



ADMINISTRATOR GUIDE

Kiwi CatTools

Version 3.12.4

© 2024 SolarWinds Worldwide, LLC. All rights reserved.

This document may not be reproduced by any means nor modified, decompiled, disassembled, published or distributed, in whole or in part, or translated to any electronic medium or other means without the prior written consent of SolarWinds. All right, title, and interest in and to the software, services, and documentation are and shall remain the exclusive property of SolarWinds, its affiliates, and/or its respective licensors.

SOLARWINDS DISCLAIMS ALL WARRANTIES, CONDITIONS, OR OTHER TERMS, EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, ON THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION NONINFRINGEMENT, ACCURACY, COMPLETENESS, OR USEFULNESS OF ANY INFORMATION CONTAINED HEREIN. IN NO EVENT SHALL SOLARWINDS, ITS SUPPLIERS, NOR ITS LICENSORS BE LIABLE FOR ANY DAMAGES, WHETHER ARISING IN TORT, CONTRACT OR ANY OTHER LEGAL THEORY, EVEN IF SOLARWINDS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The SolarWinds, SolarWinds & Design, Orion, and THWACK trademarks are the exclusive property of SolarWinds Worldwide, LLC or its affiliates, are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other SolarWinds trademarks, service marks, and logos may be common law marks or are registered or pending registration. All other trademarks mentioned herein are used for identification purposes only and are trademarks of (and may be registered trademarks) of their respective companies.

Table of Contents

Introduction to Kiwi CatTools	6
Get started with Kiwi CatTools	6
Features in the free edition versus the enterprise edition of CatTools	7
Installing CatTools as an application or a service	8
What is the Kiwi CatTools Manager?	12
Secure configuration for Kiwi CatTools	13
Devices supported on Kiwi CatTools	14
Kiwi CatTools Device Matrix	14
How to create a custom device	31
Add devices to Kiwi CatTools	33
Custom devices in Kiwi CatTools	40
Define Kiwi CatTools custom device type using .ini file	42
Define Kiwi CatTools custom device script using .txt file	54
Custom device script cl. variables and functions for Kiwi CatTools	58
Testing your custom device	74
Device specific information	74
Unsupported devices or device activities in Kiwi CatTools	102
Activities	104
Add or edit activity details in Kiwi CatTools	104
Activities types	110
Kiwi CatTools internal functions	164
Creating a custom activity	173
Unsupported activities for a device	208
Menus	209
Kiwi CatTools File menu	209
Kiwi CatTools View menu	224
Kiwi CatTools Options menu	225
Kiwi CatTools Interface menu	241
Kiwi CatTools Help menu	243

Kiwi CatTools pane interface	245
Docking panes	245
Moving and grouping panes	246
Rearranging and saving your layout	247
Reset the pane to default	248
Right-Click short menu	248
Chapter Topics	249
Devices pane	249
Kiwi CatTools Panes Overview	260
Kiwi CatTools Activities Pane	264
Kiwi CatTools Activity log pane	265
Kiwi CatTools Compare pane	265
Kiwi CatTools Info log pane	265
Kiwi CatTools Report pane	266
Kiwi CatTools TFTP pane	267
Kiwi CatTools Display pane	267
Kiwi CatTools Mail pane	268
Event Email Notification	269
Queuing messages	269
Email server authentication	269
Custom scripting	270
Script variables in Kiwi CatTools	270
Script functions in Kiwi CatTools	277
Script editors	281
Kiwi CatTools API	283
API development environments	283
Kiwi CatTools API classes	284
Kiwi CatTools API Limitations	295
Automating the installation of Kiwi CatTools	296
Install as a standard interactive application	296
Install as a Windows service	296
Troubleshooting	297

Collect and send troubleshooting logs to SolarWinds Support	297
Best practices	298
Debug Log	298
Troubleshoot - Kiwi CatTools file name conventions	300
Troubleshoot: Device specific settings	301
Troubleshoot - Error: 70: Permission denied	301
Troubleshoot - Reporting problems	302
Troubleshoot: Remote Desktop Systems	302
Troubleshoot: Remote Authentication	303
Troubleshoot: Anti-virus Software	304
Troubleshoot: Windows XP Firewall	304
Troubleshoot: The service is running but nothing is scheduled	304

Introduction to Kiwi CatTools

Kiwi CatTools (KCT) is a network automation application that provides you with the tools to manage the configuration of your network devices, including routers, switches, and firewalls. CatTools provides support for several devices, including Cisco, 3Com, Dell, Enterasys, Extreme, Foundry, HP Junpier, and Nortel.

Kiwi CatTools includes many options for customizing your environment. For example, you can create schedules to automatically update or backup your network configuration, and you can automatically receive email notifications about network configuration changes. Some of the other activities you can do with Kiwi CatTools are:

- Manage configurations for network devices - including routers, switches, and firewalls.
- Compare a network device's current configuration to the device's start up configuration
- Generate network device configuration reports, such as port, MAC, ARP, and version details.
- Issue commands via Telnet or SSH to several devices at the same time.
- Change the configuration of devices at pre-scheduled times.
- Change all of your monitored network device passwords in one go.
- Manage both IPv4 and IPv6 devices from vendors such as Cisco, HP, Huawei, and 3Com.

This configuration management tool also facilitates writing custom device scripts using VBScript, includes a built-in TFTP server, and supports SSH, Telnet, and other network protocols.

What's new in Kiwi CatTools?

To view the latest features and improvements for Kiwi CatTools, review the current [release notes](#).

Get started with Kiwi CatTools

For information about downloading and installing Kiwi CatTools, including the system requirements and port requirements, see the [Kiwi CatTools Installation Guide](#).

i After you initially install Kiwi CatTools, all features are available during a 14-day trial period. When the trial period ends, you must enter a license key to continue to access the licensed features.

To upgrade to the latest version, see the [Kiwi CatTools Upgrade Guide](#). When upgrading to the latest version of KCT, your devices, scheduled actions, and other configurations are preserved.

If you are new to Kiwi CatTools, see the [Kiwi CatTools Getting Started Guide](#). This guide walks you through examples of common configuration tasks to begin using Kiwi CatTools.

Learn more

See the following sections in the guide to learn more about:

- [Configuring the devices to monitor and update with Kiwi CatTools.](#)
- [Creating scheduled activities to perform internal functions against monitored devices.](#)
- [Exploring the KCT menu interface.](#)
- [Exploring the KCT panes.](#)
- [Available options for notifying you of events that occur on a device.](#)
- [Creating custom scripts to add devices or activities to the user interface.](#)
- [Understanding the KCT API environments, classes, and limitations.](#)
- [Troubleshooting tips for resolving common issues.](#)

Features in the free edition versus the enterprise edition of CatTools

When you initially [install Kiwi CatTools](#), all features are available during a 14-day trial period. After the trial period ends, you can continue to use the limited, free edition without purchasing a license. Or you can [enter a license key](#) to expand features available in the licensed edition.

Comparison of free and licensed features

	Free Edition	Licensed Edition
Maximum devices	1 device in database	Unlimited devices in database - subject to file size limitation (2 GB)
Scheduled activities	Up to 20 activities	Unlimited activities - subject to file size limitations
TFTP sessions	Up to 2 simultaneous sessions	Up to 100 simultaneous sessions
Client threads	1 client thread	Up to 30 client threads

Purchase licenses

Contact the SolarWinds sales team to purchase licenses directly from SolarWinds.

- [Online quote tool](#)
- sales@solarwinds.com
- 866.530.8100

View purchased licenses

After you purchase individual licenses, you can view your SolarWinds KCT licenses in the SolarWinds Customer Portal.

1. Access the [Customer Portal](#).
2. Click Licenses > Manage Licenses.

Next steps

After you have purchased licenses, you must activate them on a KCT server, and configure the devices to be monitored and configured by KCT.

Installing CatTools as an application or a service

During the installation of Kiwi CatTools, you have the option of installing the program as either an application or as a service.

- **Application** installs CatTools as a typical Windows software application, requiring you to log in to Windows before running the program.
- **Service** installs CatTools as a Windows software service, allowing the program to run without the need for you to log in to Windows.
 - This option also installs the CatTools Manager which is used to control the service.

Install CatTools as an application

The application version installs CatTools as an application, requiring a user to log in to Windows before opening the program.

Installing CatTools as an application is recommended if you need to run one-off or attended Activities. This option is also recommended for infrequent or part-time users who are running CatTools on their own personal computers.

Install CatTools as a service

The Service version installs CatTools as a Windows service, allowing the program to run without the need to log in to Windows. This option also installs the [CatTools Manager](#), which is used to manage your device activities and activity schedules.

You can interact with the CatTools service using the [CatTools Manager](#), which can be thought of as the GUI for the service.

When using the service version of CatTools, it is important that you run the program as an account with administrator privileges to perform the tasks necessary for CatTools to operate. In most cases, the Local System account is sufficient; however if you have difficulties getting CatTools to run as a service, you may need to use an account with higher administrator privileges.

SolarWinds recommends the service installation to Enterprise customers, as well as anyone running unattended activities, who need to make use of around-the-clock scheduling, or who are dedicating a machine specifically for CatTools to run on.

What is a service?

A service is a program that runs in the background and interacts with a machine according to different rules from a standard application. A standard application is able to run according to the limitations of the user who is logged in at the time. A service by default runs under a local system account, which normally has many restrictions in terms of file permissions.

A service runs when a system is powered on and does not require a user to be logged in to the machine to execute.

Access UNC and mapped drives

By default, the service version is logged in using a local system account. This account is different from a user account you use to access the machine running CatTools. The local system account does not identify any mapped drives that are defined and are accessible by your user account.

SolarWinds recommends that where possible, any activities run by CatTools that need to access a network location to read data do so using the full UNC path name. Otherwise, the activities report may generate incorrect results. For example, if a backup activity always returns a status of `Configuration is New` each time it runs it is likely that the CatTools service cannot access the network location defined in the activity setup for the current configuration file.

Install CatTools on a virtual machine

Kiwi CatTools can run on a virtual machine, but may require you to adjust the setup of the VM.

You can configure your virtual machines to use multiple virtual CPUs, based on the number of physical CPUs you have on your server. However, CatTools spawns multiple clients and this can slow down the operation of CatTools while the virtual machine waits for your physical processors to become free simultaneously.

For best results, you should ensure there is only one virtual CPU configured for each virtual machine.

Automatic service start up and service dependency

When CatTools is installed on a server as a service, the program is set to start automatically when your system is restarted or booted. This occurs even if a user is not logged in to the console.

Service dependencies

Under the majority of operating systems, the CatTools Service starts without issue. On some servers, the CatTools Service must wait for another system service to load before CatTools can start. In this case, the CatTools service fails to load if it is dependent on another service starting first.

If CatTools is not loading, a service dependency may be the issue. One of the following events may occur after rebooting:

- An error message is displayed: `One or more system services failed to start.`
- The CatTools Service fails to start even though the startup type for the service is set to `Automatic`.

To test your system for service dependency issues, check the status of the CatTools Service using the services manager:

1. Go to Start > Run.
2. In the dialog box, type `services.msc`.
3. Click OK.
4. Locate the CatTools service and verify the status is set to `Running`.

If your service status has not started, then your issue is potentially caused by a service dependency. For more information on service dependencies, see [Adding Service Dependencies](#).

Adding service dependencies

You can add a service dependency by two methods:

1. **Recommended:** [Modify the Windows registry to create a new value](#), listing the dependencies. This value can then be applied every time the CatTools Service is created on the system, such as if you upgrade or reinstall CatTools to your system. This is a "permanent" fix.
2. [Use `sc.exe` to create a dependency](#) specifically for an existing service. If the service is deleted, for example if CatTools is uninstalled, then the dependency is also lost.

Add a dependency using RegEdit

This method is the preferred solution as it ensures the CatTools service dependencies are applied every time the service is created. Therefore, you do not need to run through the `sc.exe` method if you need to reinstall or upgrade CatTools.

To ensure the required services have started before CatTools, modify the registry settings.

- **Section:** HKEY_LOCAL_MACHINE\SOFTWARE\Kiwi Enterprises\CatTools3
- **Value (STRING):** NTServiceDependencies
- **Default value:** Blank
- **Type:** Text string of service names, delimited by semi-colons.
For example: `ServiceName1;ServiceName2;ServiceName3`


For example, to modify the registry to establish that the LanmanWorkstation, TCP/IP, and WMI stack services are running before the CatTools service starts:

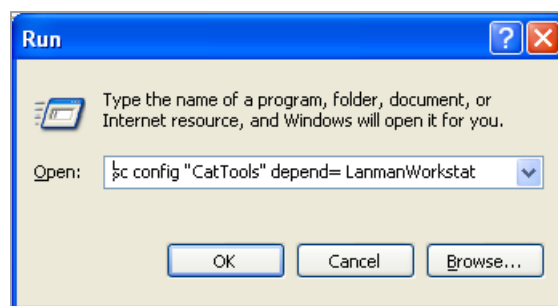
1. Uninstall CatTools.
2. Run RegEdit.
3. Locate the section key HKEY_LOCAL_MACHINE\SOFTWARE\Kiwi Enterprises\CatTools3.
4. Create a new string value of NTServiceDependencies.
5. Modify the value data to include the list of services that need to start first, for example:
`LanmanWorkstation;TCPIP;WMI`
6. Reinstall CatTools.

Add a dependency using sc.exe

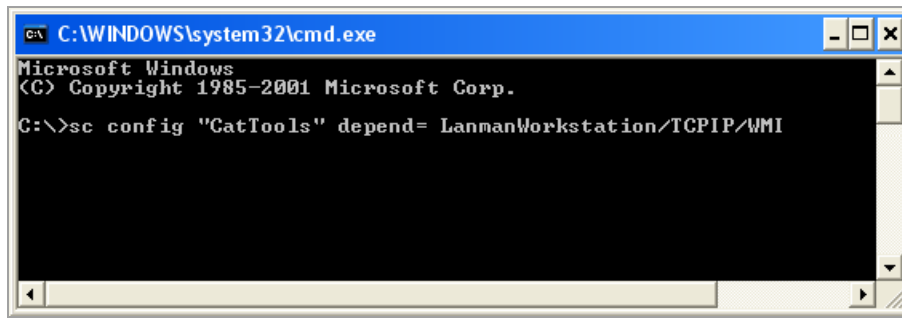
This is the easiest method to add service dependencies to the existing CatTools service. If the CatTools service is uninstalled, then the dependencies are also lost. To add the dependency, you can run a command from Start > Run dialog and enter the command. You can also run a command from the `cmd.exe` window.

For example, to add dependencies to the CatTools service for the LanmanWorkstation, TCP/IP, and Windows Management Interface (WMI) services, execute the command `sc config "CatTools" depend= LanmanWorkstation/TCPIP/WMI` using the Run dialog.

 The full command is not visible in the open text box. You can view the full command by scrolling to the right.



You can also execute the command using `cmd.exe`:



