256  ada0  GPT  (3.7G)
        40    409600      1  efi  (200M)
        409640    2008      - free -  (1.0M)
        411648  7315456      2  freebsd-zfs (3.5G)
        7727104    1192      - free -  (596K)

=>      63  7987137  da0   MBR  (3.8G)
        63      737      - free -  (369K)
        800  7986400      1  fat32   (3.8G)
Administrator@CX-3B151A$
```

2. Enter the command `ls /dev/da0*` to determine the name of the FAT32 partition.
da0s1 or da0p1.

```
ls /dev/da0s1
/dev/da0 /dev/da0s1
```

3. Enter the command `doas mkdir /mnt/usb` to create the directory `/mnt/usb`.
 4. Enter the command `doas mount -t msdosfs /dev/da0s1 /mnt/usb` in order to mount the USB flash drive under `/mnt/usb`.
 5. Use the command `cd /mnt/usb` to navigate to the mount point of the USB flash drive.
 6. Use the command `ls` to display all the folders in the directory.
- ⇒ You have successfully integrated a USB stick in TwinCAT/BSD. The settings are not permanently saved. Following a restart the USB stick must be integrated again.

USB devices can also be integrated automatically. This function was disabled ex works in order to increase the safety of TwinCAT/BSD. Further information on the automount service can be found at: <https://www.freebsd.org/doc/handbook/usb-disks.html>

8.4 Enable Automount for external data carriers

External data carriers are not automatically integrated under TwinCAT/BSD in order to increase the security of the system. The Automount function is disabled ex factory. Normally, external data carriers must be actively and manually integrated into TwinCAT/BSD, but the settings are not permanently saved and must be mounted again after a restart.

This chapter describes how to automatically and permanently mount external data carriers, such as USB sticks, optical drives, or external hard disks, with `devd-mount`. The data carrier must use a file system that is recognized by FreeBSD, such as EXFAT, FAT32, FAT16, NTFS, UFS, ISO 9660 (CDs), UDF (DVDs).

Useful commands:

- `gpart show`: Show found partitions.
- `mount`: Display which data carrier with which file system was mounted at which position.

The access authorizations can be adjusted in the configuration file `/usr/local/etc/devd-mount.conf`. By default, read/write and execute permissions are granted for the user `root` and group `operator`. For all other users, only read and execute is permitted.

Requirements:

- TwinCAT/BSD version: 14.x.xx.x, x

Proceed as follows:

1. Install the script `devd-mount` on the system with `doas pkg install devd-mount`
 2. Restart the service `devd` after installation with `doas service devd restart`.
 3. Connect the external data carrier to the industrial PC.
 4. By default, the data carrier is mounted under `/media` with its device node designation, i.e. with the name with which the data carrier is listed under `/dev`.
- ⇒ If the data carrier is removed, the mount points are retained at `/media`. If the data carrier is reinserted in the meantime or after a restart, it is mounted again at the same position.

If an additional data carrier is mounted in the meantime and the original data carrier is reinserted, the device node designation may change and a data carrier is mounted at `/media/dal` instead of `/media/da0`, for example.

8.5 Configuring the UPS software

This chapter describes how to configure the Beckhoff UPS CU81x0-0xx0 for communication via the USB interface. The required UPS software or the package `TcUpsSoftware` is already included in the current TwinCAT/BSD version. For older versions, the package can be installed later using `doas pkg install TcUpsSoftware`.

Note that the UPS software is pre-installed, but the UPS service is not started automatically at system startup. The behavior of the UPS in the event of a power failure is defined in the configuration file `TcUpsSoftware.conf` under `/usr/local/etc/TcUpsSoftware.conf`

Table 6: UPS software: Settings in the configuration file.

Option	Value	Description
ShutdownOnBatteryEnable	[0 or 1]	The option is enabled by default [=1]. In the event of a power failure, the industrial PC is shut down properly.
ShutdownOnBatteryWait	[0 ... 43200 seconds]	If <code>ShutdownOnBatteryEnable</code> is enabled, the industrial PC is shut down after the set time in seconds.
TurnUpsOffEnable	[0 or 1]	If this option is enabled [=1], the UPS is shut down after the industrial PC has been shut down.
TurnUpsOffWait	[0 ... 600 seconds]	If <code>TurnUpsOffEnable</code> is enabled, the UPS is shutdown after the industrial PC has been shut down and the time set in this option has elapsed.

Proceed as follows:

1. Open the configuration file `rc.conf` with the command `doas ee /etc/rc.conf`
 2. Add line `TcUpsSoftware_enable="YES"` to the configuration file so that the service is started automatically each time the system is started.
 3. Open the configuration file `TcUpsSoftware.conf` with the command `doas ee /usr/local/etc/TcUpsSoftware.conf` and enter the required values for the UPS hold time.
 4. Start the UPS service with the command `doas service TcUpsSoftware start`
- ⇒ The UPS software controls the Beckhoff UPS CU81x0-0xx0 in the event of a power failure according to the settings in the configuration file `TcUpsSoftware.conf` and can be disabled again for the current session with the command `doas service TcUpsSoftware stop`.

If the UPS service is to be disabled permanently, the entry `TcUpsSoftware_enable="YES"` must be removed from `rc.conf`.

8.6 Disable real-time Ethernet

This chapter demonstrates how to disable real-time Ethernet if you do not need real-time communication and want to use the interfaces instead for traditional full-bandwidth Ethernet communication.

Real-time Ethernet is enabled by default. Use the command `TcRteConfig show` to call up the current configuration for the interfaces:

```
Administrator@CX-3B151A:~ % TcRteConfig show
sysctl:
dev.igb.1.iflib.tc_rte.mode: 0
dev.igb.1.iflib.support_tc_rte: 1
dev.igb.1.iflib.disable_tc_rte: 0
dev.igb.0.iflib.tc_rte.mode: 0
dev.igb.0.iflib.support_tc_rte: 1
dev.igb.0.iflib.disable_tc_rte: 0
```

Depending on the device, different network adapters can be displayed. In the example shown here, the network adapters are called `igb0` and `igb1`.

Disable real-time Ethernet as follows:

1. Enter the command `doas TcRteConfig disable igb.1` in the console.
Real-time Ethernet is disabled for the `igb.1` interface.

```
/boot/device.hints:  
dev.igb.1.iflib.disable_tc_rte="1"  
Administrator@CX-3B151A:~ %
```

2. Restart the Industrial PC with the command `shutdown -r now` to apply the settings.

⇒ After the restart, real-time Ethernet is disabled for the `igb.1` interface. The command `doas TcRteConfig enable igb.1` can be used to re-enable real-time Ethernet. Here, too, the settings are only applied after a restart.

8.7 Start services automatically (Autostart)

This section shows how to start applications and services automatically after booting. A suitable entry in the `/etc/rc.conf` file is required at TwinCAT/BSD. The Mosquitto MQTT broker is shown as an example.

The `/etc/rc.conf` file generally contains information about the system configuration such as the local host name, configuration details for possible network interfaces and which services should be started when the system starts up.

This step works only with applications and services that bring appropriate rc scripts to be started as a service or daemon. The scripts are stored at `/etc/rc.d`. For your own applications without such rc scripts, there is further information on how to create these rc scripts at <https://docs.freebsd.org/en/articles/rc-scripting/>.

Requirements:

- Mosquitto MQTT broker.

Proceed as follows:

1. Enter the `doas ee /etc/rc.conf` command in the console.
The `rc.conf` file opens.
 2. In the editor navigate to the end of the file and create the following entry:
`mosquitto_enable="YES"`
 3. Press **[Esc]** and save the changes.
- ⇒ The Mosquitto MQTT broker is automatically started at the next system startup.

8.8 Changing the shell

A shell is a command line interface that enables the user to interact with TwinCAT/BSD and execute commands. TwinCAT/BSD already contains some shells at delivery, including the `sh` shell set as standard. The user can switch to another shell at any time, for example the `tcsh` shell, or install additional shells.

The shells differ in the built-in functions or are preferred by users if they make their daily work easier, for example by auto-completion of file names. Which shell is used by the user is in the end a question of personal preference.

All existing shells in the system are listed under `/etc/shells`.

Proceed as follows:

1. Enter the command `chsh -s tcsh` in the console to switch to the `tcsh` shell.
 2. Log in again with the command `login <username>`.
- ⇒ The `tcsh` shell is now selected as default shell for the logged in user. You can change the shell again at any time with `chsh -s`.

8.9 Change keyboard language

If you want to change the keyboard language, the quickest way is to use the `kbdmap` command and it applies to the current session. Create an entry in the `rc.conf` file, so that the language settings remain even after a restart.

Proceed as follows:

1. Open the `rc.conf` file with the `doas ee /etc/rc.conf` command
 2. Navigate with the arrow keys to the end of the file and add the following line:
`keymap="de.kbd".`
 3. Use an appropriate country code. In this sample, `de` is used as the country code for German.
- ⇒ The language settings in the `rc.conf` file are permanently applied. Restart TwinCAT/BSD with `shutdown -r now` for the changes to take effect. Remove the entry to restore the default settings.

8.10 Synchronize time with NTP

The internal time of an industrial PC is never completely accurate, so the Network Time Protocol (NTP) provides a way to determine and set the exact time. To get the current time, an NTP client is used to synchronize the local system time with a time server. For this purpose, Beckhoff offers a global NTP server pool which provides the current time:

```
ntp.beckhoff-cloud.com
```

TwinCAT/BSD is preconfigured to use this NTP server pool to determine the current time. The configuration of the NTP client is performed via the configuration file:

```
/etc/ntp.conf
```

You will find the entry for the Beckhoff NTP server pool `ntp.beckhoff-cloud.com` there. If you want to use your own NTP server, replace the entry with the address of your NTP server.

Notice If it is a Windows NTP server, the entry `tos maxdist 30` in the configuration file `/etc/ntp.conf` is also required. The distance at which an NTP server can be located and how long a packet takes to be transmitted is specified in "*tos maxdist*". The Windows time server only works if the maximum value is specified, in this case this is "30".

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface](#) [► 65]).

The screenshot shows the Beckhoff Device Manager web interface. On the left is a sidebar with navigation icons for Device, Hardware, Software (highlighted in red), TwinCAT, and Security. In the center, there are icons for OS, System (highlighted in red), and Filesystem. The main panel displays the 'NTP Server' configuration. It includes a 'Servername' field with 'ntp.beckhoff-cloud.com', a 'Refresh Rate' dropdown set to '30 minutes', and a 'Local Date/Time' section with fields for Date (17.05.2022), Time (07:49:14), and UTC Offset (+00:00). Below this is a 'Timezone' section with a dropdown set to 'UTC'. Each main section has a status icon (power, checkmark, or X).

Fig. 16: NTP server settings in the Beckhoff Device Manager.

Beckhoff NTP server pool

The Beckhoff NTP server pool is a global server pool with several time servers per geographical region. To use only the best available servers for your region, simply use the global address `ntp.beckhoff-cloud.com` and the right servers for your region will automatically be made available.

This service is only available for Beckhoff Industrial PCs and may only be used elsewhere with the express permission of Beckhoff. Beckhoff does not guarantee the continuous availability of the time servers. Accordingly, Beckhoff assumes no liability for a time server failure.

<https://www.beckhoff.com/ntp-pool>

9 Remote access

9.1 Beckhoff Device Manager: web interface

With the Beckhoff Device Manager, Beckhoff Industrial PCs can be diagnosed and important system values monitored in order to avoid downtimes. With a sophisticated system diagnostics, it is thus possible to detect critical conditions at an early stage, such as an impending heat collapse due to the failure of a fan or insufficient cooling in the control cabinet. The Beckhoff Device Manager is only available for Beckhoff Industrial PCs, since it requires a customized BIOS.

Both remote access from another PC and access from a PLC to the Beckhoff Device Manager are supported. This allows the Device Manager to be configured easily and intuitively.

Web interface

The web interface can be accessed via a standard web browser. To do this, enter the IP address or the host name of the industrial PC in the search bar.

- Example with IP address: <https://169.254.136.237>
- Example with host name: <https://CX-16C2B8> or with BTN number for new devices: <https://BTN-000zf650>



Fig. 17: Start page of the Beckhoff Device Manager.

On the start page of the Beckhoff Device Manager you can either start the Device Manager or the TwinCAT/BSD web console. The web console gives you access to the TwinCAT/BSD console, allowing you to operate the industrial PC without a monitor.

Access to the web interface is protected by the login data and the default password of the administrator. You should change this password to prevent unauthorized access to the system. Access data on delivery:

- User name: Administrator
- Password: 1

The host PC and the industrial PC must be in the same network and the firewall must allow access via port 443 (HTTPS). Port 443 is enabled ex works. Depending on the structure and configuration of your network (proxy server, etc.), the host name may not be resolved. We therefore recommend using the IP address of the industrial PC.

Device Manager: First page

The Beckhoff Device Manager is started after login. The first page provides a basic overview of the device. From here you can directly access the hardware, software, TwinCAT and security sections. This allows you to check the hardware and software in a targeted manner.

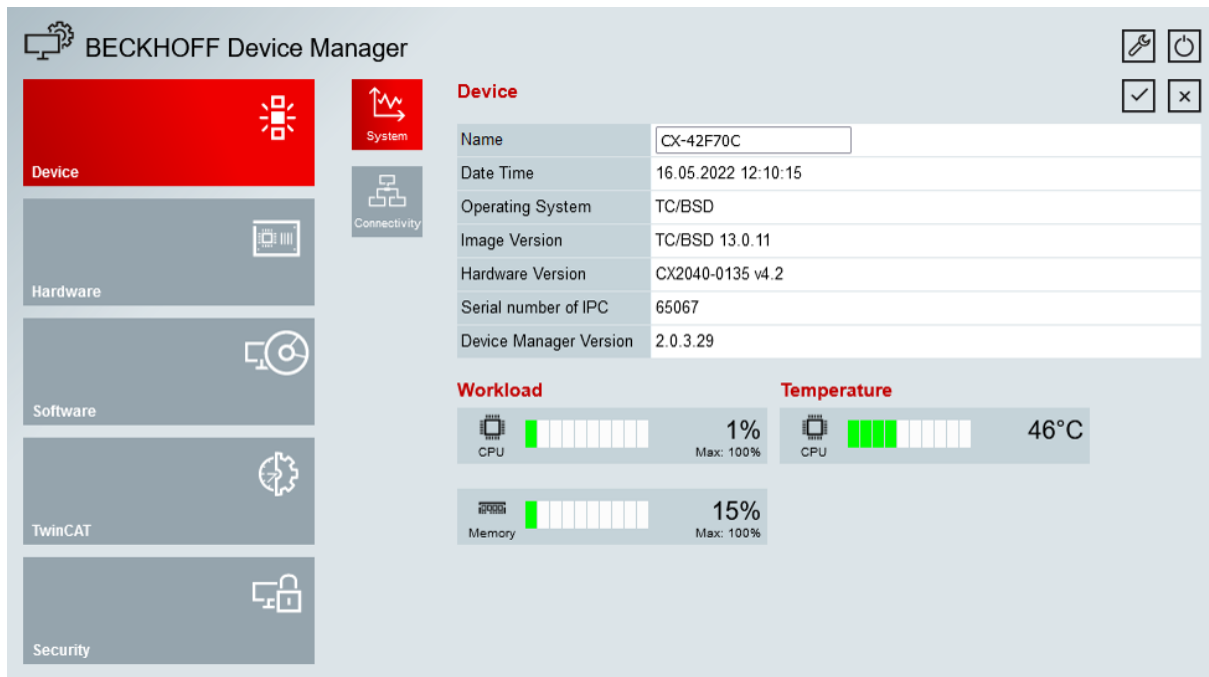


Fig. 18: First page of the Beckhoff Device Manager.

From the home page, navigate further in the menu to configure the industrial PC. Note that modifications only become active once they have been confirmed. It may be necessary to restart the industrial PC.

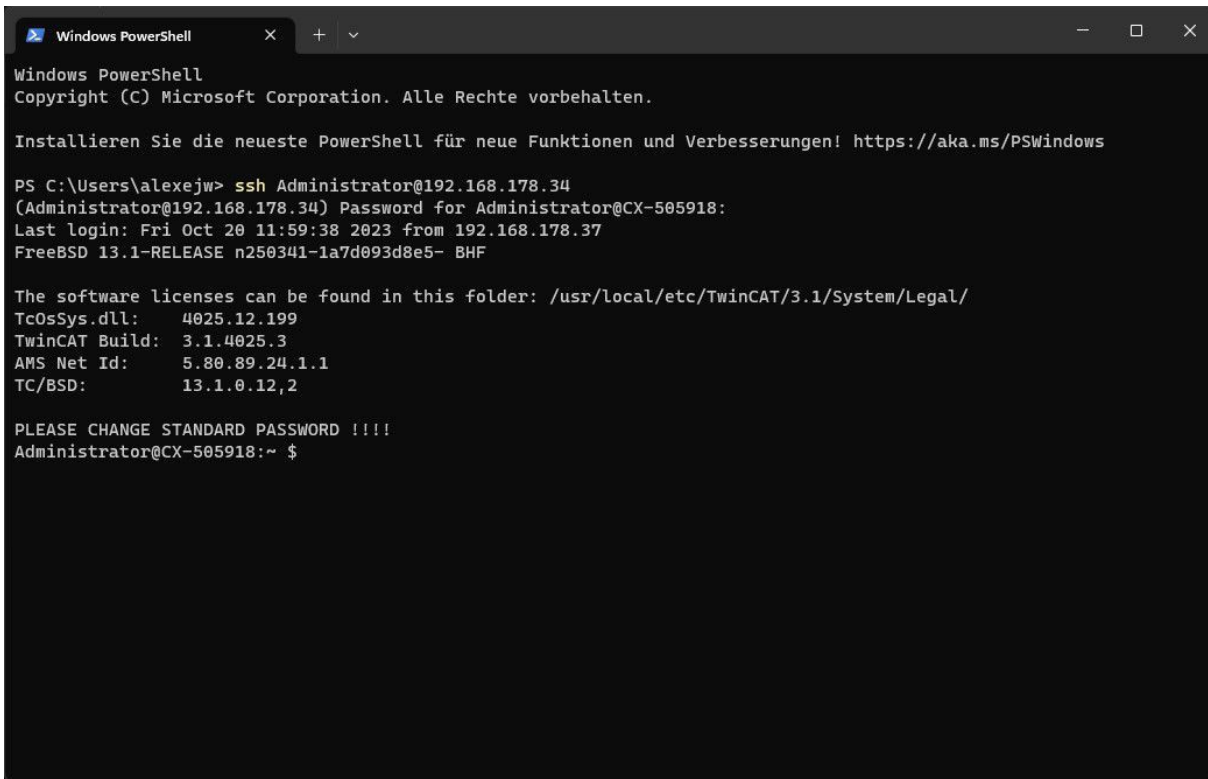
Setting options

The following information and settings are available for TwinCAT/BSD:

- General device information: Listing of the TwinCAT/BSD and the TwinCAT version. In addition to this, information about the load of the CPU and the main memory is displayed.
- Network settings: The network settings can be changed independently of the console and, for example, the IP addresses of the Ethernet interfaces can be changed or DHCP can be enabled or disabled.
- Storage media and file system: Both the free storage space and the lifetime of the storage media are displayed.
- TwinCAT/BSD packages: Listing of the installed Beckhoff packages with version information, which are either relevant for TwinCAT or the TwinCAT/BSD operating system.
- TwinCAT: In the TwinCAT menu ADS routes can be managed and new ADS routes can be created.
- User management: New users or user groups can be created.

9.2 Remote access with SSH

If you have an SSH client on your host PC, you can use this for the connection, as OpenSSH is now available by default on all recent operating systems. For example, in the current Windows 10 versions, this is a Windows function that can be used via PowerShell. To do this, use the command `ssh Administrator@<bsd-ip>`.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Alle Rechte vorbehalten.

Installieren Sie die neueste PowerShell für neue Funktionen und Verbesserungen! https://aka.ms/PSWindows

PS C:\Users\alexewjw> ssh Administrator@192.168.178.34
(Administrator@192.168.178.34) Password for Administrator@CX-505918:
Last login: Fri Oct 20 11:59:38 2023 from 192.168.178.37
FreeBSD 13.1-RELEASE n250341-1a7d093d8e5- BHF

The software licenses can be found in this folder: /usr/local/etc/TwinCAT/3.1/System/Legal/
TcOsSys.dll: 4025.12.199
TwinCAT Build: 3.1.4025.3
AMS Net Id: 5.80.89.24.1.1
TC/BSD: 13.1.0.12,2

PLEASE CHANGE STANDARD PASSWORD !!!!
Administrator@CX-505918:~ $
```

Fig. 19: Remote access via SSH using Windows PowerShell.

If you experience problems in establishing a connection via SSH, you can edit the SSH settings and comment out restrictive SSH settings (see: [Editing the SSH settings](#) [► 72]).

Access via PuTTY

PuTTY is an open source software with which a connection can be established via Secure Shell (SSH), Telnet, remote login or a serial interface.

Use PuTTY to establish an SSH connection to an industrial PC with TwinCAT/BSD when running Windows. After successful connection a console is started with which remote access commands can be issued that are subsequently executed on the industrial PC.

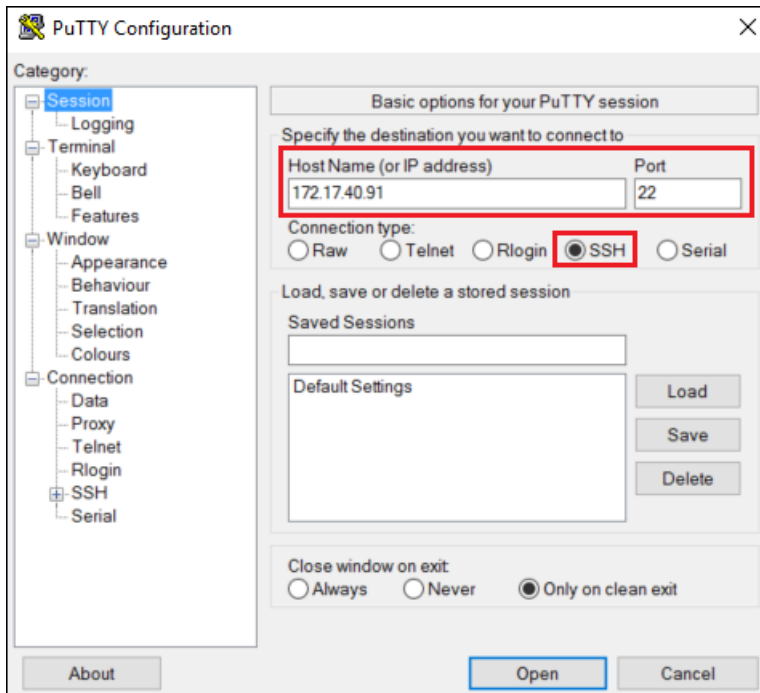
Requirements:

- Download PuTTY client at: <https://www.putty.org/>
Required minimum version: 0.70
- The local PC (development computer) and the industrial PC (TwinCAT/BSD) must be connected the same network or directly to each other via an Ethernet cable.

Start an SSH connection as follows:

1. Start the PuTTY client.

2. Enter the host name or the IP address of the industrial PC in **Host Name (or IP address)**.



3. Enter the appropriate port in **Port**. For SSH this is usually port 22.
4. Activate the option **SSH** under **Connection type** and click **Open**. The console is started.

login as:

5. Enter the login data for TwinCAT/BSD. Default:
Login: Administrator
Password: 1

⇒ You have successfully established an SSH connection to an industrial PC with TwinCAT/BSD using the PuTTY client.

9.3 Managing files with the WinSCP client

9.3.1 Starting and using the WinSCP client

WinSCP (Windows Secure Copy) is an open source software with which a connection can be established via FTP, FTPS, SCP, SFTP, WebDAV or S3h.

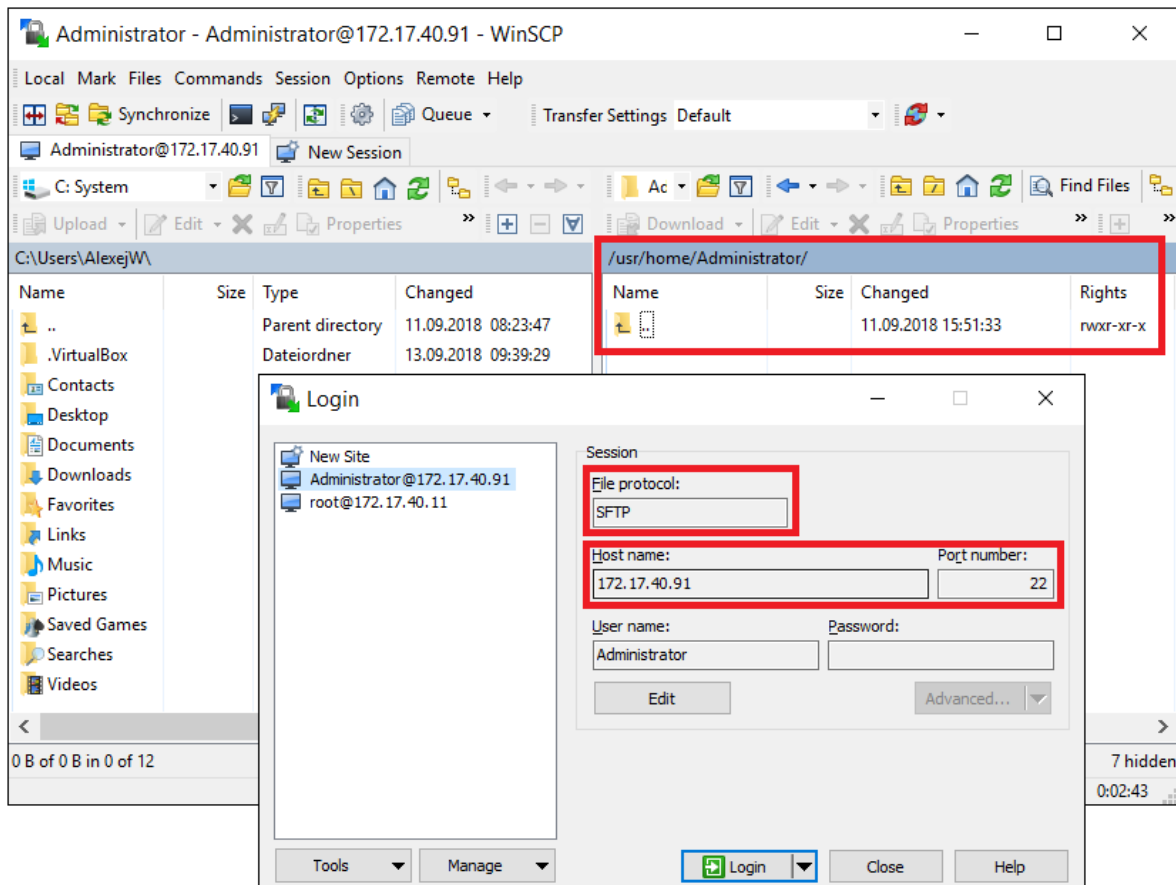
Use WinSCP to establish an SFTP connection to an Industrial PC with TwinCAT/BSD when running Windows. Following successful connection, a graphical user interface is started with which it is possible to securely transfer data and files to the Industrial PC with TwinCAT/BSD. With WinSCP, files can be copied into TwinCAT/BSD directories, new directories can be created and files can be edited.