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows
(C) Copyright 1985-2001 Microsoft Corp.
C:\>sc config "CatTools" depend= LanmanWorkstation/TCP/IP/WMI
```

This example ensures that the LanmanWorkstation, WMI, and TCP/IP stack services are running before the CatTools service starts. The service dependencies that need to be added for your system depend on your specific machine and what services are set to run on startup.

- The CatTools service must be stopped before configuring dependencies.
- The CatTools service must be contained within double quotes " ". The service name has a space within it that may cause errors if not contained in double quotes.
- When adding multiple dependencies, separate each individual dependency using a forward slash (/).
- You must include the space ' ' between depend= and the first dependency service name. If you do not include this space, the dependency is not applied, although it appears to work.

Remove a dependency

To remove a dependency, uninstall CatTools, reboot your system, and reinstall CatTools. The modified CatTools service dependency is removed and replaced with the default from the CatTools installer.

! You can also use the command: `sc config "CatTools" depend= none`. However, this can leave the CatTools service in an unusable state, as the dependencies are not properly deleted. If this occurs, the only solution is to uninstall CatTools, reboot the system, and reinstall CatTools again.

What is the Kiwi CatTools Manager?

The Kiwi CatTools Manager is an interface that is used to talk to, or manage, the CatTools service.

The application version of CatTools is split into two parts:

1. The service runs in the background processing the data and automatically runs enabled activities in CatTools.

2. The manager is the user interface that you can interact with to manage your devices and activities.

The CatTools Manager interface takes your request and passes it on to the CatTools service. The service then processes the request and returns a reply which the manager interprets and displays to the user.

Secure configuration for Kiwi CatTools

This document describes best practices for securing Kiwi CatTools (KCT).

Best practices

Create an encryption password

Enable a database encryption password when KCT is run to limit unauthorized access. Once the password is enabled, the system will prompt you to enter your password every time you open KCT. See [Change encryption password](#) for details.

Review your file transfer protocol

SolarWinds does not recommend usage of Trivial File Transfer Protocol (TFTP) in KCT due to potential security vulnerabilities. If TFTP must be used, it is highly recommended to take precautions and limit its exposure to your local network. See [TFTP server security options](#) for details.

Manage sensitive data in exported devices files

Delete or securely store the data contained in the device list `.txt` file after [exporting devices to a tab delimited file](#). Data contained in the exported devices `.txt` file may contain information that is subject to security vulnerabilities.

Devices supported on Kiwi CatTools

Kiwi CatTools supports a wide range of different manufacturer and model devices.

For an up-to-date list of the devices currently supported and the activities supported for each device type, see the [device matrix](#). For more information on adding a device using a pre-defined device type, see the [Devices | Add section](#) in the [Panels](#) chapter.

This chapter covers:

- [Creating a custom device](#) - How to create your own custom device script and add it to the CatTools device type list.
- [Device specific information](#) - Important information regarding specific devices supported on CatTools.
- [Unsupported devices or device activity](#) - Information regarding unsupported devices or device activity.

Kiwi CatTools Device Matrix

Kiwi CatTools supports a wide range of different manufacturer and model devices and works with the most common and up-to-date CLI syntax. Use the device matrix to determine whether or not a device is supported to the level you require. The list provides a current list of device types that function with KCT and also identifies the level of support that each of those scripts contains:

- [Activities](#)
- [Reports](#)
- [Connections](#)

If you are unable to get your device to work properly with CatTools, it is likely due to one of the following issues:






- The device is not supported by CatTools.
- The activity you are trying to run is not configured correctly.
- The device is using a different CLI syntax than what is expected by CatTools.

SolarWinds recommends inquiring via THWACK, the SolarWinds online community for additional information, or contact support.

Configuring device variations

There are a number of [device specific](#) script variables that CatTools uses during the execution of certain device scripts. To avoid errors and to run device specific variables, device variations are provided in CatTools. For information on how to configure these device variations, see [How to use variations in Kiwi CatTools](#).





































Legend

	Supported by CatTools.
	Not currently supported by CatTools. Please inquire on THWACK.
	Not supported by the device.
	Partial support.
	Unknown. No analysis was possible.

Activities

The following Activities require no device login:

- [Connectivity Ping test](#)
- [DB Backup CatTools](#)
- [DB Update Device password field](#)
- [DB Update Device text field](#)
- [System Delete files](#)



























Devices	Update Banner	Inter-device Ping	Connectivity Login Test	Update Password	Backup Running Config	Backup Config TFTP	Send Commands	Modify Config	TFTP Upload
3Com.Switch.5500									
3Com.Switch.SSII									
Accton.Switch									
Addpac.APOS									

Adtran.Netvanta.General									
Alcatel.Switch.Omnistack									
AlliedTelesis.AlliedWare.Plus									
AlliedTelesis.Switch.8000									
AlliedTelesis.Switch.8500									
APC.AOS									
APC.AOS.CLI									
Aruba.ArubaOS.General									
ATI.Router.General									
BelAir.SwitchRouter.Wireless									
Bluecoat.Cacheflow									
Brocade.Switch									
Checkpoint.VPN									
Cisco.ACE									
Cisco.CallManager									
Cisco.Firewall.ASA									
Cisco.Firewall.IDS									
Cisco.Firewall.PIX									
Cisco.MDS.Fibre									
Cisco.NXOS									

Cisco.Older.VPN3002									
Cisco.Other.ACNS									
Cisco.Other.CSS									
Cisco.Other.CUE									
Cisco.Other.LocalDirect or									
Cisco.Other.VPN3000									
Cisco.Router.General									
Cisco.Router.noenable									
Cisco.SCE									
Cisco.SmallBusiness									
Cisco.Switch.1900									
Cisco.Switch.CatOS									
Cisco.Switch.IOS									
Cisco.Terminal.Server									
Cisco.V4.1.VPN3000									
Cisco.VPN									
Cisco.WAE									
Cisco.Wireless.Lan									
Cisco.WLSE									
Citrix.NetScaler.General									
Citrix.NetScaler.General									
Crossbeam.COS									

Crossbeam.UTM									
Cygwin									
Dell.Switch.CLI									
DLink.Switch.General									
DLink.Wireless									
Enterasys.Matrix.Switch									
Enterasys.MatrixN.Switch									
Enterasys.Router.General									
Enterasys.Router.XSR1800									
Enterasys.Securestack									
Enterasys.Wireless.Controller									
Extreme.Switch.General									
F5.BigIP									
F5.BigIP.GTM									
F5.BigIP.TMSH									
FiberLogic.General									
Force10.General									
Fortinet.FortiOS									
Fortinet.FortiOS.General									
Foundry.Switch.General									

Foursticks.NP.Gateway									
GarrettCom.Switch.General									
Generic.Device									
HP.Switch.2500									
HP.Switch.Aruba									
HP.Switch.Others									
Huawei.General									
IBM.AIX.General									
IronPort.Security.General									
Juniper.App.Accelerator-DX									
Juniper.Application.Accelerator									
Juniper.Netscreen.Firewall									
Juniper.Router									
Lancom.ISDN.Router									
Lantronix.EDS									
Linux.RedHat.Bash									
McData.Fibre									
Meru.Controller.MC									
Mikrotik.Router									
Motorola.Router.CMTS									
Motorola.Vanguard									

MRV.Switch.General									
MultiCom.Firewall.General									
NEC.Univerge.IX									
NEC.Univerge.Switch									
NetApp.FAS.General									
Netgear.Switch.General									
Netopia.DSL									
Nokia.ADSL.M1122									
Nortel.Application.Switch									
Nortel.ARN.General									
Nortel.Router.VPN									
Nortel.Secure.Router									
Nortel.Switch.Ethernet									
Nortel.Switch.NoCLI									
Nortel.Switch.Passport									
Nortel.Wireless.Switch									
Occam.General									
Packeteer.Packetshaper									
PaloAlto.FireWall									
Pannaway.BAR									
Pannaway.BAS									

Radware.AppDirector									
Radware.WSD									
Redback.Router.General									
Riverbed.Steelhead									
Riverstone.Router.General									
Sidewinder.Firewall									
SonicWall.SonicOS									
Sun.SunOS.General									
Symbol.WS2000									
Thomson.Speedtouch									
Trapeze.Wireless.Lan									
Xirrus.Wireless									
ZyXEL.Switch									

Reports

The following Reports require no device login:

- [SNMP System Summary](#)
- [Compare Two Files](#)
- [X-Ref.Port MAC ARP](#)

Devices	Version Report	Compare Running Startup	Port Report	MAC Report	ARP Report	CDP Neighbours Report	Error Info Report
3Com.Switch.5500							
3Com.Switch.SSII							

Accton.Switch							
Addpac.APOS							
Adtran.Netvanta.General							
Alcatel.Switch.Omnistack							
AlliedTelesis.AlliedWare.Plus							
AlliedTelesis.Switch.8000							
AlliedTelesis.Switch.8500							
APC.AOS							
APC.AOS.CLI							
Aruba.ArubaOS.General							
ATI.Router.General							
BelAir.SwitchRouter.Wireless							
Bluecoat.Cacheflow							
Brocade.Switch							
Checkpoint.VPN							
Cisco.ACE							
Cisco.CallManager							
Cisco.Firewall.ASA							
Cisco.Firewall.IDS							
Cisco.Firewall.PIX							
Cisco.MDS.Fibre							
Cisco.NXOS							
Cisco.Older.VPN3002							
Cisco.Other.ACNS							
Cisco.Other.CSS							
Cisco.Other.CUE							

Cisco.Other.LocalDirector							
Cisco.Other.VPN3000							
Cisco.Router.General							
Cisco.Router.noenable							
Cisco.SCE							
Cisco.SmallBusiness							
Cisco.Switch.1900							
Cisco.Switch.CatOS							
Cisco.Switch.IOS							
Cisco.Terminal.Server							
Cisco.V4.1.VPN3000							
Cisco.VPN							
Cisco.WAE							
Cisco.Wireless.Lan							
Cisco.WLSE							
Citrix.NetScaler.General							
Citrix.NetScaler.General							
Crossbeam.COS							
Crossbeam.UTM							
Cygwin							
Dell.Switch.CLI							
DLink.Switch.General							
DLink.Wireless							
Enterasys.Matrix.Switch							
Enterasys.MatrixN.Switch							
Enterasys.Router.General							

Enterasys.Router.XSR1800							
Enterasys.Securestack							
Enterasys.Wireless.Controller							
Extreme.Switch.General							
F5.BigIP							
F5.BigIP.GTM							
F5.BigIP.TMSH							
FiberLogic.General							
Force10.General							
Fortinet.FortiOS							
Fortinet.FortiOS.General							
Foundry.Switch.General							
Foursticks.NP.Gateway							
GarrettCom.Switch.General							
Generic.Device							
HP.Switch.2500							
HP.Switch.Aruba							
HP.Switch.Others							
Huawei.General							
IBM.AIX.General							
IronPort.Security.General							
Juniper.App.Accelerator-DX							
Juniper.Application.Accelerator							
Juniper.Netscreen.Firewall							
Juniper.Router							

Lancom.ISDN.Router							
Lantronix.EDS							
Linux.RedHat.Bash							
McData.Fibre							
Meru.Controller.MC							
Mikrotik.Router							
Motorola.Router.CMTS							
Motorola.Vanguard							
MRV.Switch.General							
MultiCom.Firewall.General							
NEC.Univerge.IX							
NEC.Univerge.Switch							
NetApp.FAS.General							
Netgear.Switch.General							
Netopia.DSL							
Nokia.ADSL.M1122							
Nortel.Application.Switch							
Nortel.ARN.General							
Nortel.Router.VPN							
Nortel.Secure.Router							
Nortel.Switch.Ethernet							
Nortel.Switch.NoCLI							
Nortel.Switch.Passport							
Nortel.Wireless.Switch							
Occam.General							
Packeteer.Packetshaper							

PaloAlto.FireWall							
Pannaway.BAR							
Pannaway.BAS							
Radware.AppDirector							
Radware.WSD							
Redback.Router.General							
Riverbed.Steelhead							
Riverstone.Router.General							
Sidewinder.Firewall							
SonicWall.SonicOS							
Sun.SunOS.General							
Symbol.WS2000							
Thomson.Speedtouch							
Trapeze.Wireless.Lan							
Xirrus.Wireless							
ZyXEL.Switch							

Connections

Other connection methods are also supported by certain devices. See [Connecting via a session](#) in the Kiwi CatTools Administrator Guide.

Devices	Login Telnet	Login SSH	Login ConnectVia
3Com.Switch.5500			
3Com.Switch.SSII			
Accton.Switch			
Addpac.APOS			
Adtran.Netvanta.General			

Alcatel.Switch.Omnistack			
AlliedTelesis.AlliedWare.Plus			
AlliedTelesis.Switch.8000			
AlliedTelesis.Switch.8500			
APC.AOS			
APC.AOS.CLI			
Aruba.ArubaOS.General			
ATI.Router.General			
BelAir.SwitchRouter.Wireless			
Bluecoat.Cacheflow			
Brocade.Switch			
Checkpoint.VPN			
Cisco.ACE			
Cisco.CallManager			
Cisco.Firewall.ASA			
Cisco.Firewall.IDS			
Cisco.Firewall.PIX			
Cisco.MDS.Fibre			
Cisco.NXOS			
Cisco.Older.VPN3002			
Cisco.Other.ACNS			
Cisco.Other.CSS			
Cisco.Other.CUE			
Cisco.Other.LocalDirector			
Cisco.Other.VPN3000			
Cisco.Router.General			

Cisco.Router.noenable			
Cisco.SCE			
Cisco.SmallBusiness			
Cisco.Switch.1900			
Cisco.Switch.CatOS			
Cisco.Switch.IOS			
Cisco.Terminal.Server			
Cisco.V4.1.VPN3000			
Cisco.VPN			
Cisco.WAE			
Cisco.Wireless.Lan			
Cisco.WLSE			
Citrix.NetScaler.General			
Citrix.NetScaler.General			
Crossbeam.COS			
Crossbeam.UTM			
Cygwin			
Dell.Switch.CLI			
DLink.Switch.General			
DLink.Wireless			
Enterasys.Matrix.Switch			
Enterasys.MatrixN.Switch			
Enterasys.Router.General			
Enterasys.Router.XSR1800			
Enterasys.Securestack			
Enterasys.Wireless.Controller			

Extreme.Switch.General			
F5.BigIP			
F5.BigIP.GTM			
F5.BigIP.TMSH			
FiberLogic.General			
Force10.General			
Fortinet.FortiOS			
Fortinet.FortiOS.General			
Foundry.Switch.General			
Foursticks.NP.Gateway			
GarrettCom.Switch.General			
Generic.Device			
HP.Switch.2500			
HP.Switch.Aruba			
HP.Switch.Others			
Huawei.General			
IBM.AIX.General			
IronPort.Security.General			
Juniper.App.Accelerator-DX			
Juniper.Application.Accelerator			
Juniper.Netscreen.Firewall			
Juniper.Router			
Lancom.ISDN.Router			
Lantronix.EDS			
Linux.RedHat.Bash			
McData.Fibre			

Meru.Controller.MC			
Mikrotik.Router			
Motorola.Router.CMTS			
Motorola.Vanguard			
MRV.Switch.General			
MultiCom.Firewall.General			
NEC.Univerge.IX			
NEC.Univerge.Switch			
NetApp.FAS.General			
Netgear.Switch.General			
Netopia.DSL			
Nokia.ADSL.M1122			
Nortel.Application.Switch			
Nortel.ARN.General			
Nortel.Router.VPN			
Nortel.Secure.Router			
Nortel.Switch.Ethernet			
Nortel.Switch.NoCLI			
Nortel.Switch.Passport			
Nortel.Wireless.Switch			
Occam.General			
Packeteer.Packetshaper			
PaloAlto.FireWall			
Pannaway.BAR			
Pannaway.BAS			
Radware.AppDirector			

Radware.WSD	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Redback.Router.General	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Riverbed.Steelhead	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Riverstone.Router.General	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Sidewinder.Firewall	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
SonicWall.SonicOS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Sun.SunOS.General	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Symbol.WS2000	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Thomson.Speedtouch	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Trapeze.Wireless.Lan	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Xirrus.Wireless	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ZyXEL.Switch	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

More information on adding devices

For more information on adding a device, see:

- [Adding a device](#) - how to add a supported device
- [Creating a custom device](#) - How to create your own custom device script and add it to the CatTools device type list.
- [Device specific information](#) - Important information regarding specific devices supported on CatTools and [configuring device variations](#).
- [Unsupported devices or device activity](#) - Information regarding unsupported devices or device activity.

How to create a custom device

Quick reference guide to creating a custom device in CatTools

This is a simple step-by-step guide on creating a new custom device in CatTools. It contains brief instructions on how to add a custom device type to the CatTools GUI, and how to create a custom device script file.

The custom device templates files provided by CatTools and referenced in this guide, are based on a Cisco Router device. It is likely that further modification will be required to the device script (.txt) file once you have created it, in order to get the script to work successfully with your specific device.

Further information regarding the [custom device type file \(.ini\)](#), [custom device script file \(.txt\)](#), and [code examples](#) of how to use the CatTools internal functions in your device script, can be found in the other sub pages of this chapter.

Steps to create a custom device:

1. Create a custom device type file (.ini)

Take a COPY of the Custom.Device.Template.ini in the \Templates sub folder of the CatTools root directory and save to the \Devices sub folder, giving it a new file name using the syntax: *Custom.Manufacturer.Type* (e.g. Custom.Cisco.FirewallPIX).

2. Edit the .ini file

Using a text file editor (such as Notepad), open the .ini file created in [step 1](#). You must change the following items in the .ini file from the template default values:

[device]

name=**Custom.Manufacturer.Type**

id=**4000**

Change 'name' item to a UNIQUE device name. Example Custom.Cisco.Router Note: do not use spaces, use a 'period' mark (.) instead.

Change 'id' item to a UNIQUE number (i.e. one not used in any other device .ini file). Number must be within the range of 4000 to 4999.

[item_Name]

default=**Unique device name**

Change the 'default' item to a UNIQUE device name.

3. Save & restart

Once you have made your changes to the .ini file, save it back to the \Devices sub folder and close.

Restart CatTools to populate the 'Device type' drop-down field in the device setup screen with your new custom device type.

4. Create a custom device script file (.txt)

Take a COPY of the Custom.Device.Template.txt in the \Templates sub folder of the CatTools root directory and save to the \Scripts sub folder.

Save it with a file name using the value entered for the [device] section 'name' item in [step 2](#) above. Ensure you retain the .txt file type suffix.

Example:

(device .ini file settings)

[device]

name=Custom.Cisco.FirewallPIX

Script file name must therefore be: Custom.Cisco.FirewallPIX.txt

5. Edit the .txt file

Using a text file editor*, open the .txt file created in step 4. Follow the instructions in the SCRIPT NOTES section of the .txt file to begin customizing the script for your device.

Note: although the .txt files can be opened and edited using 'NotePad', a syntax highlighting script editor may make reading and editing the .txt file much easier.

6. Save & test

Once you have made your initial changes to the .txt file, save it back to the \Scripts sub folder and close.

To test your custom device, create an activity that is supported by your custom device script and run it manually using the 'Run Now' button. ([Device.ConnectivityTest.Login](#) is a good starting point activity as every device must be able to log in successfully in order to perform any of the

other more complex activities).

Check the [Info Log pane](#) for errors and edit your .txt file as necessary.

See [Testing your custom device](#) for more information and tips.

Creating custom device checklist:

- Create device type file (.ini)
- Create device script file (.txt)

Add devices to Kiwi CatTools

After installing CatTools and completing the CatTools setup wizard, you have two ways to add a device to CatTools:

- Add a device using the setup wizard
- Add a device from the Device Pane

Add a device using the setup wizard

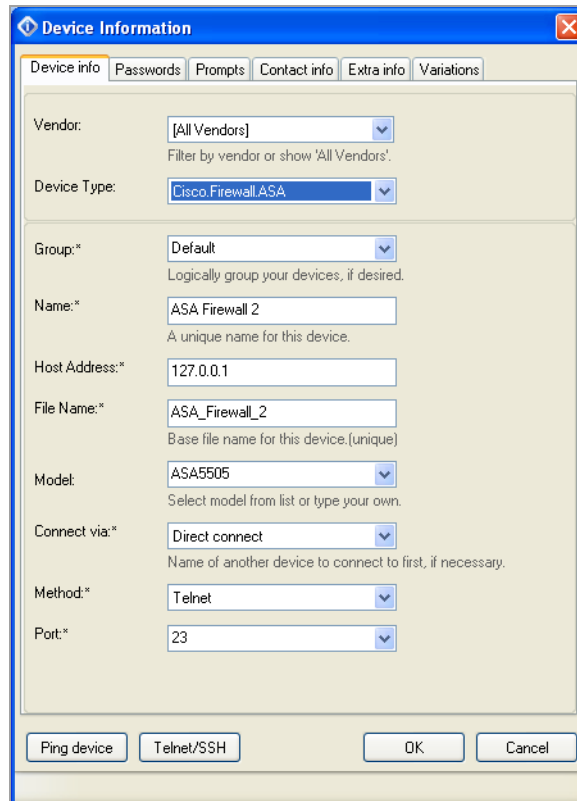
Use the [Device setup wizard](#) on the Options menu to add device to CatTools. The wizard guides you through the addition of new devices to the KCT database.

Add a device from the Device pane

From the device pane in CatTools, ensure you are on the Devices tab and click Add. The Device Information wizard has five, or six depending on selected options, tabs that need to be completed to add your new device. Fields marked with an asterisk (*) next to the field name must be completed to add the device.

Throughout the process, you can ping your device and test defined telnet / SSH session settings:

- Ping device - Sends ICMP Ping packets to the highlighted device.
- Telnet/SSH - Uses the defined Telnet or SSH client program to start a manual session with the highlighted device. The Telnet and SSH client programs can be defined in the [CatTools Setup dialogue](#).



Device Information

Device info | Passwords | Prompts | Contact info | Extra info | Variations

Vendor: [All Vendors] (Filter by vendor or show 'All Vendors')

Device Type: Cisco Firewall.ASA

Group:* Default (Logically group your devices, if desired)

Name:* ASA Firewall 2 (A unique name for this device)

Host Address:* 127.0.0.1

File Name:* ASA_Firewall_2 (Base file name for this device.(unique))

Model: ASA5505 (Select model from list or type your own)

Connect via:* Direct connect (Name of another device to connect to first, if necessary)

Method:* Telnet

Port:* 23

Ping device | Telnet/SSH | OK | Cancel

Device info tab

Configure the identification information for a device, such as the vendor, device type, model, and port. Once you have identified the device, ping the device to confirm a successful connection.

Vendor	Use the drop down list to filter the Device Type list box by vendor or choose all vendors to show all Device Types.
Device Type	Use the drop down list to select the CatTools device type from the provided list.
Group	Enter or select the name of a group you want this device to belong to. This is a free form field where you can simply create a new group by typing new text. Use group names to group your devices into logical or physical categories.
Name	Give the device a unique name.
Host Address	Enter the IP address of the new device (in standard <code>aaa.bbb.ccc.ddd</code> format, or use a hostname).
File Name	This is the base file name CatTools uses for the device's data and reports. This field reflects the device name, but transforms any characters that are not usable in a file name.
Model	Use the drop down list to select the device Model. This is a free form field so you can enter any text you like. This field is only for documentation and has no effect on the operation of CatTools - the Device Type determines how CatTools handles a session with a device.

Connect via Use the drop down list to select a device to connect via. The default is `Direct connection`. You only need to specify another device when direct access to the desired device is not possible. CatTools allows you to hop from one device to another using Telnet or SSH to reach your final destination. For example, if your device is behind an access list, but a Linux box has access to that device, you can connect via the Linux box first, then launch a telnet or SSH session to the destination device from there.

Note: When using a Cisco router as a jump point, it is recommended that you disable "logging synchronous" in the Line VTY section of the config. This can cause problems when trying to telnet out from the router.

By default, most Cisco routers would have been configured with 5 lines: `line vty 0 4`. CatTools is multi-threaded and can support, [depending on your edition](#), up to 30 client threads (connections). If you have created an activity for more than five devices, which all connect via the router, you may end up with timeout errors or connection failures as a result of all the available router lines being used.

To prevent these errors occurring you can:

- A. Increase the number of VTY lines available on your main connect via router.
- B. Set the client threads to use `5 threads`.
- C. Use a Linux box as a connect via device instead of a router.

Method Use the drop down list to select connection method. The default is `telnet`.

Note: that when using SSH there may well be a specific variant of it that is required to connect with the specific device. For instance, Netscreen devices supporting SSH2 require the variant `SSH2-nopty` to connect successfully with CatTools.

Port Enter the port number the selected connection method is to use. The telnet default is `23` and typically this does not need to be changed. SSH connections should use port `22`.

Passwords tab

Provide the information to login to or enable a device, and other passwords necessary to monitor the device through CatTools.

VTY Password The Virtual Terminal or initial login password.

Enable Password The enable mode or privileged password.

Privilege Level	<p>Set the enable mode privilege level, if required. Typically, this is not required and should be left blank. On a Cisco router, there are 16 privilege levels (0 to 15). Standard enable mode puts you in level 15.</p> <p>The privilege level is appended to the enable command when the command is sent to the device, so if data is present, but not required, it may cause an error when the device tries to authenticate enable mode.</p>
Console Password	The console password, if using a direct COM port connection.
Username	The device user name. This can be a user name type such as AAA, TACACS, RADIUS or Local username.
Password	The device password. This can be a password type such as AAA, TACACS, RADIUS or Local password.
SSH Username	The SSH Username, if required for an SSH device configuration.
SSH Password	Enter the SSH Password, if required for an SSH device configuration.
SNMP Read	Enter the SNMP Read community name for your device. The SNMP Read community default name is <code>public</code> . It is recommended you change your SNMP Read community name to something else. This field is required for running the <code>SNMP.System.Summary</code> activity otherwise the error <code>SNMP timeout or unknown error occurs</code> .
SNMP Write	Enter the SNMP Write community name for your device. The SNMP Write community default name is <code>private</code> . It is recommended you change your SNMP.
Initial login requires password	Check this box if the initial login to this device requires a password (typically the VTY or initial console password).
Initial login requires username/password	Check this box if the initial login to this device requires both a username and password.
Enable mode requires username/password	Check this box if the login to enable mode on this device requires both a username and password. The value from the Username field is used along with the Enable password previously configured. If you select this check box, the Username field and Enable password field must be completed.

Prompts tab

Configure the command line prompts used by a device, if they differ from the default. Entries are only required in this tab if a device has non-standard prompts configured.

VTY Prompt	The non-standard VTY login prompt for this device. The VTY prompt refers to the text that the device prompts you with when asking for the initial login password. For example: <code>Password:</code>
Enable Prompt	The non-standard Enable prompt for this device. The enable prompt refers to the text that the device prompts you with when asking for the Enable password. For example: <code>Password: or Super user login:</code>
Console Prompt	The non-standard Console prompt for this device. The console prompt refers to the text that the device prompts you with when asking for the initial console login password. For example: <code>Password:</code>
Username Prompt	The non-standard Username prompt for this device. The username prompt refers to the text that the device prompts you with when asking for the initial login user name. This is normally used along with a password for AAA/TACACS/RADIUS or Local authentication. For example: <code>Username: or Please enter login name:</code>
Password Prompt	Enter the non-standard Password prompt for this device. The Password prompt refers to the text that the device prompts you with when asking for the initial login password, normally used along with a username for AAA/TACACS/RADIUS or Local authentication. For example: <code>Password: or Please enter login password:</code>

Contact info tab

Describes contact information of personnel responsible for a specific device.

Address 1 / Address 2 / Address 3	Physical location of the device. This can be seen on the Devices list if the "Location" column has been enabled.
Contact Name	The name of the person responsible for this device.
Contact Phone	How to contact the person responsible for this device.
Contact E-mail	How to contact the person responsible for this device.
Contact Other	Any additional contact info.
Alert e-mail	Who to notify by e-mail of any alarms or alerts for this device.

Extra info tab

Additional device specific information and fields. These fields are all free form and you can enter any text you wish.

Serial number	The serial number of this device.
---------------	-----------------------------------

Asset Tag	Asset tag information.
Identification	Identification info for this device.
Other info	Any other serial number information.
Activity Specific1 / Specific2	Information specific to a particular activity.

On Palo.Alto.FireWall devices, you can use the Activity Specific2 field to receive data for the Backup.Runnig.Config command in an XML format. Set the Activity2 field as xml as shown:

The screenshot shows a 'Device Information' dialog box with the following fields and values:

- Serial number:
- Asset Tag:
- Identification:
- Other info:
- Activity Specific1:
- Activity Specific2: (indicated by a red arrow)


At the bottom of the dialog, there are four buttons: 'Ping device', 'Telnet/SSH', 'OK', and 'Cancel'.

Variations GUI tab

This tab is only visible on supported devices. In the Variations tab, you can specify user-defined device variations. For more information, see [Device Variations](#).


Custom devices in Kiwi CatTools

CatTools provides the ability to create custom device types and script files, should your device not be supported by one of the predefined device types.

 An understanding of and experience in Visual Basic scripting is required in order to successfully add custom scripts to CatTools. The sample code template files found the `/Templates` sub folder of the CatTools root directory provides help file documentation and comments for adding custom devices.

To add support for a custom device in CatTools, two fields are required:

1. The device type `.ini` file, which defines:
 - The device type name
 - The device ID
 - The user interface field values and default values which are displayed in the device form when adding or editing a device
2. The device script `.txt` file, which contains:
 - The device type's specific code to allows CatTools to log in to the device
 - Enter and exit different modes,
 - Activities to be performed - such as configuration backups, sending CLI commands, modifying device configurations
 - Parse command output data for reports

 The device script file also contains function calls and references to variables within the internal CatTools program code. These are prefixed with `cl`.

For a list of `cl` functions and variables, see [cl. variables and functions](#). These functions and variables can be used to develop your custom device script.

Create a custom device

The custom device template files provides by CatTools and referenced below are based on a Cisco Router device. Additional modifications to the device script `.txt` file are required once you have created it to customize the script to work successfully with your specific device.

For further information regarding customizing devices see:

- [Custom device type files \(.ini\)](#)
- [Custom device script files \(.txt\)](#)
- [Code examples of internal CatTools functions](#)

1. Create a custom device type .ini file


- a. Copy the `Custom.Device.Template.ini` file located in the `\Program Files (x86)\CatTools3\Templates` sub folder of the CatTools root directory.
- b. Save the copy to the `\Program Files (x86)\CatTools3\Devices` sub folder
- c. Enter a new file name using the syntax `Custom.[Manufacturer].[DeviceType]`. For example, `Custom.Cisco.FirewallPIX`.

2. Edit the .ini file


- a. Using a text file editor, such as Notepad, open the `.ini` file created in step 1.
- b. Change the following default values in the `.ini` file:

```
[device]
name=Custom.[Manufacturer].[Type]
id=4000
```

- Change `name` to a unique device name. For example: `Custom.Cisco.Router`

 Do not use spaces. Use a period (.) instead.

- Change `id` to a unique number that is not being used by any other device `.ini` file.

 The number must be within the range of 4000 to 4999.

```
[item_Name]
default= Unique device name
```

3. Save and restart CatTools

- a. After completing changes to the `.ini` file, save it to the `\Devices` sub folder and close.
- b. Restart CatTools to populate the Device type drop-down field. Your custom device type is now available in the device setup screen.

4. Create a custom device .txt script file


- a. Make a copy of the `Custom.Device.Template.txt` file located in the `\Templates` sub folder of the CatTools root directory.
- b. Save the copy of the file to the `\Program Files (x86)\CatTools3\Scripts` sub folder. Save the file using the value entered for `[device] name` in Step 2.

Ensure you retain the `.txt` file type suffix.

For example, if in the device `.ini` file settings, `[device] name=Custom.Cisco.FirewallPIX`, the script `.txt` file name must therefore be: `Custom.Cisco.FirewallPIX.txt`.

5. Edit the .txt file

- a. Using a text file editor, open the `.txt` file create in step 4.
- b. Follow the instructions under the SCRIPT NOTES section of the `.txt` template to customize the script for your device.

 Although you can open `.txt` files using NotePad, a syntax highlighting script editor can make reading and editing the file easier.

6. Save and test

- a. After you have made your edits to the `.txt` file, save the file to the `\Scripts` sub folder and close.
- b. To test your custom device, create an activity in CatTools that is supported by your custom device script. Run it manually using the Run Now button.
Using `Device.ConnectivityTest.Login` is recommended for testing since every device must be able to log in successfully to perform other activities.
- c. Check the Info Log pane for errors and edit your `.txt` file as necessary.

See [Testing your custom device](#) for more information and tips.

Define Kiwi CatTools custom device type using .ini file

Devices types are defined within CatTools using text (`.ini`) files. They are stored in the `\Program Files (x86)\CatTools3\Devices` sub folder within the root CatTools directory. Each `.ini` file represents a separate item in the `Device type` drop-down list field in the Device information setup form.

When CatTools starts, it searches for all the `.ini` files in the `\Program Files (x86)\CatTools3\Devices` sub folder and reads their contents. All the device types found are then available for selection in the Device type drop-down list when defining devices.

To add a new custom device type, you need to create a new `.ini` file defining the device and its characteristics.

.ini file sections overview

The contents of an `.ini` file are divided up within `[...]` sections. The main sections inside an `.ini` file are as follows:

Section	Description	Example
[info] section	This is required and identifies the <code>.ini</code> file as a CatTools file.	<pre>[info] cookie=CatTools version=3 author=SolarWinds</pre>
[device] section	<p>This is required and defines the name used within the CatTools user interface (i.e. the Device type drop-down list field) and within scripting. It also defines the unique key (id) of the device type in the CatTools database.</p> <p>Although the file name of the <code>.ini</code> file may be similar (or the same) to the <code>[device]</code> section name item, it is the name item that defines the device name within the user interface and not the <code>.ini</code> file name.</p> <p>CatTools reserves id's from 0 to 3999 for predefined device types. You can use the number range from 4000 to 4999 for the custom device types you create, however you must ensure the number is unique.</p>	<pre>[device] name=Custom.Cisco.Router id=4000</pre>

Section	Description	Example
[item] sections	<p>Each item section defines the input fields used within the CatTools user interface for the Device setup screens.</p> <p>Each input field in the CatTools user interface has a separate [item_xxx] type section.</p> <p>The name item sets the text (or label) to be displayed next to the input or selection field.</p> <p>The default item sets the default value to be used when adding a new device to the database.</p> <p>The required item tells the user interface if the field must contain valid data or not (for example: must be an item from within a predefined list or whether the user can enter their own values). 0=not required, 1=required.</p> <p>The info item is the description associated with this field. This text is displayed in the status bar of the user interface when a field has focus.</p> <p>The above [item_group] section is an example of a standard text box input field within the user interface.</p> <p>When the input field is to be a list box, you need to define the list box contents using a list item within the item section (see below).</p>	<p>[item_group]</p> <p>name=Group</p> <p>default=Default</p> <p>required=1</p> <p>info="The logical group that this device belongs to."</p>
[item_model]	<p>The list item contains a comma delimited string of items to populate the list box with. The default field specifies which item from the list to show as the default when adding a new device.</p> <p>Check box input fields can only accept values of 0 or 1. Any value other than a 1 is considered a 0. A check box example is below.</p> <p>The check box is defaulted to 1 (checked).</p>	<p>name=Model</p> <p>default=Other</p> <p>required=0</p> <p>info="The device model number."</p> <p>list=501,515,Other</p>

Section	Description	Example
[item_require_vty_login]	The size (length of the data that can be entered) of each field is determined by the design of the CatTools database. Any data entered via the CatTools user interface that is too long for the field is truncated accordingly.	name=Initial login requires password default=1 required=0 info="This device requires an initial password for access"

Custom device type .ini template

CatTools provides a starting point template file to help assist in the creation of a new custom device type. The template file is called `Custom.Device.Template.ini` and can be found in the `\Program Files (x86)\CatTools3\Templates` sub folder. This template file is included as part of the predefined device types in CatTools.

As for all of the CatTools predefined device types, it may be subject to updates and modifications with each new release of the CatTools product. For this reason, you should never modify the template file itself in order to create a new custom device type.

If you need to create a new custom device type, make a copy the template file and save it to the `\Devices` sub folder, giving it a new file name.

Creating your custom device type .ini file


When adding a new custom device type to CatTools, the first step is to make a copy of the template file `Custom.Device.Template.ini` in the `\Templates` sub folder and save it to the `\Devices` sub folder with a new file name. This gives you a new starting point device type file to work with. When naming the file, use the naming convention: `Custom.Manufacturer.Type`

Custom	Use the text "Custom" to distinguish between custom device types and CatTools predefined types. Custom devices are also grouped together in the Device type drop-down list field
Manufacturer	The manufacturer or supplier of the device. For example, Cisco, Nortel, 3Com, Juniper, Foundry, and so on.

Type The type of device added, such as a router, switch, or firewall.

SolarWinds does not recommend using the exact model number for the device as Type, as you can reuse your custom script for different models of the same Manufacturer and Type. For example: PIX model 501 and Cisco PIX model 515 can use the script `Custom.Cisco.Firewall` or `Custom.Cisco.PIX`.

Different models of similar device types can be specified within the device .ini file in the `[item_Model]` section. Alternatively you can add the model to the model drop-down list field at runtime.

 The Model field values in the CatTools predefined device types have no impact on the way the associated device script file (.txt) behaves.


Although SolarWinds recommends you follow this naming standard, the idea of the [custom device script file](#) is to give the user as much flexibility as possible while remaining within the limits of the application. If you need to have model specific behavior within your custom device script, you need to code the device script file accordingly.

Editing the .ini file

Open the custom device .ini file to make the required changes to the values within each section.

The custom .ini file contains a number of items values highlighted for emphasis. SolarWinds recommends that any values that are not highlighted be left at its original setting.

Highlight Color	Meaning
Red	The most important item values which must be changed are highlighted in red. These values must be unique otherwise your device type may not appear in the CatTools <code>Device type</code> drop-down list field.
Green	Items values which may be changed if desired are highlighted in green.
Blue	Additional comments are highlighted in blue.

 Do not include any additional comments in your .ini file

You can customize some of the items within a section, such as the name and info items, to make them more relevant to your device.

- Name item: the field label that is displayed within the form.
- Info item: the text that is displayed in the status bar when you select the field in the form.

After you have made your amendments to the `.ini` file, save it back to the `\Devices` sub folder. Restart CatTools for your new custom device type to be available in the Device type drop-down list field.

Default values

[info]


- `cookie=CatTools`
- `version=3`
- `author=SolarWinds`

Enter your name or leave as the default value.

[device]

- `name=Custom.Manufacturer.Type`

Change to a unique name. For example: `Custom.Cisco.Router`

 Do not use spaces. Use a period (.) instead.


- `id=4000`

Select a number within the range of 4000 to 4999. The number used must be unique.

Device information

[item_Group]

- `name=Group`
- `default=Default`

 Enter a default Group for the device or leave as the default.

- `required=1`
- `info="The logical group that this device belongs to."`

[item_Name]

- `name=Name`
- `default=Unique device name`

 Enter a unique device name.

- `required=1`

- info="A unique name for this device. e.g. sales-router or head-office-3500."

[item_HostAddress]


- name=Host Address
- default=127.0.0.1
- required=1
- info="IP address or host name of the device."

[item_Filename]


- name=File Name
- default=
- required=1
- info="The base file name to use for this device (unique)."

[item_Model]

- name=Model
- default=Custom

 Enter a default `Model` to use from the list below or leave as the default.

- required=0
- info="The device model number."
- list=Custom

 Additional `Model` numbers must be separated by commas. For example:
`Custom,501,515`

[item_ConnectVia]

- name=Connect via
- default=Direct connect
- required=1
- info="The name of another device to connect to first."

[item_Telnet]

- name=Method
- default=Telnet

i Enter a default `Method` to use from `list` below. If your device only uses SSH, change the default accordingly.

- required=1
- list=Telnet, SSH1, SSH2, SSH2-nopty, SSH1-DES, SSH1-3DES, SSH1-Blowfish, Cisco SSH

i Amend `list` accordingly for the device's available connection methods.

- info="Connection method to use."

[item_TelnetPort]

- name=Port
- default=23

i Enter a default port from the list below. Port field value automatically changes when Method field is changed in the user interface.

- required=1
- list=23,22
- info="Port number to use."

Device passwords

[item_VTYPass]

- name=VTY Password
- default=
- required=0
- info="VTY password."

[item_EnablePass]

- name=Enable Password
- default=
- required=0
- info="Enable or privilege password."

[item_PrivilegeLevel]

- name=Privilege Level
- default=

- required=0
- info="Sets the enable mode privilege level. (Not required in most cases)"

[item_ConsolePass]

- name=Console Password
- default=
- required=0
- info="The console (com port connection) password."

[item_AAAUsername]

- name=Username
- default=
- required=0
- info="AAA/TACACS/RADIUS/Local username."

[item_AAAPassword]

- name>Password
- default=
- required=0
- info="AAA/TACACS/RADIUS/Local password."

[item_SSHPassword]

- name=SSH Password
- default=
- required=0
- info="SSH password."

[item_SSHUsername]

- name=SSH Username
- default=
- required=0
- info="SSH username."

[item_SNMPRead]

- name=SNMP Read
- default=
- required=0
- info="SNMP Read community name."

[item_SNMPWrite]

- name=SNMP Write
- default=
- required=0
- info="SNMP Write community name."

[item_RequireVTYLogin]

- name=Initial login requires password
- default=1
- required=0
- info="This device requires an initial password for access"

[item_LoginUsesAAA]

- name=Initial login requires username/password
- default=0
- required=0
- info="The initial access requires a username and password"

[item_EnableUsesAAA]

- name=Enable mode requires username/password
- default=0
- required=0
- info="Enable mode access requires a username and password"

Device Prompts

[item_VTYPrompt]

- name=VTY Prompt
- info="Expected VTY prompt from the device. (Only required if non standard prompt is used)"

[item_EnablePrompt]

- name=Enable Prompt
- info="Expected enable mode prompt from the device. (Only required if non standard prompt is used)"

[item_ConsolePrompt]

- name=Console Prompt
- info="Expected console prompt from the device. (Only required if non standard prompt is used)"

[item_AAAUserPrompt]

- name=Username prompt
- info="Expected Username prompt from the device or AAA server. (Only required if non standard prompt is used)"

[item_AAAPassPrompt]

- name=Password prompt
- info="Expected AAA Password prompt from the device or AAA server. (Only required if non standard prompt is used)"

Device contact information

[item_Address1]

- name=Address1
- default=
- required=0
- info="Location of the device"

[item_Address2]

- name=Address2
- default=
- required=0
- info="Location of the device"

[item_Address3]

- name=Address3
- default=
- required=0
- info="Location of the device"

[item_ContactName]

- name=Contact Name
- default=
- required=0
- info="The name of the person responsible for this device."

[item_ContactPhone]

- name=Contact Phone
- default=
- required=0
- info="How to contact the person responsible for this device"

[item_ContactEmail]

- name=Contact E-mail
- default=
- required=0
- info="How to contact the person responsible for this device"

[item_ContactOther]

- name=Contact Other
- default=
- required=0
- info="Any additional contact info"

[item_AlertEmail]

- name=Alert e-mail
- default=
- required=0
- info="Who to notify by e-mail of any alarms or alerts for this device"

[item_SerialNumber]

- name=Serial number
- default=
- required=0
- info="The serial number of this device"

[item_AssetTag]

- name=Asset Tag
- default=
- required=0
- info="Asset tag information"

[item_Identification]

- name=Identification
- default=

- `required=0`
- `info="Identification info for this device"`

[item_SerialOther]

- `name=Other info`
- `default=`
- `required=0`
- `info="Any other serial number information"`

[item_ActivitySpecific1]

- `name=Activity Specific1`
- `default=`
- `required=0`
- `info="Information specific to a particular activity"`


[item_ActivitySpecific2]

- `name=Activity Specific2`
- `default=`
- `required=0`
- `info="Information specific to a particular activity"`

Define Kiwi CatTools custom device script using .txt file

Device scripts are defined within CatTools using .txt files and are stored in the `\Program Files (x86)\CatTools3\Scripts` sub folder within the root CatTools directory.

When an activity is run, CatTools reads the name item value in the `[device]` section from the device type .ini file to determine which device script .txt file it needs to use. Therefore each .txt file must be given the same file name as the name item in the corresponding .ini file.

 SolarWinds recommends that the .ini file and .txt files have the same file name, apart from the file extension.

For all the predefined device types in CatTools, the associated device script .txt files are encrypted. They are encrypted for two reasons:

- To protect our intellectual property.
- To prevent unauthorized modification of files which may cause the scripts to fail at run time.

Custom device script .txt template

CatTools provides a starting point template file, based on a Cisco Router, to help assist in the creation of a new custom device script. The template file is called `Custom.Device.Template.txt` and can be found in the `\Program Files (x86)\CatTools3\Templates` sub folder within the root CatTools directory.

If you need to create a new custom device script, make a copy the template file and save it to the `\Scripts` sub folder giving it a new file name.

i This template file is included as part of the predefined device scripts in CatTools, but is unencrypted. It is subject to updates and modifications with each new release of the CatTools product. For this reason, you should never modify the original template file when creating a new custom device script.

Creating your custom device script .txt file

When adding a new custom device script to CatTools, the first step is to copy the template file `Custom.Device.Template.txt` in the `\Templates` sub folder and save it to the `\Scripts` sub folder, giving it a file name the same as the value entered for the name item within the device type `.ini` file `[device]` section, and end with a `.txt` file type suffix. This gives you a new starting point device script file to work with.

For example, to create a new custom device script for a Cisco ASA Firewall device, you may have created an `.ini` file `Custom.Cisco.FirewallASA.ini` which contains the name item in the `[device]` section of `Custom.Cisco.FirewallASA`. Therefore, the custom script file must be named `Custom.Cisco.FirewallASA.txt`.

Editing the .txt file

The next step is to open the `.txt` file and make any necessary changes in order to customize the script to work with your device. The `.txt` file is well documented, providing instructions and assistance in making your modifications. There are a few important sections at the top of the script file which require further mention.

Visual Basic module name

You may see the following line at the very top of the script: `Attribute VB_Name = "Dev_CustomDeviceTemplate"`.


This is the Visual Basic module name for the device script file when viewing within the Visual Basic, or equivalent, development environment. This line typically does not appear when editing the file in a VB development environment, but may in other environments.

"Dev_CustomDeviceTemplate" is the name given to the custom template file module. If you have a number of custom device files, consider changing this default name to reflect your device script file name. By doing so, you can then open multiple custom device script files in your VB development environment.

If you are using a text based editor to modify your scripts, such as Notepad, changing the module name is optional. To avoid confusion you may prefer to change the name anyway.

Option Explicit

The Option Explicit statement enforces a level of compliance for the scripting code. You must explicitly declare all variables using the Dim statement, or, if re-dimensioning, using ReDim. If you attempt to use an undeclared variable name, an error occurs at compile time.

 Device script files use the Visual Basic Scripting language (VB Script). There is no declaration of the variable `types`.

Constant declarations

The script constants (`Const`) are declared below the Option Explicit statement.

- **Private Const SCRIPT_NAME = "Device Template"**

The `SCRIPT_NAME` constant is a reference given to the script file. This reference appears within the Info Log pane and `Infolog.txt` file whenever the script is called.

- **Private Const DEVICE_**

The `DEVICE_` constants are constants which define device configuration or responses from the device. For example: `DEVICE_USERNAMEPROMPT = "Username: "` implies that every time there is a prompt from the device to enter a user name, we expect to see a "Username: " prompt.

- **Private Const COMMAND_**

The `COMMAND_` constants are constants which define commands that CatTools sends to the device in order to perform specific tasks or actions when running an activity. For example: `COMMAND_ENTERENABLEMODE = "enable"` tells CatTools to send the `enable` command, instructing the device to enter `enable` mode.

Notes

--- SCRIPT NOTES ---

The Script Notes section provides useful information and lists tasks that are required to be carried out during the initial script creation phase. You can edit this section as and when you feel it is appropriate. An example may be deleting the initial setup tasks block of text from the Script Notes section after you have carried out the initial script setup tasks, as it no longer applies.

Any complex routines or script specific behavior which may be of assistance to any person maintaining the device scripts, should be mentioned in the Script Notes section. They should be either fully documented in the Script Notes section, or a reference made to the relevant function where they are fully documented.

--- DEVICE NOTES ---

The Device Notes section is an area where you can enter any device specific behavior or quirks which may be of assistance to any person maintaining the device scripts. An example may be: "The device you are creating the custom script for only supports SSH2".

Required functions

There are a number of functions that must exist within your device script in order for CatTools access the device and run activities.

Function	Action
Login	Authentication to the device
SendPostLoginCommands	Commands to be sent after successful login to device
EnterEnableMode	To enter privileged mode
SendPostEnterEnableModeCommands	Commands to set the environment after successfully entering enable mode
SendSessionTerminationCommands	To send necessary commands to exit or logout of the device

The `SendPost...` functions do not necessarily have to contain any commands, however, the function must exist and return `TRUE`. Likewise, if your device does not have the concept of an Enable mode, the device script still requires the `EnterEnableMode` function. However, the function only needs to include a single line of code sending the return value of `TRUE`.

Unsupported activities

The device template file contains a number of activities functions which have a call to the internal CatTools client function [cl.CatToolsNoSupport](#). You should call `cl.CatToolsNoSupport` so that a message is written to the Info Log to inform you that the activity is not supported.

Most activities only require the one line of code call to the `cl.CatToolsNoSupport` function. For example:

```
Function Activity_UpdatePassword()
Call cl.CatToolsNoSupport
End Function
```

Some activities also require a second line which supplies a return value to the CatTools main activity script:

```
Function Activity_TFTPUpload(iActionType, sTFTPServer, sTFTPFile,  
sResponseFile, bDoSaveToNVRAM)  
Call cl.CatToolsNoSupport  
Activity_TFTPUpload = "Not supported"  
End Function
```

If you decide to add support in your custom device script for any of these activities, you need to ensure that you replace the call to the `cl.CatToolsNoSupport` function with your new lines of code and update the return value, if one exists.

Saving the script file

Before CatTools can access the script file you have built, you must save it to the `\Scripts` folder. Normally, you do not have to stop and restart CatTools in order for script file code changes to be picked up. There may be occasions during testing in which you have to restart CatTools if the changes are not recognized.

Testing your custom device scripts

After you have the scripts set, test the new device with CatTools by setting up activities using the new device.

See [Testing your custom device](#) for more information and tips.

Send us your working scripts

If you would like us to consider adding your customized device script file as a predefined device type to ship with the product, please contact us using the [Technical Support](#) form on our [web site](#).

There are no guarantees as to when or if the device will be added to the predefined device types in CatTools, as there are a number of factors to consider, such as the number of requests for the device, complexity of the script, and technical resources available. However scripts that are sent in are cataloged for future reference.

Custom device script cl. variables and functions for Kiwi CatTools

The custom device template script contains many references to the internal CatTools client variables and calls to client functions. These variables and functions can be identified by their 'cl.' prefix.

Although the functions and variables are not documented within the template script itself, to understand what each one is used for and, in the case of the functions, understand the parameters they require and what their return values are, each variable and function exposed in the custom device template file is detailed below.

Variables

Device configuration variables (read from the device setup screen tabs)

Device Info tab

Variable	Description	Field Type
cl.CurDevName	Unique name assigned by user to refer to device.	Name field
cl.CurDevTelnet	The device connection method, i.e. Telnet or SSH.	Method field
cl.CurDevGroup	The logical group that this device belongs to.	Group field
cl.CurDevHostAddress	IP address or host name of the device.	Host Address field
cl.CurDevModel	The device model /series number.	Model field
cl.CurDevType	The CatTools device type.	Device Type field
cl.CurDevFilename	The base file name to use for this device (unique)	File Name field
cl.CurDevConnectVia	The name of another device to connect to first	Connect via field
cl.CurDevTelnetPort	Port number to use	Port field

Passwords tab

Variable	Description	Field Type
cl.CurDevVTYPass	Device VTY password	VTY password field
cl.CurDevEnablePass	Device Enable password	Enable password field
cl.CurDevPrivilegeLevel	Device Enable mode privilege level	Privilege level field
cl.CurDevAAAUsername	Device Username	Username field

Variable	Description	Field Type
cl.CurDevAAAPassword	Device Password	Password field
cl.CurDevRequireVTYLogin	Device requires Password only to log in (Initial login requires password tick box field)	
cl.CurDevLoginUsesAAA	Device requires Username and Password to log in (Initial login requires username/password tick box field)	
cl.CurDevEnableUsesAAA	Device requires Username and Password to enter Enable mode (Enable mode requires username/password tick box field)	
cl.CurDevConsolePass	Device console (com port connection) password	Console password field
cl.CurDevSSHUsername	SSH username	SSH Username field
cl.CurDevSSHPassword	SSH Password	SSH Password field
cl.CurDevSNMPRead	SNMP Read community name	SNMP Read field
cl.CurDevSNMPWrite	SNMP Write community name	SNMP Write field

Prompts tab

Variable	Description	Field Type
cl.CurDevVTYPrompt	Device VTY custom prompt assigned by user to device	VTY Prompt field
cl.CurDevEnablePrompt	Device Enable mode custom prompt assigned by user to device	Enable Prompt field
cl.CurDevConsolePrompt	Device Console port custom prompt assigned by user to device	Console Prompt field

Variable	Description	Field Type
cl.CurDevAAAUserPrompt	Device AAA Username custom prompt assigned by user to device	Username Prompt field
cl.CurDevAAAPassPrompt	Device AAA Password custom prompt assigned by user to device	Password Prompt field

Contact info tab

Variable	Description	Field Type
cl.CurDevAddress1	Location of the device	Address1 field
cl.CurDevAddress2	Location of the device	Address2 field
cl.CurDevAddress3	Location of the device	Address3 field
cl.CurDevContactName	The name of the person responsible for this device	Contact Name field
cl.CurDevContactPhone	How to contact the person responsible for this device	Contact Phone field
cl.CurDevContactEmail	How to contact the person responsible for this device	Contact Email field
cl.CurDevContactOther	Any additional contact info	Contact Other field
cl.CurDevAlertEmail	Who to notify by e-mail of any alarms or alerts for this device	Alert e-mail field

Extra Info tab

Variable	Description	Field Type
cl.CurDevSerialNumber	The serial number of this device	Serial Number field
cl.CurDevAssetTag	Asset tag information	Asset Tag field
cl.CurDevIdentification	Identification info for this device	Identification field
cl.CurDevSerialOther	Any other serial number information	Other info field
cl.CurDevActivitySpecific1	Information specific to a particular activity	Activity Specific 1 field
cl.CurDevActivitySpecific2	Information specific to a particular activity	Activity Specific 2 field

Other variables

Variable	Description
cl.RxBuffer	String of response data sent from the device
cl.ScheduleNumber	The current schedule number
cl.DeviceHostnameID	Device host name as recovered from the device after successful login
cl.DeviceVTYPrompt	The host name and ending with <code>DEVICE_STANDARDPROMPT</code>
cl.DeviceEnablePrompt	The host name and ending with <code>DEVICE_PRIVILEGEDPROMPT</code>
cl.DeviceConfigPrompt	The host name and ending with <code>DEVICE_CONFIGPROMPT</code>

Functions

The follow table lists a summary of functions available in custom device script files.

General and File

Variable	Description
cl.Initialise	Initializes (or defaults) cl. variables
cl.Delay	Force CatTools to pause for a given amount of time
cl.GetUniqueDeviceFileName	Generates unique filename in the <code>\ClientTemp</code> folder based on the device
cl.DBMetaCmd	Checks if a command is a database meta command, and then processes it
cl.UtilityMetaCmd	Checks if a command is a utility meta command, and then processes it
cl.Log	Sends a line of text to the <code>Infolog.txt</code> file and Info Log pane
cl.LogToFile	Writes data to a file

Activity

Variable	Description
cl.CatToolsNoSupport	Called for activities not supported by the device script
cl.DBCheckScheduleOption	Determine whether a particular option has been selected within a given activity

Device

Variable	Description
<code>cl.FlushRxBuffer</code>	Clears the <code>cl.RxBuffer</code> string. Used to clear previous response data before receiving next data
<code>cl.DetermineHostname</code>	Establishes and sets the device hostname and prompts <code>cl.variables</code>
<code>cl.SendData</code>	Sends text to device
<code>cl.SendAndWaitForEcho</code>	Sends text to device and waits for echo
<code>cl.SendAndWaitForPrompt</code>	Sends text to device, waits for an echo and then device prompt
<code>cl.WaitForData</code>	Waits for a specific data string to returned from the device
<code>cl.WaitForMultData</code>	Waits for any one of a range of specific data strings to be returned from the device

Text Manipulation

Variable	Description
<code>cl.ReplaceText</code>	Replace a substring within a given string of data, with a new substring
<code>cl.TextRemoveBlankHeaderLines</code>	Remove blank header lines from the beginning of a string (e.g. a device output buffer)
<code>cl.TextRemoveLinesContainingText</code>	Remove lines which contain a specific substring of text
<code>cl.TextInText</code>	Return start position of a substring within a string
<code>cl.TextRemoveTextUpTo</code>	Trim a text from a string up to or including a specified substring

Details and examples of functions

The functions listed are further detailed below with their input parameters and, where applicable, examples of their usage are provided.

`cl.Initialise()`

This function initializes (or defaults) [cl.variables](#) as follows:

```

cl.DeviceHostnameID = ""
cl.DeviceVTYPrompt = ""
cl.DeviceEnablePrompt = ""
cl.DeviceConfigPrompt = ""
cl.WaitForEcho = True
cl.WaitForTime = 0
    
```

cl.Delay(IDuration)

This function forces CatTools to pause for a given amount of time.

It has one input parameter:

IDuration	Amount of time to wait for (in seconds)
-----------	---

cl.GetUniqueDeviceFileName(sFolderName, sPrefixName) As String

This function generates a unique file name in the `\Program Files (x86)\CatTools3\ClientTemp` folder based on the device. The function has a return value of String being the path and a unique file name.

It has two input parameters passed by value, and two optional parameters:

sFolderName	String value to append to beginning of file name. This parameter can accept an empty string "".
-------------	---

sPrefixName	String value to append after sFolderName within the file name
-------------	---

Example use case

Write an output to a temporary file in the `\ClientTemp` folder:

```

' create the unique filename using only the prefix 'CLI'
sClientResultsFile = cl.GetUniqueDeviceFileName("", "CLI")
' write the data in the RxBuffer to a file in \ClientTemp folder using the
filename just created.
cl.LogToFile sClientResultsFile, cl.RxBuffer
    
```

cl.DBMetaCmd(sCmd) As Long

This function checks a command to see if it is a database meta command. If it is, then CatTools tries to process it. A [database meta command](#) enables you to directly update a field in a table in the CatTools database as part of an Activity. They are not run on a device. The syntax for a database meta command is [%ctDB:tablename:fieldname:value](#)

The function has one input parameter:

sCmd	String value of the command being evaluated
------	---

Function return values:

0	Command if not a database meta command
1	Command evaluates as a database meta command and is processed successfully (a debug message also sent to the Info Log)
-1	Command evaluates as a database meta command, but fails to be processed (an error message also sent to the Info Log)

Example use case:

Check whether command to send is a database meta command. If it isn't (i.e. it is a device command) then send it to device.

```

lRetVal = cl.DBMetaCmd(sCmd)
If lRetVal = 0 Then 'not a database meta command so send it
cl.FlushRxBuffer
' send command to device and wait for echo
bRetVal = cl.SendAndWaitForEcho(sCmd)
If bRetVal then
' command has been echoed, so lets execute it
cl.SendData vbCr
End if
End If
    
```

cl.UtilityMetaCmd(sCmd) As Long

This function checks a command to see if it is a utility meta command. If it is, then CatTools tries to process it. A [utility meta command](#) enables you to set various options for use in the Device.CLI.Send commands and Device.CLI.Modify Config activities. The activities are not run on a device. Utility meta commands change the default behavior of some of the internal CatTools functions.

The function has one input parameter:

sCmd	String value of the command being evaluated
------	---

Function return values:

0	command if not a utility meta command
---	---------------------------------------

1	command evaluates as a utility meta command and is processed successfully (a debug message also sent to the Info Log)
-1	command evaluates as a utility meta command, but fails to be processed (an error message also sent to the Info Log)

Example use case

Check if the command to send is a utility meta command. If the command is not meta, then send it to the device.

```

lRetVal = cl.UtilityMetaCmd(sCmd)
If lRetVal = 0 Then 'not a utility meta command so send it
cl.FlushRxBuffer
' send command to device and wait for echo
bRetVal = cl.SendAndWaitForEcho(sCmd)
If bRetVal then
' command has been echoed, so lets execute it
cl.SendData vbCr
End if
End If
    
```

cl.Log(iPriority, sMessage)

This function sends a line of text to the `Infolog.txt` file and Info Log pane.

It has two input parameters:

iPriority	Integer range 1 to 4 representing the logging 'level' for the line being sent. 1=Error, 2=Warning, 3=Information, 4=Debug
sMessage	Message text of the line being sent

Example use case

Log a level 4 (debug) message sending the text "Login Device Template: Custom Device 1". Assume `Const SCRIPT_NAME` from the Device Template script and that you have named your device Custom Device 1 in the Name field of the Device Info tab of the device setup screen.

```
cl.Log 4, "Login " & SCRIPT_NAME & ": " & cl.CurDevName
```

cl.LogToFile(sFilename, sData, bAppend)

This function writes data to a file.

It has three input parameters:

sFilename	String of the name and path of the file to write to
sData	String of data to write
bAppend	Boolean value (default or if unspecified is <code>False</code>). If <code>True</code> , then the file is appended. Otherwise it is overwritten

Example use case

Save current device response to `temp.txt` file on the `C:\` drive.

```
cl.LogToFile "c:\temp.txt", cl.RxBuffer, 1
```

cl.CatToolsNoSupport()

This function is called for those activities not currently supported by the device script.

cl.DBCheckScheduleOption(IScheduleNumber, IOptionNumber) As Long

This function is used to determine whether particular options have been selected within a given activity.

It has two input parameters:

IScheduleNumber	The current schedule number as a Long
IOptionNumber	The option item within the activity we are checking as a Long

Example use case

Check an activity, in this case a [Device.CLI.Send commands](#) activity, to see if the output of the command should be emailed.

```
lRetVal = cl.DBCheckScheduleOption(cl.ScheduleNumber, 8)
If lRetVal = 1 then
'code to email commands goes here...
End if
```

cl.FlushRxBuffer()

This function clears the `cl.RxBuffer` string. This function is normally used to clear out any previous device response data before receiving the next chunk of response data.

cl.DetermineHostname(Optional ByVal vStandardPrompt As Variant, Optional ByVal vPrivilegedPrompt As Variant, Optional ByVal vConfigPrompt As Variant) As Boolean

This function establishes and sets the [cl.variables](#) listed below and returns a value of `True` if successful.

cl.DeviceHostnameID	The host name of the device (e.g. DevHost123), used as 'seed' for the prompts below.	
cl.DeviceVTYPrompt	The host name and ending with <code>DEVICE_</code> <code>STANDARDPROMPT</code>	example: ??DevHost123>
cl.DeviceEnablePrompt	The host name and ending with <code>DEVICE_</code> <code>PRIVILEGEDPROMPT</code>	example: ??DevHost123#
cl.DeviceConfigPrompt	The host name and ending with <code>DEVICE_</code> <code>CONFIGPROMPT</code>	example: ??DevHost123 (

It has three optional input parameters passed by value:

vStandardPrompt	Device standard mode prompt (or VTY prompt). If none specified then ">" is used as the default
vPrivilegedPrompt	Device privileged mode prompt (or Enable prompt). If none specified then "#" is used as the default
vConfigPrompt	Device configuration mode prompt (or VTY prompt). If none specified then "(" is used as the default

cl.SendData(sDataToSend)

This function sends the specified text to the device.

It has one input parameter:

sDataToSend	string of the data to be sent to the device
-------------	---

Example use case:

Send a carriage return to the device

```
cl.SendData vbCr
```

cl.SendAndWaitForEcho(sDataToSend) As Boolean

This function sends a text string to the device and waits for an echo. By waiting for an echo of the text string it ensures that the string we are expecting to send is actually the string that is sent. For example, there may be a case in which we send a string and the device doesn't echo back with it in its entirety, or it gets substituted / corrupted.

The function has a Boolean return value: `True` if string is echoed back as expected, `False` if not.

It has one input parameter:

<code>sDataToSend</code>	String of the data being sent to the device and expected to be echoed back
--------------------------	--

This function is sometimes followed by the function `cl.SendData vbCr` which then executes the echoed string on the device.

Example use case

Send a command to show the device running configuration and wait for an echo. If echoed, execute the command.