Requirements:

- Download WinSCP from: <https://winscp.net/>
Minimum version required: 5.13.2
- The local PC (development computer) and the Industrial PC (TwinCAT/BSD) must be connected the same network or directly to each other via an Ethernet cable.

Start a connection as follows:

1. Start the WinSCP client.
The login window appears.



2. Select the SFTP protocol in **File protocol**.
 3. Enter the IP address and the port number of the Industrial PC in **Host Name** and **Port number**.
 4. Enter the login data for TwinCAT/BSD and click on **Login**.
- ⇒ You have successfully established an SFTP connection to an Industrial PC with TwinCAT/BSD and can securely transfer data and files to the Industrial PC. The TwinCAT/BSD directories are shown on the right-hand side of the graphical user interface.

9.3.2 WinSCP as the root

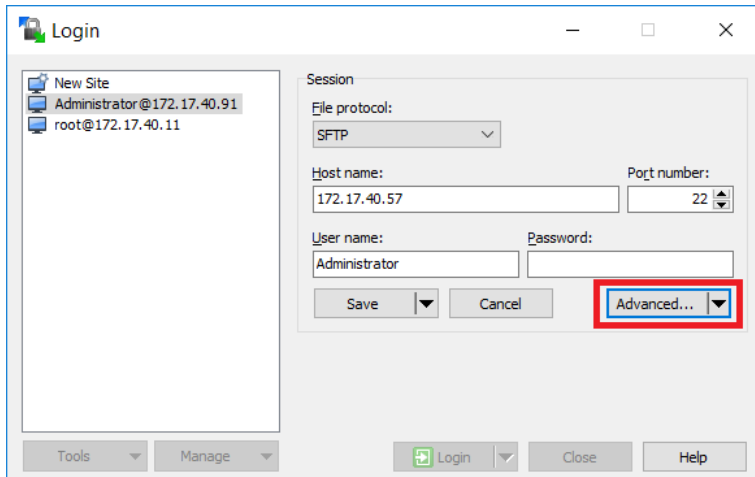
With the entry `doas /usr/libexec/sftp-server`, WinSCP starts the SFTP server with root rights. This allows additional settings to be made and configuration files to be adapted.

Requirements:

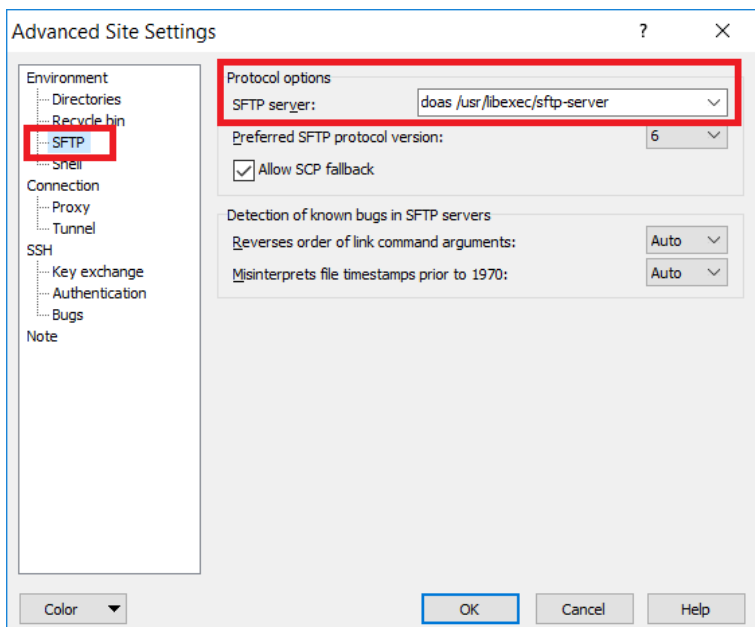
- WinSCP client has been installed.

Proceed as follows:

1. Start the WinSCP client.
The login window appears.
2. Click on **Advanced** to open further settings.



3. Click on **SFTP** in the tree view on the left and enter the value `doas /usr/libexec/sftp-server` in **SFTP server**.



4. Save the settings for the administrator account.

⇒ Then log in with the administrator account. You now have access with root rights via WinSCP.

9.3.3 Opening and editing files

With the WinSCP client you can open and edit files with the help of a graphical interface. Please note that you can only edit files for which you have the required access rights.

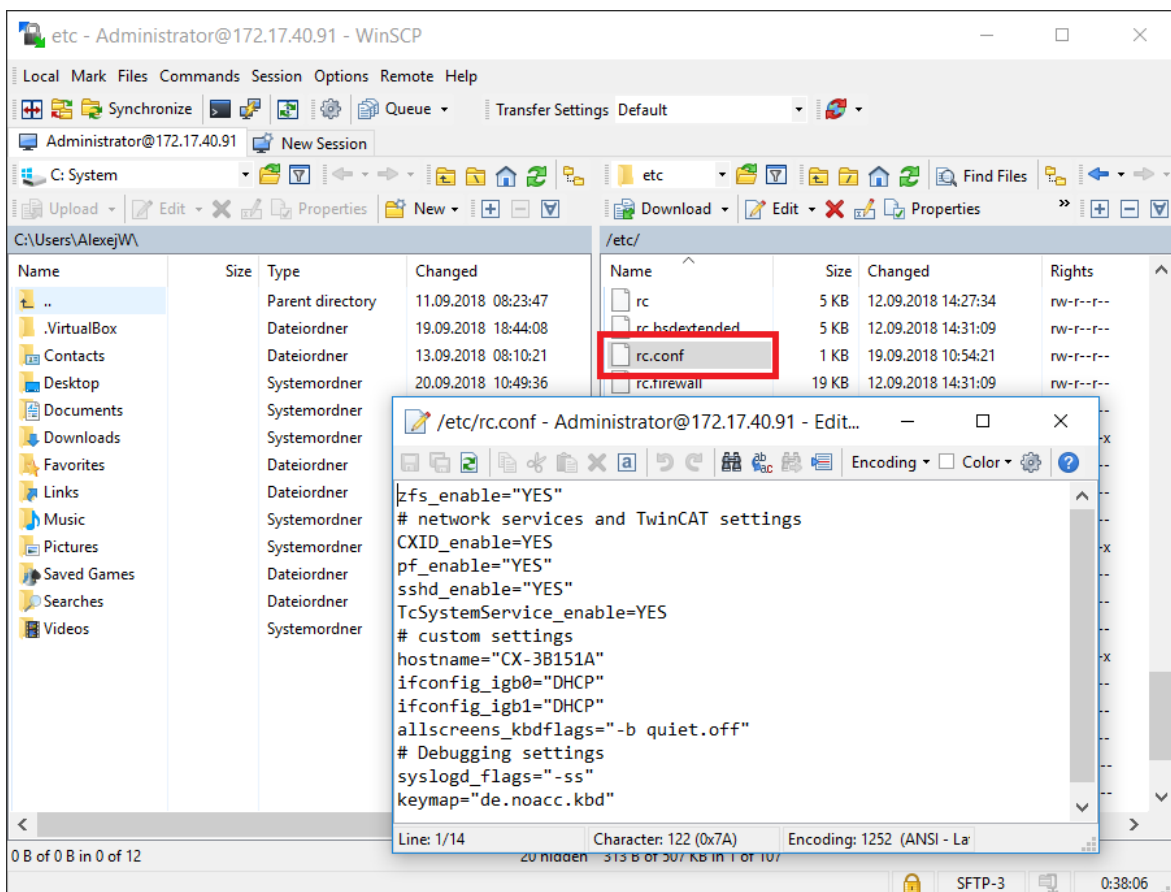
Taking the configuration file `rc.conf` as an example, this work step shows how files can be opened and edited with the WinSCP client.

Requirements:

- WinSCP-Client (see: [Starting and using the WinSCP client \[► 69\]](#)).
- Activate root rights for WinSCP (see: [WinSCP as the root \[► 70\]](#)).

Proceed as follows:

1. Start the WinSCP client.
The login window appears.
2. Enter the login data for TwinCAT/BSD and click on **Login**.
3. Navigate to the directory `/etc` and double-click on the file `rc.conf`.
The file is opened in the WinSCP editor.



4. Alternatively you can right-click on the file and open it with the editor of your choice.
 5. As soon as you save changes, the changes will be transferred to TwinCAT/BSD.
- ⇒ You have successfully opened and edited a file. In this way you can manage all files with the WinSCP client.

9.4 Editing the SSH settings

SSH is restrictively configured in TwinCAT/BSD. Current encryption methods are used. If you experience problems in establishing a connection via SSH, you can edit the SSH settings and comment out restrictive SSH settings.

Note that in doing so the restrictive settings from Beckhoff for a secure network connection will be canceled. Beckhoff recommends the use of a different software for an SSH connection to the TwinCAT/BSD or to update the existing software.

Requirements:

- Access rights to the file `sshd_config`

Proceed as follows:

1. Enter the command `doas ee /etc/ssh/sshd_config` in the console.
The file `sshd_config` is opened.

2. Comment out the following four lines to cancel the restrictive SSH settings.

```
#Ciphers chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-  
gcm@openssh.com,aes256-gcm@openssh.com  
#HostKeyAlgorithms ssh-rsa,rsa-sha2-256,rsa-sha2-512,ssh-ed25519  
#KexAlgorithms diffie-hellman-group14-sha256,diffie-hellman-group16-sha512,diffie-hellman-  
group18-sha512,curve25519-sha256,curve25519-sha256@libssh.org  
#MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm@openssh.com,umac-128-etm@openssh.com
```

3. Restart the SSH server with the command `doas service sshd restart` in order to confirm the settings.

⇒ You can restore the restrictive SSH settings at any time by removing the comments again.

10 TwinCAT/BSD Hypervisor

The TwinCAT/BSD Hypervisor enables virtual machines to run under TwinCAT/BSD.

The TwinCAT/BSD Hypervisor is based on the FreeBSD hypervisor `bhyve(4)`. The integration of `bhyve` in TwinCAT/BSD enables simultaneous and efficient execution of virtual machines and TwinCAT PLC and motion applications on the same industrial PC.

This documentation provides an overview of various features of the TwinCAT/BSD Hypervisor.

10.1 Device and feature support

Running virtual machines with `bhyve(8)` requires an industrial PC with a current Intel®- or AMD™ CPU that supports hardware-assisted virtualization.

The table [Device support for TwinCAT/BSD Hypervisor, device and GPU passthrough](#), [73] gives an overview of current industrial PCs that enable the execution of virtual machines with the TwinCAT/BSD Hypervisor and whether the Device passthrough or GPU passthrough features are supported. This overview is a technical information. It does not indicate whether ordering options for pre-installed virtual machines are available for that device. The available ordering options can be found in the price list or on the Beckhoff homepage.

Table 7: Device support for TwinCAT/BSD Hypervisor, device and GPU passthrough.

TwinCAT/BSD devices	Hypervisor	Device passthrough	GPU passthrough
CX51x0	Yes	No	No
CX52x0	Yes	Yes	No
CX20x2	Yes	Yes	No
CX20x3	Yes	Yes	No
C601x-0010	Yes	No	No
C601x-0020	Yes	Yes	No
C601x-0030	Yes	Yes	No
C602x-0000	Yes	Yes	No
C602x-0010	Yes	Yes	Yes
C603x-0060	Yes	Yes	No
C603x-0070	Yes	Yes	Yes*
C603x-0080	Yes	Yes	Yes
C6040-0090	Yes	Yes	Yes

* BIOS version ≥ 0.37 and activation of power management (PM Support) in BIOS required.

10.2 Start and manage virtual machines

Virtual machines are started and managed with the programs `bhyve` and `bhyvectl`. Before a virtual machine can be started with `bhyve`, the kernel module `vmx.ko` must be loaded:

```
doas kldload -n vmx.ko
```

So that this step does not have to be repeated after each restart, the kernel module can already be loaded during the system startup of TwinCAT/BSD by setting `vmx_load="YES"` in the `/boot/loader.conf`:

```
doas sysrc -f /boot/loader.conf vmx_load="YES"
```

Once the kernel module is loaded, a virtual machine can be started by calling `bhyve`:

```
bhyve [OPTIONS] <vm_instance>
```

The parameters `[OPTIONS]` determine the configuration of the virtual machine, which can be used, for example, to specify the number of virtual CPUs used, the size of the main memory or the storage location of local files. The last parameter `<vm_instance>` of the call specifies the virtual machine instance name.

Start VM instance with simple basic configuration

A UEFI-based virtual machine with two virtual CPUs and 2 GB of main memory can be started with the following command:

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 2G \
-s 0,hostbridge \
-s 31,lpc \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd \
-l com1,stdio \
-A -H -P \
samplevm
```

In simplified terms, the basic configuration of the VM instance is as follows:

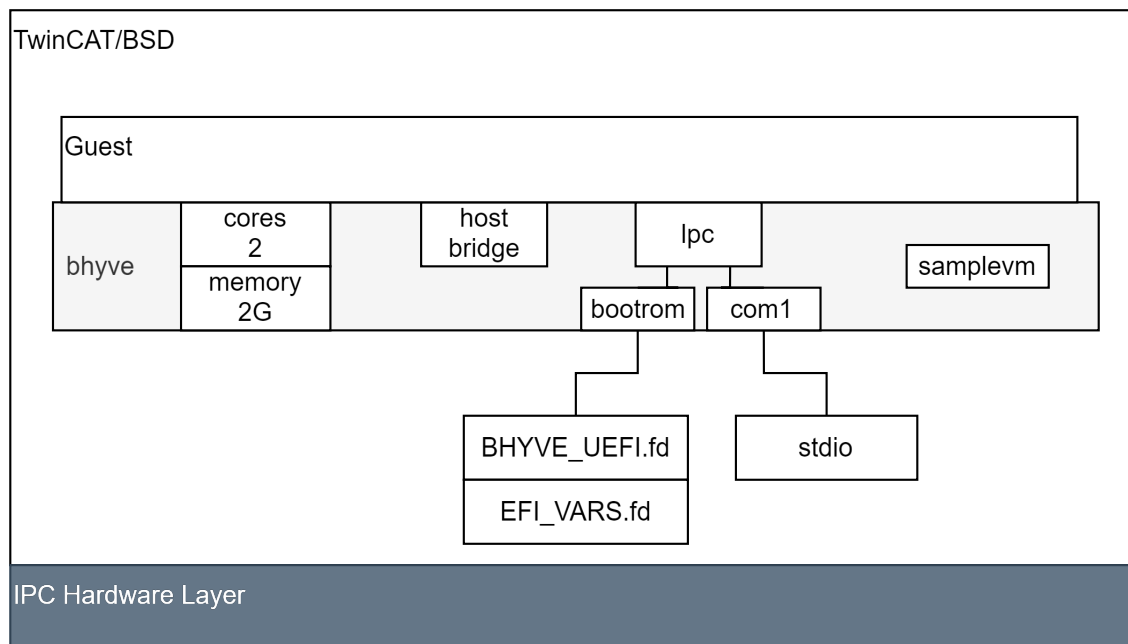


Fig. 20: Basic configuration of a VM instance.

As soon as the Bhyve process is started with the parameters, the virtual machine with the instance name `samplevm` is available. Bhyve starts the UEFI firmware stored as `bootrom`. The UEFI shell is then output via the virtual interface `com1` and redirected to the standard streams of the TwinCAT/BSD host via the parameter `stdio`, so that the UEFI shell is displayed on the command line:

```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70 (BHYVE, 0x00010000)
map: No mapping found.
Press ESC in 1 seconds to skip startup.nsh or any other key to continue.
Shell>
```

By entering `reset -s` in the UEFI shell, the virtual machine can be shut down again. The Bhyve process is then terminated with the return value 1. To restart the virtual machine, the complete Bhyve command must be called with the same parameters.

Explanation of the parameters

The meaning of the individual parameters can be called with `bhyve -h`. Detailed descriptions of the parameters can be found in the manual for [bhyve](#). Alternatively, the manual can be called up via the command line with the command `man bhyve`.

The parameters used in the sample above are briefly explained below.

Parameter	Description
<code>-c sockets=1,cores=2,threads=1</code>	Configuration of the virtual CPU topology. In this sample, a CPU socket with two cores and one thread per core.

Parameter	Description
-m 2G	Main memory available to the virtual machine. In this sample 2 GB.
-s 0,hostbridge	A virtual host bridge to connect the virtual CPU to the virtual PCI bus. By convention, the host bridge should always be configured at PCI address -s 0:0:0 (-s 0 for short).
-s 31,lcp	LPC/PCI ISA bridge for connecting emulated LPC devices. By convention, the LPC/PCI ISA bridge should always be configured at PCI address -s 0:31:0 (-s 31 for short).
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd	An emulated bootrom on the LPC bus. The UEFI firmware is passed as ROM in the file /usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd.
-l com1,stdio	A serial interface on the LPC bus whose inputs and outputs are redirected to bhyve's standard input and output streams.
-A	Creates bhyve ACPI tables for the virtual machine.
-H	Releases the virtual CPU thread when an HLT instruction is detected. Otherwise, the virtual CPUs will use 100% of the host CPUs.
-P	Forces the virtual guest CPU to terminate when a PAUSE instruction is detected.

Parameters starting with `-s` are used to configure virtual PCI slots to which in turn emulated PCI devices can be assigned (for examples see: [Advanced VM configuration](#) [► 78]). Parameters starting with `-l` are used to configure emulated LPC devices behind the LPC/PCI-ISA bridge.

Manage virtual machines

Started virtual machines are listed as Bhyve processes on the TwinCAT/BSD host. Accordingly, running virtual machines can be listed with `ps (1)`:

```
ps -a | grep bhyve
7048 0 SC 0:31.06 bhyve: samplevm (bhyve)
7642 0 SC 0:01.83 bhyve: debian11 (bhyve)
```

Running virtual machines can be shut down or terminated via the TwinCAT/BSD host by sending signals via `kill (1)` to the respective Bhyve process. The TERM signal can be used to send an ACPI shutdown request to the virtual machine to trigger the virtual machine shutdown:

```
doas kill -s TERM $(pgrep -f "bhyve: samplevm")
```

If the virtual machine does not respond to ACPI shutdown requests, the KILL signal can be used to terminate the Bhyve process directly:

```
doas kill -s KILL $(pgrep -f "bhyve: samplevm")
```

After a Bhyve process has been terminated, its return value (exit code) can be queried by the shell variable `$?`:

```
echo $?
```

Return values greater than 1 indicate that the virtual machine could not be shut down properly. If the virtual machine is to be restarted or the configuration of a VM instance has been changed between `bhyve` calls, the VM instance must first be removed via `bhyvectl`:

```
doas bhyvectl --vm=samplevm --destroy
```

Virtual machine instances are listed as device files at `/dev/vmm` and can be further managed using `bhyvectl`. Via `ls -al /dev/vmm` it can also be determined which virtual machines are currently created on the TwinCAT/BSD host:

```
ls /dev/vmm
samplevm
```

10.3 Use shell scripts

Virtual machines can be easily started and managed using shell scripts. Scripted VM applications allow configurations to be saved persistently and reused after a restart. In combination with further instructions and shell scripts, any VM applications can be set up under TwinCAT/BSD.

The following sample script shows the basic structure of a scripted VM application:

```
# root permissions are required to run VMs
if test "$(id -u)" -ne 0; then
printf "%s must be run as root\n" "${0##*/}"
exit 1
fi

# Default values for VM configuration
vm_name="samplevm"

# Ensure that kernel modul vmm.ko is loaded
kldload -n vmm.ko

while true; do
# destroy former VM instance to ensure we start
# with a clean VM configuration
if test -e "/dev/vmm/${vm_name}"; then
bhyectl --vm="${vm_name}" --destroy
fi

# start a simple UEFI based VM instance
_bhyve_rc=0
bhyve \
-A -H -P \
-c sockets=1,cores=1,threads=1 \
-m 1G \
-s 0:0,hostbridge \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd \
-l com1,stdio \
-s 31:0,lpc \
"${vm_name}"
_bhyve_rc=$?

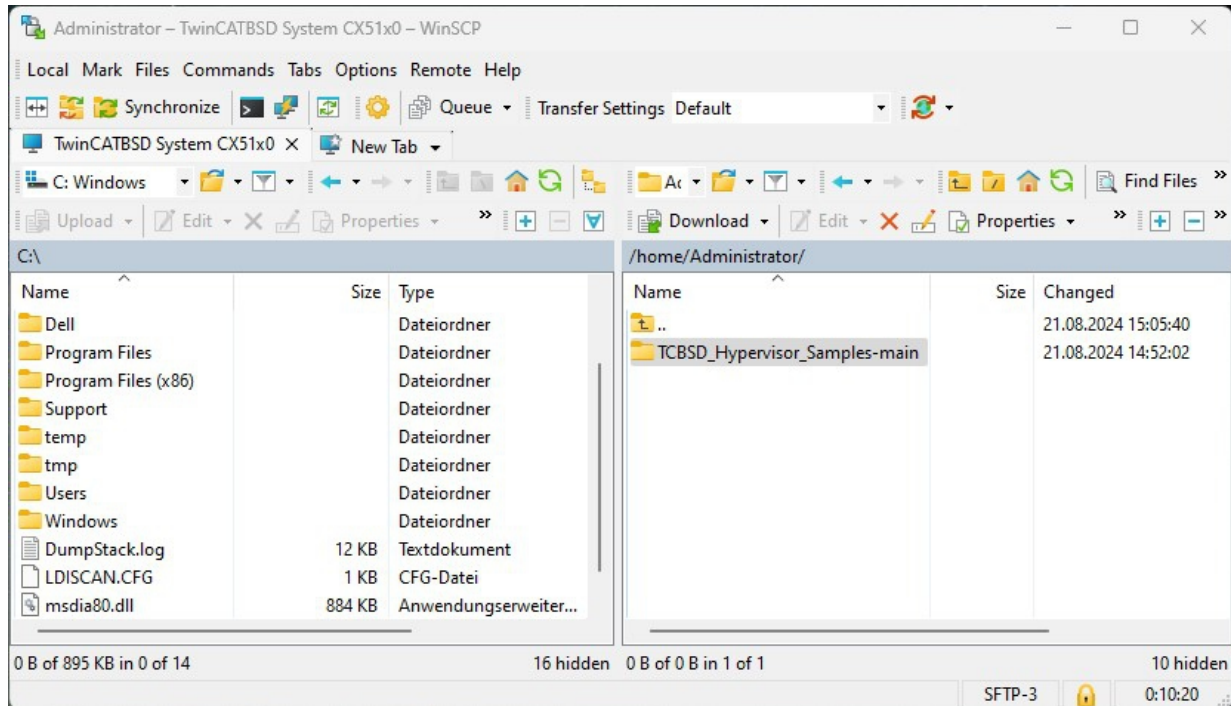
# according to bhyve man pages the return codes indicates
# how the VM was terminated:
# 0: rebooted
# 1: powered off
# ...
# 4: exited due to an error
if test "${_bhyve_rc}" -ne 0; then
printf "bhyve exited with return code: %s\n" "${_bhyve_rc}"
break
fi
printf "Restarting %s\n" "${vm_name}"
done
```

The sample script can be saved to a text file on the TwinCAT/BSD host and executed. Alternatively, the sample script can be downloaded from GitHub repository at https://github.com/Beckhoff/TCBSD_Hypervisor_Samples and copied to the TwinCAT/BSD host.

Proceed as follows:

1. Download the sample script at https://github.com/Beckhoff/TCBSD_Hypervisor_Samples.

- Copy the entire folder on the TwinCAT/BSD host to the directory `/home/Administrator` by using the WinSCP program, for example.



- Alternatively, you can use the following command to load and unpack the sample script directly under TwinCAT/BSD:

```
fetch -o /home/Administrator/main.zip \
https://github.com/Beckhoff/TCBSD_Hypervisor_Samples/archive/refs/heads/main.zip \
&& unzip -d /home/Administrator main.zip
```

- Navigate to the new directory with `cd /home/Administrator/TCBSD_Hypervisor_Samples-main/basic_vm_script`.
- Enter the command `doas make` to install the sample script `samplevm`. In addition to the installation, the file permissions are set, making the sample script executable. Without the `doas make` command, file permissions must be set manually to run the sample script.

- Finally, enter the command `doas samplevm` to run the sample script.

⇒ The virtual machine boots into the UEFI shell, which is output to the command line.

```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70 (BHYVE, 0x00010000)
map: No mapping found.
Press ESC in 1 seconds to skip startup.nsh or any other key to continue.
Shell>
```

You can return to the command line by shutting down the virtual machine with the `reset -s` command. To start virtual machines as a system service under TwinCAT/BSD, shell scripts can be used in combination with the `rc` framework and thus virtual machines can be managed as a system service or started automatically at system startup (see: [Autostart shell scripts](#) [► 78]).

10.4 Autostart shell scripts

To manage a virtual machine as a system service or to start it automatically at system startup, the call of `bhyve` or a shell script can be included in the `rc` framework.

The GitHub repository: https://github.com/Beckhoff/TCBSD_Hypervisor_Samples/tree/main/vm_autostart includes sample files that demonstrate how to incorporate a simple VM configuration into the `rc` framework using shell scripts. The sample in the directory `vm_autostart` contains the appropriate files for this.

If the sample script has been loaded onto the TwinCAT/BSD host as described in chapter [Use shell scripts \[► 76\]](#), you can change to the directory with `cd /home/Administrator/TCBSD_Hypervisor_Samples-main/vm_autostart`.

```
vm_autostart
├── Makefile
├── rc.d
│   └── samplevm
└── samplevm
```

The sample script `samplevm` shown in the chapter [Use shell scripts \[► 76\]](#) has been extended by `start`, `stop` and `status` parameters to be able to start and stop a VM configuration with VNC access via the command line.

The `vm_autostart/rc.d/samplevm` shell script is used to integrate the `vm_autostart/samplevm` shell script into the `rc` framework.

Controlling the VM system service with the sample script:

1. Navigate to the directory with `cd /home/Administrator/TCBSD_Hypervisor_Samples-main/vm_autostart`
 2. Enter the command `doas make` to install both files from the `vm_autostart` directory on the TwinCAT/BSD host.
 3. Then enter `doas service samplevm enable` to enable the VM instance as a system service for autostart via `service(8)`.
 4. After including the shell script as a system service, the virtual machine can be started with the `doas service samplevm start` command.
 5. The command `doas service samplevm status` shows whether the virtual machine has been restarted and with which process ID it is running.
- ⇒ In this sample, the virtual machine can be accessed via a VNC client via TCP port “5900”. From now on, the virtual machine is also restarted after a reboot of the TwinCAT/BSD host and is available for use.
1. The virtual machine can be stopped again with the command `doas service samplevm stop`.
 2. The autostart of the VM is disabled again with the command `doas service samplevm disable`.

The sample script is a first starting point and shows how virtual machines can be started, managed and automated under TwinCAT/BSD. The sample script `vm_autostart/samplevm` can be customized and extended as needed to achieve a desired VM configuration. The chapter [Advanced VM configuration \[► 78\]](#) explains other parameters that can be used to extend the configuration of a virtual machine.

For detailed information about creating `rc.d` scripts, see the FreeBSD Handbook chapter [Practical rc.d scripting in BSD](#).

10.5 Advanced VM configuration

10.5.1 ZFS data sets as storage location for virtual machines

Using ZFS datasets offer the possibility to use functions and properties of ZFS like quotas, compression, block sizes or snapshots for VM applications.

The following call creates a file system as location for the virtual machine `samplevm` and mounts the file system in the directory structure at `/vms/samplevm`:

```
doas zfs create -o mountpoint=/vms/samplevm zroot/samplevm
```

The file system can now be used to back up files for virtual drives, EFI variables, or other VM-related data.

Backup points from virtual hard disks via ZFS snapshots

ZFS snapshots can be applied to ZFS datasets to create backup points of virtual machines (see also: [ZFS volumes as data memory for virtual hard disks \[► 83\]](#)). The state of a virtual machine can thus be backed up at a specific point in time and restored if necessary.

A snapshot of the ZFS dataset `zroot/samplevm` can be created via [zfs-snapshot\(8\)](#):

```
doas zfs snapshot zroot/samplevm@latest
```

`@latest` defines the name of the snapshot.

To restore the ZFS dataset and the files stored in it to the state of the snapshot `@latest`, the following command can be used

```
doas zfs rollback zroot/samplevm@latest
```

During the build and restore process, the VM instance should be shut down.

More detailed information about the Z file system and the use of ZFS datasets and snapshots can be found in the chapter [The Z file system in the FreeBSD documentation](#)

10.5.2 UEFI-based virtual machines

UEFI-based virtual machines can be started by the parameters `-l, bootrom, <efi-rom>[, <efi-vars>]`. For `<efi-rom>`, the path to an EFI ROM file must be specified. Optionally, the path to a file can be specified at `<efi-vars>`, which in turn serves as a location for EFI virtual machine variables.

Files with EFI variables should be created per VM instance. The following command creates a copy of the file `BHYVE_BHF_UEFI_VARS.fd` to be used for the VM instance `samplevm`.

```
doas cp /usr/local/share/uefi-firmware/BHYVE_BHF_UEFI_VARS.fd /vms/samplevm/EFI_VARS.fd
```

The file `EFI_VARS.fd` is then passed as `<efi-vars>` parameter to the `bhyve` call.

Additionally, the parameter `fwcfg=qemu` should be appended. This allows the firmware to access bhyve's dynamically generated ACPI tables.

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,/vms/samplevm/EFI_VARS.fd,fwcfg=qemu \
-l com1,stdio \
-s 0:0,hostbridge \
-s 31:0,lpc \
-A -H -P \
samplevm
```

10.5.3 VNC-based interaction with virtual machines

NOTICE

Unsecured TCP port

Incoming connections on TCP port 5900 are not blocked by the firewall in this sample. Set up a secure and encrypted connection and secure TCP port 5900 via SSH as soon as the operation takes place in an unsecured network.

With Virtual Network Computing (VNC) it is possible to control a virtual machine on a TwinCAT/BSD host via a network connection. For this, `bhyve` provides an integrated VNC server to interact with VM instances.

Virtual machine graphical output and user input to the virtual machine can be transmitted via the integrated VNC server by configuring the virtual machine with a frame buffer device `fbuf`. The following options can be passed to the frame buffer device `fbuf` to configure the VNC server:

```
fbuf,[rfb=ip-and-port][,w=width][,h=height][,vga=vgaconf][,wait][,password=password]
```

The following call starts the virtual machine `samplevm` with a frame buffer device at PCI slot 2. The configuration options specify that the VNC server listens for connections on TCP port 5900 of the TwinCAT/BSD host. In addition, the image resolution of the frame buffer is set to 1024x768 pixels. Further configuration options of the `fbuf` device can be found in the [bhyve man pages](#)

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 31,lpc \
-A -H -P \
samplevm
```

Depending on the VNC client, mouse pointer positions may not be passed accurately. The `bhyve` call can then be extended to include a `xhci`, `tablet` device configuration.

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 3,xhci,tablet \
-s 31,lpc \
-A -H -P \
samplevm
```

The integrated VNC server does not support transport layer security. Incoming TCP connections on port 5900 are blocked by default by the TwinCAT/BSD packet filter `pf(8)`. Incoming connections can be allowed by configuring the packet filter (see: [Firewall](#)).

10.5.4 Virtual drives

Virtual machines can be configured with virtual drives (block storage devices). These can in turn be used as virtual hard disks (nvme, ahci-hd or virtio-blk) or as a virtual CD-ROM drive (ahci-cd).

The following call starts the virtual machine `samplevm` with an emulated NVMe drive and a virtual AHCI CD-ROM drive:

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 2G \
-l bootrom,/vms/samplevm/BHYVE_BHF_UEFI.fd,/vms/samplevm/EFI_VARS.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 3,xhci,tablet \
-s 10,nvme,/vms/samplevm/disk0.img \
-s 15,ahci-cd,/vms/samplevm/os-installer.iso,ro \
-s 31,lpc \
-H -P -A \
samplevm
```

The resulting VM configuration looks as follows:

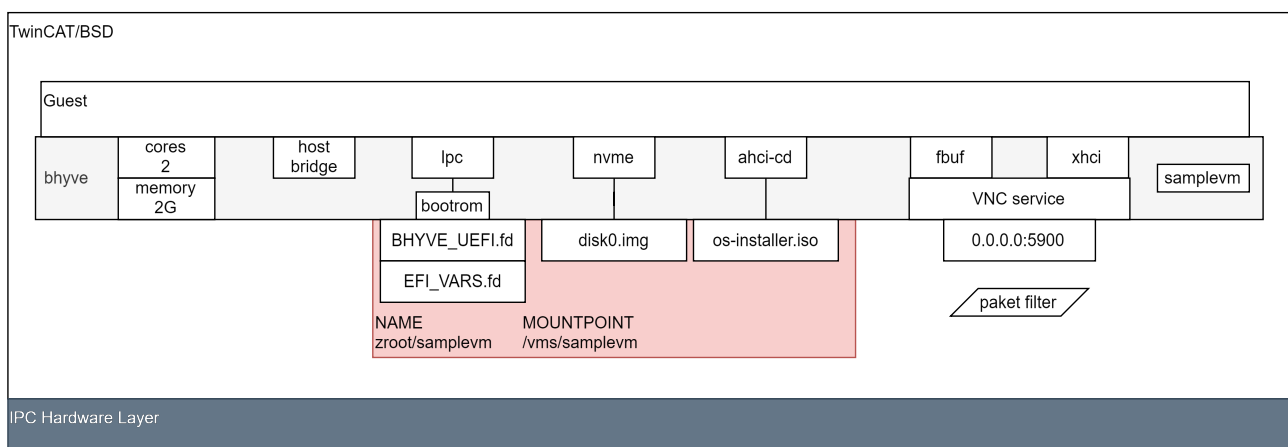


Fig. 21: VM instance with virtual drives.

Both drives use regular files as disk image or ISO-CDROM on the TwinCAT/BSD host as data memory. Both files must exist before `bhyve` is called. Alternatively, block devices such as ZFS volumes can also be transferred as data memory (see: [ZFS volumes as data memory for virtual hard disks](#) [► 83]).

To make the configured drives known to guest operating systems via ACPI, the parameter `-A` must also be passed.

In the sample above, the disk image file `/vms/samplevm/disk0.img` is used as data memory for the virtual hard disk (see: [Disk image files as data memory for virtual hard disks](#) [► 81]).

The file `/vms/samplevm/os-installer.iso` is only accessed for reading. If the memory image of `os-installer.iso` corresponds to a bootable ISO image, the installation of an operating system can be started within the virtual machine, for example (see: [Installing Debian Linux as a guest operating system](#) [► 93]).

In this sample, both files are stored in the directory `/vms/samplevm`, which belongs to the previously created ZFS dataset `zroot/samplevm`. The ZFS dataset therefore serves as a storage location for the persistent data of the virtual machine. This means that ZFS snapshots can be used to back up and restore data.

10.5.4.1 Disk image files as data memory for virtual hard disks

Disk image files are regular files in which the contents of virtual hard disks can be stored. An empty disk image file of maximum 20 GB can be created with the help of `truncate(1)` as follows:

```
truncate -s 20G /vms/samplevm/disk0.img
```

The created `disk0.img` file can then be passed to the `bhyve` call as a backend for a virtual hard disk:

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd \
-s 0,hostbridge \
-s 10,nvme,/vms/samplevm/disk0.img \
-s 31,lpc \
-H -P -A \
samplevm
```

10.5.4.2 Use of installation media (ISO images)

Installation programs for operating systems are often made available for download as ISO images via websites. Under TwinCAT/BSD `fetch(8)` can be used to download an ISO image from a website.

The following call loads the Debian ISO image `debian-11.5.0-amd64-netinst.iso` from the website [cdimage.debian.org](https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/debian-11.5.0-amd64-netinst.iso) and saves it locally in the file `os-installer.iso`.

```
fetch -o os-installer.iso https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/debian-11.5.0-amd64-netinst.iso
```

The downloaded ISO file can then in turn be used as media in a virtual CD-ROM drive (`ahci-cd`) of a virtual machine.

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 3,xhci,tablet \
-s 15,ahci-cd,/usr/home/Administrator/os-installer.iso,ro \
-s 31,lpc \
-A -H -P \
samplevm
```

10.5.4.3 ZFS volumes as data memory for virtual hard disks

ZFS volumes are a type of ZFS data sets and are listed as block devices at `/dev/zvol/zroot`. A ZFS volume can be used as data memory for virtual drives in order to take advantage of ZFS data sets such as snapshots, clones or compression.

The following command creates the ZFS volume `zroot/vms/samplevm/disk0` in the ZFS pool `zroot` with 20 GB

```
doas zfs create -V 20G zroot/vms/samplevm/disk0
```

The following call starts the `samplevm` virtual machine with an emulated NVME hard disk that uses the ZFS volume `zroot/vms/samplevm/disk0` as data storage, which is available under the directory `/dev/zvol/zroot/vms/samplevm/disk0`.

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 3,xhci,tablet \
-s 10,nvme,/dev/zvol/zroot/vms/samplevm/disk0 \
-s 31,lpc \
-A -H -P \
samplevm
```

10.5.5 Virtual machine network configuration

Virtual machines can be configured with virtual network controllers to connect the virtual machine to a network. Virtual machines use `tap(4)` or `vmnet(4)` network interfaces of the TwinCAT/BSD host, which in turn are managed under TwinCAT/BSD with the help of `ifconfig(8)`.

The following command creates a new `vmnet(4)` instance:

```
doas ifconfig vmnet create
vmnet0
```

Similarly, `tap(4)` instances can be created

```
doas ifconfig tap create
tap0
```

`tap(4)` and `vmnet(4)` network interfaces can be created via `cloned_interfaces` at system startup. To do this, the instances of the `cloned_interfaces` listing can be added to the rc configuration:

```
doas sysrc cloned_interfaces+="vmnet0 tap0"
```

The created `vmnet(4)` or `tap(4)` instances (in this case `vmnet0` and `tap0`) can then be used as an Ethernet endpoint for a virtual machine to exchange Ethernet packets between the TwinCAT/BSD host and the virtual machine environment.

To do this, the `bhyve` call is started for one or more emulated `virtio-net` devices that use the previously created network interfaces as endpoints. A MAC address is generated for each network interface of the virtual machine. Optionally, each network interface can also be given a defined MAC address with `,mac=xx:xx:xx:xx:xx:xx`.

The following command starts a virtual machine with two virtual network controllers that use the above-mentioned instances `vmnet0` and `tap0` at PCI slot `-s 20` and `-s 21` as endpoints on the host side and receive defined MAC addresses:

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 3,xhci,tablet \
-s 20,virtio-net,vmnet0,mac=58:9c:fc:02:34:25 \
-s 21,virtio-net,tap0,mac=58:9c:fc:03:5e:ec \
```

```
-s 31,lpc \
-A -H -P \
samplevm
```

Thus, virtual machines are always connected to external networks via `tap(4)` or `vmnet(4)` devices.

The configuration with virtual network controllers is as follows:

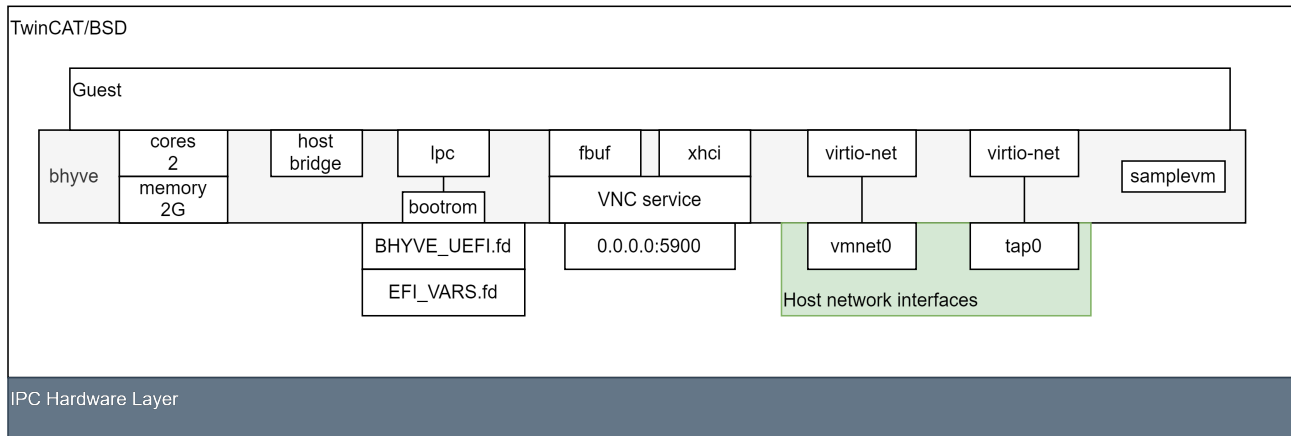


Fig. 22: Configuration of a VM instance with virtual network controllers.

Depending on the application, a virtual machine can also be configured with additional network controllers in order to be integrated into different networks. The connection of a virtual machine to a network is then determined by the configuration of the respective `tap(4)` or `vmnet(4)` devices on the TwinCAT/BSD host. This results in different possibilities to realize the communication of virtual machines into a network:

1. [Host-Only network \[► 85\]](#)
2. [NAT network \[► 87\]](#)
3. [Bridge network \[► 86\]](#)
4. [Ethernet device passthrough \[► 90\]](#)

10.5.5.1 Host-Only network

In a Host-Only network configuration, network packets are only exchanged between the virtual machine and the TwinCAT/BSD host. A `vmnet(4)` interface is created on the TwinCAT/BSD host for this purpose. The `vmnet(4)` instance serves as a backend for an emulated virtio-net or e1000 Ethernet controller of the virtual machine. Within the guest system, the Ethernet controller is recognized as a network interface. The IP addresses of the host and guest network interfaces are then configured to create a Host-Only network.

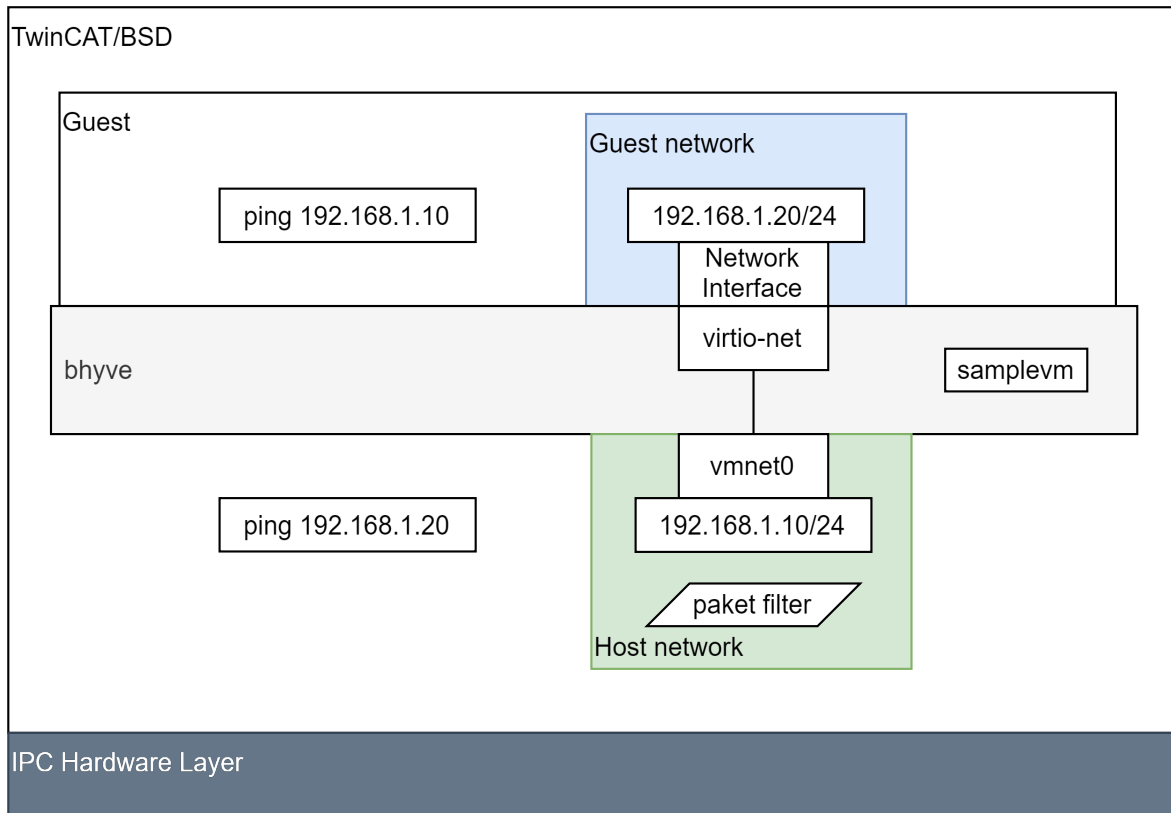


Fig. 23: VM instance with a Host-Only network configuration.

To configure a Host-Only network, a `vmnet(4)` interface is created on the TwinCAT/BSD host with `ifconfig` and this network interface is assigned a private IP address.

```
doas ifconfig vmnet create inet 192.168.1.10 netmask 255.255.255.0
```

So that the `vmnet` instance is already created at system startup, it can be stored in the `rc` configuration:

```
doas sysrc cloned_interfaces+="vmnet0"
doas sysrc ifconfig_vmnet0="inet 192.168.1.10 netmask 255.255.255.0"
```

The `vmnet(4)` device is then transferred to the virtual machine as a backend for a virtual network controller of type `virtio-net` (see figure above).

Within the guest system, the virtual network interface must be configured to be on the same IP network as the `vmnet0` interface of the TwinCAT/BSD host (sample above: 192.168.1.0/24). Afterwards the Host-Only communication can be checked with ping requests between TwinCAT/BSD host and guest system.

The following call starts the virtual machine `samplevm` with a `virtio-net` based network controller at PCI address 20. The previously configured `vmnet0` instance is passed as the backend.

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 20,virtio-net,vmnet0 \
-s 31,lpc \
-A -H -P \
samplevm
```

10.5.5.2 Bridge network

NOTICE

Connection failure and limited availability in the network

Changes to the filter rules affect the availability of the TwinCAT/BSD host, the virtual machine as well as their services in the network.

In a Bridge network, a physical network interface of the TwinCAT/BSD host (e.g. `igb0`) is connected to a `tap(4)` device via a `bridge(4)` device. The `tap(4)` device in turn serves as a backend for a network interface of the virtual machine (see: [Virtual machine network configuration \[► 83\]](#)).

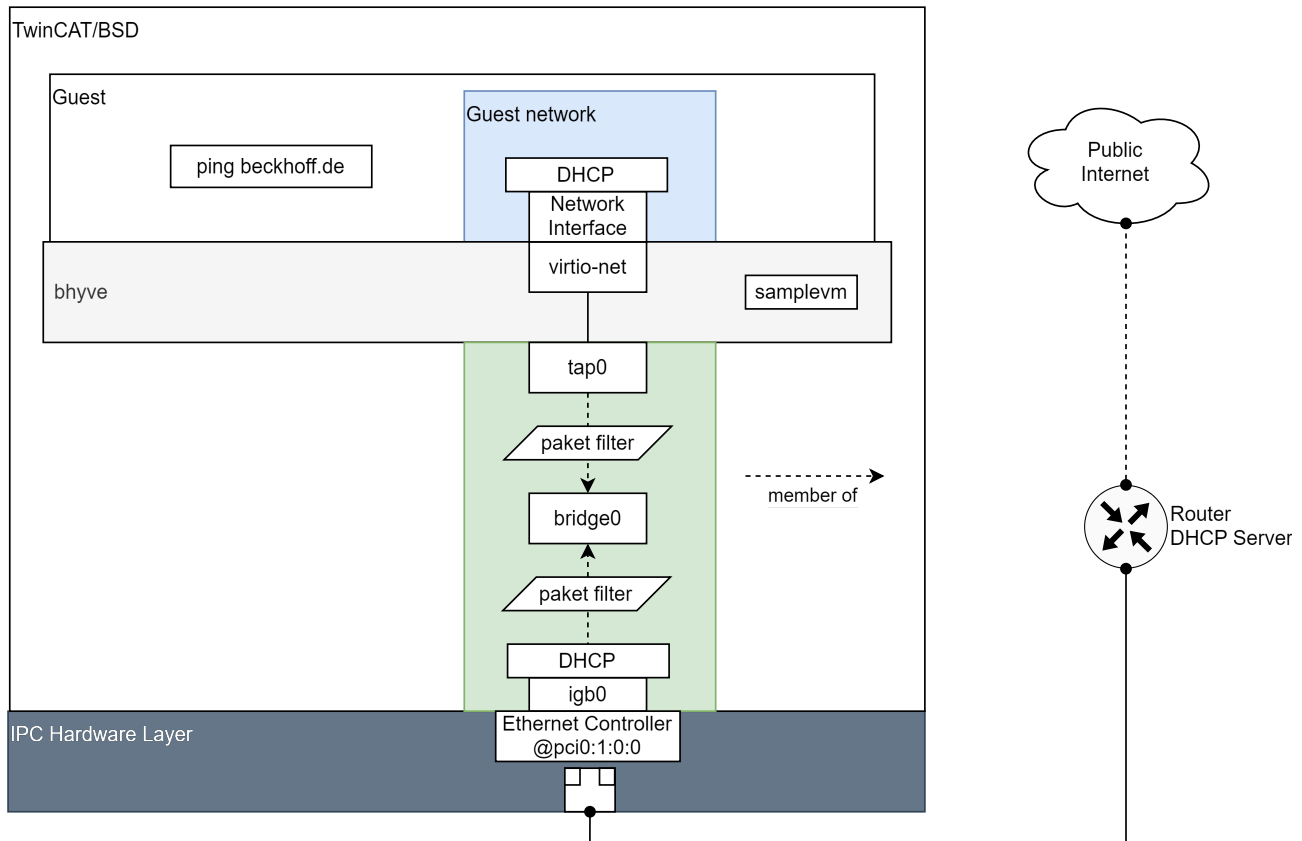


Fig. 24: VM instance with a Bridge network configuration.

Network communication of the virtual machine is thus bridged to the physical network interface of the TwinCAT/BSD host via the respective `tap(4)` device by a `bridge(4)` device on Ethernet level.

Configuration of the network components

A `bridge(4)` device is created on the TwinCAT/BSD host using `ifconfig(8)`:

```
doas ifconfig bridge create
```

The output `bridge0` appears.

Likewise, a `tap(4)` device is created as the backend for the virtual network interface:

```
doas ifconfig tap create
```

The output `tap0` appears.

To forward network packets between a physical network interface of the TwinCAT/BSD host and a `tap(4)` device via the `bridge0`, the corresponding devices must become members of the `bridge0`.

The following call makes the physical network interface `igb0` of the TwinCAT/BSD host and the `tap0` device members of the `bridge0` instance:

```
doas ifconfig bridge0 addm igb0 addm tap0 up
```

Depending on the industrial PC or Ethernet interface used, the naming of the network interface under TwinCAT/BSD may vary and be displayed as `em0`, `em1` or `igb1`, for example.

So that the `bridge0` configuration is already created at system startup, it can be stored in the `rc` configuration:

```
doas sysrc cloned_interfaces+="bridge0 tap0"
doas sysrc ifconfig_bridge0="addm igb0 addm tap0 up"
```

Via `ifconfig bridge0` the members of the `bridge0` can be checked:

```
ifconfig bridge0
bridge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
ether 58:9c:fc:10:ff:e1
id 00:00:00:00:00:00 priority 32768 hellotime 2 fwddelay 15
maxage 20 holdcnt 6 proto rstp maxaddr 2000 timeout 1200
root id 00:00:00:00:00:00 priority 32768 ifcost 0 port 0
member: tap0 flags=143<LEARNING,DISCOVER,AUTOEDGE,AUTOPTP>
        ifmaxaddr 0 port 5 priority 128 path cost 2000000
member: igb0 flags=143<LEARNING,DISCOVER,AUTOEDGE,AUTOPTP>
        ifmaxaddr 0 port 1 priority 128 path cost 2000000
groups: bridge
nd6 options=9<PERFORMNUD,IFDISABLED>
```

The corresponding `bhyve(8)` command uses only the `tap0` instance to connect the virtual machine to the Bridge network:

```
doas bhyve \
-c sockets=1,cores=1,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0.0:5900,w=1024,h=768 \
-s 3,xhci,tablet \
-s 20,virtio-net,tap0 \
-s 31,lpc \
-A -H -P \
samplevm
```

Filter rules in the Bridge network

By default, the packet filter `pf(8)` under TwinCAT/BSD blocks the exchange of network packets at a `bridge(4)` device.

The filtering behavior on `bridge(4)` devices can be disabled via `sysctl(8)`. By setting the variables `net.link.bridge.pfil_member` and `net.link.bridge.pfil_bridge` to 0:

```
doas sysctl net.link.bridge.pfil_member=0
doas sysctl net.link.bridge.pfil_bridge=0
```

To set the settings persistently, the following lines must be added to the `/etc/sysctl.conf` file:

```
net.link.bridge.pfil_member=0
net.link.bridge.pfil_bridge=0
```

For more information, see the man pages for `bridge(4)`, `sysctl(8)`, and `sysctl.conf(5)`.

Alternatively, filter rules can be defined for the `bridge(4)` and its members `pf(8)` to control packet exchange in the bridge network (see: [Firewall](#) [► 32]).

10.5.5.3 NAT network

A NAT network can be used to send requests from a private VM network (for example, a Host-Only network) to an external network.