```

bRetVal = cl.SendAndWaitForEcho("show running-config")
If bRetVal then
cl.SendData vbCr
End if
    
```

cl.SendAndWaitForPrompt(sDataToSend) As Boolean

This function sends a text string to the device and waits for an echo. If successfully echoed, the string is executed and the function waits for a valid device prompt to be returned (standard or privileged prompts). This function is normally called when executing a known valid command on a device with no output, for example, `term len 0` to turn off output paging.

The function has a Boolean return value: `TRUE` if string is echoed back and you then receive one of the expected prompts after the command has been executed, and `FALSE` if not.

It has one input parameter:

<code>sDataToSend</code>	String of the data being sent, echoed and then executed on device
--------------------------	---

cl.WaitForData(sData, lTimeout) As Boolean

This function waits for a given amount of time for the specific string data to be returned from the device. The function has a Boolean return value: `True` if string is found within the timeout period specified, `False` if not.

It has two input parameters:

sData	The string of data we are waiting to receive from the device
lTimeOut	The amount of time to wait for (in seconds)

Example use case

Send the command to exit config mode and then check if we have been returned to the Enable mode prompt within a 30 seconds timeout.

```
cl.SendData Chr(26) ' i.e. CTRL-Z
If cl.WaitForData('HostName1#', 30) = False Then
cl.Log 4, "Failed to exit Configure Terminal mode"
End if
```

cl.WaitForMultData(rgMult, Optional iChoices = 0, Optional lTimeOut = 0) As Long

This function waits for a given amount of time for any one of string data items defined in the range, to be returned from the device within the specified amount of time. The function has a return value of Long representing which item of string data it has found first. It returns 0 if none of the items are found within the given timeout value.

It has three input parameters:

rgMult	The range of possible string data items we are expecting back from the device
iChoices	(Optional) The count of the number of items we are looking for - excludes rgMult (0) item. Optional as this is now calculated by the function itself if not specified
lTimeOut	(Optional) The amount of time to wait for (in seconds). Optional parameter, if not specified then default CatTools internal timeout value is applied

Example use case

Wait for one of the device prompts to be returned within 30 seconds.

```
rgMult(1) = "HostName1>"
rgMult(2) = "HostName1#"
rgMult(3) = "HostName1(Config)"
iChoices = 3
lTimeOut = 30
iRetVal = cl.WaitForMultData(rgMult, iChoices, lTimeOut)
If iRetVal = 0 Then
cl.Log 4, "Failed to receive device prompt"
End if
```

To satisfy this case, you can also send:

```
cl.WaitForMultData(rgMult)
```

which uses the default timeout within CatTools, or:

```
cl.WaitForMultData(rgMult, , lTimeOut)
```

which uses your specified timeout value. The function works out the `iChoices` value. If you are using the constant `COMMAND_TIMEOUT` to specify your timeout, then you can increase the timeout using:

```
Const COMMAND_TIMEOUT = 30 '(i.e. in seconds)
iTimeOutMultiplier = 2 '(increase by a factor of 2)
cl.WaitForMultData(rgMult, , COMMAND_TIMEOUT * iTimeOutMultiplier)
```

cl.ReplaceText(sData, sFind, sReplace) As String

This function replaces a substring within a given string of data, with a new substring. It replaces all substring occurrences with the new substring. The function has a return value of String being the new string of data with replacements. If no replacements are made, then the return string is the same as the original string. The function returns a null string if the length of the input parameter `sData` is 0.

It has three input parameters:

sData	String to be searched
sFind	The substring we want to replace
sReplace	The new substring we want to replace with

Example use case 1

Replace all occurrences of the substring "old text" with "new data" within the string object `sData`.

```
sData = cl.ReplaceText(sData, "old text", "new data")
```

Example use case 2

Remove nulls from within the string object `sData`.

```
sData = cl.ReplaceText(sData, Chr(0), "")
```

cl.TextRemoveBlankHeaderLines(ByVal sData) As String

This function is used to remove blank header lines from the beginning of a string, such as a device output buffer. It is used primarily for massaging the configuration data output from a device to create the backup file. It works through from the top of the output buffer string, removing lines containing just a carriage return <CR> or a line-feed <LF> (or both). The function has a return value of String being the 'trimmed' buffer string.

It has one input parameter passed by value:

sData String we are manipulating

Example use case

Trim all the blank lines from the top of the buffer string object sConfigData.

```
sConfigData = cl.TextRemoveBlankHeaderLines(sConfigData)
```

cl.TextRemoveLinesContainingText(ByVal sData, ByVal sFind) As String

This function is used to remove lines from within the device output buffer which contain a specific substring of text. It is used primarily for massaging the output data from a device to create a text file or report. The function has a return value of String being the new buffer with the lines containing the substring removed.

It has two input parameters passed by value:

sData String to be searched

sFind Substring we are searching for

Example use case

Remove all config lines of data from the buffer string object sConfigData, containing the device paging prompt text for example --More--.

```
Const DEVICE_MORETEXT = "--More--"
sConfigData = cl.TextRemoveLinesContainingText(sConfigData, DEVICE_MORETEXT)
```

cl.TextInText(ByVal sData, ByVal sFind) As Integer

This function returns the start position of a substring within a string. If nothing is found, or the length of either input strings are 0, then the function returns 0.

It has two input parameter passed by value:

sData	String to be searched
sFind	Substring we are searching for

Example use case

Find the start position of the device header text. If the value is 0, the start position has not been found, so send a line to the Info Log.

```
Const DEVICE_CONFIGHEADERTEXT = "Generating configuration:"
iRetVal = cl.TextInText(sConfigData, DEVICE_CONFIGHEADERTEXT)
If iRetVal = 0 then
cl.Log 4, "Failed to find device configuration header text"
End if
```

cl.TextRemoveTextUpTo(ByVal sData, ByVal sFind, Optional bAndIncluding As Boolean = False, Optional bForwards As Boolean = True) As String

This function is used to trim a text from a string up to or including a specified substring. The trim can work in either direction, such as from the start of a file to the end, or from the end to the start. The function has a return value of String, being the new string with the relevant text removed, or if the substring `sFind` cannot be found then the original string `sData` is returned.

It has two input parameters passed by value, and two optional parameters:

sData	String to be searched
sFind	Substring we are searching for
bAndIncluding	(Optional) Boolean value: if set to <code>True</code> trims up to and including the substring text <code>sFind</code> . If <code>False</code> (default) then the substring <code>sFind</code> is not trimmed.
bForwards	(Optional) Boolean value: if set to <code>True</code> (default) trims from the start, or top, of the string we are searching. If <code>False</code> , the function trims from the end (or bottom) of the string backwards.

Example use case

Check output for header text and if found remove everything up to and including the header text.

```
Const DEVICE_CONFIGHEADERTEXT = "Generating configuration:"
If cl.TextInText(sConfigData, DEVICE_CONFIGHEADERTEXT) > 0 Then
sConfigData = cl.TextRemoveTextUpTo(sConfigData, DEVICE_CONFIGHEADERTEXT,
True, True)
End If
```

Testing your custom device

Testing your custom device scripts is relatively straight forward.

After you have the device type [.ini](#) file and the device script [.txt](#) file set up, you can test the new device with CatTools by setting up activities using the new device type. To test that you can access the device, SolarWinds recommends using the [Device.ConnectivityTest.Login](#) activity. Nearly all of the activities that run on a device require logging in to the device. If the [Device.ConnectivityTest.Login](#) activity fails, it is likely that the other activities also fail.

Aids to test your device

There are several testing aids available in CatTools:

1. Info Log messages

Within the Info Log pane you see messages appear while an activity is running. The level of messages that appear can be filtered by the drop-down list near the bottom of the Info Log window. Selecting level 4 `Show Debug events and above`, displays any `cl.log` messages generated during the running of an activity. The client script function `cl.Log 4,"message"` is found throughout device scripts to display level 4 messages which help assist in troubleshooting device specific issues.

i All Info Log messages are logged to a file called `InfoLog.txt` in the CatTools root folder. This file can get very large very quickly. To purge the file, from the CatTools File Menu select `File > Delete > Delete Infolog.txt`.

2. Debug log

Under the File menu of CatTools there is an entry titled `Enable Capture Mode`. This turns on the logging of the conversation between CatTools and the end device, creating a disk file in the `\Debug` sub folder of the communications.

This file can be useful when the script does not appear to be communicating correctly with the device after entering a command. The scripting mechanism waits for known prompts when issuing a command to the device, but eventually times out if an expected prompt does not appear. The debug log capture mode file shows what the device actually sent to CatTools, and you can adjust your code accordingly.

Device specific information

This sub-chapter provides further information on various different devices that help explain the device behavior or helps in configuring the device in CatTools.

- [Device variations configuration](#)
- [Configure Dell devices for CatTools](#)
- [Configure backup activity for ASA dap.xml file](#)
- [Configure backup activity for 3Com superstack switches](#)
- [Connect to a Cisco Terminal server](#)
- [Connect to a device using session method](#)
- [Connect to a Netscreen device using SSH2 protocols](#)
- [Add a CiscoVPN device](#)
- [Add a Cisco WAAS device](#)
- [Backup a device via SFTP or SCP](#)

Device Variations

What are variations?

There are a number of device specific script variables that CatTools uses during the execution of device scripts. If the characteristics of your device do not match these device specific variables then the script does not work correctly. The variations system provides a way to override these device specific variables on a device by device basis.

Which devices support variations?

The `Generic.Device` script is a basic script designed to be a starting point for the use of variations. However, some of the other scripts also support the use of variations. In the device information panel a variations tab is available if the [device supports variations](#). You can access the [variations GUI](#) via the Variations tab.

The screenshot shows the 'Device Information' dialog box with the following fields and values:

- Vendor: [All Vendors]
- Device Type: Cisco Firewall.ASA
- Group: Default
- Name: ASA Firewall 2
- Host Address: 127.0.0.1
- File Name: ASA_Firewall_2
- Model: ASA5505
- Connect via: Direct connect
- Method: Telnet
- Port: 23

Buttons at the bottom: Ping device, Telnet/SSH, OK, Cancel.

Overview of how to use variations

To use variations on a specific device a variations file for that device must be placed in the `\Program Files (x86)\CatTools3\Variations` sub folder of the CatTools install directory. The variations file must be named the same as the Device File Name in the CatTools Device Information > Device info pane. The file must have a `.txt` extension.

For example if your device was `Generic Device 1` with a File Name of `Generic_Device_1` your variations file would be called `Generic_Device_1.txt`. You can manually create this file using notepad or you can use the [variations GUI](#).

An individual variations file takes precedence over a group variations file.

Applying variations to a group of devices

It is possible to apply a single variations file to all devices of the same script type that belong to the same group. To do this, name the variations file the same as the device script type, for example, `Generic.Device.txt`. Place the file in a folder with the same name as the group the devices belong to. This folder should be place in the `Variations\Groups` folder.

For example, to apply a variations file to all devices of the `Juniper.Router` device type that belong to a group called `HeadOffice` your directory structure would be:

```
...CatTools\Variations\Groups\HeadOffice\Juniper.Router.txt
```

You can manually create this file using notepad or you can use the [variations GUI](#).

Variations File Contents

The variations file should include only the items you need to override. Outlined below are the items that can be overridden. The value within " " is the value that you should change to your device specific value.

Login Prompts

These are the prompts that the device displays when it asks for a user name or a password.

- `DEVICE_USERNAMEPROMPT = "Username:"`
- `DEVICE_PASSWORDPROMPT = "Password:"`

Mode Prompts

These are the characters which appear at the end of the hostname in the various modes the device has. So for example, if at your VTY prompt `Cisco2950>` was displayed the prompt would be `>`.

- `DEVICE_STANDARDPROMPT = ">"`
- `DEVICE_PRIVILEGEDPROMPT = "#"`
- `DEVICE_CONFIGPROMPT = "("`

i Note that in this case, for the `config` prompt only the first symbol that appears to the right of the hostname in config mode is selected. This would match the extended config modes. In this example, it would match `Cisco2950(config)` and `Cisco2950(Config-if)` because `Cisco2950(` appears in both of these prompts.

More Text (Paging prompts)

This is the text that is displayed at the end of a page when device paging is turned on.

i You must include all of the `more` paging prompt text. The `Device.Backup.Running Config` activity strips the more text from the saved config. If you have not set the full `more` paging prompt text, then the saved config may contain additional undesirable or invalid lines of config.

- `DEVICE_MORETEXT = "--More--"`

Invalid / Incomplete command

These are the messages output by the device when an invalid / incomplete command is entered.

i You do not have to include all of the `error` text but enough so that it can be uniquely recognized and distinguished from other device output.

- `DEVICE_INVALIDCOMMAND = "% Invalid input detected at"`
- `DEVICE_INCOMPLETECOMMAND = "% Incomplete command"`

Yes/No Text

This is the text that appears when the device asks you a question which requires a yes or no answer, for example: `Reload Config (y/n)`. You should not include all of the text, but just enough of the text so that it can be uniquely recognized and distinguished from other device output.

- `DEVICE_YESNOTEXT = "(y/n)"`

More Text Keystroke

This is the key that is pressed to continue from a paging prompt.

- `MORETEXT_KEYSTROKE=" "`

Device Commands

These are commands that are issued to the device during execution of certain parts of the script. The commands to enter the `leave enable / privileged exec (enable) mode`.

- `COMMAND_ENTERENABLEMODE = "enable"`
- `COMMAND_EXITENABLEMODE = "disable"`

The commands to enter or leave config mode.

- `COMMAND_ENTERCONFIG = "configure terminal"`
- `COMMAND_EXITCONFIG = "CTRL-Z"`

The commands to disable / enable paging.

- `COMMAND_DISABLEPAGING = "terminal len 0"`
- `COMMAND_ENABLEPAGING = "terminal len 24"`

The commands to show the running config and the start up config.

i Note: SolarWinds recommends avoiding the use of abbreviated commands such as `"sh run"` as some of the scripts look for parts of the command, including `show`, and use this to determine whether to increase the timeout for that command.

- `COMMAND_RUNNINGCONFIG = "show running-config"`
- `COMMAND_STARTUPCONFIG = "show startup-config"`

The commands to save changes, disconnect from the device and ping another device.

- `COMMAND_SAVENVRAM = "write mem"`
- `COMMAND_DISCONNECT = "exit"`
- `COMMAND_PING = "ping"`

Backup Ignore Text

The `Device.Backup.Running Config` activity defines certain items that are ignored as changes when the current and previous configs are compared. This variation allows you to add additional device specific items to the ignore text. Unlike the variations above you can have multiple `Backup Ignore Text` items in your variations file. You cannot use `'|'` to delimit multiple items in a single line. Each ignored item must be on a new line. See [Ignore Text](#) for more information.

- `BACKUP_IGNORETEXT = "^! Last configuration change at"`
- `BACKUP_IGNORETEXT = "<Time"`

Pre-Login Keystroke

Some devices require keystrokes to be sent prior to login to wake up the device. You can specify more than one `Pre Login Keystroke` item so if there is a case in which you needed to press the space bar twice prior to logging in, you must include two `Pre login keystrokes` lines, each sending a single space.

- `PRE_LOGIN_KEYSTROKE = "CTRL-Y"`
- `PRE_LOGIN_KEYSTROKE = "CTRL-Z"`
- `PRE_LOGIN_KEYSTROKE = "CR"`
- `PRE_LOGIN_KEYSTROKE = " "`
- `PRE_LOGIN_KEYSTROKE = "Y"`

Pre-Login Message

Some devices that require pre-login keystrokes prompt with a message. If this is the case, the pre-login message should be set. This ensures the keystrokes are not sent until the message prompt is displayed.

- `PRE_LOGIN_MESSAGE = "Press CTRL-Y to continue"`

Post-Login Keystroke

Some devices require keystrokes to be sent after to login. For example, at a menu to get to the CLI prompt. You can specify more than one `Post Login Keystroke` item. For example, if you needed to press the space bar twice after log in, include two `Post login keystrokes` lines, each sending a single space.

- `POST_LOGIN_KEYSTROKE = "CTRL-Y"`
- `POST_LOGIN_KEYSTROKE = "CTRL-Z"`
- `POST_LOGIN_KEYSTROKE = "CR"`
- `POST_LOGIN_KEYSTROKE = " "`
- `POST_LOGIN_KEYSTROKE = "Y"`

Post-Login Message

Some devices that require post-login keystrokes prompt with a message. If this is the case, the post-login message should be set to ensure the keystrokes are not sent until the message is displayed. This should be part of the text that appears in a menu prior to keys strokes being sent to enter the CLI.

- `POST_LOGIN_MESSAGE = "Press C to enter the CLI"`

Full Prompt Values

CatTools normally determines the device host name and the specified prompts determine the full VTY, enable and config prompt. Should CatTools not be able to do this for your device you can specify the full values. If any of the below four values are used as an override CatTools does not try to determine the host name for the device. Generally speaking, if you intend to use any of the below you should specify values for all four items.

- `FULL_HOSTNAMEID = "Cisco2950"`
- `FULL_VTYPROMPT = "Cisco2950>"`
- `FULL_ENABLEPROMPT = "Cisco2950#"`
- `FULL_CONFIGPROMPT = "Cisco2950 ("`

Data Timeout

When CatTools waits for a response from a device it waits for a specified period of time, normally 30 seconds, and if no data is returned, timeouts. For certain commands which are known to produce more output, such as show commands, this default timeout is increased using a multiplier, normally by 4. If you have a device which is particularly slow, you can increase the default timeout from 30 seconds.

- `RESPONSE_TIMEOUT= "30"`

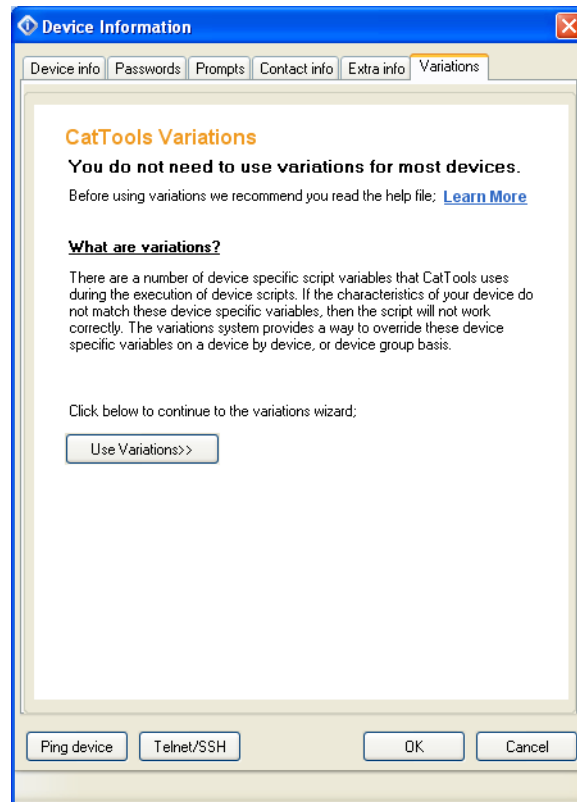
Strip Characters

Some devices include spurious escape or ANSI or Null characters in their output. Functionality to remove these from the buffer as the data is captured can be turned on if required.

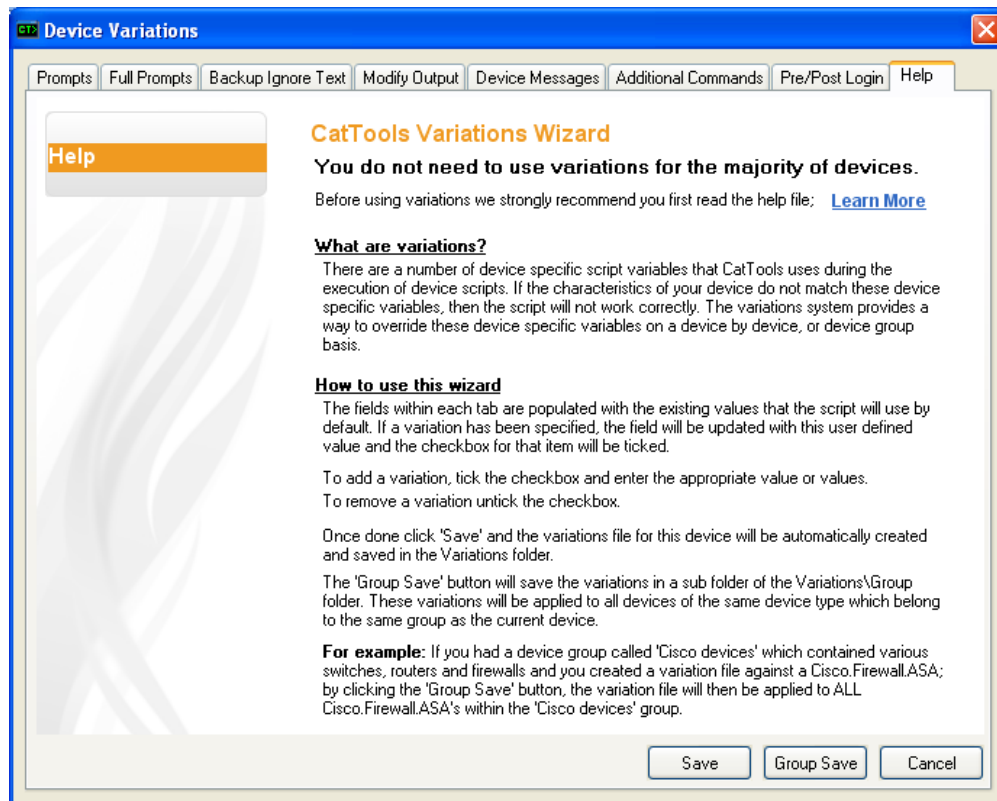
- `RESPONSE_STRIP_VT100ESC = "1"`
- `RESPONSE_STRIP_ANSICHARS="1"`
- `RESPONSE_STRIP_NULLS="1"`

Device Variations GUI

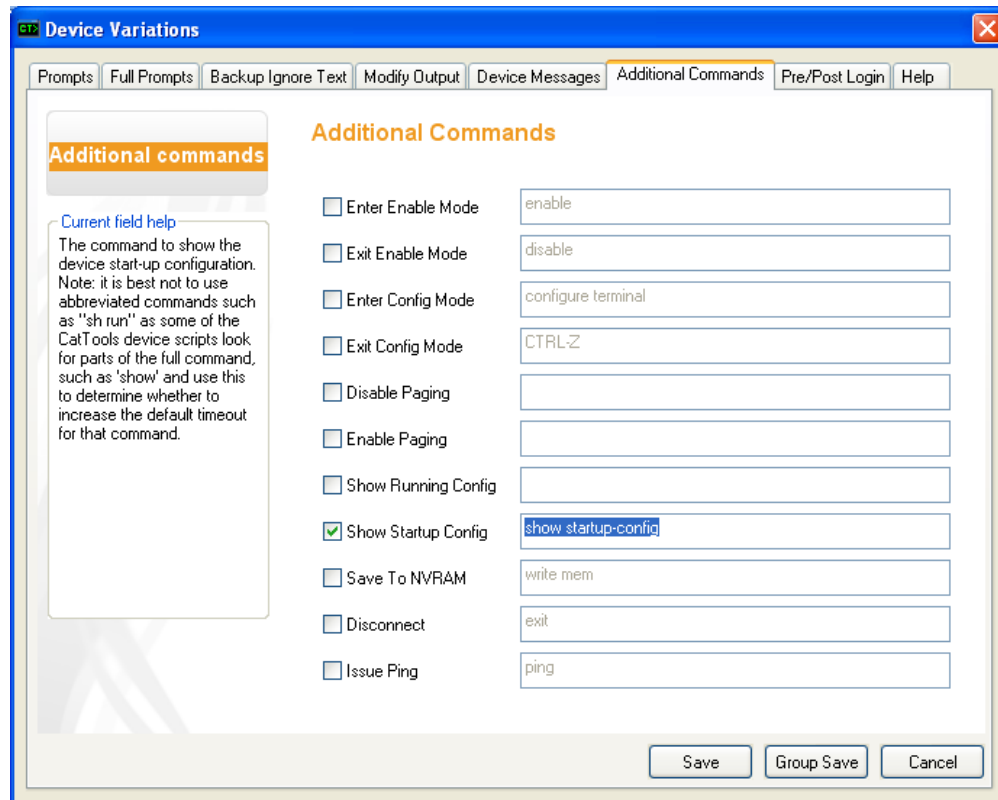
Although you can manually create a variations file in Notepad, the variations GUI helps to automate this process. If the device script supports variations then the Variations tab will be available on the Device Information panel. See [Device Variations](#) for more information on the individual variations.




To open the Variations GUI, click `Use Variations` on this tab. The Variations GUI opens, defaulting to the Help tab.



The tabs on the GUI show any variations currently being used. If a field shows a grayed out value, this is the value that the underlying script is currently using. To select a variation, select the check-box and enter the variations value.



 When your cursor is placed in a text-box field, the Current field help panel in the left pane is populated with additional helpful information about that field.

Saving a device variation

After selecting and entering values for the fields you want to override, click the Save button. The variations file is created and saved to the variations folder for that device.

Applying a device variation to a group of devices

Click the Group Save button to apply the variations to all devices of the same device type as the current device and that belong to the same group as the current device.

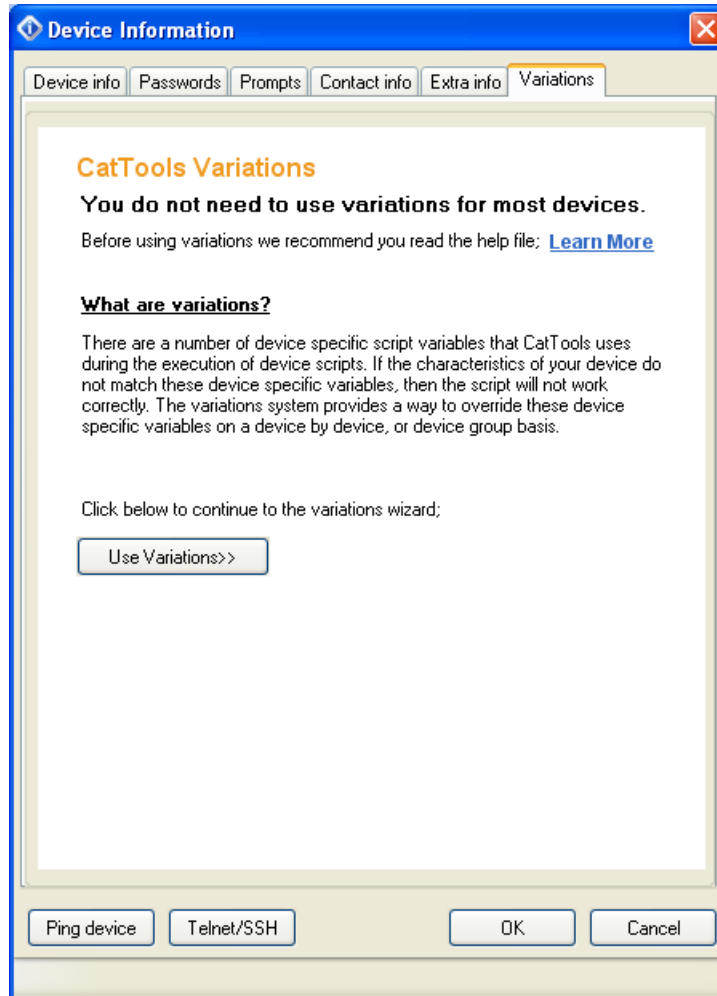
Removing variations

To remove all variations you can delete the variations file from the variations folder or sub folder. You can also remove all the variations in the variations GUI, and save or save the group as appropriate.

How to use variations in Kiwi CatTools

There are some device specific variables that need to be used when running activities in CatTools. If they do not match, an error is thrown in CatTools. To avoid this and also to run device specific variables, device variations are provided in CatTools. When adding a device in CatTools, the variation tab is displayed if the device supports variations.

To start configuring the variation options for a device, click on the Variations tab in Device Information, and then click on Use Variations. This opens the device's variations configuration wizard.



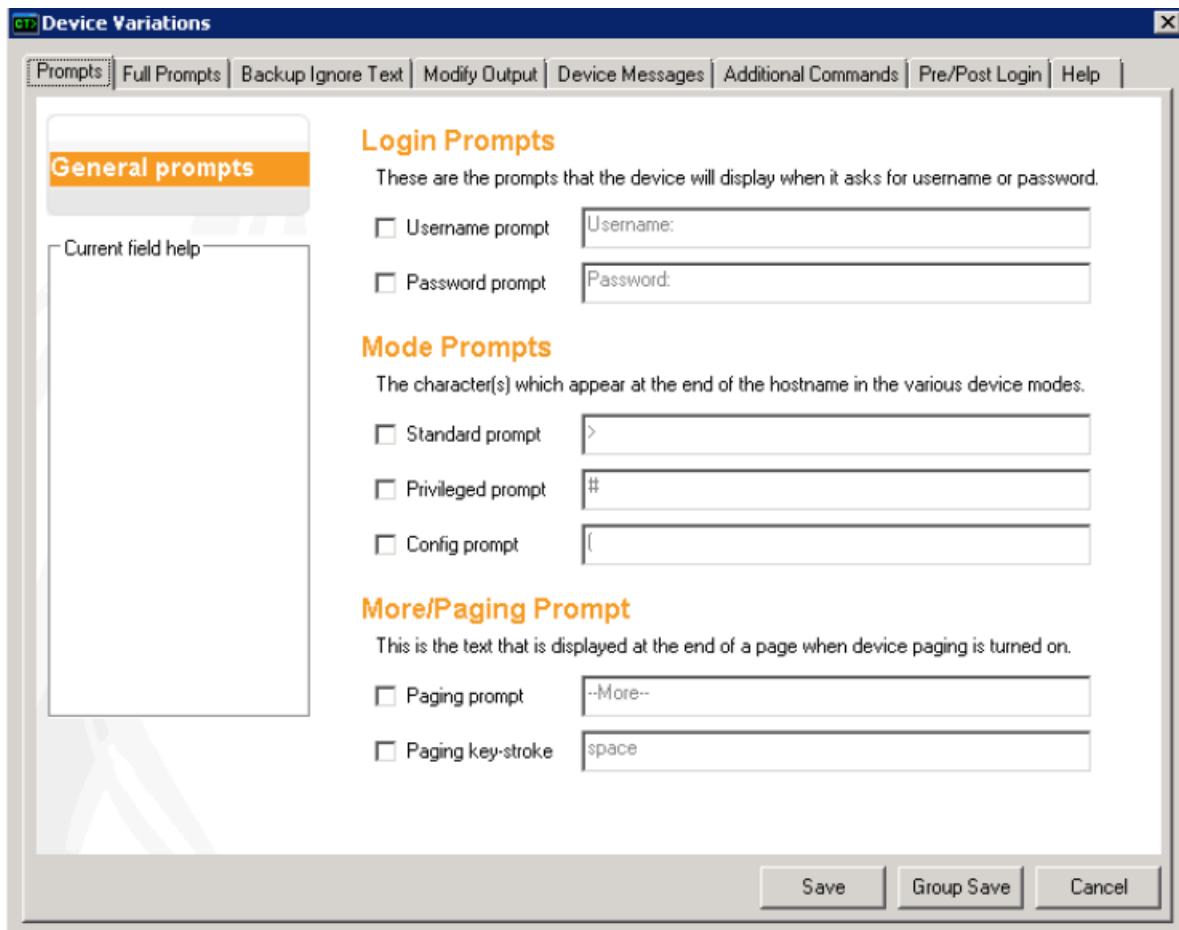
Device variations supported by CatTools

Alcatel.Router	Alcatel.Switch.OmniStack	ApcAOS	APC.AOS.CLI
BluecoatCacheFlow	Brocade.Switch	BluecoatCacheflow	Cisco.CallManager
Cisco.Firewall.ASA	Cisco.NXOS	Cisco.Router.General	Cisco.Small.Business
Cisco.Switch.IOS	Cisco.Wireless.Lan	Dell.Switch.CLI	Dlink.Switch.General

Enterasys.Secure.Stack	Enterasys.Wireless.Controller	F5.BigIP	F5.BigIP.TMSH
Fortinet.FortiOS	Generic.Device	Juniper.Router	Lantronix.EDS
Mikrotik.Router	Nortel.Switch.Ethernet	PaloAlto.FireWall	SonicWall.SonicOS
HP.Switch.2500	RedHat Linux	Huawei General	Checkpoint VPN

Prompts tab

On the Prompts tab, you can enter the applicable information in the fields under the Login Prompts, Mode Prompts, and More / Paging Prompts sections.




The screenshot shows the 'Device Variations' window with the 'Prompts' tab selected. The window has a menu bar with 'Prompts', 'Full Prompts', 'Backup Ignore Text', 'Modify Output', 'Device Messages', 'Additional Commands', 'Pre/Post Login', and 'Help'. On the left, there is a 'General prompts' section with a 'Current field help' area. The main content area is divided into three sections:

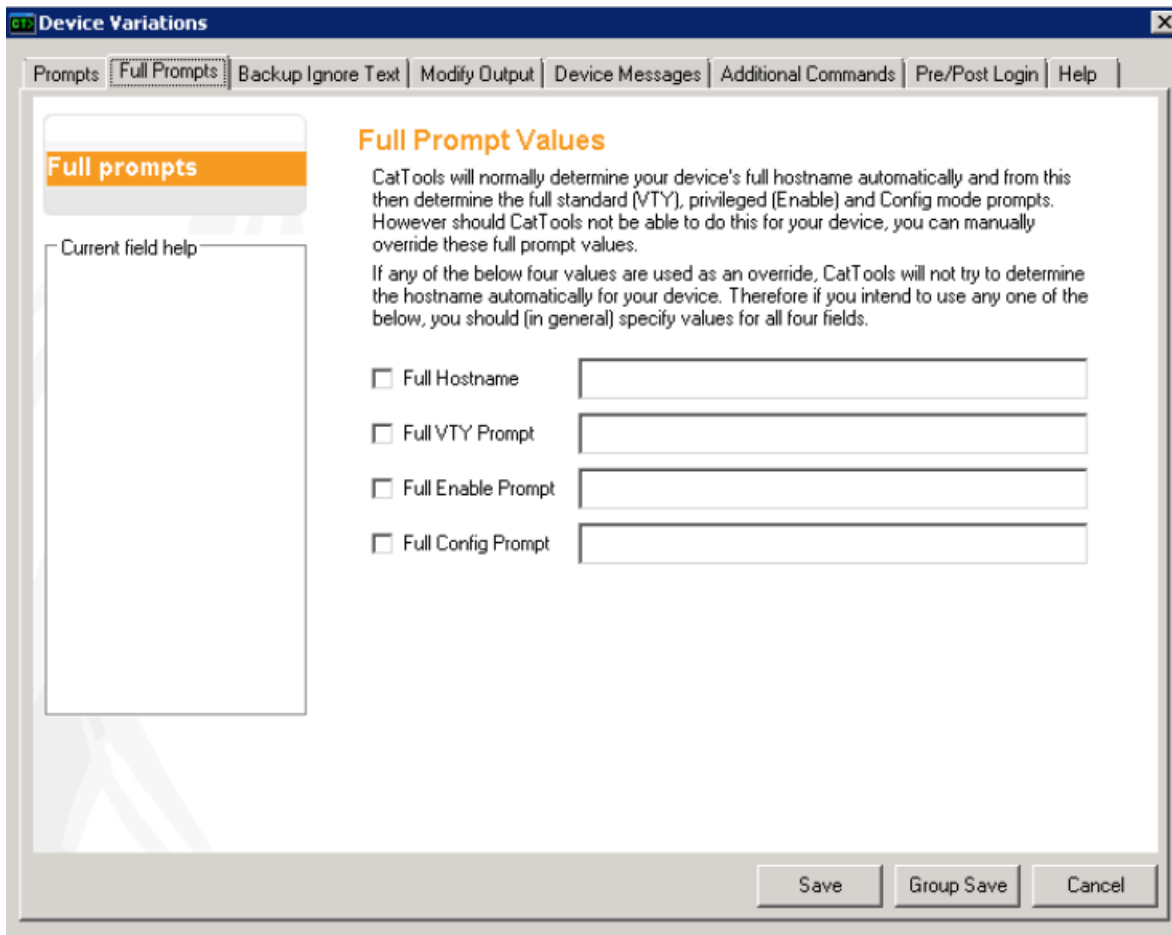
- Login Prompts:** Described as 'These are the prompts that the device will display when it asks for username or password.' It contains two checkboxes: 'Username prompt' (with a text field containing 'Username:') and 'Password prompt' (with a text field containing 'Password:').
- Mode Prompts:** Described as 'The character(s) which appear at the end of the hostname in the various device modes.' It contains three checkboxes: 'Standard prompt' (with a text field containing '>'), 'Privileged prompt' (with a text field containing '#'), and 'Config prompt' (with a text field containing '(').
- More/Paging Prompt:** Described as 'This is the text that is displayed at the end of a page when device paging is turned on.' It contains two checkboxes: 'Paging prompt' (with a text field containing '--More--') and 'Paging key-stroke' (with a text field containing 'space').

At the bottom right, there are three buttons: 'Save', 'Group Save', and 'Cancel'.

Full Prompts tab

From the Full Prompts tabs, you can manually enter the prompt values for standard, privileged and config modes.

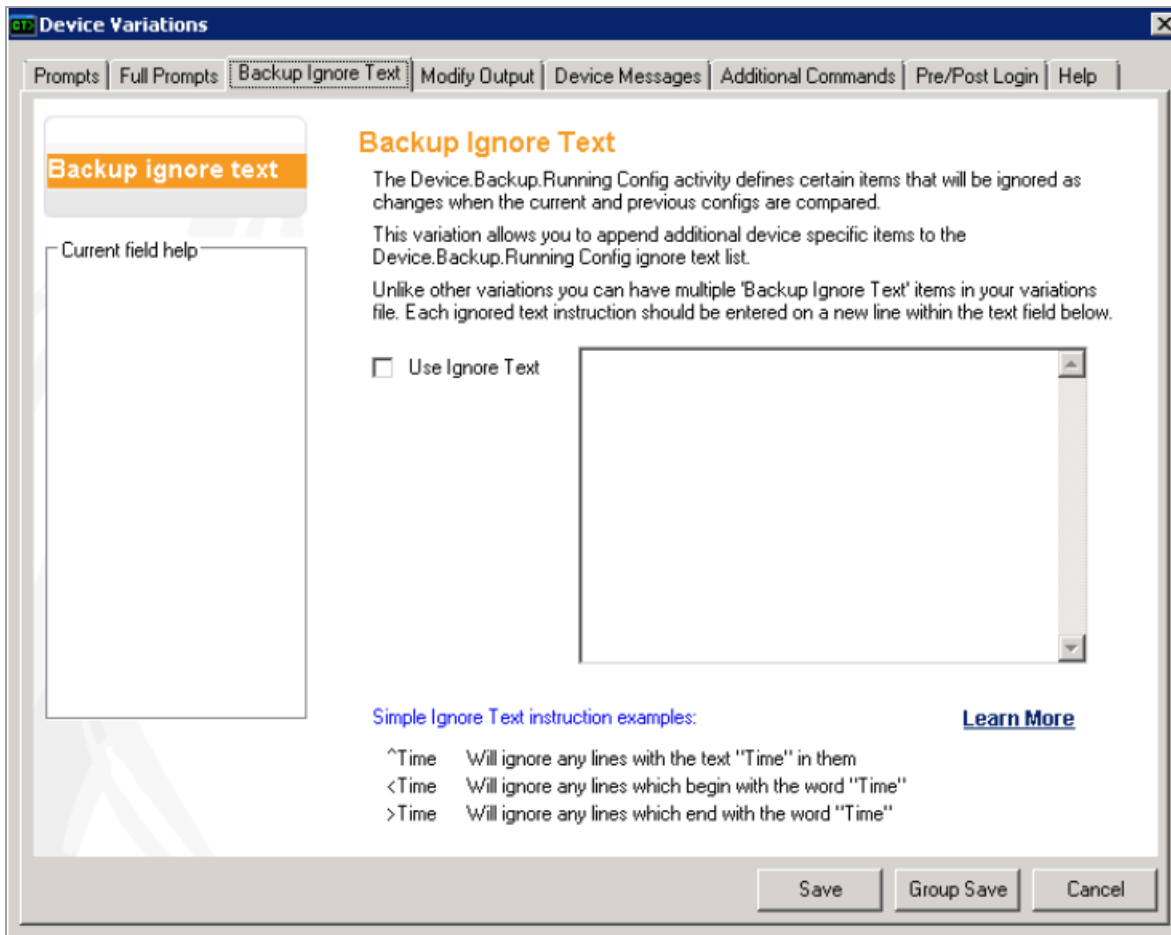
 If any of the prompts are used, then CatTools does not determine the device's host name automatically. It is advisable to enter values for all four prompt fields.



Backup Ignore Text tab

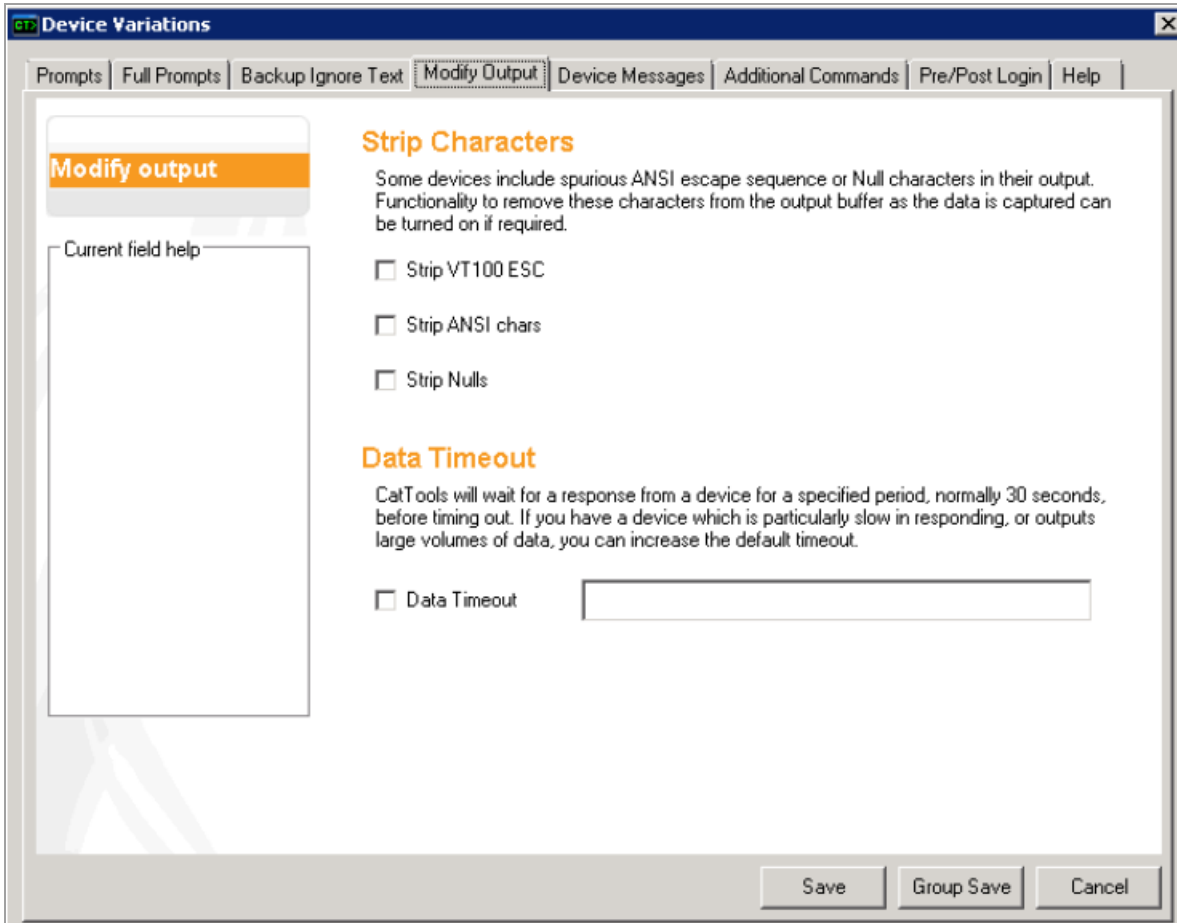
The backup ignore text tab allows you to append additional values to the `Backup.Running Config` ignore text list. You can use the following prompts to specify the text to be ignored when configurations are compared:

- `{start} {end}`: Message between the start and end is ignored.
- `[block] [block]`: Message or content to be ignored fully.



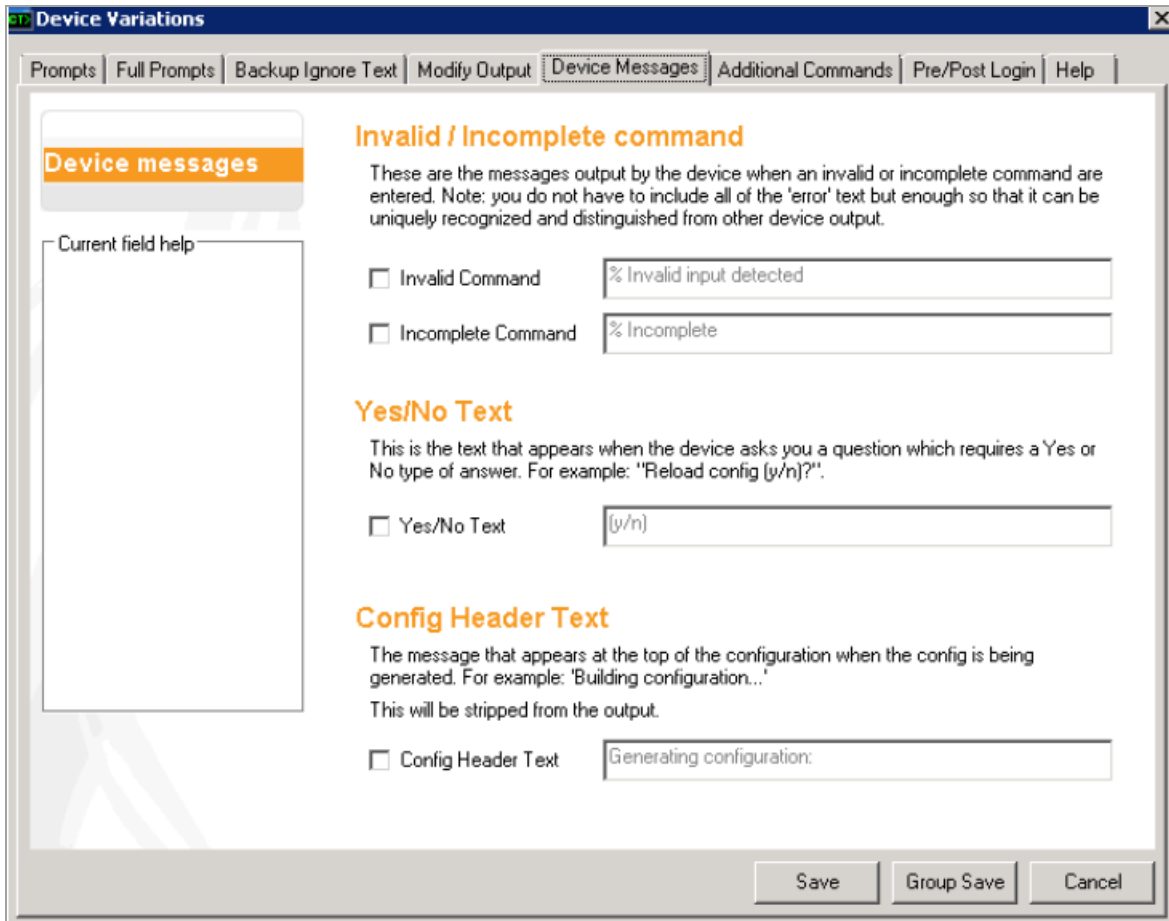
Modify Output tab

The Modify Output tab is used to remove ANSI characters from the output buffer once the data is captured. You can also increase the timeout for a device if you anticipate that it may run exceptionally slow.



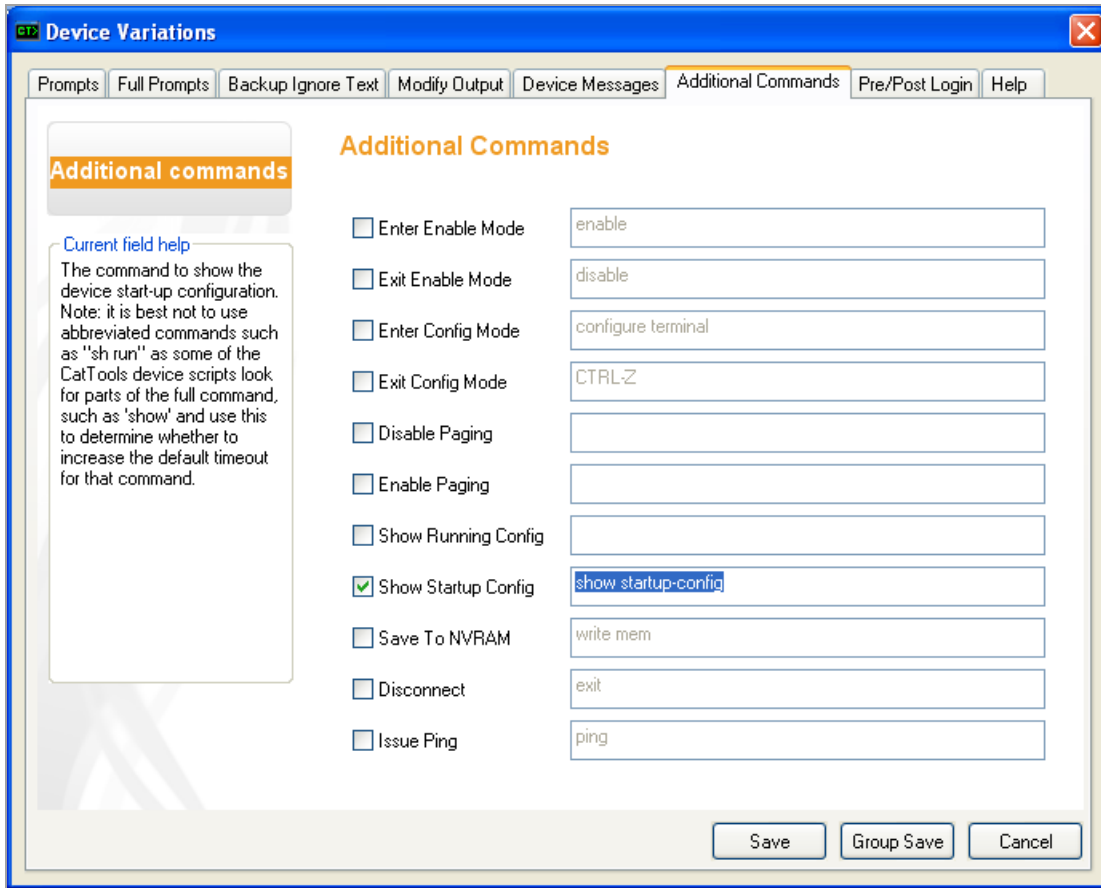
Device Messages tab

The Invalid / Incomplete command section identifies the new value given for the invalid or incomplete value entered. The Yes / No text section specifies the value to be given when a yes or no question is asked by the device. The Config Header text section is for stripping text from the config output any messages you expect to be displayed at the top of the configuration for the device.



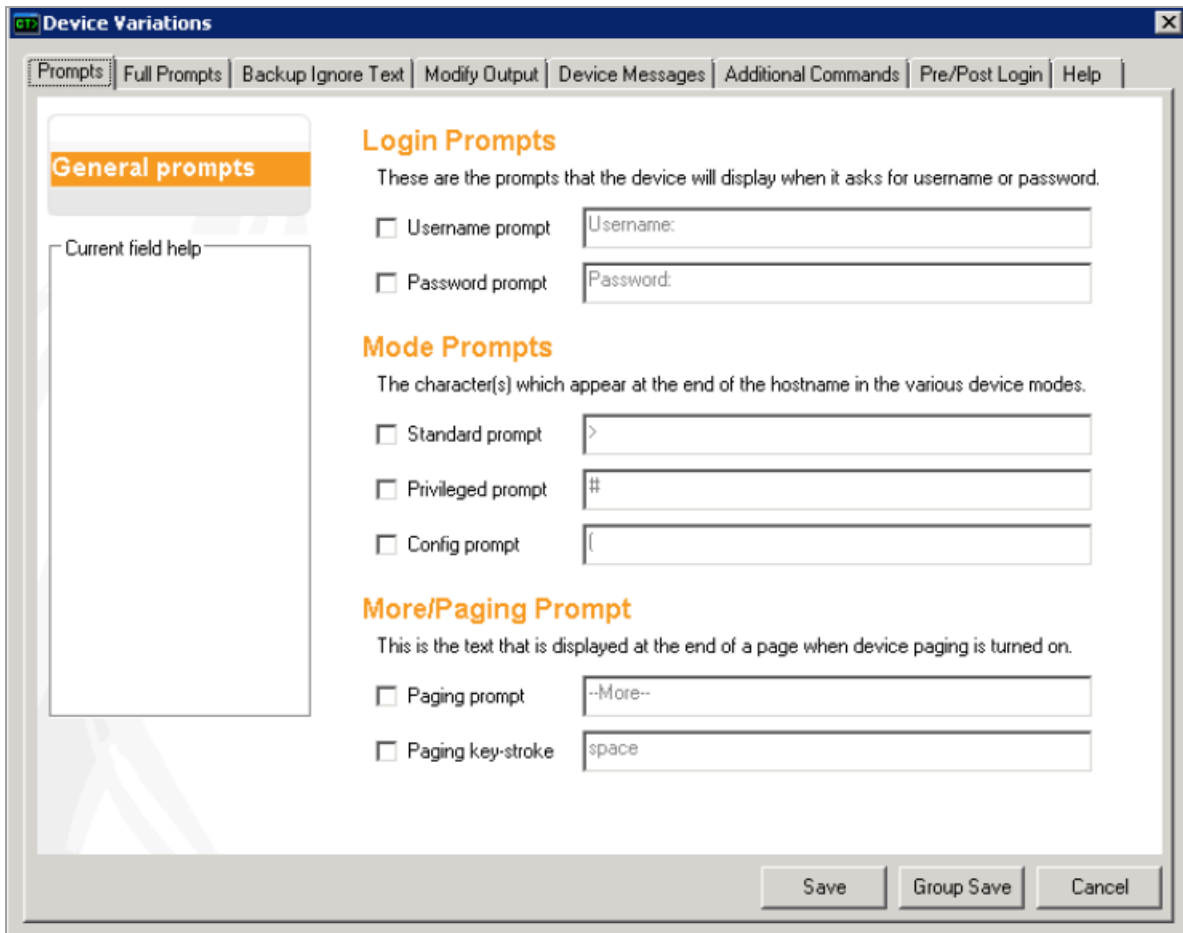
Additional Commands tab

These help in giving additional commands which may need to be given to the device.



Pre/Post-Login tab

You can set required keystrokes and messages pertaining to a device before or after the device logs in.



Device Variations

Prompts | Full Prompts | Backup Ignore Text | Modify Output | Device Messages | Additional Commands | Pre/Post Login | Help

General prompts

Current field help

Login Prompts
These are the prompts that the device will display when it asks for username or password.

Username prompt Username:

Password prompt Password:

Mode Prompts
The character(s) which appear at the end of the hostname in the various device modes.

Standard prompt >

Privileged prompt #

Config prompt (

More/Paging Prompt
This is the text that is displayed at the end of a page when device paging is turned on.

Paging prompt --More--

Paging key-stroke space

Save Group Save Cancel

Pre-login keystroke

Some devices require a keystroke to be sent prior to logging in to a device.

Pre-login message

Pre-login keystrokes prompts with a message. This message can be set here.

Post-login keystroke

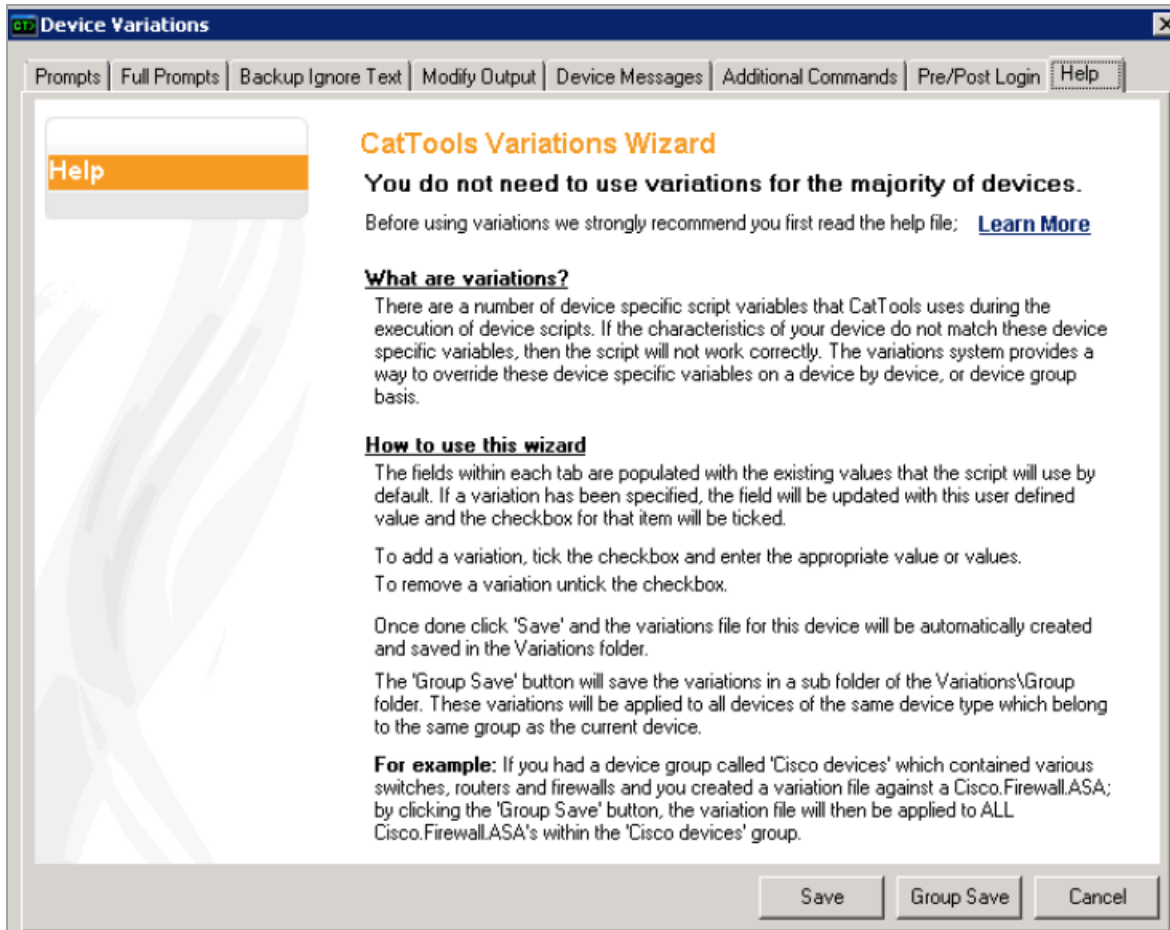
Some devices require a keystroke to be sent after logging in to a device.

Post-login message

A post-login keystroke prompts with a message. This message can be set here.

Help tab

The general help tab opens the general help guide for configuring device variations.



Using Dell devices in CatTools

On some Dell devices, such as PowerConnect switches 5224, 5324, 6024, and so on, the device may be interrupted with system or port status messages when capturing device output for configuration backups or reports. Because these messages are sent to the terminal window, they are captured along with the rest of the data output and can therefore cause reporting issues or problems with the configuration backup activity compare function.

To prevent these messages from occurring, you change to the device global configuration by sending the following command when in configuration mode:

```
no logging console
```

This command prevents all system messages from being written to the console / terminal window.

To turn on system messaging again, execute the command:

```
logging console level
```

The `level` limits the logging of messages displayed on the console to a specified level:

- emergencies
- alerts
- critical
- errors
- warnings
- notifications
- informational
- debugging

SSH2 connection issues

There appears to be an issue with the implementation of SSH2 for the Dell operating system version 2.0.0.12. When using the SSH2 connection method in CatTools to connect to a Dell device and the Username and Password fields values are set correctly, users may still see the Info Log error: `Failed to determine hostname - timeout`.

Verify the version of the Dell operating system running on the device. This issue is isolated to the 2.0.0.12 version of the Dell operating system. Versions prior to this are not effected. The problem has been acknowledged by Dell and, as of March 2008, is expected to be resolved in a firmware release. For information on the firmware edition that resolves this issue, contact Dell support directly.

This issue is not specifically limited to CatTools, and may also effect other clients used to connect to the device using the 2.0.0.12 version of the Dell operating system. For example, PuTTY 0.60 has the same problem, although PuTTY 0.58 does not.

Configure ASA `dap.xml` file backup

The Dynamic Access Policies (DAP) configuration of ASA v8.0 is stored in a file called `dap.xml` on the flash memory. It is not stored as part of the running config file or startup config file. A normal CatTools backup activity does not backup this file.

To backup this file with CatTools

To backup this file with CatTools, you will need to create a second activity using trivial file transfer protocols (TFTP) to transfer the file off your device.

1. Enable the Device.Backup.Running Config activity. A green check mark will appear inside the check box when it is enabled.

2. Edit the [Device.CLI.Send commands](#) activity and click on Options. Insert the following commands and click OK.

```
copy disk0:/dap.xml tftp://tftpserver-name/dap.xml
dap.xml
tftpserver-name
%ctDeviceName.%ctDateISO.dap.xml
```

 This will automate a manual command to backup the dap.xml file.


3. Run both activities to complete the backup.

Configure backup for 3Com superstack switches

Kiwi CatTools supports many 3Com superstack switches: 610 / 630 / 1100 / 3300 / 3300 FX / 3300XM / 3300TM / 3300SM / 3300MM. However, it has been found that 3Com switches can have issues when backing up the configuration of your switch.

Although telnet is not the most elegant, nor the most efficient, way to obtain the configuration of a 3Com switch, there aren't many other practical options. By capturing the output of various display commands, CatTools provides a readable list of configured settings in case of switch failure.

Since 3Com does not provide a "show config" type command, the CatTools backup method is the next best thing. Because there are several commands required to obtain a complete list of settings, the backup of your 3Com switch can take up to five minutes per switch. A future version of CatTools may use SNMP or HTTP to obtain the config.

 Occasionally, while accessing the 3Com switch CLI, the switch may perform a warm reset at odd times and for no obvious reason. This reset causes the telnet session to disconnect. All network traffic stops flowing until the switch has finished its start-up sequence which can take approximately 15 seconds. The switch appears to reset more often when the telnet session originates from a high speed port (100 MBps). Once the switch has warm started, it is more prone to crashing again. A power reset (removal of the power cord from the switch) does help to return the switch to a stable state and reduce the chance of a crash occurring.

The crash does not appear to be associated with any particular command and occurs with all versions of switch agent software, up to and including version 2.62.

Using an Admin user account to login instead of a Security user account reduces the occurrence of crashes.

Recommended backup plan

Since the chance of a switch crash, backup of the devices should only occur during times of low traffic use or after hours and you should reduce the frequency of your backups. Only make a backup when you believe the configuration has changed. Use the device Admin user account instead of the Security user account to login.

At this stage the config always appears to have changed. This is because various packet counters and stats are included in the captured config. These values change with each backup. A future version of CatTools may parse the captured data better and produce a cleaner list.

What commands are issued?

CatTools issues the following commands and captures the output to a file:

1. `bridge display`
2. `bridge multicastFiltering routerPort list`
3. `bridge port address list all`
4. `bridge port summary all`
5. `ethernet summary all`
6. `feature resilience detail`
7. `feature trunk summary`
8. `ip interface display`
9. `snmp trap display`
10. `system display`
11. `system inventory`
12. `system module display`
13. `system security access display`
14. `system security user display`
15. `bridge vlan summary all`
16. `bridge vlan detail`
Defined VLAN numbers
17. `bridge port detail`
1 to number of switch ports

18. feature trunk detail

1 to number of trunks

Device setup

When setting up your switch device, ensure that you enter the username and passwords for the device into the appropriate fields. Leave the VTY and enable passwords blank as they are not used for 3Com superstack switches. The standard 3Com usernames are:

- Monitor
- Admin
- Manager
- Security

You need to use the Security login if you want to make changes to passwords or user accounts.

Accessing switch stacks

A switch stack consists of up to four switches connected via matrix cables. Each switch has a unit number from 1 to 4 which identifies it. The IP address is the same for each device in the stack. Once logged in, you switch between the stack units by issuing the command `system unit N`, where `N` is the number of the stack unit you want to session to. For example, `system unit 3` switches to the third stack unit.

Refer to the '[Connecting via a session](#)' section as to how to set up the devices in CatTools.

Sending commands to the CLI via CatTools


By using the [Device.CLI.Send commands](#) at enable prompt menu, you can send commands to the 3Com switch CLI. The registered version of CatTools allows you to capture the output to a file.

For example, to change the password for the user "admin" use the commands:

```
system security user modify admin
newpass
newpass
private
```

Another example of a CLI command is adding port 3 to VLAN 2 with tagging:

```
bridge vlan addport 2 3 802.1Q
```

 To drop back a menu item, use the command `q` (quit).

Connecting via a Cisco Terminal Server

Kiwi CatTools 3.11 supports a Cisco Terminal Server device type. This device type is designed to enable CatTools to connect to devices physically connected to a terminal server type device via the console port. It should support authentication to both the terminal server device and to the device connected to it, or to just one or other of the devices. It should support either local or remote authentication. This type of connection is sometimes known as reverse telnet.

Set up a terminal server

1. Open Kiwi CatTools.
2. From the Devices pane, click Add to open the device wizard.
3. Add a new device to CatTools using the Cisco Terminal Server device type.
4. Set the device Host Address to the correct IP address of the device.
5. Set the Connect via to `Direct connect`.
6. Set the authentication on this device to the passwords required when using a port of the terminal server.

Use the terminal server to connect a device

To use the terminal server to connect to a device make the following settings on the device that is attached to it:

1. Set the CatTools device type to the type that corresponds to the actual device type.
2. Set the Connect via to `connect via the terminal server device`.
3. Set the Method to `telnet`.
4. Set the Port to the port on the terminal server the device is connected to, for example Async Port 1 = TCP port 200.
5. Set the authentication required to access the device itself.

Run an activity when connected to the terminal server

To run an activity on the device that is connected to the terminal server, just select the device itself.

When CatTools wants to connect to the device, it knows to connect to the terminal server first. It connects to the terminal server using the port that is defined to the device that is connected to via the terminal server. If any authentication is required to use the port, it uses the authentication details from the terminal server. After CatTools authenticates to the terminal server, it sends a CR to it to activate the port. It then authenticates to the device behind the terminal server using the credentials set to that device.

There are several things to bear in mind when utilizing a terminal server with CatTools.

- The connection is normally via a comm type port. The device may well be significantly slower to access from CatTools.
- The terminal server must be directly connected to by CatTools. CatTools does not support the use of a terminal server that has to be connected to via another device.
- If you need to connect to the terminal server itself to say issue commands or back it up, define a separate device to CatTools using its normal device type, such as Cisco router general for a Cisco 2509 router. Set its device details up with the correct Host Address, connection method, and the like. Set the port to the normal connection method port, such as port 23 for telnet.
- After disconnecting from a device on the terminal server, the port connection may be left open before timing out. The duration of the timeout depends on the settings on the device.

Where the device requires authentication, and you try to reconnect to it before the port connection has timed out, CatTools may fail to connect with a message "Did not receive VTY entry prompt ...". This happens because CatTools is set to send authentication to the device and the device does not require it as the port connection is still available. You can unset the authentication options for the device in CatTools, and it should then connect.



If this is a problem, you may want to set the exec timeout (Cisco router) to `value` low to ensure the console connection does not stay up for long periods of time.

Connecting to a device via a session

The Session connection method was added to provide support for the many and varied methods by which inter-connectivity can be achieved. Connections using this method can be achieved from one host device to any virtual device, blade, session or module.

To make use of this feature you need to setup two devices in CatTools. One that represents the parent device and one that represents the virtual device or session contained within it. Connection to the parent device is as normal. To connect to the virtual device you need to implement the following three fields on the device form.

1. Host Address: used to execute any command that may be necessary to connect to the next device. Examples include:
 - `Session 15`
 - `Changeto System`
 - `Session slot 4 proc 1`
 - `Telnet`
2. ConnectVia: set to the device you need to connect to first in order to establish the session.
3. Method: set to `Session`.

This feature is only available to specific devices:

- Cisco.Router.General
- Cisco.Switch.IOS
- Cisco.Switch.CatOS
- Cisco.Firewall.ASA
- Cisco.Firewall.PIX
- Cisco.ACE
- 3COM.Switch.SSII

Set up a connection via a session

The following example illustrates how this concept works: A Cisco 6509 has a firewall service module that you want to backup. The 6509 is CatOS and the firewall services module (FWSM) is Cisco IOS.

1. Add a device called `My 6509` as device type `Cisco.Switch.CatOS`.
2. Set the necessary username and password information.
3. Set the Host Address field to the IP of the device.
4. Set the Connect Via field to `Direct Connect`.
5. Set the Method field, in this example, to `Telnet`.
6. Create a second device called `My FWSM` as device type `Cisco.Switch.IOS`.
7. Set the Connect Via field option to `My 6509`.
8. Set the Method field to `Session`.
9. In the Host Address field, enter the command that is necessary to access the module. In this example set the field to `Session Slot 4 Proc 1`.
10. Configure login details on the Passwords tab.
11. To backup the FWSM, create a backup activity and assign `My FWSM` to it.

When CatTools runs the backup activity it identifies that to get to `My FWSM` it needs to go through `My 6509`. Therefore, CatTools logs into `My 6509` first. Once logged in, CatTools issues your custom command to establish the connection and backs up the device as defined by the `Cisco.Switch.IOS` device type.

Disclaimer regarding connecting via a session

Normal communication between CatTools and the connected device is a process of sending commands and receiving known responses. In this instance, we know what responses to expect because we know the OS of the device we are connected to.

However, the Session connection method can be used to connect from one device to another. In most cases these parent-child devices are on the same operating system but in some cases they won't be. For that reason the logic behind this particular connection routine is different versus all other devices. It works in reverse.

As we do not know what a login prompt, if one even exists, might look like on the second device we can not use that prompt as a known good response. Instead we look for the known bad responses of the first device at the time the connection command is issued. If anything other than a known error is received it is assumed that the response is the banner or login or prompt from the second device. At this point control is handed over to the script of the device-type of the second device to process authentication.

It is possible that the connection is established and an error message is returned that is not trapped for. In this scenario CatTools thinks that it is on the second device and starts performing the activity as scheduled. If the operating systems are similar enough the commands may still be valid and the instructions are processed on the wrong device.

For this reason, SolarWinds strongly recommends that you thoroughly test all Session connections before putting them into production and that you monitor them closely.

SSH2 for Cisco and Netscreen

Many Cisco IOS devices support SSH2 version 12 and later. Netscreen devices from version 5 and later of the IOS also support SSH2.

To enable CatTools to connect to a Netscreen device use a variant of the SSH2 protocol called `SSH2-notty`.

This is in relation to the pseudo terminal that is used in `*nix` when using SSH connections to a command processor, but may not be required by network devices.

Add a Cisco VPN device

When adding a Cisco VPN device in CatTools, SolarWinds highly recommends you first attempt setting your device up as device type `Cisco.VPN`. The `Cisco.VPN` device type is more intelligent in determining which commands it needs to send to navigate the menu driven system, which the Cisco VPN devices use. By using the `Cisco.VPN` device type, it can handle any differences within the number systems used for accessing each menu, sub menu or menu item.

If you continue to have issues when running a configuration backup of a Cisco VPN, contact us using the [Technical Support](#) form.

Differences between Cisco VPN Versions

For reference, Cisco VPN devices normally have a menu interface that CatTools accesses by sending a string of menu item numbers that invokes the display of the config.


- Cisco 3002 use 2.5.1 as the menu sequence.

 The device type `Cisco.Older.VPN3002` uses this by default.

- Cisco VPN IOS versions prior to 4.0 use 2.6.3 as the menu sequence.

 The device type `Cisco.Other.VPN3000` uses this string by default.


- Cisco VPN IOS 4.1, uses 2.8.3 as the menu sequence.

 The device type `Cisco.V4.1.VPN3000` uses this by default.

- The intermediate Cisco VPN IOS version 4.0 2.7.3 as the menu sequence.

 None of the other three Cisco VPN device types outlined previously supports 2.7.3.

The `Cisco.VPN` device type should be able to handle any of the above command variations, so always try this device type first. If the `Cisco.VPN` device type is not able to backup your device, try another VPN type. If none of the VPN3000 device types backup your device, use the alternate command field of the backup activity and enter 2.7.3.

 If you use an alternate command, it overrides the default commands for all selected devices in the activity.


Cisco WAAS

In CatTools, you can create a process to enable the backup of the database from a Cisco Wide Area Application Service (WAAS). The general procedure used may be modified for other devices which also create a local backup file which then needs to be transferred to the host machine.

The process involves running two external scripts from the `Device.CLI.Send commands` activity:


1. The first script creates the backup file and stores the file name in a scripting dictionary for use later.
2. The second script issues commands to use file transfer protocol (FTP) to transfer the file to the local machine. It relies on the host machine running its own FTP server.

These two scripts can be found in [this downloadable zip folder](#). The files should be copied into the UserScripts folder in the main CatTools folder.

 You may need to create the /UserScripts folder in the \Program Files (x86)\CatTools3 folder.

1. Add Cisco WAAS to CatTools as a new device using the `Cisco.Switch.IOS` script.
2. Create a new `Device.CLI.Send commands` activity.
3. Add the following lines to the list of commands on the Options panel:


```
%ctRunExternalScript("C:\Program
Files\CatTools3\UserScripts\WAASDumpName.txt","cms database backup")
%ctRunExternalScript("C:\Program
Files\CatTools3\UserScripts\FTPCopyWAASDump.txt","<ftpHostname>
192.168.1.10 <ftpUsername>
admin_backup <ftpPassword> mypassword <ftpFileDirectory> backup.dump")
```

 Some items, such as the user name and password, need to be changed to the correct values for your FTP server. The scripts themselves are well commented and form the basis for further user customization scripts.

Backup a device via SFTP or SCP

Your device or network security policies may require transferring of files securely over secure file transfer protocols (SFTP) or over secure copy protocols (SCP). Although CatTools does not provide a specific activity to do this (similar to the [Device.Backup.TFTP](#) activity), you may still be able to transfer using the [Device.CLI.Send commands](#) activity.

If you require an SFTP/SCP Server, you can download the [SolarWinds SFTP/SCP Server](#) from the SolarWinds website. The SolarWinds SFTP/SCP server runs as a service, but some basic configuration may be necessary to ensure the SFTP/SCP server behaves in a way that works best within your environment.

- 
- An SFTP/SCP Server is not the same as a file transfer protocol (FTP) server. SFTP/SCP and FTP are different protocols. You cannot connect to a SFTP/SCP server with an FTP client.
 - Not all CatTools device type scripts support SFTP/SCP transfers via the `Device.CLI.Send commands` activity. Some devices prompt with verification messages, for example "Address or name of remote host"; "Destination filename"; "Destination username", which may need device script updates to handle. If you find your device does not work, contact the SolarWinds [Technical Support Team](#).

Unsupported devices or device activities in Kiwi CatTools

If your device is not included on the list of CatTools supported device types, and is not covered by one of the CatTools generic scripts, don't worry. Support for additional device types are being added to Kiwi CatTools all the time.

For an up to date list of the devices types currently available in CatTools and the activities supported for each of the device types, see the [device matrix](#).

If the device type required to support your device is not listed in the matrix, review how to request support for your device, and familiarize yourself with the details SolarWinds requires in order to look into adding support for your device. You can also inquire via [thwack](#), the SolarWinds online community site, for further information on your device.

Get support for devices added to CatTools

As of Kiwi CatTools 3.3.2, there is now a facility to create your own custom device scripts. If there is an immediate need to backup the configuration of a device currently not supported by CatTools, you may be able to add support for the device yourself rather than waiting for it to be added in a future CatTools release.

For more information, see [Create a custom device](#).

Requesting support for your device

If you would like us to consider adding your device then please [inquire on thwack](#) to let us know your requirements. When considering what devices to add we look at the following factors:

- **If SolarWinds can get network access to a device.**

This is by far the fastest way to add a new device. If you can send support a user name and password for your device, and support can log into it from their labs, then adding support for it is so much easier.

- **How many requests SolarWinds has for support of that device type.**

Even with limited resources and time, SolarWinds strives to help as many people as possible. A device with a large number of unique requests helps SolarWinds identify those devices that the majority of CatTools users desire support for.

- **How much information SolarWinds has on the device.**

If the support team cannot gain network access to a device, the next best thing is to see how the device works by examining logs captured from it. It is recommend you capture logs using a tool such as [PuTTY](#).

The more information about the device you can provide SolarWinds support with the better.

Sample device responses to report when requesting device support

The following are examples of the sort of responses we need to see from your device. If the action is not obvious, such as a `Ctrl+Z` keystroke, then this should be explained:

- A successful login
- A failed login. Showing responses to bad user name and password entries
- An invalid command being entered
- An incomplete command being entered
- Successfully entering and exiting enable/privileged mode
- Successfully entering and exiting configuration mode
- How to save changes to the configuration, such as write memory or copy running startup
- How to display the configuration (both running and startup)
- A show version type command
- The necessary commands to show Port / MAC / ARP information
- Results of a Ping command. Success, failure, and unreachable.
- Commands to enable and disable paging
- A `--More--` prompt (or equivalent paging prompt)
- A Yes / No prompt (or equivalent confirmation prompt)

Requesting support for a device activity

If there is a listing to support your device, but the activity you require is not currently supported by the device type, please [contact us via thwack](#), providing us with as much detail as you can and, if possible, include a PuTTY capture of the relevant command(s).

Activities

Kiwi CatTools Activities enable you to perform automated actions against your network devices monitored by CatTools. An Activity is an internal task that CatTools performs against one or many of your network devices that you have entered into the CatTools database. Activities can be scheduled to run at specific times by configuring the activity schedule, or they can be run immediately by using the Run Now button on the Activities pane.

The configured launches a client to go out and query your devices and obtain the information needed to complete the activity's task. The activity then collects all of the information returned and acts upon it, either storing it for future use or immediately analyzing the data. Finally, the activity reports the results to you via email notification.

See the following topics for more information:

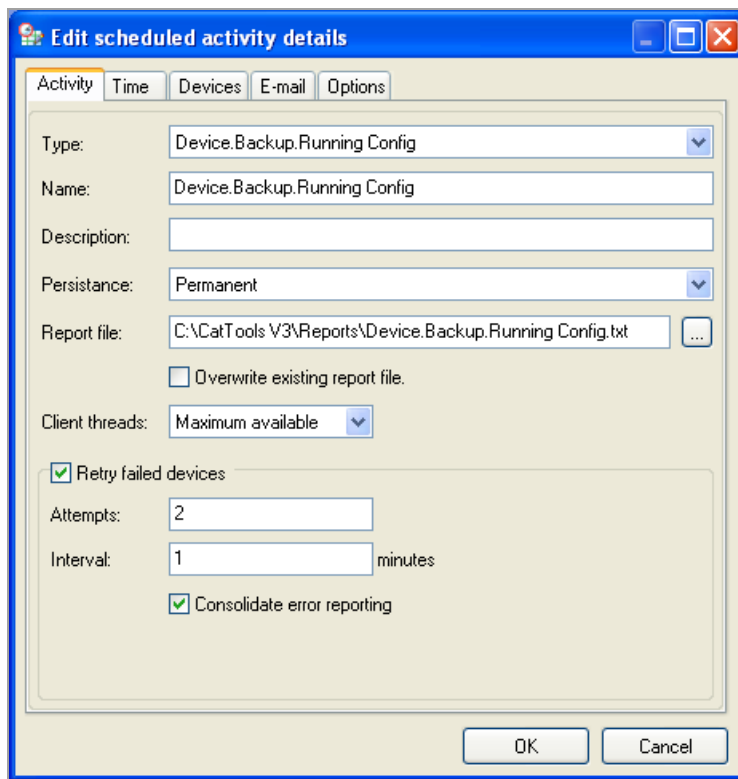
- [Add and edit scheduled activity details](#)
- [Activities list](#)
- [Internal functions](#)
- [Create a custom activity](#)
- [Unsupported activities for a device](#)

Add or edit activity details in Kiwi CatTools

This section provides an overview of the interface used to add and edit an activity.

Activity Tab

The Activity tab is where you define what sort of activity you are creating as well as its core details.



Type The [type of Activity](#) you wish to set up.

Persistence Toggles the lifetime of the activity. Activities can be permanent, run once and deactivated, or run once and deleted from activities.

Report file The path to a text file detailing the run history of the activity.

Overwrite existing report file Indicates the report file is overwritten each time the activity runs.
Leaving the checkbox unselected means the report file is appended when the activity runs.

Client threads The number of simultaneous activities for load balancing purposes. This can also be thought of as the maximum number of concurrent devices that CatTools talks to at any time. This setting is only valid for activities that use clients to go out and talk to devices. The more threads that are running, the faster CatTools completes the activity.

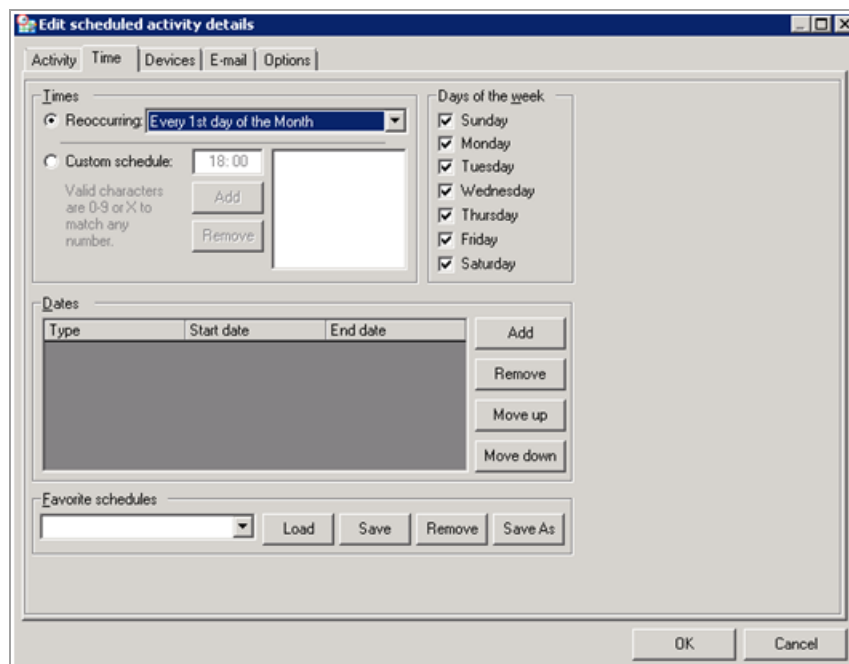
 The free edition of Kiwi CatTools limits the maximum number of threads to 1.

Retry failed devices Retry running the activity against any devices that may have failed in the course of the activity. The activity pauses for an interval of x minutes between each attempted retry. During this time the activity is still classed as running / busy.

Attempts	The number of times a activity retry attempt is made against a failed device. Valid range is 1 to 10 attempts.
Interval	The number of minutes the activity pauses between each retry. Valid range is 1 to 60 minutes.
Consolidate error reporting	<p>Specify how Level 1 errors are reported. This option is beneficial if the priority is whether an error occurred, not necessarily what the specific error is.</p> <p>For example, you are backing up three different devices and one of the devices rebooted and fails to connect on the first attempt. The device connection is successful on the second attempt, however your primary concern is determining that a backup of the device was obtained. You are not concerned about the initial failure since you know it was caused by a reboot. In this example, you want to have this option ticked. If do you want to see all errors that occurred on all retry attempts, then you should uncheck this option.</p>

Time tab

The Time tab lets you set up the scheduling of the activity. You may select a Reoccurring schedule, or one that runs the activity at specific times.



Reoccurring Opt to run your activity during a the pre-configured time.

- If you select **Now** the activity runs immediately upon saving.
- If you select **Never** the activity must be manually run from the activities pane.

Days of the week	The days of the week on which the activity runs.
Dates	<p>Specific dates to start and end the running of an activity.</p> <p>For example, your team is completing the setup of 10 new devices on January 15 of this year. You want to backup the configuration of all 10 devices, at one time, the following day. To do so, you can schedule CatTools to automatically run the backup configuration activity on January 16. The activity runs only on that specific date and does not run again unless manually activated.</p>
Custom schedule	<p>Custom schedule dates and times can be added, removed, or re-ordered.</p> <p>Exercise care when using this option:</p> <ul style="list-style-type: none"> • If an activity does not start running after you have configured it, check that the current date on your system is not before the Start date you have set. • If an activity stops running, check that the current date on your system is past the End date you have set.
Favorite schedules	<p>You can set up Favorite schedules that can be applied to different activities.</p> <p>Once you have created a schedule you can save it using the Save As button, and re-use this schedule for multiple activities by Loading it as required.</p>
Monthly	<p>Two options for monthly scheduling:</p> <ul style="list-style-type: none"> • First day of the month • Every first selected day of the month

For example, you need to setup the configuration backup activity to automatically run on the 1st of February.

1. Under Times, select Every 1st day of the Month for the reoccurrence.

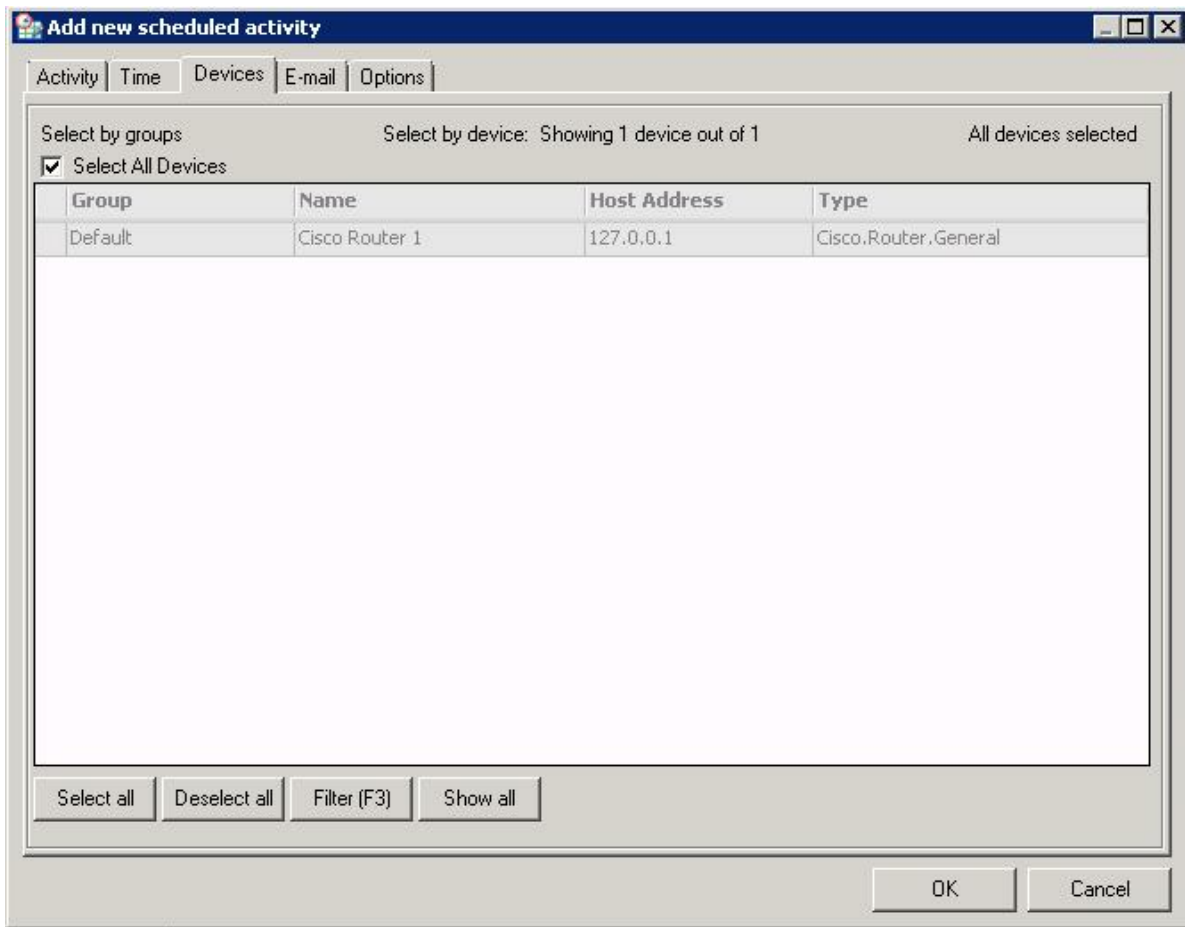
This specifies that the activity is only run on the first day of the month.

2. For Days of the week, select the checkbox next to each day of the week.

Since the date February 1 can fall on any day of the week, since the year is not specified, it is important that every day of the week is selected. If you do not select, for example, Sunday, and one year February 1 falls on a Sunday, the activity will not run.

Devices tab

From the Devices tab, you can associate your devices with the configured activity. Each device has a tick box to indicate which devices are currently associated with this activity.



There are a number of methods you can use to select devices:

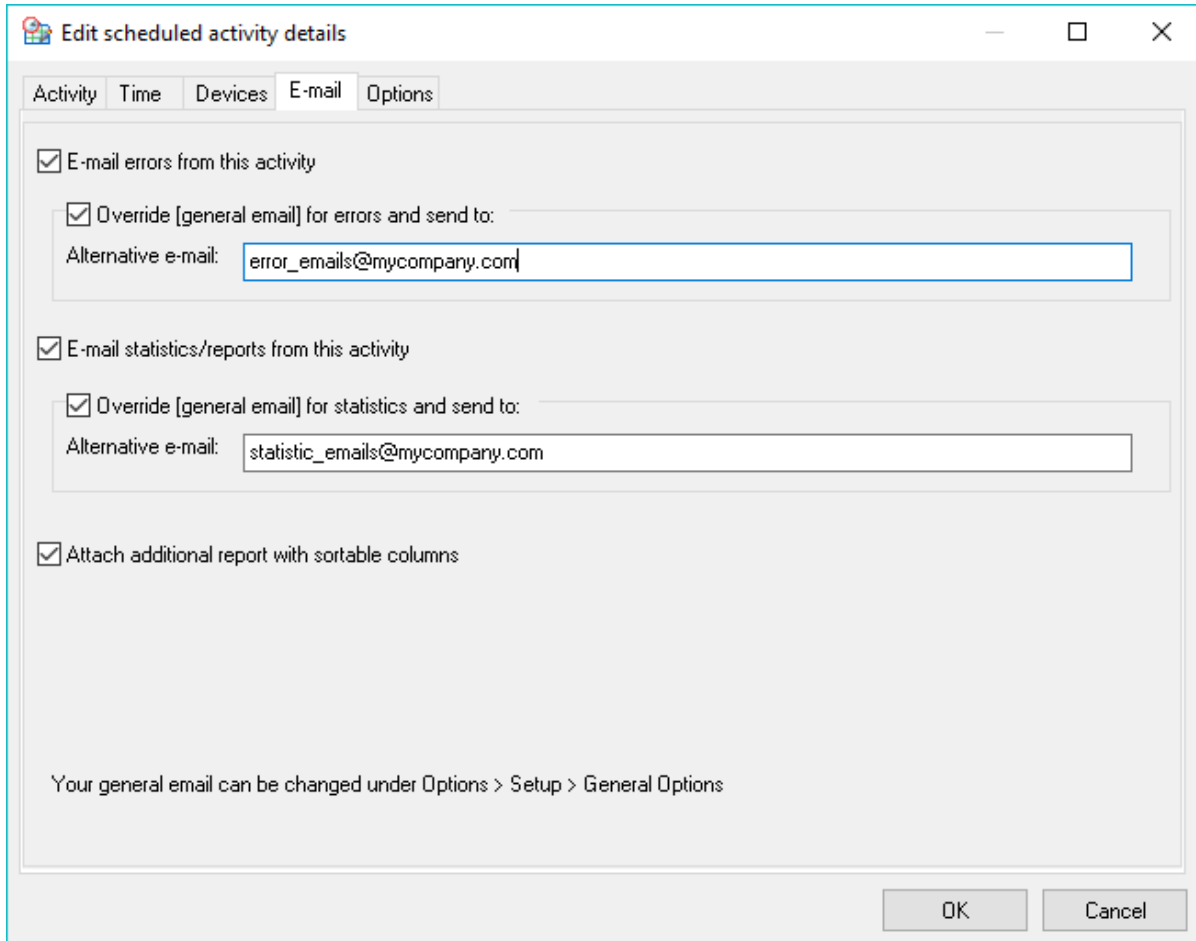
- Click on the check box for each device.
- Double click on the row containing the device.
- Select a number of devices using standard Windows list selection methods, and then right click to show a context menu. Selecting the Enable selected option checks all devices that are highlighted.
- Use the Filter method to display a group of devices, such as all devices whose group contains "Head Office". When the selected devices are displayed, click the Select All button to check all the displayed devices.
- The [Select All Devices](#) option allows all devices within the database (both current and future devices) to be added to this activity.

i Note on Filter and Select All options:

- **Filter**- Opens the "Database Filter" window, so you can filter the display of devices. The text entered in the Contains field is case sensitive. For more information, see [Filter devices in the Kiwi CatTools database](#).
- **Show all**- Disables any active filters and shows all devices in the database.

Email tab

The E-mail tab allows you to select notification alerts for the current activity.



There are two types of email notifications: errors and reports. By default, both email types are sent to the recipient email address set in the general [CatTools Setup](#). You can override this setting by selecting an Alternative e-mail address.

Error emails	Sent when a level 1 critical error is triggered at any point in the activity by any one of the devices the activity is being run against. To receive this notification, select "Email errors from this activity".
Report emails	Sent at the end of the activity and summarizes the results of the activity. An optional HTML formatted report can also be attached to the email, by checking the Attach additional report with sortable columns check box. To receive this notification, select "Email statistics / reports from this activity".

Options tab

The Options tab is specific to the activity type and the available options vary depending on the activity being configured. Refer to the specific activity type for more details.

- [DB.UpdateDevice.Password field](#)
- [DB.UpdateDevice.Text field](#)
- [Device.Backup.Running Config](#)
- [Device.CLI.Modify Config](#)
- [Device.CLI.Send commands](#)
- [Device.ConnectivityTest.Login](#)
- [Device.ConnectivityTest.Ping](#)
- [Device.InterDevice.Ping](#)
- [Device.TFTP.Upload Config](#)
- [Device.Update.Banner](#)
- [Device.Update.Password](#)
- [Report.ARP table](#)
- [Report.CDP Neighbors table](#)
- [Report.Compare.Running Startup](#)
- [Report.Compare.Two files](#)
- [Report.Error info table](#)
- [Report.MAC address table](#)
- [Report.Port info table](#)
- [Report.SNMP.System summary](#)
- [Report.Version table](#)
- [Report.X-Ref.Port MAC ARP](#)

Activities types

The sections below include the type and description of all activities that can be selected in CatTools.

Database Activities

[DB.Backup.CatTools](#)

Creates a backup copy of the CatTools database file and save it to a specified location. This activity allows you to automate the backing up of your CatTools database.

It does not run against your physical network devices as most of the other activities do. Instead it takes a backup of your existing CatTools database and makes a backup to the specified location. It also tries to squeeze the backup after completing the copy.

[DB.UpdateDevice.Password field](#)

Updates all passwords contained in the CatTools database device password field. This activity allows you to maintain your passwords in the CatTools database device password fields.

It does not run against your physical network devices as all the other Activities do but is instead used to do batch processing against multiple devices within the CatTools database Device table.

For example, your network devices use a RADIUS or TACACS server for remote authentication. You have just changed the password on the RADIUS/TACACS server and you now need to update all of your devices within CatTools with this new information. Instead of editing each device individually you would create a new `DB.UpdateDevice.Password` field activity which applies the new password to all of your Devices at once.

[DB.UpdateDevice.Text field](#)

Updates the CatTools database device text field with specified values. This activity allows you to update any of the text fields in the CatTools database Device table.

It does not run against your physical network devices as all the other Activities do but is instead used to do batch processing against multiple devices within the CatTools database Device table.

For example, your network devices use a RADIUS or TACACS server for remote authentication. You have just changed the user name on the RADIUS/TACACS server and you now need to update all of your devices within CatTools with this new information. Instead of editing each device individually you would create a new `DB.UpdateDevice.Text` field activity which applies the new user name to all of your Devices at once.

Device Activities


[Device.Backup.Running Config](#)

Captures and saves the current running configuration of selected devices to a specified location.

This activity makes a backup of a device's running configuration and compares it to the current config file on disk. If there are differences, the current config is moved to the "Dated Configs" folder and the file name is appended with the current date. The newly downloaded config then becomes the current config. An HTML "diff" report is created in the "Reports" folder and a copy is e-mailed to the nominated person.

Device.Backup.TFTP

Transfers files from a device to a TFTP server with an option to compare the content with the current existing file on a disk. This activity is designed replicate the `Device.Backup.Running config` activity via TFTP, however it is not limited to the device-config. You can specify other files, such as `VLAN.dat`. You also have the option whether to do the compare. Some files may be in binary format and so can't be compared. You can also specify your own command set to TFTP the file.

 SolarWinds recommends using the `Device.Backup.Running config` activity to backup your config files. The `Device.Backup.TFTP` activity may serve as a useful alternative in certain circumstances.

Device.CLI.Modify Config

Modifies the configuration of selected devices by sending CLI commands. This activity allows you to modify the configuration of device or a selection of devices.

CatTools logs in to each selected device in turn, execute the configuration changes you describe, and will log any failures to the report file: `...\Reports\Device.CLI.Modify Config.txt`

Whereas the `Device.CLI.Send commands` activity does this from the normal command line prompt (or Enable mode if that option is checked) the `Device.CLI.Modify Config` activity executes commands in the Config mode (Config terminal).

Device.CLI.Send commands

The `Device.CLI.Send commands` activity allows you to issue a series of CLI commands to the selected devices as if you were typing them from the normal command line prompt, or Enable mode, if that option is checked.

Device.ConnectivityTest.Login

The `Device.ConnectivityTest.Login` activity allows you to log in to a selection of devices. This activity is similar to [Device.ConnectivityTest.Ping](#) in that it does not change anything on the device but simply confirms connectivity plus the ability to log in.

Device.ConnectivityTest.Ping

The `Device.ConnectivityTest.Ping` activity allows you to ping a selection of devices. CatTools pings each selected device in turn, and logs any failures to the Report file `\Reports\Device.ConnectivityTest.Ping.txt`

The ping test is a simple Internet Control Message Protocol (ICMP) Echo request, sent from the machine running CatTools to the selected device(s). It tests the operation of your network between CatTools and the selected device, but only at the IP level. This activity is a useful test of your network, or your users' ability to use the network, depending on your specific network design. The activity report file provides a historical record of the status of the tested devices.

[Device.InterDevice.Ping](#)

The Device.InterDevice.Ping activity allows you to ping a list of hosts from the selected devices. CatTools logs into each device in turn and performs a ping test to each device listed. Alternatively, it can ping multiple selected devices.

CatTools creates a report in the form of a table, listing each device and the results of each ping originated by that device. Under the Options tab, you can also check the option to have CatTools report only on failed ping requests. This keeps the report as small as possible and draws attention only to fault conditions.

[Device.TFTP.Upload Config](#)

The Device.TFTP.Upload Config activity enables you to upload text config files to the specified device.

[Device.Update.Banner](#)

The Device.Update.Banner applies a banner to your devices. CatTools passes the banner command to each selected Device in turn. This activity supports the use of [Command Variables](#).

[Device.Update.Password](#)

The Device.Update.Password activity enables you to change some of the passwords on devices.

A report gives you the status of each device selected for the activity after the activity is completed. The report shows both the new and old passwords. These passwords are shown by default in plain text on the report. You may elect to hide the reported passwords by selecting the [Hide password change report passwords](#) option in the Misc tab in the CatTools Setup dialogue.

Report Activities

[Report.ARP table](#)

The Report.ARP table activity builds a report using the ARP table from selected devices as source data. On a Cisco IOS device it uses a simple "Show IP ARP" command and builds a table of results over time.

This activity automatically indexes MAC addresses against IP addresses and device interfaces, and then resolves their host names via DNS, if required. By default, the table is updated after each run of the activity, providing a historical record of the devices attached to your network over time. Each entry is time stamped, and "First Seen" and "Last Seen" columns included in the report.

[Report.CDP Neighbors table](#)

The Report.CDP.Neighbors table activity produces a report of all networked devices that can be seen by the specified device using the CDP neighbor command.

[Report.Compare.Running Startup](#)

The Report.Compare.Running Startup activity compares the running and the startup configs of your device(s) and reports on the differences found.

The activity works by first connecting to your selected devices, then issuing the commands which show the running and startup configs. These configs are saved into the `ClientTemp` folder while the activity is running. When the configs have been retrieved from the devices, a compare is run against the two captures and a report is generated.

[Report.Compare.Two files](#)

The Report.Compare.Two files activity runs a compare against two files which you define and reports on the differences it finds.

[Report.Error info table](#)

The Report.Error info table activity collects and reports error counter information for all interfaces of each selected device.

[Report.MAC address table](#)

The Report.MAC address table activity gathers MAC address entries (CAM table entries) from switches and bridging routers and can build a table over time.

CatTools gathers information from every command that succeeds and ignores any command that does not produce a valid result.

[Report.Port info table](#)

The Report.Port info table activity creates a snapshot of the interface configuration and state of the selected devices and stores the data in a tab delimited report file. The data is stored in a text file. The information gathered includes name, VLAN, port type (Ethernet, Fast Ethernet, Full Duplex, etc), status, and speed.

[Report.SNMP.System summary](#)

The Report.SNMP.System summary activity produces a summary report of data available from any device that supports standard system SNMP.

[Report.Version table](#)

The Report.Version table activity creates a tab delimited report of all the software and hardware version information from the selected devices.

[Report.X-Ref.Port MAC ARP](#)

The Report.X-Ref.Port MAC ARP activity creates a cross reference report of MAC and IP addresses against ports on a network. If IP addresses have been resolved to host names on the ARP report, the host names are reported as well.

This report collects all port information from the Port report and matches MAC and ARP information against the ports. These report activities **must** be run before the cross reference report can be run. The report does not require any devices to be selected. It uses the devices selected for the port report as its starting point.

System Activities

[System.File.Delete](#)

The System.File.Delete activity allows you to automate the deletion of files. It is primarily designed to allow the management of the dated config files, although it can be run against any folder. The activity does not run against your physical network devices.


DB.Backup.CatTools

The **DB.Backup.CatTools** activity automates backing up your CatTools database file. It does not run against your physical network devices as most of the other Activities do. Instead it takes a copy of your existing CatTools database and makes a backup to the specified location. It also attempts to squeeze the backup after completing the copy.

Setting up the activity for DB.Backup.CatTools

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Specify the database backup location using the [Filename Variables](#) as part of the file name.

DB.UpdateDevice.Password field


The **DB.UpdateDevice.Password field** activity allows you to maintain your passwords in the CatTools database device password fields. It does not run against your physical network devices as all the other Activities do but is instead used to do batch processing against multiple devices within the CatTools database Device table.

For example, your network devices use a RADIUS or TACACS server for remote authentication. You have just changed the password on the RADIUS/TACACS server and you now need to update all of your devices within CatTools with this new information. Instead of editing each device individually you create a new **DB.UpdateDevice.Password** field activity which applies the new password to all of your Devices at once.

Setting up the activity DB.UpdateDevice.Password field


For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

VTY Password The CatTools database field value for the VTY Password field.

AAA Password The CatTools database field value for the AAA Password field.

 The AAA Password field in the CatTools database corresponds to the Password field on the [Passwords](#) tab of the Device setup form. This field can be used to store AAA / TACAC / RADIUS / Local authentication or SSH passwords.

Enable Password The CatTools database field value for the Enable Password field.

Console Password The CatTools database field value for the Console Password field.

SNMP Read Allows you to change the CatTools database field value for the SNMP Read field.

SNMP Write Allows you to change the CatTools database field value for the SNMP Write field.

Stop on error The activity immediately quits the job and returns control to CatTools if an error is encountered at any stage of processing.

DB.UpdateDevice.Text field


The **DB.UpdateDevice.Text field** activity allows you to update any of the text fields in the CatTools database Device table. It does not run against your physical network devices as other Activities do, but is used to do batch processing against multiple devices within the CatTools database Device table.


For example, your network devices use a RADIUS or TACACS server for remote authentication. You have changed the username on the RADIUS/TACACS server and now need to update all of your devices within CatTools with this new information. Instead of manually updating the username for each device individually, the DB.UpdateDevice.Text field activity applies the new username to all of your connected devices at once.

Setting up the activity for DB.UpdateDevice.Text field

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Database Field(s)	Select an item from the device details you wish to update in the database. Additional options can be configured based on dropdown menu selections.
	<div style="border: 1px solid #add8e6; padding: 5px;"> <p> The Username\Password database field option updates the AAA Username field in the CatTools database. The AAA Username database field corresponds to the Username field on the Passwords tab of the Device information/setup form. This field can be used to store AAA / TACAC / RADIUS / Local authentication or SSH usernames.</p> </div>
New database value	Enter the value for the above selected database field.
Stop on error	An error encountered at any stage of the processing the activity immediately quits the job and returns control to CatTools.

Device.Backup.Running Config

The **Device.Backup.Running Config** activity creates a backup of a device's running configuration and compare it to the current config file on disk. If there are differences, the current config is moved to the "Dated Configs" folder and the filename is appended with the current date. The newly downloaded config then becomes the current config. An HTML "diff" report is created in the "Reports" folder and a copy is e-mailed to the nominated person.


The Device.Backup.Running Config activity completes the following tasks:

1. A connection is made via SSH or telnet to the device.
2. A command, such as "Show running configuration", is issued.
3. The configurations are collected and stored on disk in a temporary file.
4. A check is made to see if there is an existing config file on disk for this device.
5. If there is an existing config file, the existing file and the new file are compared for differences.
 - a. A text and HTML report is made of any differences and stored in the "Reports" folder. A copy of the difference report is sent via e-mail.
 - b. The existing config file is moved into the "Dated configs" folder and appended with the current date.
 - c. The new config file is put in the "Configs" folder and becomes the current device backup.
6. If there is no existing config file, the generated config is placed in the "Configs" folder as the current device backup.


Setting up the activity for Device.Backup.Running Config

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Use alternate command The command issued to backup the running config depends on the device type. For example, a Cisco router is sent the command `Show running-config`. If you wanted to send a different command, enter it in this field.

 The command must be suitable for all the devices selected for this activity. If your alternate command is only suitable for a Cisco VPN, then make sure only Cisco VPN devices are selected under the devices tab.

Current config file Specifies the folder and file name of where to store the current device configuration. If an existing file exists a comparison is completed between the two files. If there are no changes, no files is modified.

The default location is