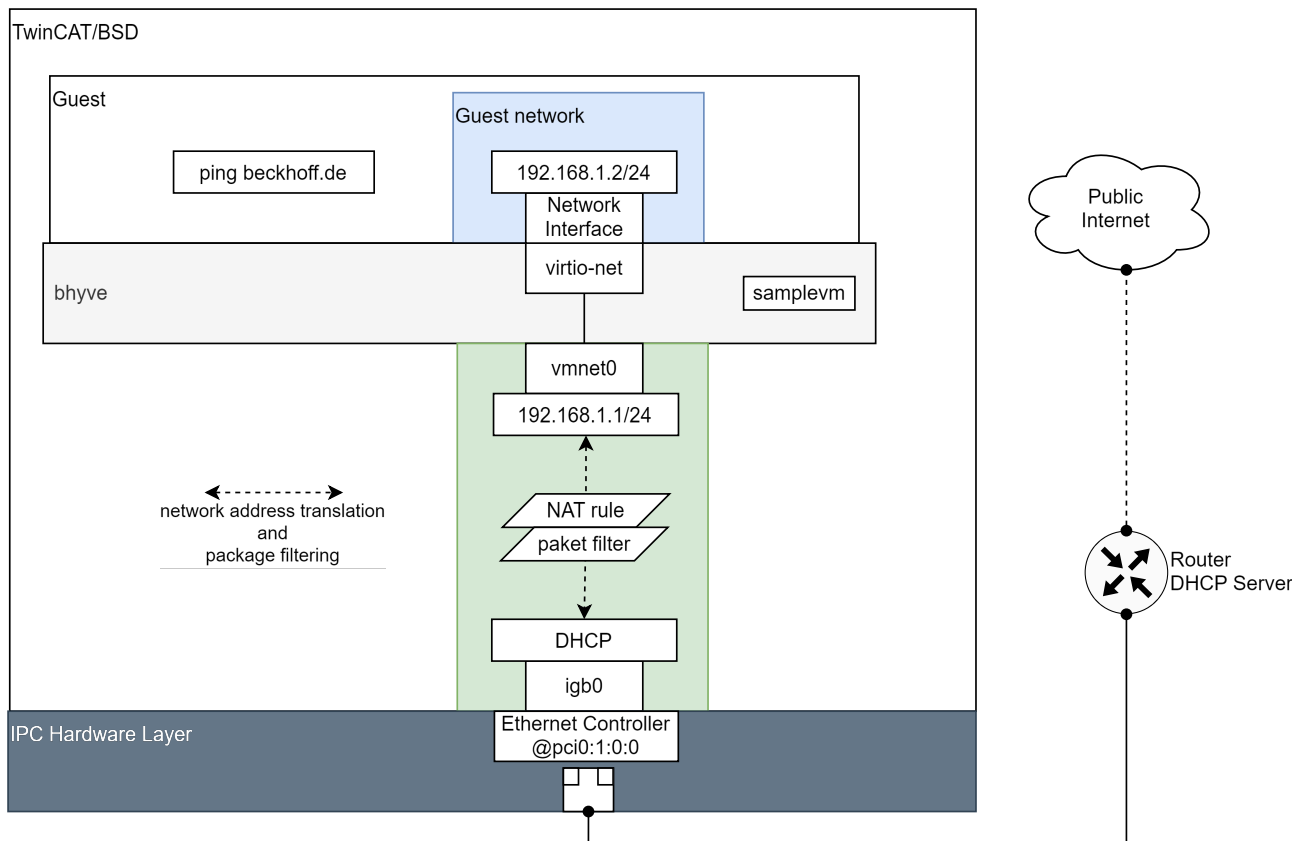


Fig. 25: VM instance with a NAT network configuration.

Under TwinCAT/BSD the forwarding of IP packets between network interfaces must be activated for this:

```
doas sysctl net.inet.ip.forwarding=1
```

To save this setting persistently `net.inet.ip.forwarding=1` can be added to the file `/etc/sysctl.conf`. In addition, the translation of private network addresses to an external network requires appropriate network address translation (NAT) rules in `pf(8)`.

The following sample uses the `vmnet0` configuration from chapter [Host-Only network](#) [► 85] for the private network between virtual machine and TwinCAT/BSD host.

```
ifconfig vmnet0
vmnet0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=80000<LINKSTATE>
ether 58:9c:fc:10:56:5b
inet 192.168.1.1 netmask 0xfffff00 broadcast 192.168.1.255
groups: vmnet
media: Ethernet autoselect
status: no carrier
nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
```

The IPC is connected to an external network via the physical network interface `igb0`:

```
ifconfig igb0
igb0: flags=8863<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=4a004a9<RXCSUM,VLAN_MTU,JUMBO_MTU,VLAN_HWCSUM,LRO,RXCSUM_IPV6,NOMAP>
ether 00:01:05:62:3b:b0
inet 172.17.98.154 netmask 0xfffff00 broadcast 172.17.98.255
media: Ethernet autoselect (1000baseT <full-duplex>)
status: active
nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
```

For the translation of private addresses to the external network, the following `pf` rule can first be saved in a text file (`samplevm.nat.conf`) and then loaded via `pfctl(8)`:

```
nat on igb0 from vmnet0:network to any -> (igb0)
doas pfctl -a "bhf-nat/samplevm-nat" -f samplevm.nat.conf
```

Additionally, incoming network traffic should be allowed into the private network:

```
pass from vmnet0:network to any keep state
```

The rule set can in turn be saved in a text file and loaded via `pfctl(8)`:

```
doas pfctl -a "bhf/bhyve/samplevm " -f samplevm.filters.conf
```

Once both rule sets have been loaded, the virtual machine can be started with `vmnet0` as the backend for the virtio-net-based network controller:

```
doas bhyve \  
-c sockets=1,cores=1,threads=1 \  
-m 2G \  
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \  
-s 0,hostbridge \  
-s 20,virtio-net,vmnet0 \  
-s 31,lpc \  
-A -H -P \  
samplevm
```

Within the guest operating system, communication into the external network can be checked with ping requests:

```
ping beckhoff.com  
  
PING beckhoff.com (18.195.44.45) from 192.168.1.2 : 56(84) bytes of data.  
64 bytes from ec2-18-195-44-45.eu-central-1.compute.amazonaws.com (18.195.44.45): icmp_seq=1 ttl=245  
time=7.44 ms  
64 bytes from ec2-18-195-44-45.eu-central-1.compute.amazonaws.com (18.195.44.45): icmp_seq=2 ttl=245  
time=7.27 ms  
64 bytes from ec2-18-195-44-45.eu-central-1.compute.amazonaws.com (18.195.44.45): icmp_seq=3 ttl=245  
time=7.36 ms  
^C
```

It should be noted that the virtual network interface in the guest operating system is assigned a network address in the range of the `vmnet0` network (192.168.1.0/24 see above). In addition, the `vmnet0` address (192.168.1.1) must be entered as the default gateway and name server addresses must be stored in order to resolve public domain names such as `beckhoff.com`.

10.5.5.4 Ethernet device passthrough

Industrial PCs with IOMMU support allow physical Ethernet devices to be explicitly assigned to a virtual machine. The virtual machine can thus be connected to a network directly via the physical Ethernet device, without network packets being switched via the TwinCAT/BSD host. The general procedure for assigning PCI devices is described in the chapter [PCI device passthrough](#) [► 90].

The following figure shows how the Ethernet device at PCI address 3:0:0 is passed through to the samplevm to connect the virtual machine to an external network.

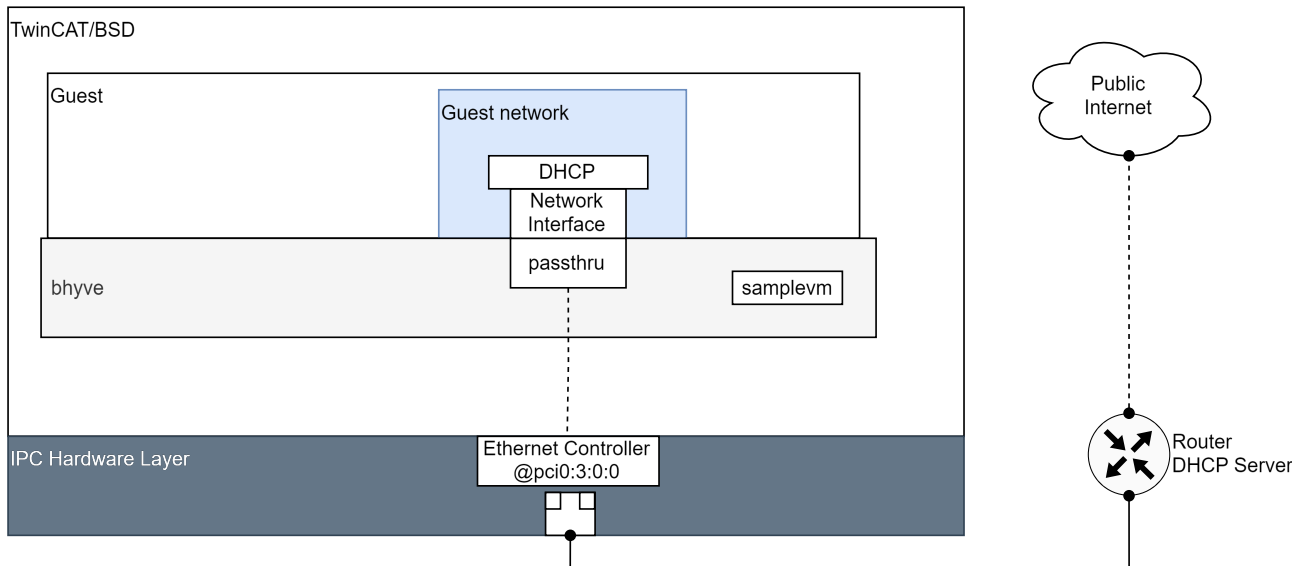


Fig. 26: VM instance with an Ethernet device passthrough configuration.

The following bhyve call assigns the Ethernet device at the physical PCI address **3:0:0** to the virtual PCI address **20** of the virtual machine:

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 2G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 20,passthru,3/0/0 \
-s 31,lpc \
-A -H -P -S\
samplevm
```

10.5.6 PCI device passthrough

Industrial PCs with IOMMU virtualization functions allow physical PCI devices to be explicitly assigned to a virtual machine (see table: [Device support for TwinCAT/BSD Hypervisor, device and GPU passthrough.](#) [► 73]). PCI devices such as the GPU, network interfaces or USB controllers can be explicitly assigned to a virtual machine as `passthru` devices.

To assign a PCI device to a virtual machine, its PCI address is needed first. The command `pciconf -l` lists all PCI devices and their addresses.

```
$ pciconf -l
...
vgapci0@pci0:0:2:0: class=0x030000 rev=0x00 hdr=0x00 vendor=0x8086 device=0x3e92 subvendor=0x8086
subdevice=0x2212
xhci0@pci0:0:20:0: class=0x0c0330 rev=0x10 hdr=0x00 vendor=0x8086 device=0xa36d subvendor=0x8086
subdevice=0x7270
igb0@pci0:1:0:0: class=0x020000 rev=0x03 hdr=0x00 vendor=0x8086 device=0x1533 subvendor=0x8086
subdevice=0x1533
...
```

In the following sample, the three listed devices are to be assigned to a virtual machine.

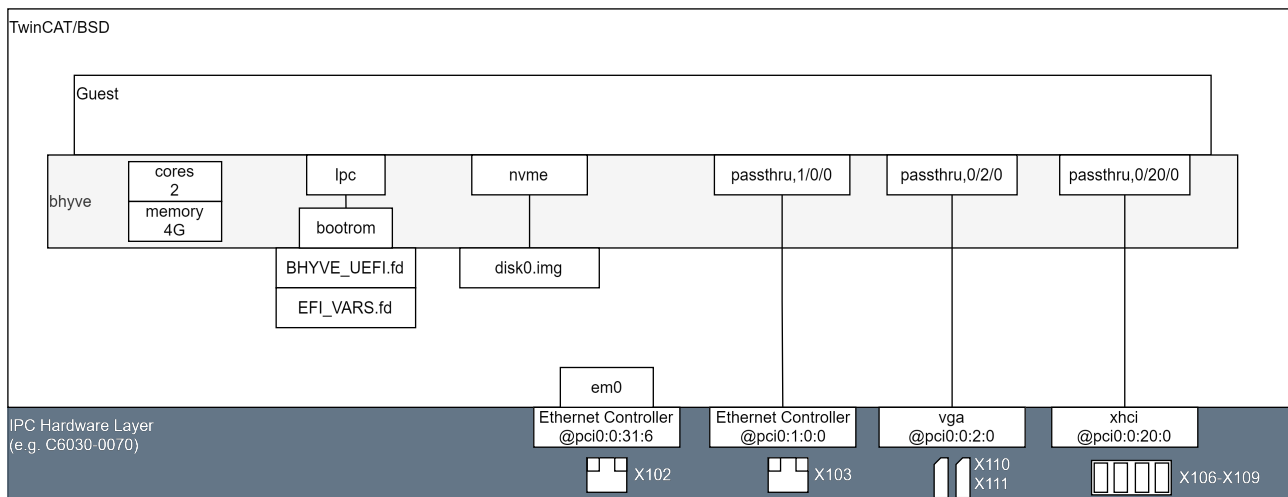


Fig. 27: Configuration of a VM instance with PCI device passthrough.

Device	Description	Address
vgapci0@pci0:0:2:0	GPU	pci0:0:2:0
igb2@pci1:0:0:0	Ethernet Controller	pci1:0:0:0
xhci1@pci0:20:0:0	USB Controller	pci0:20:0:0

To isolate the devices from the TwinCAT/BSD host, devices with `devctl` are assigned the `ppt` (PCI PassThrough) driver.

```
doas devctl set driver -f pci0:0:2:0 ppt
doas devctl set driver -f pci0:0:20:0 ppt
doas devctl set driver -f pci0:1:0:0 ppt
```

To set the drivers already at system boot the PCI addresses can be added as `pptdevs` to the `/boot/loader.conf` file:

```
pptdevs="1/0/0 0/2/0 0/20/0"
```

A new output of `pciconf -l` now shows that the `ppt` drivers have been assigned to the devices:

```
$ pciconf -l
...
ppt0@pci0:0:2:0: class=0x030000 rev=0x00 hdr=0x00 vendor=0x8086 device=0x3e92 subvendor=0x8086
subdevice=0x2212
ppt1@pci0:0:20:0: class=0x0c0330 rev=0x10 hdr=0x00 vendor=0x8086 device=0xa36d subvendor=0x8086
subdevice=0x7270
ppt2@pci0:1:0:0: class=0x020000 rev=0x03 hdr=0x00 vendor=0x8086 device=0x1533 subvendor=0x8086
subdevice=0x1533
...
```

The PCI devices can now be passed to `bhyve` with the parameter `-s [slot],passthru,[slot/bus/function]`. The values for `[slot/bus/function]` refer to the PCI addresses of the `pciconf -l` output. Since passthrough devices use fixed memory addresses, `bhyve` must also be passed the flag `-s` as a parameter to disable memory swapping for the process.

The `bhyve` call with assigned on-board GPU, Ethernet and USB controller results for the sample as follows:

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 4G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0:0,hostbridge \
-s 2:0,passthru,0/2/0 \
-s 3:0,passthru,0/20/0 \
-s 10:0,nvme,/vms/samplevm/disk0.img \
-s 20:0,passthru,1/0/0 \
-s 31:0,lpc \
-A -H -P -S \
samplevm
```

Extended parameters for the pass-through of the integrated GPU

The pass-through of the integrated GPU on supported IPCs requires extended bhyve parameters so that the EFI firmware `BHYVE_BHF_UEFI.fd` can correctly initialize the integrated graphics card during the boot process of the virtual machine and make it available to the guest operating system.

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 4G \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd,fwcfg=qemu \
-s 0:0,hostbridge \
-s 2:0,passthru,0/2/0,rom=/vms/samplevm/gop.rom \
-s 3:0,passthru,0/20/0 \
-s 10:0,nvme,/vms/samplevm/disk0.img \
-s 20:0,passthru,1/0/0 \
-s 31:0,lpc \
-o pci.0.31.0.pcireg.vendor=host \
-o pci.0.31.0.pcireg.device=host \
-o pci.0.31.0.pcireg.revid=host \
-o pci.0.31.0.pcireg.subvendor=host \
-o pci.0.31.0.pcireg.subdevice=host \
-A -H -P -S \
samplevm
```

The `passthru` parameter for the integrated graphics card (`-s 2:0,passthru,0/2/0`) is extended by `,rom=/vms/samplevm/gop.rom`.

The file `gop.rom` is required for the initialization of the graphics card by the EFI and can be created with the following command:

```
doas gop-dump /vms/samplevm/gop.rom
```

In addition, as shown above, the PCI registers of the emulated LPC bridge must be set to the value `host` when calling `bhyve`:

```
-o pci.0.31.0.pcireg.vendor=host \
-o pci.0.31.0.pcireg.device=host \
-o pci.0.31.0.pcireg.revid=host \
-o pci.0.31.0.pcireg.subvendor=host \
-o pci.0.31.0.pcireg.subdevice=host \
```

10.5.7 Pass-through of input devices

Input via mouse and keyboard can be transferred to the virtual machine via a `virtio-input` device. In this case, the entire USB controller does not have to be passed through via PCI Device Passthrough so that other USB interfaces are still available to the TwinCAT/BSD host.

To forward input to the virtual machine via `virtio-input`, the input events of the connected device `/dev/input/eventN` must be transferred to the `virtio-input` device.

The following sample call transfers the integrated GPU using PCI-Device Passthrough and the input events `/dev/input/event4` and `/dev/input/event3` via `virtio-net`.

```
doas bhyve \
-c sockets=1,cores=2,threads=1 \
-m 4G \
-l bootrom,/vms/samplevm/UEFI.fd,/vms/samplevm/EFI_VARS.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,passthru,0/2/0,rom=/vms/samplevm/gop.rom \
-s 3,virtio-input,/dev/input/event4 \
-s 4,virtio-input,/dev/input/event3 \
-s 10,nvme,/vms/samplevm/disk0.img \
-s 31,lpc \
-o pci.0.31.0.pcireg.vendor=host \
-o pci.0.31.0.pcireg.device=host \
-o pci.0.31.0.pcireg.revid=host \
-o pci.0.31.0.pcireg.subvendor=host \
-o pci.0.31.0.pcireg.subdevice=host \
-A -H -P samplevm
```


Identification of input events of connected USB devices

The `sysctl kern.evdev.input` command can be used to list connected input devices and their IDs in order to determine the assignment between the input device and `/dev/input/event<ID>`.

The following sample shows a USB mouse (`ums0`) with Input ID 4 and a USB keyboard (`ukbd0`) with Input ID 3.

```
kern.evdev.input.4.uniq: 2057366C5943
kern.evdev.input.4.phys: ukbd1
kern.evdev.input.4.id: { bustype = 0x0003, vendor = 0x046d, product = 0xc093, version = 0x0000 }
kern.evdev.input.4.name: Logitech Advanced Corded Mouse M500s, class 0/0, rev 2.00/53.00, addr 2
kern.evdev.input.3.uniq:
kern.evdev.input.3.phys: ukbd0
kern.evdev.input.3.id: { bustype = 0x0003, vendor = 0x046d, product = 0xc328, version = 0x0000 }
kern.evdev.input.3.name: Logitech USB Keyboard, class 0/0, rev 1.10/86.00, addr 1
kern.evdev.input.2.uniq:
kern.evdev.input.2.phys: acpi_button0
kern.evdev.input.2.id: { bustype = 0x0019, vendor = 0x0000, product = 0x0000, version = 0x0001 }
kern.evdev.input.2.name: Sleep Button
kern.evdev.input.1.uniq:
kern.evdev.input.1.phys: sysmouse
kern.evdev.input.1.id: { bustype = 0x0006, vendor = 0x0000, product = 0x0000, version = 0x0000 }
kern.evdev.input.1.name: System mouse
kern.evdev.input.0.uniq:
kern.evdev.input.0.phys: kbdmux0
kern.evdev.input.0.id: { bustype = 0x0006, vendor = 0x0000, product = 0x0000, version = 0x0000 }
kern.evdev.input.0.name: System keyboard multiplexer
```

Accordingly, as in the sample above, the input events from the mouse and keyboard can be transferred to the virtual machine.

10.6 Installing Debian Linux as a guest operating system

In the following sample Debian is to be installed in a virtual machine under TwinCAT/BSD. The shell scripts from the GitHub repository https://github.com/Beckhoff/TCBSD_Hypervisor_Samples/tree/main/vm_autostart can be used as a template.

```
TCBSD_Hypervisor_Samples/vm_autostart
├── Makefile
├── rc.d
│   └── samplevm
└── samplevm
```

Requirements for the VM setup:

To run Debian as a guest operating system without a graphical desktop, the VM should provide two virtual cores and 2 GB RAM. The VM should be UEFI-based in order to boot guest operating systems. The Debian installation should be done via a "net installer" ISO-CDROM image. For Internet access, the VM is connected to the host's network via a Bridge network.

To interact with the Debian installer, a VNC connection to the VM should be enabled. The required scripts, the files for the UEFI and virtual data carriers (CDROM ISO and hard disk) should also be saved on a ZFS dataset so that the state can be backed up via a ZFS snapshot after the operating system has been installed and restored at a later date.

An overview of the configuration is shown in the following figure:

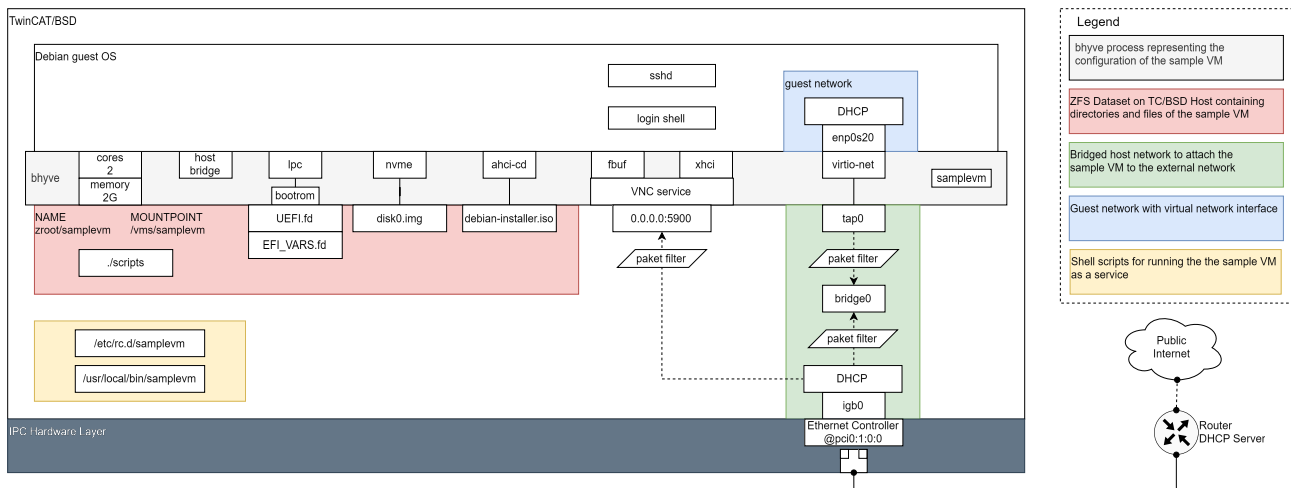


Fig. 28: Debian Linux sample VM

Prerequisites:

- Root rights are assumed for all the following commands. Switch to a root shell via `doas sh`.

Proceed as follows to set up the configuration:

1. Create a ZFS data sets as storage location for virtual machines [► 78] configuration:

```
zfs create -p -o mountpoint=/vms/samplevm zroot/samplevm
```

2. Download and unzip the GitHub sample scripts to /vms/samplevm/scripts:

```
fetch -o /home/Administrator/main.zip \
https://github.com/Beckhoff/TCBSD_Hypervisor_Samples/archive/refs/heads/main.zip
unzip -d /vms/samplevm /home/Administrator/main.zip
mv /vms/samplevm/TCBSD_Hypervisor_Samples-main/vm_autostart /vms/samplevm/scripts
rm -rf /vms/samplevm/TCBSD_Hypervisor_Samples-main /home/Administrator/main.zip
```

3. The next step is to adapt the bhyve call within /vms/samplevm/vm_autostart/sample. The file can be opened and edited with ee:

```
ee /vms/samplevm/scripts/samplevm
```

4. Search for the bhyve call and adjust the parameters as follows:

```
bhyve \
-c sockets=1,cores=2,threads=1 \
-m 2G \
-l bootrom,/vms/samplevm/UEFI.fd,/vms/samplevm/EFI_VARS.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 2,fbuf,rfb=0.0.0:5900,w=1280,h=1024 \
-s 3,xhci,tablet \
-s 10,nvme,/vms/samplevm/disk0.img \
-s 15,ahci-cd,/vms/samplevm/debian-installer.iso,ro \
-s 20,virtio-net,tap0 \
-s 31,lpc \
-A -H -P -w \
"${vm_name}"
```

5. Close the editor with ESC and save the changes to the file.

6. The adapted bhyve call references files in the program parameters that are not yet available on the TwinCAT/BSD host. The necessary files are therefore created in the next steps.

7. For the installation of the operating system the Debian "network install" CD-ISO should be used. The ISO file can be downloaded with `fetch(8)`, as described in the chapter [Use of installation media \(ISO images\)](#) [► 82], and later passed to the bhyve call as a disk of an ahci-hd device:

```
fetch -o /vms/samplevm/debian-installer.iso https://cdimage.debian.org/mirror/cdimage/archive/12.0.0/amd64/iso-cd/debian-12.0.0-amd64-netinst.iso
```

8. To install Debian on a virtual hard disk, the following command creates an empty disk image, which is used as a backend for the emulated nvme device in the bhyve call above:

```
truncate -s 20G /vms/samplevm/disk0.img
```

9. Debian uses EFI variables to store information about bootable disks. Therefore, a copy of the file `/usr/local/share/uefi-firmware/BHYVE_BHF_UEFI_VARS.fd` as well as the UEFI firmware itself should be stored in the ZFS dataset directory for the virtual machine:

```
cp /usr/local/share/uefi-firmware/BHYVE_BHF_UEFI.fd /vms/samplevm/UEFI.fd
cp /usr/local/share/uefi-firmware/BHYVE_BHF_UEFI_VARS.fd /vms/samplevm/EFI_VARS.fd
```

10. For the Debian installation, the virtual machine requires an Internet connection. For this purpose, a **Bridged network** [► 86] is created on the TwinCAT/BSD host as shown in the figure above:

```
ifconfig bridge create name bridge0
ifconfig tap create name tap0
ifconfig bridge0 addm tap0 addm igb0 up

sysrc cloned_interfaces+="bridge0 tap0"
sysrc ifconfig_bridge0="addm tap0 addm igb0 up"

echo "net.link.bridge.pfil_bridge=0" >> /etc/sysctl.conf
echo "net.link.bridge.pfil_member=0" >> /etc/sysctl.conf
sysctl -f /etc/sysctl.conf
```

11. All files referenced in the program parameters of the bhyve call adapted above are now available on the TwinCAT/BSD host. Accordingly, the samplevm script can be started in the directory `/vms/samplevm/scripts` :

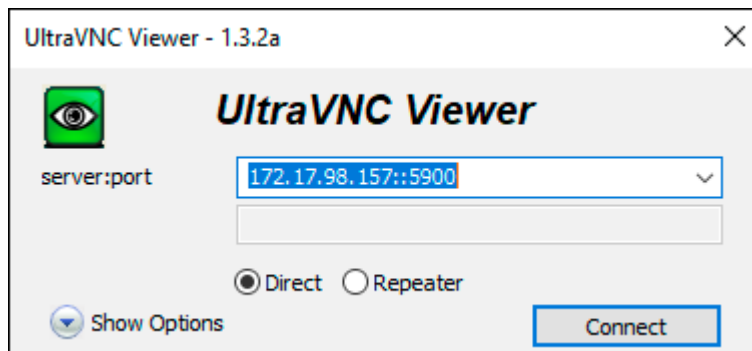
```
cd /vms/samplevm/scripts
sh -x samplevm start
```

12. The `-x` flag outputs the executed commands in the script on the command line. As soon as the `bhyve (8)` process is executed, the following output should appear on the command line:

```
+ bhyve_rc=0
+ bhyve -c 'sockets=1,cores=2,threads=1' -m 2G -l 'bootrom,/vms/samplevm/UEFI.fd,/vms/samplevm/EFI_VARS.fd,fwcfg=qemu' -s 0,hostbridge -s '2,fbuf,rfb=0.0.0:5900,w=1280,h=1024' -s 3,xhci,tablet -s 10,nvme,/vms/samplevm/disk0.img -s 15,ahci-cd,/vms/samplevm/debian-installer.iso,ro -s 20,virtio-net,tap0 -s 31,lpc -A -H -P -w samplevm

fbuf frame buffer base: 0x12ec16e00000 [sz 16777216]
```

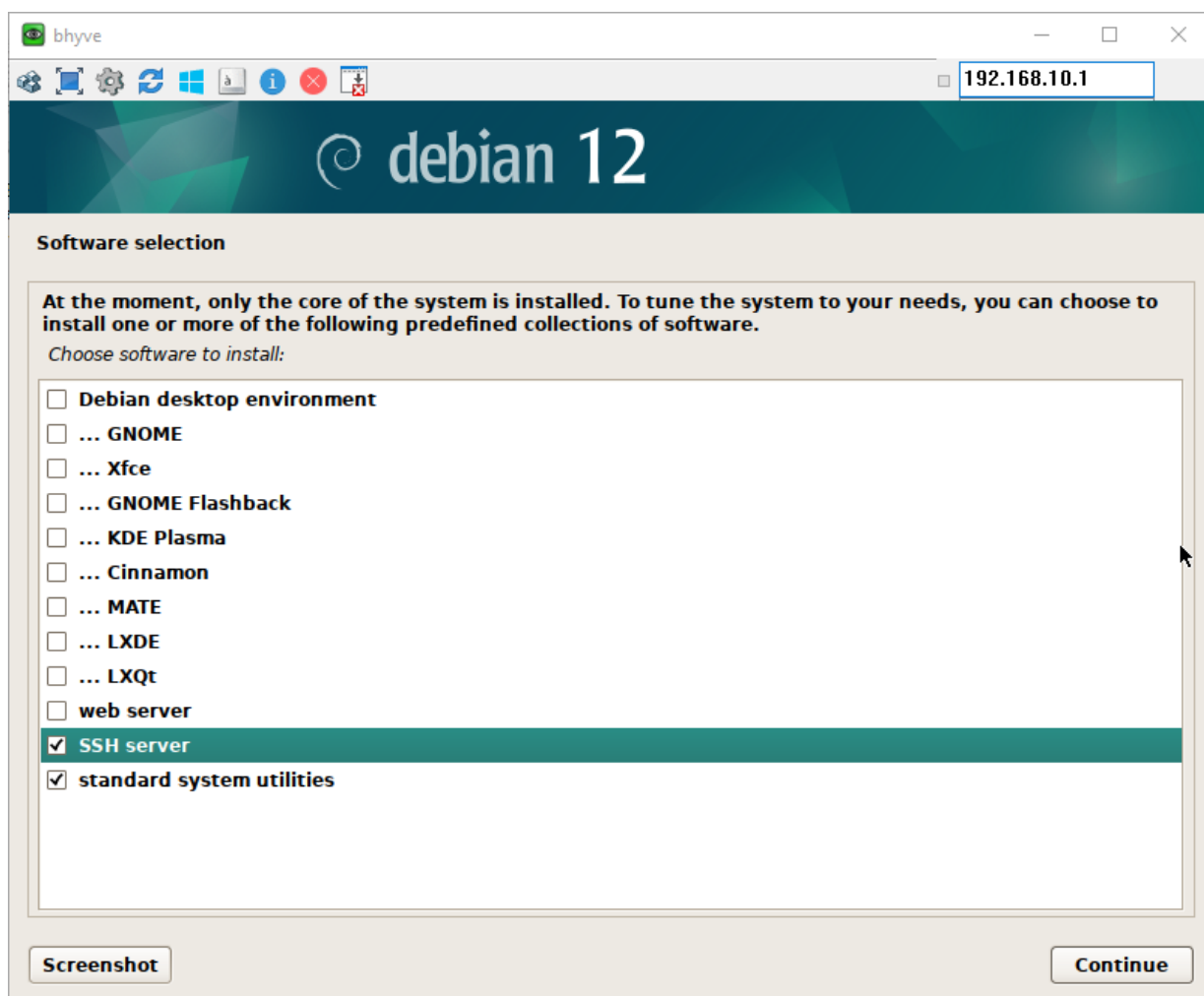
13. Now a VNC client, such as **Ultra-VNC** can be used to connect to the virtual machine:



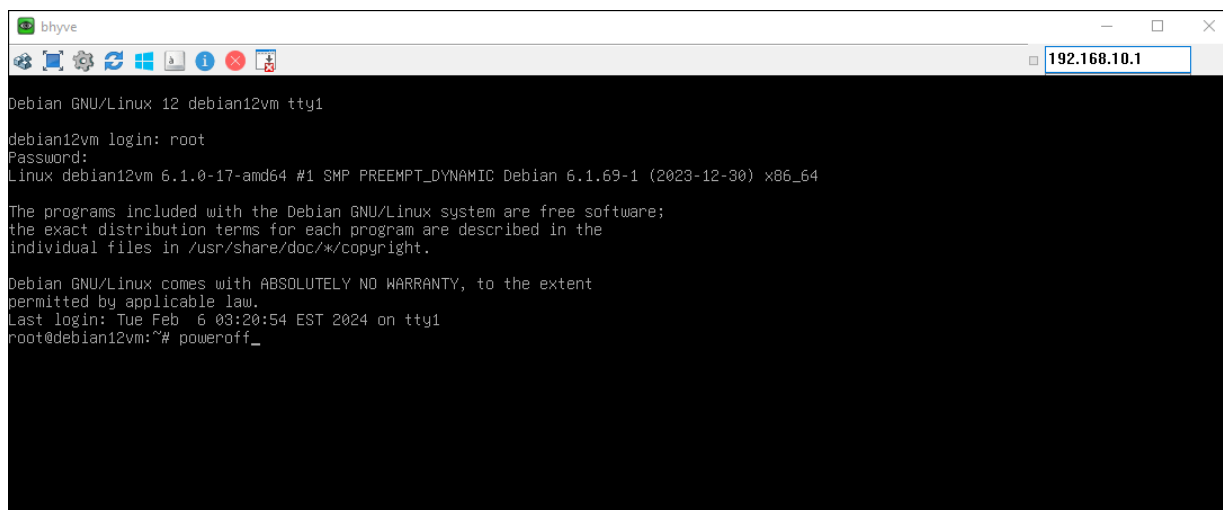
14. After a successful connection via a VNC client, the installation menu is displayed. Start the installation of Debian:



15. Since Debian is to be installed without a graphical desktop, the options "Debian Desktop Environment" and "... GNOME" can be deselected. The "SSH server" option must be selected so that the Debian command line can later be accessed via an SSH client.



16. As soon as the installation of Debian is complete, the virtual machine is restarted. This will terminate the VNC connection. Once the connection is restored, you can use VNC to run the Debian operating system in the virtual machine.
17. The command `ip --brief address` can be used to output the network address of the virtual machine in the network. The network address can be used, for example, to establish an SSH connection with an SSH client for remote access to the Debian operating system.
18. The VM should be shut down for the final steps.



19. If a connection to the VM can be established via SSH and the operating system is installed, the bhyve call can be adapted because neither the VNC connection nor the Debian installer are required for the next boot processes:

```
bhyve \
-c sockets=1,cores=2,threads=1 \
-m 2G \
-l bootrom,/vms/samplevm/UEFI.fd,/vms/samplevm/EFI_VARS.fd,fwcfg=qemu \
-s 0,hostbridge \
-s 10,nvme,/vms/samplevm/disk0.img \
-s 20,virtio-net,tap0 \
-s 31,lpc \
-A -H -P -w \
"${vm_name}"
```

20. To save the current state of the configuration, a ZFS snapshot of the dataset can be created with the following command. The current state of the autostart scripts, as well as the files for the virtual hard disks and the UEFI of the virtual machine, are backed up and can be restored to the current state:

```
zfs snapshot zroot/samplevm@os-installed
```

The VM configuration can be registered as a service via the Makefile. This allows the script to be stored in Autostart and managed via service(8):

```
cd /vms/samplevm/scripts
make install
service samplevm enable
service samplevm status
service samplevm start
service samplevm status
service samplevm stop
```

10.7 Windows as guest operating system

With the integrated hypervisor, the TwinCAT/BSD operating system provides the option of running Windows in a virtual machine (VM for short).

This functionality is particularly interesting for customers and users who cannot do without Windows but still want to use TwinCAT/BSD with all TwinCAT 3 runtime functions. TwinCAT/BSD including Windows VM can either be ordered ex factory or subsequently installed to existing industrial PCs.

In both cases, a valid license is required to operate Windows in a virtual machine.

Ordering options for ex factory installation

The following ordering options make it possible to order a TwinCAT/BSD Industrial PC ex factory with a preconfigured and licensed Windows VM:

- [C9900-S620: Windows 10 pre-installed with Device Passthrough \[► 99\]](#) for industrial PCs with integrated graphics card and preconfigured graphics output.
- [C9900-S621: Windows 10 pre-installed without Device Passthrough \[► 100\]](#) and without preconfigured graphics output.

Subsequent installation on existing industrial PCs

Delivered industrial PCs can be subsequently converted to TwinCAT/BSD (C9900-S60x) with optional Windows VM (C9900-S62x). The upgrade process can be initiated via the Beckhoff Service. All necessary files are provided by Beckhoff Service.

- [Subsequent installation of Windows VM with vm-installer \[► 101\]](#)

10.7.1 C9900-S620: Windows 10 pre-installed with Device Passthrough

With the C9900-S620 ordering option, TwinCAT/BSD-based industrial PCs are supplied with a preconfigured and licensed Windows 10 VM. With the configuration, the integrated graphics card, the USB controller and an Ethernet interface are explicitly passed through to the Windows VM via Device Passthrough. This allows the Windows environment to be shown on a display and operated via USB input devices (mouse, keyboard, multi-touch).

The following figure shows the schematic configuration of the C9900-S620 ordering option.

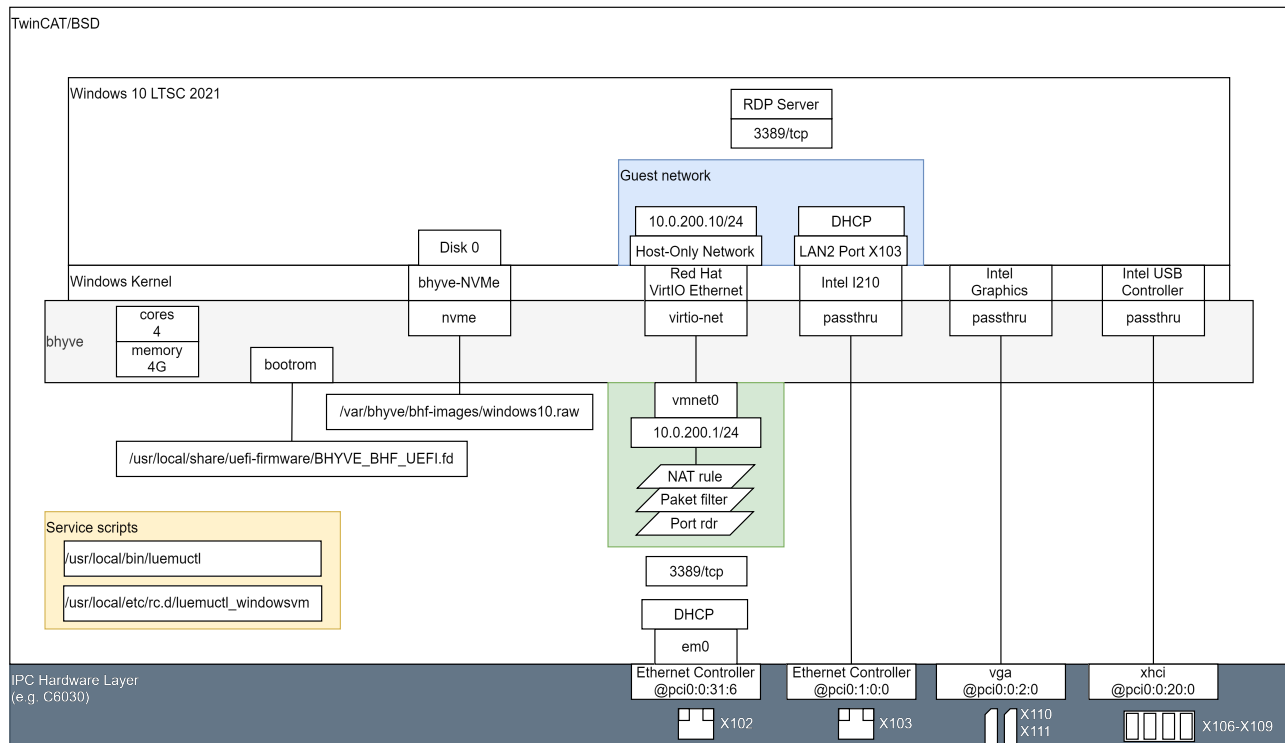


Fig. 29: C9900-S620 ordering option with preconfigured Windows 10 VM with Device Passthrough.

The Windows VM is assigned 4 CPU cores and 4 GB of main memory by default. The VM boots in UEFI mode and loads the installed Windows operating system from disk 0. The virtual disk 0 has a size of 30 GB and contains the Windows C:\ partition in addition to the boot partition. The corresponding disk file can be found on delivery at `/var/bhyve/bhf-images/windows10.raw`.

Windows has two configured network interfaces ex factory. The services of the TwinCAT/BSD host can be accessed via the Host-Only network (web console, SSH, etc.). At the same time, the network configuration enables outgoing network communication via the host's network using a NAT rule between the network of `vmnet0` and `em0`.

In the delivery state, port forwarding for the remote desktop protocol is also stored. The Windows VM can thus be accessed via remote desktop client using the IP address of the TwinCAT/BSD host.

The network interface LAN2 Port X103 directly accesses the Ethernet controller on port X103 of the IPC. Accordingly, a direct connection can be established between the connected network and the Windows VM.

The autostart of the Windows VM is controlled ex factory via the program `/usr/bin/luemuctl` and the corresponding rc script `/usr/local/etc/rc.d/luemuctl_windowsvm`. Accordingly, the commands `service luemuctl_windowsvm [enable|disable|start|stop|status]` are available for managing the service.

Further information on `luemuctl` can be found on the man pages `man luemuctl` and `man luemuctl-service`.

To configure the Windows VM with Device Passthrough, the industrial PC must meet the following minimum requirements:

- Processor options Intel® Core™ i3, i5 or i7 of the ninth or eleventh generation

- Memory expansion to at least 8 GB DDR4 RAM
- Data carrier size at least 80 GB
- C9900-B617: Special BIOS to support GPU Passthrough for Intel® Core™ of the ninth generation

10.7.2 C9900-S621: Windows 10 pre-installed without Device Passthrough

With the C9900-S621 ordering option, TwinCAT/BSD-based industrial PCs are supplied with a preconfigured and licensed Windows 10 VM. This ordering option is intended for use cases in where the Windows VM is to be accessed via remote desktop and therefore does not require Device Passthrough from devices or can be run on IPCs that do not support Device Passthrough.

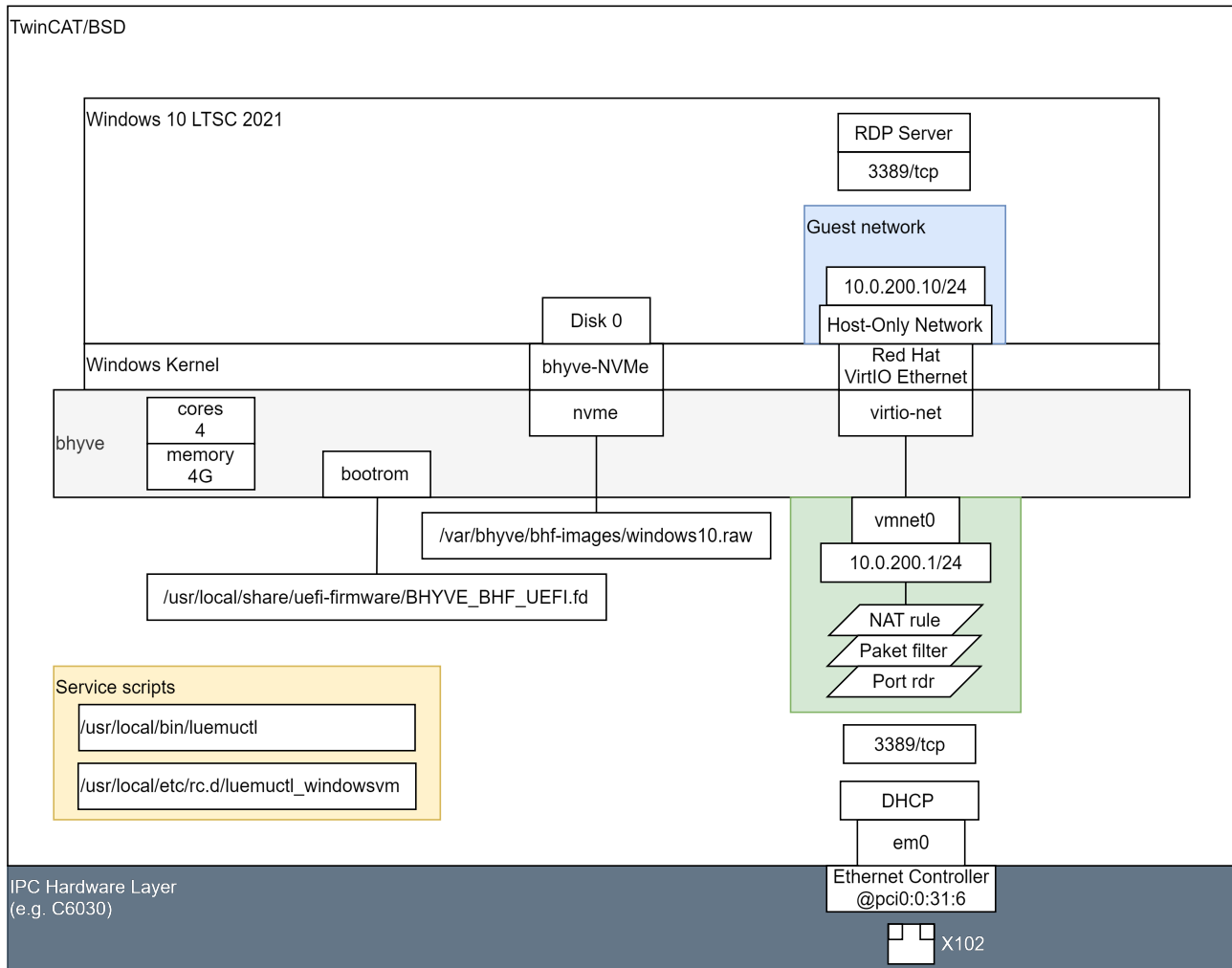


Fig. 30: C9900-S621 Ordering option with preconfigured Windows 10 VM.

The Windows VM is assigned 4 CPU cores and 4 GB of main memory by default. The VM boots in UEFI mode and loads the installed Windows operating system from disk 0. The virtual disk 0 has a size of 30 GB and contains the Windows C:\ partition in addition to the boot partition. The corresponding disk file can be found on delivery at `/var/bhyve/bhf-images/windows10.raw`.

Windows has a configured network interface ex factory. The services of the TwinCAT/BSD host can be accessed via the Host-Only network (e.g. web console, SSH, etc.). At the same time, the network configuration enables outgoing network communication via the host's network through a NAT rule between the network of `vmnet0` and `em0`.

In the delivery state, port forwarding for the remote desktop protocol is also stored (TCP port 3389). The Windows VM can thus be accessed via remote desktop client using the IP address of the TwinCAT/BSD host.

The autostart of the Windows VM is controlled ex factory via the program `/usr/bin/luemuctl` and the corresponding rc script `/usr/local/etc/rc.d/luemuctl_windowsvm`. Accordingly, the commands `service luemuctl_windowsvm [enable|disable|start|stop|status]` are available to manage the service.

Further information on `luemuctl` can be found on the man pages `man luemuctl` and `man luemuctl-service`.

For use without GPU passthrough in combination with a Windows VM, the industrial PCs must meet the following minimum requirements:

- Processor options Intel® Core™ i3, Core™ i5 or Core™ i7 of the ninth or eleventh generation, CX20x2 with Intel® Xeon®-D, CX20x3 and CX56xx with AMD Ryzen™
- at least 8 GB DDR4 RAM
- at least 80 GB storage medium

10.7.3 Subsequent installation of Windows VM with `vm-installer`

The `-S620` and `-S621` ordering options can be subsequently installed on existing TwinCAT/BSD devices using the program `vm-installer`.

A valid TwinCAT/BSD and Windows VM license is required. Both licenses can be requested as a license upgrade from Beckhoff Service (service@beckhoff.com).

When you order the Windows VM, you will be provided with a personal download link for the Windows disk image.

You can load the image directly onto the TwinCAT/BSD host with the following `fetch(8)` command. Alternatively, the loaded file can also be copied to the TwinCAT/BSD system using WinSCP.

```
fetch https://fileexchange.beckhoff.com/fop/dMtKokIr/IN-0410-0612-08-1-154031.raw.zstd
```

The compressed disk image can then be unzipped with `unzstd(8)` and written to the `windows.raw` file as follows:

```
unzstd -o windows.raw IN-0410-0612-08-1-154031.raw.zstd
```

The unpacked disk image (`windows.raw`) is used by the program `vm-installer` to configure a virtual machine according to the `S620` or `S621` option.

The program `vm-installer` is installed with the following `pkg` call:

```
doas pkg install -y vm-installer
```

To install a Windows VM according to the ordering option `C9900-S620` with `vm-installer` on the system, the following call can be used:

```
doas vm-installer install \  
--logfolder="/var/log/windowsvm" \  
--nat-if=em0 \  
--passthru \  
windows.raw
```

To install a Windows VM according to the ordering option `C9900-S621`, the following call can be used:

```
doas vm-installer install \  
--logfolder="/var/log/windowsvm" \  
--nat-if=em0 \  
windows.raw
```

In both cases, the disk image `windows.raw` is initialized and the Windows guest system is set up accordingly.

After installation, the status of the VM can be queried using the command `doas service luemuctl_windowsvm status`.

The running Windows VM can be accessed by the user via remote desktop using the IP address of the TwinCAT/BSD host.

10.7.4 Adapting the Windows VM configuration

The order ex factory or the subsequent installation of C9900-S620/-S621 with `vm-installer` use the rc script `/usr/local/etc/rc.d/luemuctl_windowsvm` to manage the VM as a service with the program `/usr/bin/luemuctl`.

Further information on `luemuctl` can be found on the man page `man luemuctl`.

The command `luemuctl run [OPTIONS] <vm_disk>` can be adapted in the file `/usr/local/etc/rc.d/luemuctl_windowsvm` in accordance with the man pages.

Alternatively, the disk image used can be referenced as a disk image in your own shell scripts if the initialized Windows environment is to be used in extended VM configurations.

11 C/C++ projects for TwinCAT/BSD

In this chapter, you will learn how to use executable code

- to compile locally on the industrial PC under TwinCAT/BSD
- or remotely compile and debug on TwinCAT/BSD® target systems.

11.1 Compiling under TwinCAT/BSD

This step shows how to generate executable code directly under TwinCAT/BSD with the LLVM compiler. For this purpose, a sample C/C++ project is created, which will use the ADS interface. To use the functions of TcAdsDll in your C/C++ project, you have to integrate the header file TcAdsAPI.h in your project.

The ADS header file is located in the following directory:

usr/local/include/TcAdsAPI.h

Sample C/C++ project *adstest.c*

```
#include <stdio.h>
#include <stdint.h>
#include "TcAdsAPI.h"

int main(){
    printf("ADS Test Sample\n");

    long nTemp;
    AdsVersion* pDLLVersion;
    nTemp = AdsGetDllVersion();
    pDLLVersion = (AdsVersion *)&nTemp;
    printf("Version: %d\n", (int)pDLLVersion->version);
    printf("Revision: %d\n", (int)pDLLVersion->revision);
    printf("Build: %d\n", (int)pDLLVersion->build);
    //printf("Ads DLL Version: %d\n", version);

    long l_port;
    l_port = AdsPortOpen();
    printf("Port opened: %ld\n", l_port);
    AdsPortClose();
    printf("Port closed\n");
    return 0;
}
```

Requirements:

- Install the developer package with the command `doas pkg install os-generic-userland-devtools`

Generate executable code under TwinCAT/BSD as follows:

1. Copy the file *adstest.c* to any TwinCAT/BSD directory. Example: */usr/local/ADSinterface*
2. Navigate to the directory with the example file *adstest.c*
3. Use the command `doas cc -c -I /usr/local/include/ -D POSIX adstest.c -o adstest.o` to compile the file *adstest.c*.
4. Link the compiled file with the ADS library with the command `cc -lpthread adstest.o /usr/local/lib/libTcAdsDll.so -o adstest`
5. Execute the file with `./adstest`.

11.2 Remote debugging of applications with Visual Studio Code (VSC)

This section describes how to use your Windows-based development environment to compile and debug applications on TwinCAT/BSD target systems.

The basic idea is that you keep all your source code on the Windows development computer. The source code editing is done on Windows and the debugging front end also runs on your normal Windows development computer. An SSH connection is used for compiling and debugging on TwinCAT/BSD. This process can be divided into the following steps:

1. Synchronize the source code from the Windows development computer to the TwinCAT/BSD target system via SSH using `rsync`.
2. Compile the application on the target system via an SSH connection.
3. Debug using VSC via an SSH connection to `gdb`, which runs on the target system.

Requirements:

- Ensure that the following packages are installed:
 - `llvm`
 - `os-generic-userland-devtools`
 - `gdb`
 - `rsync`
- Install OpenSSH for Windows: https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_install_firstuse
- Create a user key for the development computer. We recommend the algorithm `ed25519` (`ssh-keygen -t ed25519`) https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_keymanagement#user-key-generation
- Copy the public key to your target system using PowerShell:


```
(Get-Content -Encoding UTF8 $home\.ssh\id_ed25519.pub).Replace("`r`n", "`n") | ssh Administrator@172.17.66.111 'cat >> ~/.ssh/authorized_keys'
```
- Install the VSC remote debugging extension on your development computer:


```
(CTRL+p) ext install ms-vscode.cpptools
```
- Install `cwrsync` at `C:\Program Files\cwrsync` on your development computer. This software is an `rsync` implementation for Windows and is available for example at: <https://www.itefix.net/cwrsync>

VSC settings:

Now `cwrsync` and `ssh` are available, and you can continue to set up VSC for remote debugging.

1. Start by synchronizing the source code. Make sure you have a `.vscode` folder in your project directory. Then select the following from the VSC menu:

```
Terminal->Configure Tasks...->Create tasks.json from template->Others
```

2. Customize the following template according to your configuration to get a task like `sync-src-to-remote`.

```
{
  "version": "2.0.0",
  "windows": {
    "options": {
      "shell": {
        "executable": "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe",
        "args": [
          "-NoProfile",
          "-ExecutionPolicy",
          "Bypass",
          "-Command"
        ]
      }
    }
  },
  "tasks": [
    {
      "label": "sync-src-to-remote",
      "type": "shell",
      "args": [
        "-aP",
        "--exclude",
        ".git",
        "--exclude",

```

```

        "This folder should be ignored",
        "-e", "c:/Program Files'/cwrsrc/bin/ssh.exe -i ${userHome}/.ssh/id_ed25519 -o
'StrictHostKeyChecking no'",
        ".",
        "Administrator@172.17.66.111:~/${workspaceFolderBasename}/"
    ],
    "command": "c:/Program Files/cwrsrc/bin/rsync.exe",
    "problemMatcher": []
  }
}

```

3. You can test the new task as follows:

Menu->Terminal->Run Task->sync-src-to-remote

4. Now you should see something in the terminal output window and the source files should appear on your target system. If you see "permission denied" errors, try these additional rsync arguments:

```
--chmod=u=rwX --chmod=go=rX"
```

5. If you use another SSH port, add it with -p <Port>:

```

c:/Program Files'/cwrsrc/bin/ssh.exe -p 22222 -i ${userHome}/.ssh/id_ed25519 -o
'StrictHostKeyChecking no'

```

6. The next step is to build your application remotely on your target system. Add another task build-on-remote to your tasks.json.

```

{
  "version": "2.0.0",
  "windows": {
    "options": {
      "shell": {
        "executable": "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe",
        "args": [
          "-NoProfile",
          "-ExecutionPolicy",
          "Bypass",
          "-Command"
        ]
      }
    }
  },
  "tasks": [
    {
      "label": "sync-src-to-remote",
      "type": "shell",
      "args": [
        "-aP",
        "--exclude",
        ".git",
        "--exclude",
        "This folder should be ignored",
        "-e", "c:/Program Files'/cwrsrc/bin/ssh.exe -i ${userHome}/.ssh/id_ed25519 -o
'StrictHostKeyChecking no'",
        ".",
        "Administrator@172.17.66.111:~/${workspaceFolderBasename}/"
      ],
      "command": "c:/Program Files/cwrsrc/bin/rsync.exe",
      "problemMatcher": []
    },
    {
      "dependsOn": "sync-src-to-remote",
      "label": "build-on-remote",
      "group": "build",
      "type": "shell",
      "args": [
        "Administrator@172.17.66.111",
        "cd ${workspaceFolderBasename}; clang++ -g -Wall -pedantic main.cpp -o main. bin"
      ],
      "command": "ssh",
      "problemMatcher": []
    }
  ]
}

```

7. This time you can use STRG+SHIFT+B to run the new build-on-remote task because we added it to the build group.

8. Add a launch configuration to debug your application remotely. Select the following in the menu:

Run->Add configuration

9. Edit the launch.json template to fit your requirements:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "debug-on-remote",
      "type": "cppdbg",
      "request": "launch",
      "cwd": "/home/Administrator/${workspaceFolderBasename}",
      "preLaunchTask": "build-on-remote",
      "program": "main",
      "stopAtEntry": false,
      "externalConsole": false,
      "sourceFileMap": {
        "/home/Administrator/${workspaceFolderBasename}": "${workspaceFolder}"
      },
      "pipeTransport": {
        "debuggerPath": "/usr/bin/gdb",
        "pipeProgram": "ssh",
        "pipeArgs": [
          "Administrator@172.17.66.111"
        ]
      },
      "MIMode": "gdb",
      "setupCommands": [
        {
          "description": "Enable pretty-printing for gdb",
          "text": "-enable-pretty-printing",
          "ignoreFailures": true
        }
      ]
    }
  ]
}
```

10. Adjust a few parameters to your requirements:

program: should be the binary on your remote machine you want to debug
pipeTransport->pipeArgs :username and ip address for your remote machine

Now you have a launch.json file. F5 will run a debugger connected to your program on the target computer.

Notice: Ensure that you compile with debug symbols enabled ("-g").

12 TwinCAT 3

12.1 Searching for target systems

Before you can work with the Industrial PC you must connect your local computer to the Industrial PC. After that you can search for the Industrial PC with the aid of the IP address or the host name and subsequently set it as the target system.

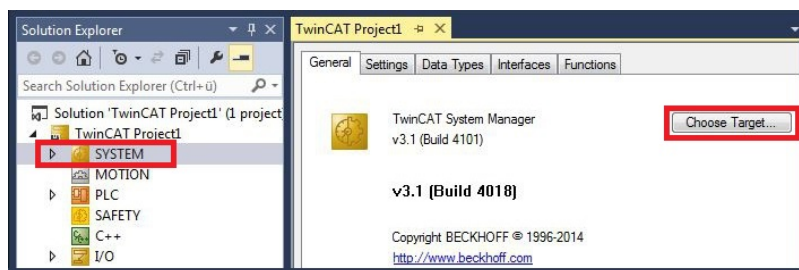
To do this the local PC and the Industrial PC must be connected to the same network or directly to each other via an Ethernet cable.

Requirements for this step:

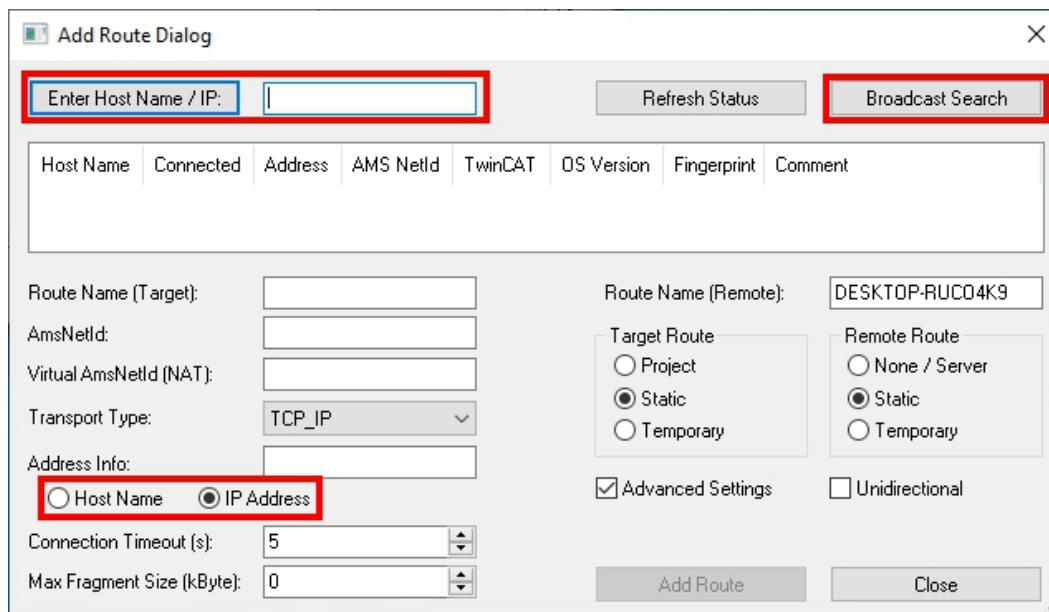
- TwinCAT 3 must be in Config mode.
- The IP address or the host name of the Industrial PC must be known.

Search for a target system as follows:

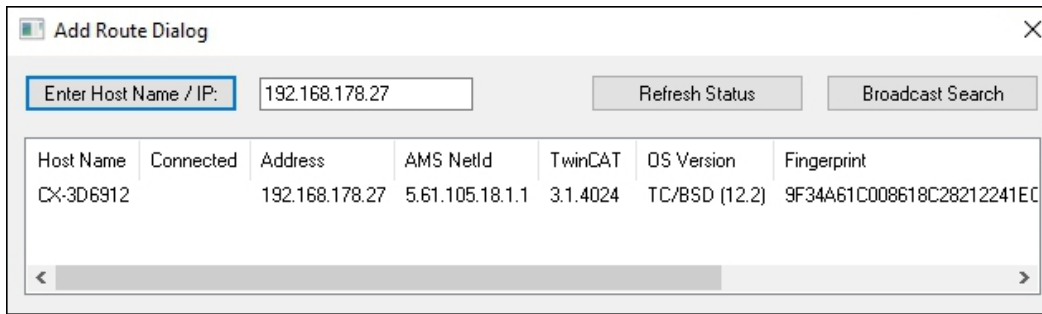
1. In the menu at the top click on **File > New > Project** and create a new TwinCAT XAE project.
2. In the tree view on the left click on **SYSTEM**, and then **Choose Target**.



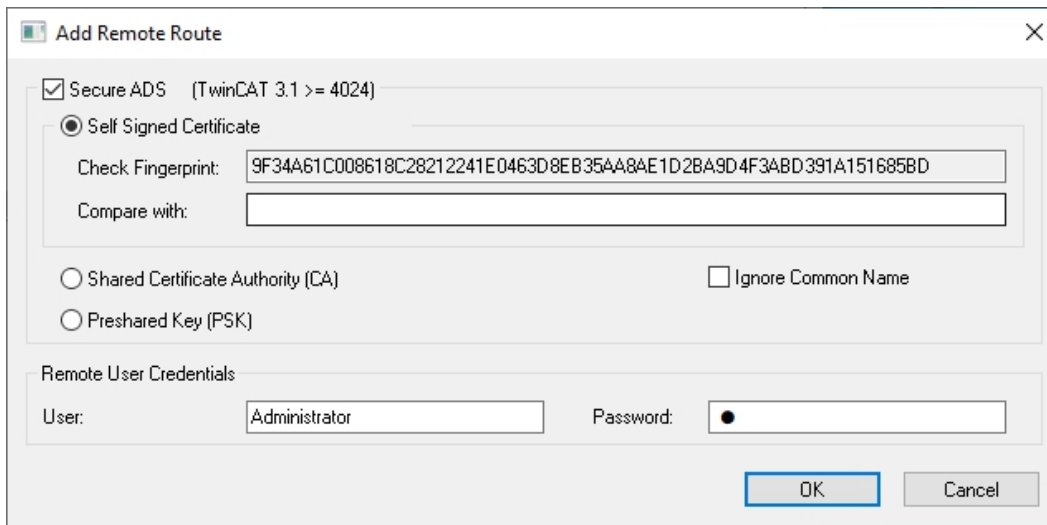
3. Click on **Search (Ethernet)**.
4. Use the **Broadcast Search** function to search for the Industrial PC. The **Broadcast Search** function does not work if the local computer and the Industrial PC are not located in the same subnet. In this case, enter the IP address of the Industrial PC under **Enter Host Name / IP** and press **[Enter]**.



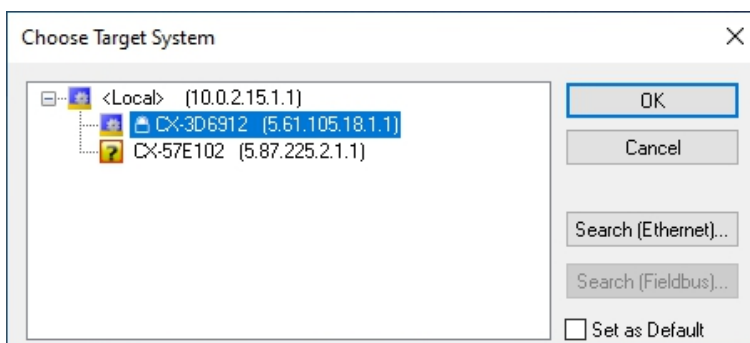
5. Mark the desired Industrial PC and click on **Add Route**.



6. Log in as "Administrator" with the password "1". Note that only Secure ADS is possible by default. For unencrypted ADS connections, you must open the corresponding ADS port in the firewall (see [Firewall](#) [p. 32]).



7. If you do not wish to search for any further devices, click on **Close** to close the Add Route window. The new device is displayed in the **Choose Target System** window.
8. Mark the Industrial PC that you wish to set as the target system and click on **OK**.



- ⇒ You have successfully searched for an Industrial PC in TwinCAT and inserted it as the target system. The new target system and the host name are displayed in the menu bar.



Using this procedure you can search for all available devices and also switch between the target systems at any time. Next, you can scan the Industrial PC.

12.2 Scan devices

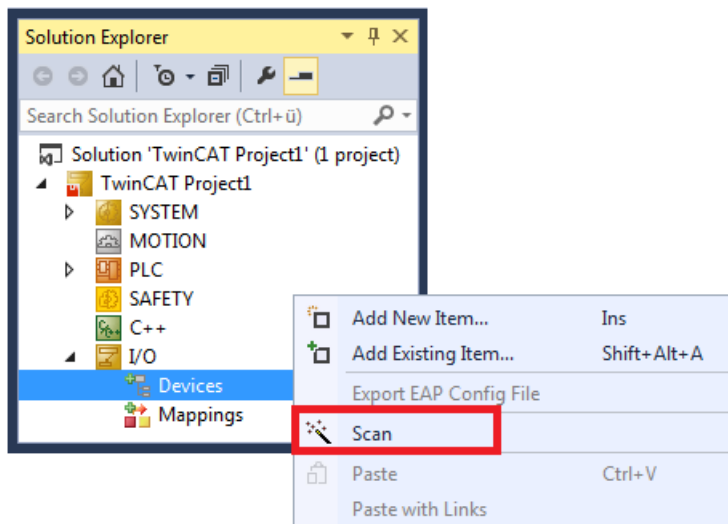
In this work step you will be shown how to scan an Industrial PC in TwinCAT and subsequently configure it.

Requirements for this step:

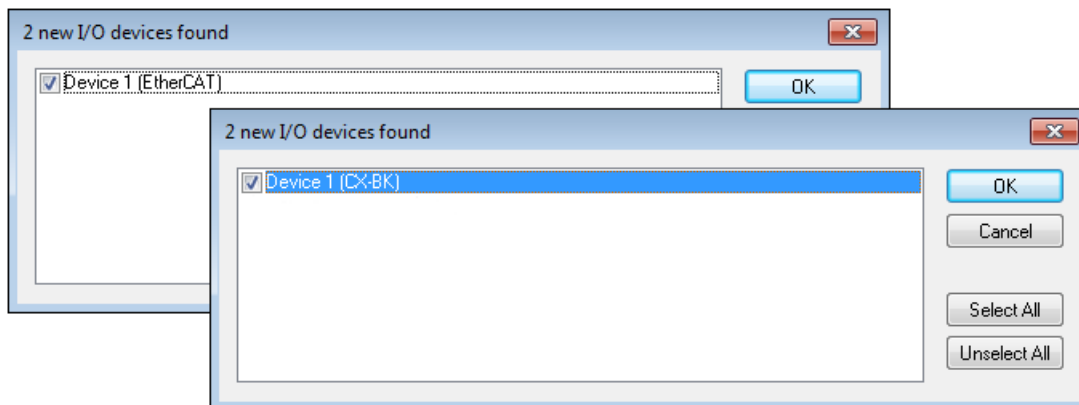
- An Industrial PC has already been selected as the target system.

Add the Embedded PC as follows:

1. Start TwinCAT and open an empty project.
2. In the tree view on the left, right-click on **I/O Devices**.
3. In the context menu click on **Scan**.



4. Select the devices you want to use and confirm the selection with **OK**.
Only devices that are actually available are offered for selection.



For Embedded PCs with connected Bus Terminals (K-bus) a Bus Coupler device (CX-BK) is displayed. With EtherCAT terminals (E-bus) the EtherCAT coupler will be displayed.

5. Confirm the request with **Yes**, in order to look for boxes.
 6. Confirm the request whether to enable FreeRun with **Yes**.
- ⇒ The Industrial PC was successfully scanned in TwinCAT and is displayed in the tree view with the inputs and outputs.

12.3 Changing the AMS NetID

This work step shows how you can change the AMS NetID of the industrial PC. Note that in doing so you will also change the address of the industrial PC in the TwinCAT network. The AMS NetID consists of 6 bytes and is represented in dot notation and in the hexadecimal system.

Requirements:

- Access rights to the file *TcRegistry.xml*

Proceed as follows:

1. Stop the TcSystemService with the command `doas service TcSystemService stop`
2. Enter the command `doas ee /usr/local/etc/TwinCAT/3.1/TcRegistry.xml` in the console.
The file *TcRegistry.xml* opens.

```
<Value Name="CurrentVersion" Type="SZ">3.1</Value>
<Key Name="System">
  <Value Name="RunAsDevice" Type="DW">1</Value>
  <Value Name="AmsNetId" Type="BIN">053B151A0101</Value>
</Key>
```

3. Change the AMS NetID under the entry `<Value Name="AmsNetId" Type="BIN">053B151A0101</Value>`.
The entry 053B151A0101 corresponds to the following AMS NetID: 5.59.21.26.1.1
 4. Press **[Esc]** and save the changes.
- ⇒ You have successfully changed the AMS NetID. Restart the TcSystemService with the command `doas service TcSystemService start`

12.4 Put TwinCAT into Run or Config mode

You can put TwinCAT to Run or Config mode directly from TwinCAT/BSD, i.e. with the help of the console. Control is provided by the tool *TcSysExe.exe*, which also provides information on licenses, different versions and system IDs. Retrieve more information with `TcSysExe.exe --help`.

Proceed as follows:

1. Enter the command `doas TcSysExe.exe --config` in the console to put TwinCAT into Config mode.
 2. Enter the command `doas TcSysExe.exe --run` in the console to put TwinCAT into Run mode.
- ⇒ The command `TcSysExe.exe --mode` displays the current TwinCAT status in the console.

12.5 Create or delete ADS routes manually

This step describes how you can manually create or delete an ADS route directly from TwinCAT/BSD. To configure ADS routes from TwinCAT/BSD, the tool `ads` can be used. The command `ads` displays all available parameters. Already existing ADS routes are listed in the file `StaticRoutes.xml`.

These settings are alternatively possible via the web interface of the Beckhoff Device Manager (see: [Beckhoff Device Manager: web interface](#) [► 65]).

Proceed as follows:

1. Create the ADS route according to the following pattern:

```
[<target[:port]>] [OPTIONS...] <command> [CMD_OPTIONS...] [<command_parameter>...]
```

2. At `<target>`, use the host name, IP address, or Ams Net Id of the target system to create a new ADS route.
3. For `<command>` use the command `addroute` and the following options:

```
--addr=<hostname> or IP address of the routes destination
--netid=<AmsNetId> of the routes destination
--password=<password> for the user on the remote TwinCAT system
--username=<user> on the remote TwinCAT system (optional, defaults to Administrator)
--routename=<name> of the new route on the remote TwinCAT system (optional, defaults to --addr)
```

4. Enter the command `ads 192.168.0.231 addroute --addr=192.168.0.1 --netid=192.168.0.1.1.1 --password=1 --routename=example.beckhoff.com` in the console.

⇒ Create new ADS routes according to the pattern shown or delete the unneeded ADS routes in the file `StaticRoutes.xml` under the entries `<Route>`.

```
---snipped---
<?xml version="1.0"?>
<TcConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <RemoteConnections>
    <Route>
      <Name>example.beckhoff.com</Name>
      <Address>192.168.0.1</Address>
      <NetId>192.168.0.1.1.1</NetId>
      <Type>TCP_IP</Type>
      <Flags>64</Flags>
    </Route>
    <Route>
      <Name>DESKTOP-RUC04K9</Name>
      <Address>192.168.40.88</Address>
      <NetId>192.168.2.15.1.1</NetId>
      <Type>TCP_IP</Type>
      <Flags>64</Flags>
    </Route>
  </RemoteConnections>
</TcConfig>
---snipped---
```

12.6 Increase heap memory

NOTICE

Oversized heap memory

If the heap memory is too large, the entire main memory is allocated, which leads to the system not working properly. Make sure that the heap memory is not selected too large in relation to the available main memory.

If the heap memory is too small for the download of a PLC project, a corresponding error is issued and the process is aborted.

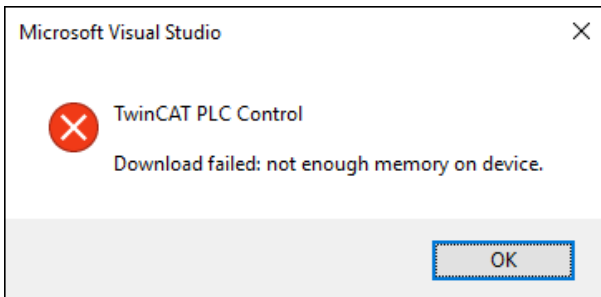


Fig. 31: Error message if the heap memory is too small.

When activating the TwinCAT project where the PLC project is in AutoStart, TwinCAT simply switches back to Config mode and there is no error message or similar.

The size of the heap memory is not automatically adjusted, but can be increased under TwinCAT/BSD for extensive PLC projects in the file `/usr/local/etc/TwinCAT/3.1/TcRegistry.xml`. To do this, the XML file must be expanded as follows:

Path: `HKEY_LOCAL_MACHINE\SOFTWARE\BECKHOFF\TWINCAT3\SYSTEM`
 Entry: `<Value Name="HeapMemSizeMB" Type="DW">{size in MB}</Value>`

Proceed as follows:

1. Stop the TcSystemService with the command `doas service TcSystemService stop`
2. Enter the `doas ee /usr/local/etc/TwinCAT/3.1/TcRegistry.xml` command in the console. The file `TcRegistry.xml` is opened.
3. Add the `<Value Name="HeapMemSizeMB" Type="DW">{size in MB}</Value>` entry to the XML file at `<Key Name="System">`.

```
---snipped---
<Key Name="System">
  <Value Name="RunAsDevice" Type="DW">1</Value>
  <Value Name="RTIMEMode" Type="DW">0</Value>
  <Value Name="AmsNetId" Type="BIN">0542F70C0101</Value>
  <Value Name="LockedMemSize" Type="DW">33554432</Value>
  <Value Name="SysStartupState" Type="DW">5</Value>
  <Value Name="HeapMemSizeMB" Type="DW">1024</Value>
---snipped---
```

4. The size is set in megabytes. In this sample, these are 1024 MB.
 ⇒ Restart the TcSystemService with the command `doas service TcSystemService start`. After the heap memory is increased to 1024 MB, the PLC project starts and the download is not aborted with an error.

12.7 Adapting the router memory

The main memory is used by TwinCAT/BSD and by TwinCAT (TwinCAT memory). The TwinCAT memory is further divided into the router memory and the PLC memory. The router memory is used for ADS communication and the PLC memory for the actual PLC program including TcConfiguration, mapping and data.

An adjustment of the router memory is only necessary if a large amount of ADS communication takes place and for this reason it becomes necessary to design the size of the router memory accordingly. By default the router memory is set in TwinCAT. The maximum value for the router memory is 1024 MB.

Make sure that the heap memory is larger than the router memory and otherwise increase the heap memory before adjusting the router memory (see: [Increase heap memory](#) [► 112]). This chapter shows how the router memory can also be customized under TwinCAT/BSD.

To do this, the XML file at `/usr/local/etc/TwinCAT/3.1/TcRegistry.xml` must be adapted as follows:

```
Path: HKEY_LOCAL_MACHINE\SOFTWARE\BECKHOFF\TWINCAT3\SYSTEM
Entry: <Value Name="LockedMemSize" Type="DW">{size in Byte}</Value>
```

Proceed as follows:

1. Stop the TcSystemService with the command `doas service TcSystemService stop`
2. Enter the `doas ee /usr/local/etc/TwinCAT/3.1/TcRegistry.xml` command in the console. The file `TcRegistry.xml` is opened.
3. Adjust the `<Value Name="LockedMemSize" Type="DW">{size in Byte}</Value>` entry in the XML file.

```
---snipped---
<Key Name="System">
  <Value Name="RunAsDevice" Type="DW">1</Value>
  <Value Name="RTimeMode" Type="DW">0</Value>
  <Value Name="AmsNetId" Type="BIN">0542F70C0101</Value>
  <Value Name="LockedMemSize" Type="DW">33554432</Value>
  <Value Name="SysStartupState" Type="DW">5</Value>
  <Value Name="HeapMemSizeMB" Type="DW">1024</Value>
---snipped---
```

4. In this sample a value of 33554432 bytes = 32 MB is set. For example, change the value to 67108864 bytes to increase the router memory to 64 MB.
- ⇒ Restart the TcSystemService with the command `doas service TcSystemService start`.

12.8 Assign isolated cores

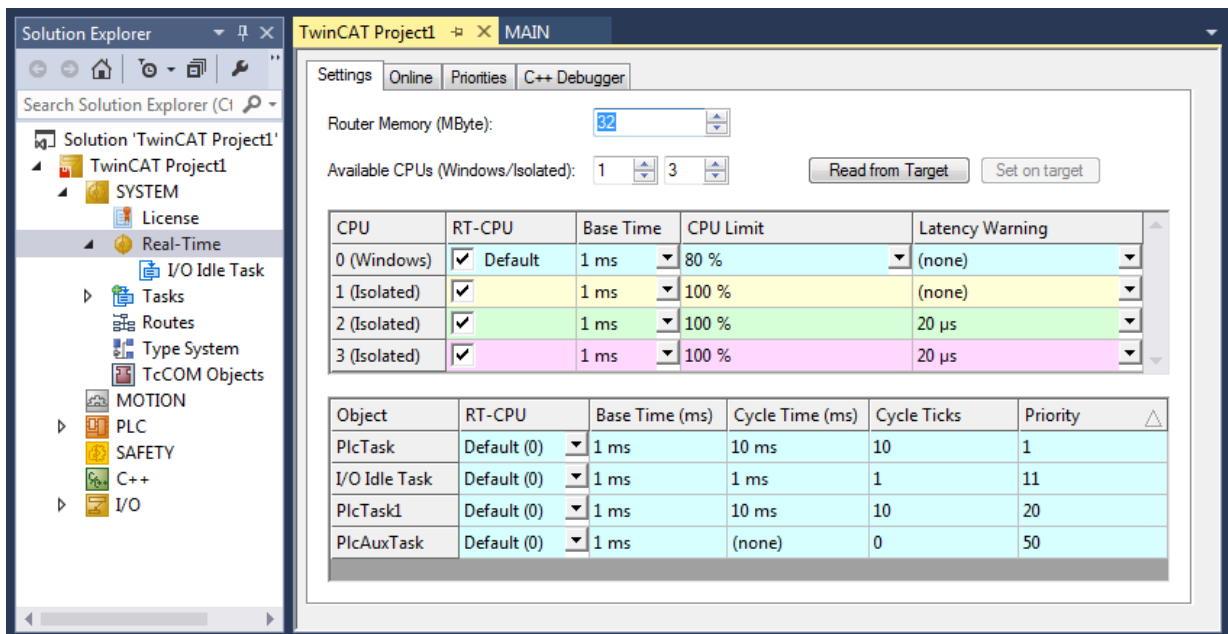
For multi-core systems TwinCAT 3 offers the possibility to isolate single cores. This allows different TwinCAT tasks to be assigned to a core isolated for real-time use. This section shows how to set isolated cores in the TwinCAT/BSD console.

Requirements:

- A multi-core Industrial PC. One CPU core (Shared) for TwinCAT/BSD and three CPU cores (Isolated) should be available for different TwinCAT tasks.

Proceed as follows:

1. With the command `TcCoreConf` an overview of the available cores and their definition Shared/Isolated can be displayed.
 2. Enter the command `doas TcCoreConf -s 1` in the console. This sets a CPU core (Shared) for TwinCAT/BSD. The remaining three CPU cores are isolated.
 3. Restart the Industrial PC with the command `shutdown -r now` to apply the settings.
 4. Then you can use the command `sysctl hw.ncpu` to display the number of CPU cores (Shared).
- ⇒ You have successfully configured one CPU core (Shared) and three CPU cores (Isolated). The settings can be controlled with the command `TcCoreConf`. You can also read out the current settings from the Industrial PC in TwinCAT 3 (XAE). To do this, click on the button **Read from Target** at **Real-Time**.



With `TcCoreConf --help` all available commands are displayed. With `doas TcCoreConf -d`, for example, all CPU cores can be reset to "Shared".

```
Administrator@CX-3B151A:~ % TcCoreConf -help
TcCoreConf:
  -s --set CPUs
    set number of shared cores
  -d --delete
    set all cores as shared core
  -f NAME, --file NAME
    set name of configuration file to change
  --rsdp ADDR
    set pointer for RSDP
  --show
    show active settings of shared/isolated cores
  --strip
    remove hints of unknown apic-ids
  --noflat
    don't add flat cpu topology setting
```

13 Restore options

Define a backup and recovery strategy for your TwinCAT/BSD system in order to restore TwinCAT/BSD in a very short time in the event of data loss or defective storage media. Backups help to minimize downtime and thus to allow work to continue without large production losses. Both a process for creating a backup copy and a process for restoring it should be defined. Security aspects should also be taken into account and, for example, the storage location where the backup is to be stored should be defined.

Beckhoff offers a simple backup solution with the TwinCAT/BSD installer stick. In addition, the `restorepoint` program makes restore points possible with TwinCAT/BSD; these restore points store the current state of the system and restore it if necessary. A variety of implementations are therefore available, with the exact definition of a backup and restore strategy left to the user.

The following scenarios are possible and are intended to help you understand the different modes of operation. However, the scenarios presented should not be considered the only way recommended by Beckhoff.

Scenario 1: Factory settings

An Industrial PC with TwinCAT/BSD is to be reset to the factory settings in case of a problem.

- The user tests and develops on an Industrial PC with TwinCAT/BSD.
- In the test and development phase, there is a problem because, for example, basic settings have been changed.
- The user solves the problem by resetting TwinCAT/BSD to the factory settings (see: [Resetting to factory settings](#) [► 116]).

Scenario 2: Series production

The test and development phase has been successfully completed. The machine manufacturer wants to start series production:

- The machine manufacturer creates a restore point (delivery state OEM) in order to be able to restore the system in the event of an error (see: [Creating a restore point](#) [► 116]). The machine manufacturer's end customer can use this restore point in case of problems.
- The machine manufacturer then activates the Write Filter to secure TwinCAT/BSD in the preconfigured state and to prevent a misconfiguration at the end customer (see: [Write filter](#) [► 26]).
- In the final step, the machine manufacturer creates a backup, which is stored as a master image and used for series production (see: [Creating a backup](#) [► 119]).

Scenario 3: Commissioning at the end customer

The machine arrives at the end customer and is to be backed up after commissioning:

- After parameterizing the machine, the end customer creates a restore point called "Commissioning" (see: [Creating a restore point](#) [► 116]).
- The end customer then activates the Write Filter in order to avoid accidental misconfiguration (see: [Write filter](#) [► 26]).
- The end customer creates his own backup (see: [Creating a backup](#) [► 119]) in order to be able to restore the system, for example, in the event of a defective data carrier (see: [Restoring a backup](#) [► 119]).

13.1 Restore point

Restore points are used to restore an old system state if TwinCAT/BSD exhibits undesirable behavior after a major system change or misconfiguration, and this behavior is not easy to rectify. The advantage of restore points is that these configuration errors are easily and quickly undone without reinstalling TwinCAT/BSD.

You define the time to create a restore point, for example, when you make a larger system change or install third-party programs. However, restore points are not a substitute for a full backup and do not protect against data loss. Regular backups are another protection measure that allows you to protect yourself from data loss due to defective storage media, for example (see: [Creating a backup](#) [► 119]).

The restore points are created and managed in the console using the `restorepoint` program. The following modes are supported by the program:

- `status`: Lists all available restore points. On delivery, a restore point named `factoryreset`, the Beckhoff factory settings, is available.
- `create`: Creates a new restore point. The name of the restore point can be set as an argument. If no name is specified, an automatically generated name is used.
- `rollback`: Return to a specific restore point. Note that all data created after the restore point will be destroyed. If no restore point is specified as an argument, the user is asked with an interactive dialog.
- `destroy`: The specified restore point is destroyed. In this mode, all existing data is preserved, but the restore point itself is deleted.

Restore points under TwinCAT/BSD are based on ZFS snapshots. As a result, they consume very little memory when they are created. Any change in the saved restore point for the current live system the user is working with is reflected in the memory space used by the restore point. Use `zfs list -t snap` to display all system snapshots.

The `USED` column shows the actual space used by the snapshot; the `REFER` column shows the space referenced by the snapshot but actually stored in other datasets. It is therefore always advisable to create a restore point before making any changes in the system, since this hardly uses any system resources. After some time and many changes between the restore point and the live system, it is recommended to delete restore points that are no longer needed in order to free up the increasing memory space used by the restore points.

13.1.1 Resetting to factory settings

You can reset TwinCAT/BSD to the factory settings at any time and restore the delivery status if, for example, the system no longer works properly after a misconfiguration.

The restore points are created and managed in the console using the `restorepoint` program. This section shows you how to reset TwinCAT/BSD to the factory settings.

Proceed as follows:

1. Enter the command `doas restorepoint rollback factoryreset` on the console.
 2. All snapshots to which the system is reset are displayed.
 3. Confirm the restoration with `[y]`.
- ⇒ The system is reset to the factory settings. After a restart, TwinCAT/BSD is in the delivery state again.

13.1.2 Creating a restore point

Memory consumption due to restore points

A restore point consumes storage space because the entire system is backed up, including kernel dumps at `/var/crash`. Clean up the system before creating a restore point or delete old restore points.

Restore points are used to restore an old system state if TwinCAT/BSD no longer works properly after a major system change or misconfiguration. Create restore points when you want to make major system changes, install programs or run tests.

The restore points are created and managed in the console using the `restorepoint` program. This section shows you how to create restore points in TwinCAT/BSD.

Proceed as follows:

1. Enter the command `doas restorepoint create` in the console.

- The restore point is created with an automatically generated name.
- Check the creation of the restore point with the command `restorepoint status` and have all restore points displayed.

```
Administrator@CX-4FAA38$ restorepoint status
last BE: zroot/ROOT/default
factoryreset
2020-08-28T08:56:14Z
2020-08-28T09:03:05Z
```

- Alternatively, use the command `doas restorepoint create your-restorepoint` in order to define your own name for the restore point.

⇒ The restore point is created and can be used at any time to reset the system (see: [Resetting to the restore point \[► 117\]](#)).

```
Administrator@CX-4FAA38$ restorepoint status
last BE: zroot/ROOT/default
factoryreset
2020-08-28T08:56:14Z
2020-08-28T09:03:05Z
your-restorepoint
```

13.1.3 Resetting to the restore point

NOTICE

Loss of data

Data and restore points created after a certain restore point are deleted when resetting to a previous restore point.

If TwinCAT/BSD no longer works properly after a misconfiguration, you can easily undo these configuration errors with the help of restore points without reinstalling TwinCAT/BSD.

Proceed as follows:

- Enter the command `restorepoint status` on the console to display all the restore points that can be used.

```
Administrator@CX-4FAA38$ restorepoint status
last BE: zroot/ROOT/default
factoryreset
2020-08-28T08:56:14Z
2020-08-28T09:03:05Z
your-restorepoint
```

- Enter the command `doas restorepoint rollback` in the console to see all existing restore points.
- Select a menu item to reset the system to a specific restore point.

```
Administrator@CX-4FAA38~ $ doas restorepoint rollback
Password:
1 factoryreset
2 2020-08-28T08:56:14Z
3 2020-08-28T09:03:05Z
4 your-restorepoint
```

- All snapshots to which the system is reset are displayed.
 - Confirm the restoration with **[y]**.
- ⇒ TwinCAT/BSD is reset to the restore point and restarted. Note that data and restore points created after the selected restore point are deleted during the reset.

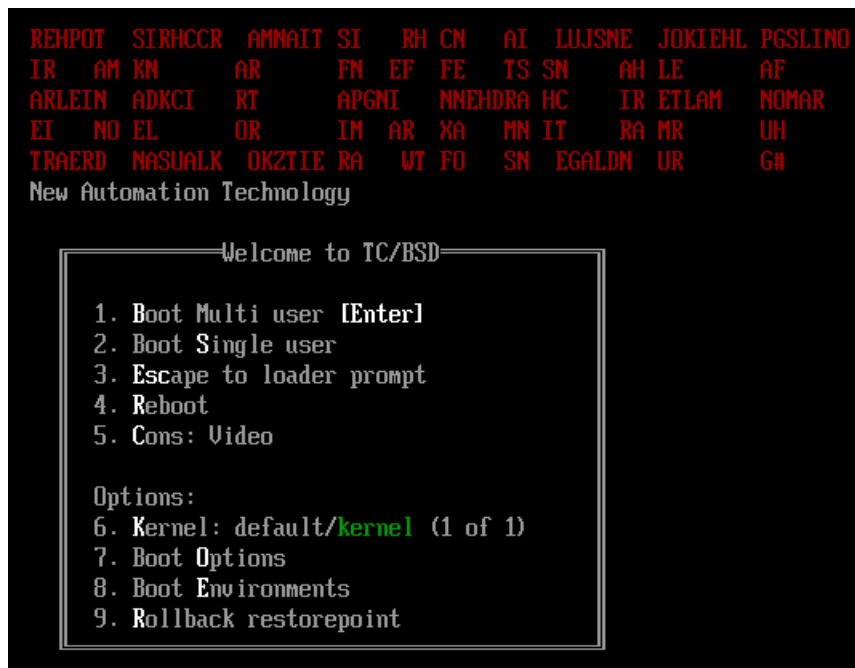
13.1.4 Using the restore boot environment

You can restore a restore point from the restore boot environment when TwinCAT/BSD no longer boots and the console is inaccessible as a result. To do this, start the boot menu during the boot process in order to switch to the restore boot environment.

Proceed as follows:

- Start the Industrial PC.

2. During the bootup, press and hold the **[Space bar]**. The boot menu appears.



3. Select the option **Rollback restorepoint**.
- ⇒ TwinCAT/BSD starts in the restore boot environment. Now you can restore the factory settings with the command `restorepoint rollback factoryreset` or use a specially created restore point (see: [Resetting to the restore point](#) [► 117]).

13.2 Backup and restore

Unlike a restore point, TwinCAT/BSD can be saved and managed as a backup copy on an external storage device by means of a backup.

This backup copy can be used to restore the system in the event of a system failure or data loss. Make regular backups from your system in order to restore your industrial PC to the state it was at the time of the backup.

13.2.1 Creating a backup

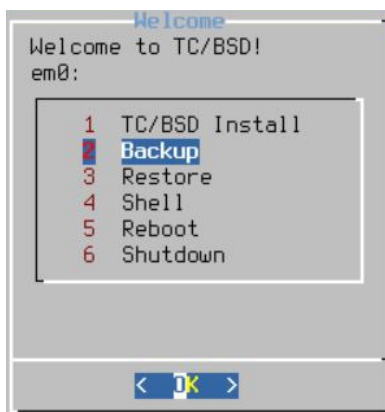
You can create and restore a backup using the TwinCAT/BSD installer stick. All backups are stored on a FAT32 partition on the USB stick. FAT32 is interoperable with Windows and FreeBSD®. This allows the backups created to be managed both with a TwinCAT/BSD system and with a Windows system.

Requirements:

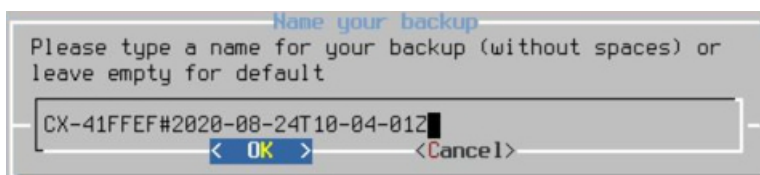
- TwinCAT/BSD installer stick (see: [Create bootable USB stick](#) [► 17]).

Create a backup as follows:

1. Connect the TwinCAT/BSD installer stick to the Industrial PC.
2. Boot the Industrial PC from the TwinCAT/BSD installer stick.
3. Open the boot menu with **[F7]** if the Industrial PC doesn't boot automatically from the USB stick.
4. Select the UEFI entry for the USB stick and confirm with **[Enter]**. The Industrial PC boots from the USB stick and the Beckhoff TwinCAT/BSD installer is run.
5. Select the option **Backup**.



6. Assign a file name to the backup or accept the default name made up of host name and timestamp.



7. Select the option **Reboot** for a reboot once the backup is complete.
- ⇒ The backups are stored on the USB stick with the respective file name. Archive the backups on the USB stick. You can also copy the backups to an external storage medium or archive them on the network.

13.2.2 Restoring a backup

● Use suitable backups to restore

i A backup can only be restored to one device within the same series, e.g. CX51x0, CX20x3, C6015 etc., otherwise incompatibilities may occur if the backup is restored to a device from a different series.

You can restore a backup with the aid of the TwinCAT/BSD installer stick. To do this, the industrial PC must be booted from the TwinCAT/BSD installer stick.

Requirements:

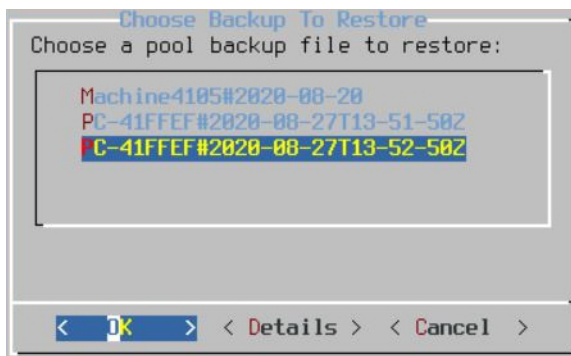
- TwinCAT/BSD installer stick (see: [Create bootable USB stick](#) [► 17]).

Proceed as follows:

1. Connect the TwinCAT/BSD installer stick to the industrial PC.
2. Boot the industrial PC from the TwinCAT/BSD installer stick.

Open the boot menu with [F7] if the industrial PC doesn't boot automatically from the USB stick.

3. Select the UEFI entry for the USB stick and confirm with **[Enter]**. The industrial PC boots from the USB stick and the Beckhoff TwinCAT/BSD installer is run. Select the **Restore** option.
4. Select the backup to be restored to the industrial PC.



⇒ Restart the industrial PC after restoring. The industrial PC is now in the state it was at the time of the backup.

13.2.3 Creating and restoring a backup from the live system

If required by your application, backups can also be created and restored from the live system, without a TwinCAT/BSD installer stick. Use the scripts TcBackup and TcRestore for this purpose.

Do not create a backup from the running system if the system is writing to the disk at the time of the backup. A backup can become corrupted if the system has write access to the disk during the backup. In other words, make sure that there are no processes running that persistently back up data and that the disk you want to restore your backup to has enough space.

Running TcBackup and TcRestore and writing to and from the file where the backup is saved must be done with root rights. In other words, execute a shell with root rights beforehand, in which you then work, or execute the single command as a string with a shell with root rights. The latter option is illustrated in the following examples.

Proceed as follows:

1. Enter the command `doas sh -c "TcBackup.sh --disk /dev/ada0 > backup.tcbkp00"` to create a backup from the ada0 disk to the Backup.tcbkp00 file.
 2. Enter the command `doas sh -c "TcRestore.sh --disk /dev/ada1 < backup.tcbkp00"` to restore a backup from the Backup.tcbkp00 file on the ada1 disk.
- ⇒ The two commands can be combined, as follows. The command `doas sh -c "TcBackup.sh --disk /dev/ada0 | TcRestore.sh --disk /dev/ada1"` creates a backup from the ada0 disk and immediately restores it to the ada1 disk.

14 Error handling and diagnostics

14.1 Using kernel messages for diagnosis

You can turn on kernel messages for your own diagnostic purposes or for Beckhoff Support. The kernel messages are displayed when booting from TwinCAT/BSD and stop at the error location. As a result, you or Beckhoff Support can locate errors.

Requirements:

- Restart required. The kernel messages can only ever be turned on during the bootup.

Proceed as follows:

1. Start the Industrial PC.
 2. Press the **[Spacebar]** during the boot process.
The boot process is paused.
 3. Press button **[6]** to select option **6. Boot Options**.
The **Boot Options** menu appears.
 4. Press button **[5]** to activate option **5. Verbose: on**.
 5. Press **[Enter]** to conclude the procedure.
- ⇒ TwinCAT/BSD continues to boot up and the kernel messages are displayed. These settings are not saved. Repeat the work steps shown if you require the kernel messages again at the next start.

14.2 Log files

Kernel, security and TwinCAT logs can be used for diagnostic purposes and are located in the following directories.

General kernel log

/var/log/messages

Open the kernel log with the following command:

```
cat /var/log/messages
```

Security log

/var/log/security

Open the security log with the following command:

```
doas cat /var/log/security
```

Filter the output of `cat` with the pipe operator and the program `grep`. The pipe operator passes the output of `cat` to the program `grep`, which can filter its input based on an expression. Use the expression `cat /var/log/messages | grep -i tc` to filter for all TwinCAT-relevant data. The option `-i` is case-insensitive.

The program `less` can be used to output and search text files. For example, **[&]** can be used to start the search function. Use **[h]** to display information on further functions. Use **[q]** to quit the program.

Another useful program pertaining to log files is `tail`. In this case only the last few entries are output. This is useful if only the most recent log entries are of interest. For example, the command `tail -5 /var/log/messages` outputs the last five entries of the log file.

14.3 Dumps

Kernel dump

The kernel dump can be found under
`/var/crash`

14.3.1 Using automatic process dump

When a process crashes, a memory dump is automatically created and stored in the file `progname.core`. The file contains the full process state at the time of the crash. The file is usually stored where the user-mode process is located: `/usr/local/bin`

The file can then be analyzed with a debugger such as `gdb`. The following section describes the local analysis of the dump directly on the TwinCAT/BSD system. Of course, the analysis can also be performed with suitable programs on the Windows development computer. To do this, copy the dump to your Windows computer using WinSCP, for example.

Requirements:

- Install the debugger `gdb` with the command `pkg install gdb`.

Proceed as follows:

1. Navigate to the appropriate directory if you are not in the same directory as the file containing the process dump.
 2. Enter the command `gdb -c <filename>` on the console to examine a process dump.
- ⇒ Enter `help` to see more information about the name of the GDB command or general information about GDB.

14.3.2 Creating a process dump manually

Use the `gcore` program to examine failure-prone programs, or if your Industrial PC is in an infinite loop or something situations. The process dump is particularly useful for taking a snapshot of a running process and analyzing processes under TwinCAT/BSD.

By default, the process dump is written to the file `core.pid`. The file can then be analyzed with a debugger such as `gdb`.

Proceed as follows:

1. Determine the process ID (`pid`) of the desired process with the command `pgrep -l <processname>`. The command `ps -A` can be used to list all processes.
 2. Enter the command `gcore <pid>` in the console. Sample: `gcore 6674`
 3. The command `gcore 6674` generates a file with the name `core.6674`
- ⇒ The file is created in the current directory. This file can then be read and analyzed with a debugger. The option `-c` can be used to specify your own file name.
Sample: `gcore -c testfile 6674`

14.3.3 Provide system information for analysis

The `bhfinfo` tool can be used to collect various system information and send it to Beckhoff Support as a ZIP file for evaluation, for example. The ZIP file contains, among other things, various version information for both TwinCAT/BSD and TwinCAT. It provides information about installed packages, running processes, network configuration, configuration files and much more.

1. Enter the command `doas bhfinfo /home/Administrator/FileOutput` in the console.
⇒ The `FileOutput` file is saved in the `/home/Administrator/` directory.
2. For example, use [WinSCP](#) [► 69] to copy the ZIP file to your Windows development computer.

- ⇒ The ZIP file contains important system information in a bundle and can be sent to Beckhoff Support for evaluation.

14.4 Using the ADS monitor

The TwinCAT ADS monitor is divided into two applications. The AMS logger records the AMS commands, while the AMS viewer displays the recorded data or can be used to control the AMS logger remotely.

The AMS logger is called up and configured with the program `tcamslog`. Additional parameters can be used to determine the maximum size of the log file or whether a ring buffer should be used, for example.

Table 8: ADS monitor, parameters of the `tcamslog` application.

Parameter	Description
-l	listen Waits for an AMS viewer connection.
-p	port Port for the AMS viewer connection. Standard: 0xbf12/48914
-c	capture Starts logging AMS commands.
-f	file Name of the log file. Standard: <code>ams.cap</code>
-d	dir Directory in which the log file is stored. The default is the current directory.
-s	size Maximum size of the log file. Standard: 15 MB
-r	ringbuffer Enabled by default. The log is distributed over two or more files.

Proceed as follows:

1. Enter the command `tcamslog -c -r -s 20 -f testlog` in the console to record AMS commands.
 2. The parameters used in the sample determine the size `[-s]`, the file name `[-f]` and the use of the ring buffer `[-r]`.
- ⇒ In the next step, load the recorded AMS commands into the AMS viewer to analyze the log file.

14.5 Analyzing network traffic with Wireshark

TwinCAT/BSD has a packet sniffer as standard. The `tcpdump` program monitors the Ethernet interfaces, records the network traffic and saves the data in a file on the industrial PC.

The saved file can then be copied to a development computer, opened with Wireshark and analyzed.

Requirements:

- Wireshark installed on development machine: <https://www.wireshark.org/download.html>
Wireshark User's Guide: https://www.wireshark.org/docs/wsug.html_chunked/

Proceed as follows:

1. Enter the `doas tcpdump -i igb1 -s 0 -w DHCP.pcap` command in the console. In this sample, `igb1` corresponds to Ethernet interface X000.
-i Ethernet interface.
-s Length of the snapshot. The value "0" sets the length to the default value of 262144 bytes.
-w File in which the output should be stored.

2. Confirm the command with the administrator password.

```
tcpdump: listening on igb1, link-type EN10MB (Ethernet), capture size 262144 bytes
```

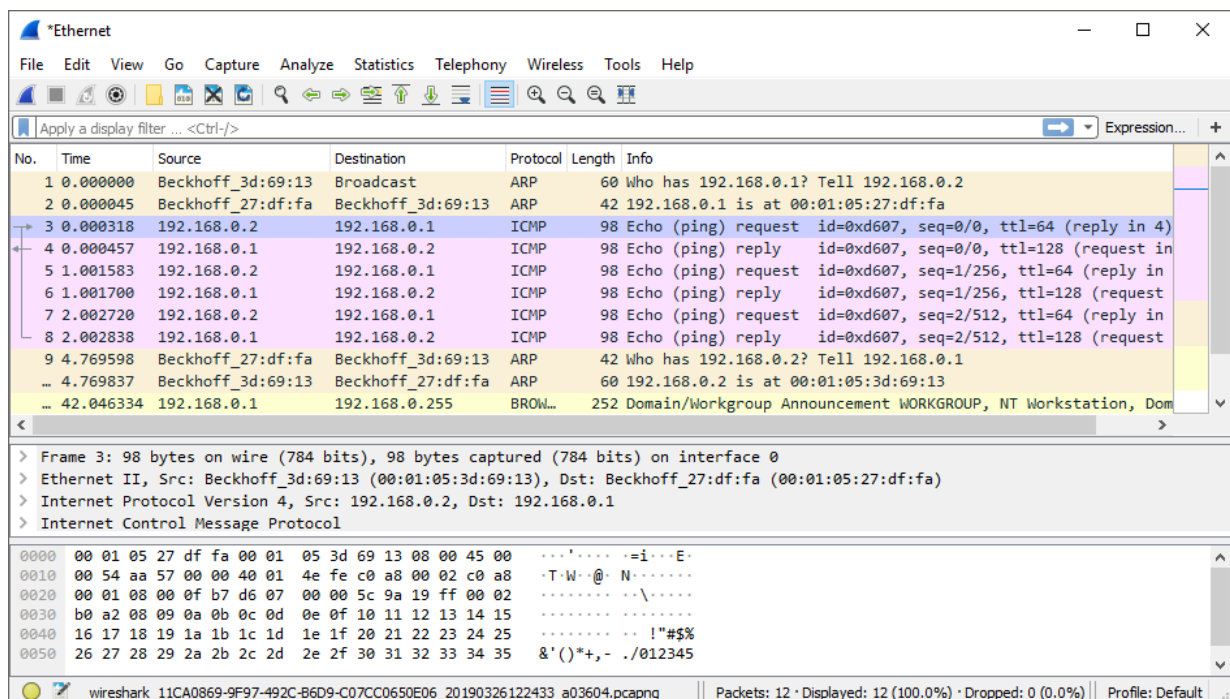
3. You can stop recording at any time by pressing **[Ctrl] + [c]**.

```
33523 packets captured
33531 packets received by filter
0 packets dropped by kernel
Administrator@CX-3B151A:~ %
```

4. In this sample, the **DHCP.pcap** file is saved in the home directory under `/usr/home/Administrator`.

5. Copy the **DHCP.pcap** file to a development computer using the WinSCP client (see: [Managing files with the WinSCP client](#) | ▶ 69|).

⇒ You have successfully recorded the network traffic on the Ethernet interface `igb1`. You can then open and analyze the **DHCP.pcap** file with Wireshark.



14.6 System repair

If TwinCAT/BSD does not boot due to an inconsistent file system or error in a configuration file, there are two ways to repair the TwinCAT/BSD installation or recover data.

For repair or data recovery, use either the

- TwinCAT/BSD installer stick you used for the installation
- or single-user mode, which you can run during system startup.

14.6.1 Booting from the USB installer stick

NOTICE

Security risk

In the default setting, every user who has physical access to the Industrial PC has root rights and thus full control over the system. Restrict access to the Industrial PC to trusted persons.

The TwinCAT/BSD installer stick can be used for repair or data recovery, for example if a faulty process prevents a system start or a faulty TwinCAT project causes a boot loop.

When you boot from the TwinCAT/BSD installer stick, you have access to the fully functional TwinCAT/BSD system installed on the USB stick. In contrast to single-user mode, this allows you to copy important data directly to the USB stick or create a backup after a repair.

Proceed as follows:

1. Boot from the TwinCAT/BSD installer stick and select the **shell** option.



2. Import the storage pool (zpool) of your TwinCAT/BSD system with the `zpool import -fR /tmp/zpool zroot` command.
 3. First, mount the default dataset with the `zfs mount zroot/ROOT/default` command. All other datasets will be mounted automatically.
- ⇒ The file systems of the broken system can now be accessed via `/tmp/zpool`.

14.6.2 Start single-user mode

NOTICE

Security risk

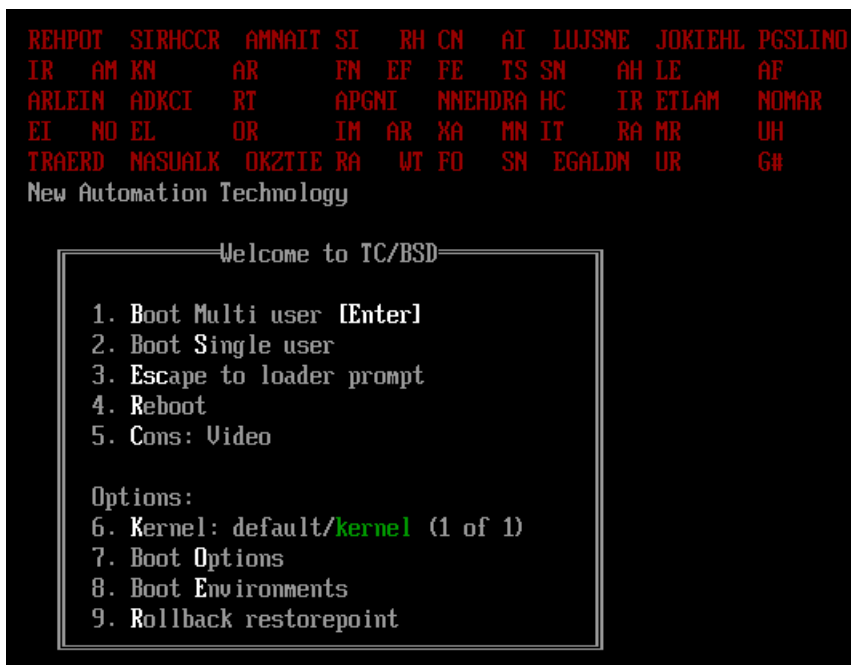
In the default setting, every user who has physical access to the Industrial PC has root rights and thus full control over the system. Restrict access to the Industrial PC to trusted persons.

Single-user mode grants full access to the local system and the configuration files and can be executed during system startup. Single-user mode is similar to safe mode under Windows and can be used for repair or data recovery, for example if a faulty process prevents a system start or a faulty TwinCAT project causes a boot loop.

In single-user mode you have no access to the network and no processes are running. Processes and the network can be started if required.

Proceed as follows:

1. During the bootup, press and hold the **[Space bar]**. The boot menu appears.
2. Select single-user mode with **[2]**.



3. The dataset `zroot/ROOT/default` is mounted automatically. All datasets can be mounted collectively with the command `zfs mount -a`, or the datasets can be mounted individually.
 4. Enable the writing rights for the dataset `zroot/ROOT/default` with the command `zfs set readonly=off zroot/ROOT/default` in order to be able to make changes to the system.
- ⇒ You can exit single-user mode again with `exit`.

15 Appendix

15.1 Important commands

This chapter summarizes and explains important and frequently used commands. The summary is intended as an aid and does not claim to be exhaustive.

15.1.1 TwinCAT

Table 9: Important commands and tools, TwinCAT.

Command	Description
ads	The tool can be used to manage ADS routes. In addition, variables can be read, written or license information, such as the system ID) can be read. Enter the command <code>ads</code> in the console to view all parameters related to the tool. Sample Create ADS route: <code>ads 192.168.0.231 addroute --addr=192.168.0.1 --netid=192.168.0.1.1.1 --password=1</code>
TcSysExe.exe	With TcSysExe.exe it is for example possible to control the TwinCAT mode from the console and to put TwinCAT into Run or Config mode. In addition, information on licenses, different versions and system IDs can be retrieved. The command <code>TcSysExe.exe --help</code> displays all available parameters.
TcRegistry.xml	In the file <code>TcRegistry.xml</code> the Ams Net Id, the HeapMemSize and LockedMemSize can be set.
StaticRoutes.xml	The file <code>StaticRoutes.xml</code> is used to configure ADS routes from TwinCAT/BSD.
TcCoreConf	With the tool it is possible to manage CPU cores and isolate CPU cores, for example. Enter the command <code>TcCoreConf --help</code> in the console to view all parameters.
TcRteConfig	This tool can be used to disable real-time Ethernet when you do not need real-time communication. Enter the command <code>TcRteConfig --help</code> in the console to view all parameters.

15.1.2 Shell

Table 10: Important commands, shell in general.

Command	Description
script	Creates a typescript of the terminal session.
which <command>	Search for the command <command> in the current directory and display where it was found.
history 20	Display the last 20 commands that were entered.
! <num>< td=""><td>Execute the command <num> again from the history.</td></num><>	Execute the command <num> again from the history.
<command1>; <command2>	Execute command 1 followed by command 2.
<command1> && <command2>	Execute <command1>, followed by <command2>, but only if <command1> was successful (\$? = 0).
<command1> <command2>	Redirect output of <command1> to input of <command2>.
<command> >&out.txt	Send both the standard output and the error output of a command to the file out.txt.
printenv	Display all environment variables.
echo \$PATH	Display individual environment variable "PATH".
setenv <variable> "value" [csh]	Sets environment variable <variable>.
unsetenv <variable> [csh]	Removes environment variable <variable>.
^C (Ctrl-C)	Terminate current command.
^U (Ctrl-U)	Delete up to the beginning of the line.
reset	Reset terminal settings.

Command	Description
exit	Exit shell.
logout	

Table 11: Important commands, job control.

Command	Description
<code>^C</code> (Ctrl-C)	Terminate current foreground process.
<code>^Z</code> (Ctrl-Z)	Suspend current foreground process. Creates a suspended job.
<code>jobs</code>	List jobs under this shell.
<code>kill %<num></code>	Terminate the job with the number <num>.
<code>fg</code> <code>fg %<num></code>	Restart suspended process in the foreground.
<code>bg</code> <code>bg %<num></code>	Restart suspended process in the background.
<code><command> &</code>	Start the command as a background job.

15.1.3 File and directory management

Table 12: Important commands, file management.

Command	Description
less <file>	The file content is read out Space bar = next page, b = previous page, q = exit / = forward search, ? = backward search, n = repeat search
grep -i <string> <file>	Shows all lines containing the specified string; -i= is case sensitive.
wc -l <file>	Counts the lines in the file.
tail -f <file>	Particularly useful for log files. The last 10 lines of the file are displayed. The parameter -f also displays newly added lines. Exit with ^C.
tail -n <file>	This can be used to adjust the number of lines that are output. Example: tail -n <file> shows only the last line of the file.
strings <file> less	Extracts strings from a binary file.
touch <file>	Creates a file if not already present, or updates the timestamp.
rm <file>	Delete file.
cp <file> <user>	Copy file.
cp <file1> <file2> ... <path>/	Copy one or more files to another directory. The trailing slash after <path> is not essential, although it prevents errors when copying a file if the path does not exist.
mv <oldname> <user>	Rename file or directory.
mv <file1> <file2> ... <path>/	Move one or more files to a directory.
ln <file> <user>	Create a hard link from <file> to <user> (both names point to the same inode of the file system). Both names must be on the same file system.
ln -s <path> <user>	Make <user> a symbolic or soft link that points to the path that can be a file or directory and can be anywhere in the file system.

Table 13: Important commands, file permissions.

Command	Description
ls -l <file>	Displays permissions for files or directories. -rwxrwxrwx For a file: r allows reading; w allows writing/attaching; x allows executing. For a directory: r allows listing of content; w allows creating or deleting files within the directory; x allows entering the directory.
ls -ld <path>	Directories are displayed like files. Without -d, the directory content is listed recursively when directories are entered.
chown <user> <path> chgrp <group> <path> chown <user>:<group> <path>	Changing the owner, group or both of a file or directory.
chmod [ugoa]+[rwx] <path> chmod [ugoa]-[rwx] <path>	Adding or removing permissions. u = user (owner), g = group, o = others, a = all (ugo) e.g. "chmod go+r file" adds the permission 'r' to 'group' and 'others'.
chmod <nnn> <path>	Change all bits simultaneously to the octal value nnn. e.g. "chmod 640 file" sets rw- for user, r-- for group, --- for others. 0 --- 1 --x 2 -w- 3 -wx 4 r-- 5 r-x 6 rw- 7 rwx
umask umask <nnn>	Show or set the file creation mask for this session; these are the permission bits that are not set for newly created files. For example, "umask 022" means that newly created files have no more than rwxr-xr-x permissions.

Table 14: Important commands, file search.

Command	Description
<code>find <path> -type f</code>	Finds all files under the specified path. Use "." for the current directory. Use the option <code>-type f</code> to only display files.
<code>find <path> -type f -name 'placeholder*'</code>	Finds all files under the specified path whose name begins with "placeholder".
<code>find <path> -type f xargs <command></code>	Find all files under the path and apply <code><command></code> to each of them.
<code>find <path> -type f -print0 xargs -0 <command></code>	Secure version of the above command, and works with file names containing spaces.

Table 15: Important commands, compressed files and archives.

Command	Description
<code>gzip -dc <file>.gz less</code> <code>bzip2 -dc <file>.bz2 less</code>	Reads a compressed text file without unpacking it on the hard disk.
<code>tar -tzf <file>.tgz or .tar.gz</code> <code>tar -tjf <file>.tbz2 or .tar.bz2</code>	Shows the contents of the compressed tar archive. Add option <code>-v</code> for more details.
<code>tar -xvzf -C <path> <file>.tgz</code> <code>tar -xvjf -C <path> <file>.tbz2</code>	Extract the contents of the compressed archive to the specified directory, otherwise to the current directory.

Table 16: Important commands, directories.

Command	Description
<code>pwd</code>	Display current directory.
<code>cd <path></code>	Change to a subdirectory of the current directory.
<code>cd ..</code>	Move up one level to the parent directory.
<code>cd /</code> <code>cd /<absolute path></code> <code>cd ~<user></code> <code>cd</code>	Change current directory: to the root directory, to an absolute path, to the home directory of a particular user, or to your own home directory.
<code>ls</code> <code>ls <path></code>	Lists the contents of the current directory or the specified directory.
<code>ls -l</code>	Lists the directory in long form.
<code>ls -a</code>	Lists all files, including hidden files.
<code>ls -d</code>	Lists the directory itself, instead of its contents.
<code>ls -ld <path></code>	Sample for the combination of flags.
<code>mkdir <path></code>	Create a directory.
<code>rmdir <path></code>	Delete an empty directory.
<code>rm -rf <path></code>	Recursively delete a directory and its entire contents.

15.1.4 System administration

Table 17: Important commands, user accounts.

Command	Description
id	Show current uid, gid and additional groups.
whoami	Display current user name only.
cat /etc/passwd	Show all user accounts.
cat /etc/group	Show all groups.
pw useradd <user> -m	Create user; -m= create home directory.
passwd passwd <user>	Set or change password for yourself or another account (administrator only).
pw usermod <user> -G wheel	Add users to the group "wheel" or simply edit /etc/group directly.
pw userdel <user> -r	Delete user; -r= remove home directory and all content.
cat /etc/master.passwd	View all accounts, including encrypted passwords.
vipw	Lock master.passwd Edit it and rebuild password databases.

Table 18: Important commands, file system.

Command	Description
mount	Display mounted file systems.
df df -h	Shows occupied and free space in all mounted file systems. Adding -h = shows 1G instead of 1048576.
du -c <path>	Adds space occupied by files or directories in the specified path or in the current directory.
mount -r -t cd9660 /dev/acd0 /cdrom	Mount device /dev/acd0 [IDE CD] in directory /cdrom; file system type is cd9660; -r=read-only.
umount /cdrom	Eject the device. The device must not be used.
fstat	List processes with open files.
cat /etc/fstab	Display file system table.
mount /cdrom	Mount /cdrom with parameters from /etc/fstab
mount -a	Mount all file systems in /etc/fstab except those marked "noauto" (this happens during normal booting, but is useful when booting in single-user mode).

Table 19: Important commands, packages.

Command	Description
pkg info	Display an overview list of the installed packages.
pkg info <package>	Display a detailed description of the package.
pkg info -l <package>*	Display a list of all files contained in the package.
pkg add <file>-1.2.3.tbz	Install package from file.
pkg add -r <package>	Install package from the default FTP server.
PACKAGEROOT="ftp://ftp.uk.freebsd.org" pkg add -r <package>	Install package from an alternative FTP server.
pkg install <package>	Installs package from remote repository or local archive.
pkg delete <package>	Uninstalling a package.

Table 20: Important commands, kernel modules.

Command	Description
kldstat	Display loaded modules.
kldload <module>	Load the named module and all modules on which it depends.
kldunload <module>	Unload module.

Table 21: Important commands, network.

Command	Description
<code>ifconfig</code>	Display all interfaces.
<code>ifconfig igb0 192.168.0.1/24</code>	Configure interface.
<code>netstat -r -n</code>	Display table with redirects.
<code>route add default 192.168.0.254</code>	Add a static default route.
<code>ping <IP-Adress></code>	Send test packages. Terminate with ^C.
<code>tracert -n <IP-Adress></code>	Send test packets and display intermediate routers.
<code>tcpdump -i igb0 -n -s1500 -X</code> <code>tcpdump -i igb0 -n tcp port 80 -w <file></code>	Displays complete packets that were sent and received via a specific interface. The second form shows only package headers to/from TCP port 80. Use the option <code>-w <file></code> to save the network dump in file <code><file></code> .
<code>/etc/rc.d/netif start</code>	Initialize network interfaces using the settings in <code>/etc/rc.conf</code> .
<code>/etc/rc.d/routing start</code>	Initialize static routes from the settings in <code>/etc/rc.conf</code> .
<code>/etc/rc.d/dhclient start</code>	Configure interfaces marked with "DHCP" in <code>/etc/rc.conf</code> .
<code>netstat -finet -n</code>	Display active network connections. Use <code>-a</code> to add listening sockets.
<code>sockstat -4 -l</code>	Displays processes that listen to IPv4 and IPv6 sockets.

Table 22: Important commands, processes.

Command	Description
<code>ps aux</code>	Show all processes.
<code>ps aux grep <processname></code>	Show all processes that correspond to the pattern <code><processname></code> . Note that <code>grep <processname></code> can be displayed itself.
<code>top</code>	Continuous display of the most active processes. Quit with q.
<code>kill <pid></code>	The process with the specified process ID is quickly cleaned up and terminated.

Table 23: Important commands, system status.

Command	Description
<code>Alt-F1 ... Alt-F8</code>	Switch between virtual consoles.
<code>date</code>	Display current date and time.
<code>ntpdate -b <server1> <server1> ...</code>	Synchronize the clock with the specified NTP servers.
<code>uptime</code>	Display time since last restart and average load.
<code>w</code>	Shows who is currently logged in.
<code>last -10</code>	Display the last 10 logins.
<code>Shutdown -r now</code>	Restart.
<code>doas shutdown -p now</code>	Switch-off.

15.1.5 Important files and directories.

Table 24: Important files and directories.

Path	Description
/boot/kernel/kernel	The kernel itself.
/boot/kernel/	
/boot/loader.conf	Kernel modules at startup. See /boot/defaults/loader.conf <pre>hint.acpi.0.disabled=1 # disable ACPI if_wi_load="YES" # load the 'wi' network driver snd_driver_load="YES" # load all sound drivers</pre>
/dev/null	The "bit bucket". To discard all output of a command (stdout and stderr): <pre># somecommand >/dev/null 2>&1 [sh]</pre>
/etc/crontab	Regular scheduled tasks.
/etc/group	Binds additional groups to users (only becomes effective after the next login).
/etc/hosts	Local assignments between IP addresses and host names.
/etc/inetd.conf	Controls services that were started by inet but do not have their own daemon processes, e.g. ftpd.
/etc/localtime	Binary file, not editable. Describes the current time zone. <pre># cp /usr/share/zoneinfo/Africa/Maputo /etc/localtime</pre>
/etc/mail/ mailer.conf	Configures which MTA is used when local processes generate emails.
/etc/make.conf	Default values for creating software applications/ports. Only available if created by user. If not present, the default values of the ports are used.
/etc/motd	"Message of the day" is displayed during login.
/etc/newsyslog.conf	Configures the automatic rotation of log files.
/etc/periodic/...	Various scripts that are executed at scheduled times.
/etc/rc.conf	Master configuration file. See /etc/defaults/rc.conf for permitted settings. <pre># network settings hostname="foo.example.com" ifconfig_igb0="192.168.0.1/24" # oder "DHCP" defaultrouter="192.168.0.254". # set clock on startup ntpdate_enable="YES". ntpdate_flags="-b ntp-1.example.net ntp-2.example.net". # activate services inetd_enable="YES" sshd_enable="YES".</pre>
/etc/rc.d/...	Startup scripts. Run as /etc/rc.d/<script> start or /etc/rc.d/<script> stop Only works if the corresponding service exists. <pre>service_enable="YES" in /etc/rc.conf</pre>
/etc/rc.local	Create this script to run additional commands at system startup.
/etc/resolv.conf	Configuring the DNS client <pre>example.com nameserver 192.0.2.1 nameserver 192.0.2.2</pre>
/etc/ssh/ sshd_config	Configure ssh daemon to allow or deny root logins, for example.
/etc/sysctl.conf	Sets runtime kernel variables at startup: <pre>net.inet.ip.forwarding=1 # if this system is a router.</pre>
/etc/syslog.conf	Configure the destinations of log messages. After the change: <pre># killall -1 syslogd</pre>

Path	Description
/etc/ttys	Configure logins on serial lines or modems.
/rescue/...	Statically linked binary files for use in emergencies.
/root	Home directory for 'root' users (still available when other file systems are not mounted).
/usr/local/etc/...	Configuration files for third-party programs (ports/packages).
/usr/share/skel/...	Placeholders that fill the home directory of a new user.
/var/db/pkg/...	Path under which the installed packages <code>pkg</code> are stored (do not change this!).
/var/log/maillog	Mail log file.
/var/log/messages	General system log file.
/var/mail/<user>	Default location for the user mailbox.
/var/run/<inetd>.pid	File with process ID of the running 'inetd' daemon.
/var/spool/mqueue/...	Sendmail queue.
/var/tmp	Temporary files; applications should write large files here and not in /tmp, as is usually the case on a larger file system.
~/.ssh/authorized_keys	Public keys corresponding to the private keys that can log on to this account using SSH RSA/DAS authentication.

15.1.6 Text Editors

Table 25: Important commands, vi editor.

Command	Description
:q! [Enter]	Exit without saving.
:wq [Enter]	Write and exit.
:wq! [Enter]	Write and exit, forces overwriting of the write-protected file.
:w filename [Enter]	Write to another file.
^L (Ctrl-L)	Redraw the screen.
^	Move to the beginning of the line.
\$	Move to the end of the line.
h j k l	Move the cursor left / down / up / right / up / down (alternative to the arrow keys).
:num [Enter]	Go to line number.
G	Go to the last line.
/pattern [Enter]	Search forward for pattern.
?pattern [Enter]	Search backwards for pattern.
n	Repeat last search.
i text ESC	Insert text before the cursor position.
A text ESC	Append text after end of line.
o text ESC	Open a new line after the current one and insert text.
x	Delete the characters under the cursor.
r char	Replaces the character under the cursor with another single character.
dd	Delete entire row.
yy	Copy ("drag") the current line.
num yy	Copying number lines, starting with the current line.
p	Paste copy buffer after the current line.

Table 26: Important commands, Easy Editor (ee).

Command	Description
ESC	Pop-up menu.
^C	Prompt.
^C quit [Enter]	Exit without saving.
^C exit [Enter]	Write and exit.
^C write [Enter]	Write to another file.
^A	Move to the beginning of the line.
^E	Move to the end of the line.
^C num [Enter]	Go to line number.
^Y string [Enter]	Search forward for string.
^X	Repeat last search.
^K	Delete entire row.

15.1.7 Documentation

Table 27: Important commands, documentation.

Command	Description
man <command> man 5 <command> man -a <command>	Displays the manual page for the command <command>. If a page with the same name exists in more than one section, you can specify the section number or -a to display pages from all sections.
man -k <string>	Search for string <string> in the manual.
man hier	Description of the directory structure
cd /usr/share/doc; ls cd /usr/share/examples; ls	Browse system documentation and samples. Note in particular /usr/share/doc/de/books/handbook/index.html
cd /usr/local/share/doc; ls cd /usr/local/share/ examples	Browse the package documentation and examples.

15.2 Bibliography

Lucas, Michael W., Absolute FreeBSD: The Complete Guide to FreeBSD, San Francisco 2019.

Lucas, Michael W., Jude, Allan, FreeBSD Mastery: ZFS, Tilted Windmill Press 2015.

The FreeBSD Documentation Project, FreeBSD Handbook, at https://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ (accessed 12 July 2019).

Schneider, Wolfram, FreeBSD Manual Pages, at <https://www.freebsd.org/cgi/man.cgi> (accessed 12 July 2019).

15.3 FreeBSD Copyright

Copyright 1992-2023 The FreeBSD Project.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The views and conclusions contained in the software and documentation are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the FreeBSD Project.

15.4 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Download finder

Our [download finder](#) contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for [local support and service](#) on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on our internet page: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline: +49 5246 963-157
e-mail: support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline: +49 5246 963-460
e-mail: service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone: +49 5246 963-0
e-mail: info@beckhoff.com
web: www.beckhoff.com

List of tables

Table 1	Change notes for the documentation.	8
Table 2	Change notes for the operating system.	8
Table 3	Description of the directories in the installer folder.	16
Table 4	Overview of important TwinCAT/BSD directories.	25
Table 5	Firewall rule for unencrypted ADS communication.	33
Table 6	UPS software: Settings in the configuration file.	61
Table 7	Device support for TwinCAT/BSD Hypervisor, device and GPU passthrough.....	73
Table 8	ADS monitor, parameters of the tcamslog application.	123
Table 9	Important commands and tools, TwinCAT.	127
Table 10	Important commands, shell in general.	127
Table 11	Important commands, job control.	128
Table 12	Important commands, file management.	129
Table 13	Important commands, file permissions.	129
Table 14	Important commands, file search.	130
Table 15	Important commands, compressed files and archives.	130
Table 16	Important commands, directories.	130
Table 17	Important commands, user accounts.	131
Table 18	Important commands, file system.	131
Table 19	Important commands, packages.	131
Table 20	Important commands, kernel modules.	131
Table 21	Important commands, network.	132
Table 22	Important commands, processes.	132
Table 23	Important commands, system status.	132
Table 24	Important files and directories.	133
Table 25	Important commands, vi editor.	134
Table 26	Important commands, Easy Editor (ee).	135
Table 27	Important commands, documentation.	135

List of figures

Fig. 1	TwinCAT/BSD bootloader during startup.	9
Fig. 2	Add-Route dialog with a TwinCAT/BSD target system and associated IP address.	10
Fig. 3	Remote access via SSH using Windows PowerShell.	11
Fig. 4	Access via the web-based console of a TwinCAT/BSD system.	12
Fig. 5	Access to the device manager of a TwinCAT/BSD system.	12
Fig. 6	Change password in the web interface of the Beckhoff Device Manager.	13
Fig. 7	Beckhoff TwinCAT/BSD installer: Start page with interactive menu.	14
Fig. 8	FAT partition and folder structure of the TwinCAT/BSD installer stick under Windows.	15
Fig. 9	Folder structure of the TwinCAT/BSD installer stick with two backup folders.	15
Fig. 10	Structure of the TwinCAT 3 Runtime under TwinCAT/BSD.	21
Fig. 11	Overview of the structure of the memory pool, including storage media and datasets.	23
Fig. 12	Datasets of the TwinCAT/BSD operating system.	24
Fig. 13	TwinCAT/BSD directory structure.	25
Fig. 14	Example of the use of the SMS function block in TwinCAT 3.	38
Fig. 15	Breakdown of the TwinCAT/BSD version.	41
Fig. 16	NTP server settings in the Beckhoff Device Manager.	64
Fig. 17	Start page of the Beckhoff Device Manager.	65
Fig. 18	First page of the Beckhoff Device Manager.	66
Fig. 19	Remote access via SSH using Windows PowerShell.	67
Fig. 20	Basic configuration of a VM instance.	74
Fig. 21	VM instance with virtual drives.	81
Fig. 22	Configuration of a VM instance with virtual network controllers.	84
Fig. 23	VM instance with a Host-Only network configuration.	85
Fig. 24	VM instance with a Bridge network configuration.	86
Fig. 25	VM instance with a NAT network configuration.	88
Fig. 26	VM instance with an Ethernet device passthrough configuration.	90
Fig. 27	Configuration of a VM instance with PCI device passthrough.	91
Fig. 28	Debian Linux sample VM.	94
Fig. 29	C9900-S620 ordering option with preconfigured Windows 10 VM with Device Passthrough.	99
Fig. 30	C9900-S621 Ordering option with preconfigured Windows 10 VM.	100
Fig. 31	Error message if the heap memory is too small.	112

Trademark statements

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.

Third-party trademark statements

AMD is a trademark of Advanced Micro Devices, Inc.

Debian is a registered trademark owned by Software in the Public Interest, Inc.

FreeBSD is a registered trademark of The FreeBSD Foundation and is used by Beckhoff with the permission of The FreeBSD Foundation.

Intel, the Intel logo, Intel Core, Xeon, Intel Atom, Celeron and Pentium are trademarks of Intel Corporation or its subsidiaries.

itec is a registered trademark of TE Connectivity Industrial GmbH.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

VNC® is a trademark of RealVNC Limited and is protected by trademark registrations and/or pending trademark applications in the European Union, United States of America and other jurisdictions.

Wireshark is a registered trademark of Sysdig, Inc.

More Information:
www.beckhoff.com/twincat-bsd

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com