```
\Configs\%GroupName%\Config.Current.Running.%BaseFile%.txt
```

The variables `%GroupName%` and `%BaseFile%` are replaced at run time. For example:

```
C:\Program
files\Cattools\Configs\Default\Config.Current.Running.Sales_
Router.txt
```

Dated config file Specifies the folder and file name of where to store dated configuration files. The current file is moved to the dated config folder when changes are detected.

The default location is `\Dated`

```
Configs\%GroupName%\Config.Dated.Running.%BaseFile%.%DateISO%-%TimeHH
MM%.txt
```

The variables `%GroupName%`, `%BaseFile%`, `%DateISO%` and `%TimeHHMM%` are replaced at run time. For example: `C:\Program files\Cattools\Dated`

```
Configs\Default\Config.Dated.Running.Cisco805.20031006-1914.txt
```

See the section [Filename Variables](#) for a list of variables you can use.

HTML compare report Specifies the folder and file name of where to store the HTML differences report. This diff report file contains a side by side comparison of the old and new configuration. Changes are highlighted in different colors.

The default location is under

```
\Reports\ConfigChanges\%GroupName%\Config.Changes.Running.%BaseFile%.
%DateISO%-%TimeHHMM%.htm
```

The variables `%GroupName%`, `%BaseFile%`, `%DateISO%` and `%TimeHHMM%` are replaced at run time. For example: `C:\Program`

```
files\Cattools\Reports\ConfigChanges\Default\Config.Changes.Running.C
isco805.20031006-1914.htm
```

Text summary diff report Specifies the folder and file name of where to store the text based diff report. This diff report file contains a list of the changes found in the old and new configs. This report is a smaller and simpler way to view the changes. Unlike the HTML report, where both the old and new configs are shown in full. The text report only shows the lines that are different.

The default location is

```
\Reports\ConfigChanges\%GroupName%\Config.Changes.Running.%BaseFile%.%DateISO%-%TimeHHMM%.txt
```

The variables `%GroupName%`, `%BaseFile%`, `%DateISO%`, and `%TimeHHMM%` are replaced at run time. For example: `C:\Program`


```
files\Cattools\Reports\ConfigChanges\Default\Config.Changes.Running.Cisco805.20031006-1914.txt
```

Only notify by e-mail if configs have changed Run configuration backups as often as you want, but only receive an e-mail notification when changes are detected between two configuration files.

Attach reports to e-mail Allows you to choose what level of diff reports you are sent via e-mail:

- HTML compare report only (default)
- Text diff report only
- Both reports

If you want full comparison reports with highlighting, use the HTML compare report. If you just want the changes, use the text report.

 If you are sending the reports via an insecure medium, such as a corporate network or internet, be aware that the HTML reports contain your full device configuration. SolarWinds recommends that you send password protected HTML reports over an insecure medium.

Zip attachments To reduce network traffic and allow for password protection, you can choose to zip the attachments. This is the recommended and default option. It keeps the e-mail smaller and places all the files in a single attachment, which can be beneficial when you have, for example, 400 devices.

Password protect zip file Protect your configs and reports by adding a password to your zip file. A password protects your files over insecure mediums while in transit to your e-mail box. SolarWinds recommends using current password strength standards when creating a password for your files.

Why it is important to backup your device configurations?

The Device.Backup.Running Config activity is one of a number of default activities provided with the CatTools program. It is designed to copy the currently running configuration from the selected device into a text file on your local file system.

This is an important distinction. Kiwi CatTools copies the configuration parameters that are currently running on the device. Many devices have their own local storage, where one or more versions of their configuration files may be stored. Most network devices managed by CatTools have a start-up config file that they load and run when the power is switched on.

Because this running config can be changed in real-time by someone logged into the device, for example, via a telnet session, the running config may no longer be the same as the start-up config. Capturing these different versions of your device configurations is exactly the sort of thing that CatTools is designed to handle.

Common scenarios

Consider the following scenario: you have a night-shift engineer who encounters a problem on your network, hurriedly telnets into a router, and makes some changes to that device that restores service to your internal customers, but in his hurry forgets to document or save these changes.

If you power cycle that device at some later time, those changes can be lost, potentially recreating the problem. Only this time, it the problem occurs during your busiest period of the day. Now you have the same problem as before, but the fix is gone and the engineer is unavailable to consult. You may have to start the whole problem-resolution process from scratch, costing your organization time, money, and credibility.

How does backing up my configurations with Kiwi CatTools benefit my organization?

There are several ways that the activity might be useful in this reasonably common scenario:


1. It is best practice in change management to always have a roll-back contingency. Naturally, such a plan is of no benefit if you cannot recover and re-install old device configurations. Using Kiwi CatTools, you can regularly capture a restorable copy of the current configuration prior to making any changes.
2. CatTools contains a Diff Report function which runs as a part of the activity. This report compares the configuration that has just been downloaded to the previous copy stored on file. If any changes are detected then it creates a report in HTML format that highlights the lines that are different. The default setting is for this report to be emailed to you as soon as it is created, as well as being stored in the Reports directory.

In the previous example, if our hypothetical night-shift engineer had run a Device.Backup.Running Config before and after they had made changes to the router, then you have a report showing exactly what changes had been made. Additionally, you have two copies of the config in two text files: one before the changes and one after.

You could use either of these files to restore the router to a previous configuration, or you can use the Diff Report to manually identify and recreate the changes.

Device.Backup.TFTP

The **Device.Backup.TFTP** activity makes a backup of the specified file and optionally compare it to the current file on disk. If there are differences, the current file is moved to the "Dated Configs" folder and the filename is appended with the current date. The newly downloaded file then becomes the current file. A HTML differences report is created in the "Reports" folder and a copy is emailed to a specified email address.

 SolarWinds recommends using the [Device.Backup.Running config activity](#) to backup your config files. However, the Device.Backup.TFTP activity serves as a useful alternative in certain circumstances.

Although this activity is designed to replicate the Device.Backup.Running config activity using Trivial File Transfer Protocols (TFTP), it is not limited to device configurations. You can specify other files, such as `VLAN.dat`. You also have the option to generate a comparison report. Some files may be in binary format and therefore cannot be compared using Kiwi CatTools. Finally you can also specify custom commands to send to TFTP the file.

The Device.Backup.TFTP activity completes the following tasks:

1. A connection is made via SSH or telnet to the device.
2. Commands to use TFTP against the specified file is issued.
3. The file is collected and stored in the TFTP folder.
4. A check is made to see if there is an existing file on disk for this device.
5. If a current TFTP file exists, it is compared to the newly downloaded file.
 - a. A text and HTML report is made of any differences and stored in the "Reports" folder. A copy of the difference report is sent to a specified recipient via e-mail.
 - b. The current file is moved to the "Dated configs" folder and appended with the current date.
 - c. The newly downloaded file is put in the TFTP folder and becomes the current file backup.
6. If a current TFTP file does not exist, the newly downloaded file is placed in the TFTP folder as the current file backup.


Setting up the activity for Device.Backup.TFTP

For information on setting up activities, see [Add / Edit scheduled activity details](#).

This activity is available in the following device scripts:

AlliedTelesis.AlliedWare.Plus	Cisco.Router.General	Cisco.Switch.IOS	Cisco.Switch.CatOS
Cisco.Firewall.PIX	Cisco.Wireless.Lan	Fortinet.FortiOS.General	Generic.Device
HP.Switch.2500	NEC.Univerge.IX	Nortel.Switch.Ethernet	Nortel.Switch.NoCLI
Bluecoat.Cacheflow	Checkpoint.VPN	F5.BigIP.TMSH	PaloAlto.FireWall
SonicWall.SonicOS			


Activity options

 Configure the options specific to the **Device.Backup.TFTP** activity in the Options tab.

Optional alternate list of commands Removes the selection from the 'File to write to TFTP Server' checkbox. Some device scripts include a default set of commands that can be used to generate and TFTP the config for this activity. However, should there be no default commands for your device you can use this box to enter your own.

For example, you may enter the command:

```
copy running-config tftp:
10.190.2.98
%ctDeviceName-Running-Config
```

 When specifying the name of the file to be written, the name must include the `%ctDeviceName` variable. CatTools uses this variable to correctly identify the file for comparison.

Enter commands in enable mode CatTools enters enable mode before issuing any commands to a device.

Answer [yes] to any confirmation prompts Some issued commands have confirmation prompts, such as: "Are you sure? Y or N". CatTools automatically answers such prompts in the affirmative, on the assumption that if you entered the commands you want them to be executed. If this option is unchecked, then the activity will enter N(for "No") in response to any command confirmation prompts.

File to write to TFTP server Use the default list of commands, where available, to TFTP the file.

Enter the name of the file you want to TFTP in the text box, for example, `running-config`, `vlan.dat`, and so on.

Current file Specifies the folder and file name of where to store the current file. By default this is in the configs folder and uses `Config.Current.Running` as part of the file name. The most common use case for this activity is to backup the config by TFTP. This is also be the file that a new TFTP file is compared against.

The default location is

```
\Configs\%GroupName%\Config.Current.Running.%BaseFile%.txt
```

The variables `%GroupName%` and `%BaseFile%` are replaced at run time. For example:

```
C:\Program
files\Cattools\Configs\Default\Config.Current.Running.Sales_
Router.txt
```

This value is where any new configuration files are stored. If an existing file exists a comparison is done against the newly downloaded file. If there are no changes, no files are modified.

Compare, move to dated file Runs a comparison between the file downloaded and an existing file, as specified by the current file. You can specify the folder and the file name of the stored dated files. The current file is moved to the dated config folder if changes are detected.

The default location is `\Dated`

```
Configs\%GroupName%\Config.Dated.Running.%BaseFile%.%DateISO%-%Time
HHMM%.txt
```

The variables `%GroupName%`, `%BaseFile%`, `%DateISO%`, and `%TimeHHMM%` are replaced at run time. For example:

```
C:\Program files\Cattools\Dated
Configs\Default\Config.Dated.Running.Cisco805.20031006-1914.txt
```

See the section [Filename Variables](#) for a list of variables you can use.

Ignore Text If you expect a new file to repeatedly contain lines of text that are different from the original file every time the comparison is run, such as a timestamp, you can add the lines to disregard in Ignore Text. More information is available on the [Ignore Text](#) field.



To ignore "certificate self-signed" changes on Cisco routers, see the [block text ignore example](#).

Only notify by e-mail if configs have changed


Run configuration backups as often as you want, but only receive an e-mail notification when changes are detected.

Attach reports to e-mail

Choose what level of diff reports you are sent via e-mail:

- HTML compare report only (default)
- Text diff report only
- Both reports

If you want full comparison reports with highlighting, use the HTML compare report. If you just want the changes, use the text report.

 If you are sending the reports via an insecure medium, such as a corporate network or internet, be aware that the HTML reports contain your full device configuration. SolarWinds recommends that you send password protected HTML reports over an insecure medium.

Zip attachment s

To reduce network traffic and allow for password protection, you can choose to zip the attachments. This is the recommended and default option. It keeps the e-mail smaller and places all the files in a single attachment, which can be beneficial when you have, for example, 400 devices.

Device.CLI.Modify Config

The **Device.CLI.Modify Config** activity modifies the configuration of device or a selection of devices. CatTools logs in to each selected device in turn, executes the configuration changes, and logs any failures to the report file: `...\Reports\Device.CLI.Modify Config.txt`

Whereas the [Device.CLI.Send commands](#) activity does this from the normal command line prompt, or Enable mode if that option is checked, this activity executes commands in the Config mode from the Config terminal.

A simple way to familiarize yourself with this activity is to do a change of interface description, using the commands in the example below:


```
int e0
description this is a test
```

Copy the above sample command and enter a different description of your choice. Run the commands in sequence and note the description change from one to the other. You can also combine the example with a [Device.Backup.Running Config](#). You can see the changes in the config files, the email reports it generates, and so on.

Setting up the activity for Device.CLI.Modify Config

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options


 Configure the options specific to this activity in the Options tab.

Config commands to be entered into the device (Max 10,000 characters)

Enter commands to change the configuration of selected devices.

For example, for a Cisco IOS router, add the commands that you manually enter in config mode after entering "config term" at the Enable prompt. If you want to change the VTY login password for the Cisco IOS router, enter the following commands in this field:

```
Line vty 0 4
password mynewpassword
```

 CatTools enters config mode and executes the commands exactly as written in this field. It is vital to write them exactly as if you were configuring the device manually via a telnet session. **Each command must be entered on a separate line.** If you enter a blank line between commands, this is applied just as if you pressed enter in a telnet session. The maximum limit is 10,000 characters.

Within the commands, you can use a number of variables that are replaced by actual values when the commands are presented to the device. See the section [Command Variables](#) for a list of variables you can use.

You may also enter meta commands in the list of commands. For more information, see [Meta Commands](#).

Or, read commands from file	<p>Reads commands from the specified text file entered.</p> <p>This option and "Config commands to be entered into the device" are mutually exclusive. If you select this option, the config commands field is deactivated, and the commands are only read from the text file.</p> <p>The rules for entering commands in the text file are the same as the config commands field. Every line is read as a command to be entered in the devices config mode. A blank line between commands is read as an <Enter>. However, the text file does not have a limit on the number of characters allowed.</p>
Save device output to file	<p>A device-specific record of each time the activity runs. All output from the device is recorded to a text file, exactly as it would appear in a telnet session. The file name is based on the device name and the date. Each device creates its own unique output file based on the %DeviceName% and %DateISO% variables.</p>
Overwrite existing capture file	<p>Overwrites the file created by the output from the last run of the Device.CLI.Modify Config activity.</p> <p>If unselected, the existing file is appended. A new file is created only when the activity's run date is different.</p>
Answer [yes] to any confirmation prompts	<p>Some issued commands have confirmation prompts, such as: "Are you sure? Y or N". CatTools automatically answers such prompts in the affirmative, on the assumption that if you entered the commands you want them to be executed.</p> <p>If this option is unchecked, then the activity will enter N(for "No") in response to any command confirmation prompts.</p>
Stop on error	<p>The activity immediately quits the job and returns control to CatTools if an error is encountered at any stage of processing.</p>
Save running config to start-up config when complete	<p>Copies the running configuration on the device to non-volatile memory, so that the modified config becomes the start-up config.</p>

When to use the Device.CLI.Modify Config activity in Kiwi CatTools

This activity is a useful network management tool for several reasons:

- Allows precise timing control over your network.
- Enables automation of these controls, yet still use exactly the same commands as you would enter if you did the task manually.

The benefit is that you learn and use the same command line instructions in both cases, and the CatTools program itself is very simple to learn to use in quite sophisticated ways because it simply applies those commands on your behalf, just as you would via telnet.

Task scheduling

The time controls are perhaps the most useful aspect of this activity. With different combinations of scheduled activities you can tune your network to the changing needs of your organization over the course of a typical business day or week.

CatTools is designed to manage the repetitive scheduled configuration tasks on your behalf. It removes the possibility of human error from such tasks. It provides you with a much greater confidence in the currency of your backup processes, and in your ability to roll back to a previous state using your historical backups. It provides for active management of your network and its condition, by enabling the remote execution of command line instructions across your network, at times determined by you.

Scheduled config changes can also be used to add or remove access-list entries at specified times, such as allowing access to the payroll system during business hours only.

Example

If you have some switch ports that are wired to a shared conference room that is only in use during the day, you could schedule a config change to shutdown the specific interfaces at 6 p.m. and then another activity to enable the interfaces at 7 a.m. This will ensure that unauthorized staff can not hack into the network from the shared conference room after hours.

How to enter different commands onto each device for Device.CLI.Modify Config

If you want to configure each of the selected devices in a different way, there are two ways to achieve this:

1. Create a separate Device.CLI.Modify Config activity for each device and set the commands to be entered in the options tab.

This method involves having a separate scheduled activity for each device you want to modify.

2. Create a single Device.CLI.Modify Config activity and select all the devices you want to modify.
 - a. Select `Or, read commands from file` option.
 - b. Specify the `%BaseFile%` value as part of the file name.
 - c. Create a separate text file for each device that contains the commands you want to have entered into the config.

Example

1. Specify a file name: `C:\CatTools\%BaseFile%.txt`
2. Select a device for the `%BaseFile%` variable. In this example, we use `CiscoRouter1`. A list of other devices and their respective base filename include:

Device name	Base filename	Config commands filename
CiscoRouter1	CiscoRouter1	C:\CatTools\CiscoRouter1.txt
CiscoRouter2	CiscoRouter2	C:\CatTools\CiscoRouter1.txt
Sales Switch	Sales_Switch	C:\CatTools\Sales_Switch.txt

i Any spaces and invalid characters are replaced by an underscore (`_`) in the base filename.

3. In the text file, enter a list of commands. For example, in the file `C:\CatTools\CiscoRouter1.txt`, enter the commands:

```
Interface s 0/0
Description New WAN link to head office
No shutdown
```

4. You can create a text file for multiple devices. Enter different commands in each text file and then run the activity. CatTools automatically matches the correct text file with the device based on the `%BaseFile%` value.

Device.CLI.Send commands

The Device.CLI.Send commands activity allows you to issue a series of CLI commands to the selected devices as if you were typing them from the normal command line prompt, or Enable mode, if that option is checked.

The Device.CLI.Send commands activity automatically performs the following actions:

1. Log in to the device.
2. Handle any terminal display window length issues.
3. Enter enable or privileged mode, if optioned.
4. Enter specified commands in sequence, and wait for a valid response.
5. Log each command entered and the device response to a text log file.


Should an error occur, the activity aborts, if optioned.

6. Disconnect from the device.

Setting up the activity for Device.CLI.Send commands

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

List of commands to be entered into the device (Max 10,000 chrs)

Enter commands to change the configuration of selected devices.


For example, some CLI commands that can be entered are:

On a Cisco router:

```
Clear counters
Clear interface
```

On a Cisco Catalyst CatOS switch:

```
Set Port 1/1 disable
Set Port 2/12-24 enable
```

 **Each command must be entered on a separate line.** If you enter a blank line between commands, this is applied just as if you pressed enter in a telnet session. The maximum limit is 10,000 characters.

You may also enter meta commands in the list of commands. For more information, see [Meta Commands](#).

Or, read commands from file

Reads commands from the specified text file entered.

This option and "Config commands to be entered into the device" are mutually exclusive. If you select this option, the config commands field is deactivated, and the commands are only read from the text file.

The rules for entering commands in the text file are the same as the config commands field. Every line is read as a command to be entered in the devices config mode. A blank line between commands is read as an <Enter>. However, the text file does not have a limit on the number of characters allowed.

Enter commands in enable mode	CatTools enters enable mode before entering any of the commands to the device.
Save device output to file	Text file containing a device-specific record of each time the activity runs. All output from the device is recorded to a text file, exactly as it would appear in a telnet session. The file name is based on the device name and the date. Each device creates its own unique output file based on the <code>%DeviceName%</code> and <code>%DateISO%</code> variables.
Overwrite existing capture file	Overwrites the file created by the output from the last run of the <code>Device.CLI.Send commands</code> activity. If unselected, the existing file is appended. A new file is created only when the activity's run date is different.
Answer [yes] to any confirmation prompts	Some issued commands have confirmation prompts, such as: "Are you sure? Y or N". CatTools automatically answers such prompts in the affirmative, on the assumption that if you entered the commands you want them to be executed. If this option is unchecked, then the activity will enter N(for "No") in response to any command confirmation prompts.
Stop on error	The activity immediately quits the job and returns control to CatTools if an error is encountered at any stage of processing.
Send output via e-mail	Sends a report of activity results to a designated email address. The report can be sent as either an attachment or included in the body of the email.
Zip attachments	To reduce network traffic and allow for password protection, you can choose to zip the attachments. This is the recommended and default option. It keeps the e-mail smaller and places all the files in a single attachment, which can be beneficial when you have, for example, 400 devices.
Password protect zip file	Protect your configs and reports by adding a password to your zip file. A password protects your files over insecure mediums while in transit to your e-mail box. SolarWinds recommends using current password strength standards when creating a password for your files.

Run external scripts in Kiwi CatTools

It is possible to run an external VB script from within the [Device.CLI.Send commands](#) activity and use the return value from the external script as part of a CLI command to be executed. When creating external scripts for use within CatTools, you will need to edit the script files using a [script editor](#).

A number of useful [Functions](#) and [Variables](#) are exposed for use by external scripts.

Supported device scripts

- Cisco.Switch.IOS
- Cisco.Router.General
- Cisco.Other.CUE
- Cisco.Wireless.LAN
- Cisco.WAE
- Aruba.ArubaOS.General

Usage

To run an external script from within the [Device.CLI.Send commands](#) activity, commands should be entered in the following format:

```
%ctRunExternalScript("PathToScript", ["VariableString"])
```

- `PathToScript` is the full path to the script to be run. It must be wrapped in quotation marks as shown.
- `VariableString` is a string containing values to pass to the external script. It must be wrapped in quotation marks as shown, but cannot contain internal quotes. `VariableString` is optional as the script may not require variables.

You can pass more than one variable to the script, within the confines of the variable string. For example you could pass "255|127|1|2" as your variable string and use the external script to parse these into individual number values.

Only one `%ctRunExternalScript` command can exist per line.

External script parameters

- The external script must include `function Main`, which is the script's entry point. Function `Main` must accept a string parameter:

```
Function Main(VariableString)
```
- The script always returns a value, even if the value is: `Null`
- The script may parse `VariableString` and separate it into individual variables for use within the script.
- The script has these [Functions](#) and [Variables](#) exposed to it.
- The script stores values for use later using the functionality exposed by a scripting dictionary.

Scenario 1: Delete files from TFTP and run config

A script is run which deletes all files from the TFTP folder before the running config is copied into it. The external script does not require any parameters and returns a `null` value. The next command `copy running tftp` is then executed.

```
%ctRunExternalScript("C:\DeleteFiles.txt")
copy running tftp
192.168.1.200
%ctGroupName/%ctDeviceName-Config.txt
```

Scenario 2: Send command to identify and enable specific port

A script is run which returns a value of the ports to enable based on the value of `StringVariable`. The variable is 5 in this case, which represents the slot number.

```
Set Port %ctRunExternalScript("C:\PortToEnable.txt","5") enable
```

After execution of the script, the command processed by the [Device.CLI.Send commands](#) activity in CatTools is:

```
Set Port 5/1-12 enable
```

Where 5/1-12 is the returned value from the function. This value is inserted into the final command used.

Scenario 3: Change hostname

The hostname is changed to a new hostname. No variables are passed.

```
Conf term
hostname %ctRunExternalScript("C:\hostnames.txt")
wri mem
```

Sample External Scripts

The following example can be applied to Scenario 2 above. The function returns a value based on the input string.

```
Function Main(inputString)
    Dim SelectionValue
    SelectionValue = Val(inputString)
    Main = ""
    Select Case SelectionValue
    Case 1
        Main = "1/1-12"
    Case 2
        Main = "2/1-12"
```

```

Case 5
Main = "5/1-12"
End Select
End Function

```

The following script example can be applied to Scenario 3 above. Although the user is not passing an input string, CatTools passes a default null string. The `inputString` variable is still required in the function definition. The function itself is referencing `cl.DeviceHostnameId`, which is one of the variables made available to external scripts.

```

Function Main(inputString)
  Main=""
  If cl.DeviceHostnameId="MyRouter1" Then Main="MyNewRouterName1"
  If cl.DeviceHostnameId="MyRouter2" Then Main="MyNewRouterName2"
End Function

```

Example Generic Script

The script below is a more complete framework which you can use as a starting point for your external scripts. This script works in the following way:

1. Send a command to the device.
2. Wait until the device finishes the output generated by the command and returns the device prompt.
3. Parse the stored output from the device and take appropriate action.

Depending on what you need to achieve you may want to add additional `mults` to the `localSendCommandSingle` function. By adding matching case statements, you can respond to output from the device not handled by the standard device script.

```

Option Explicit
' The starting place for the external script is always Function Main.
Function Main(inputString)
  Dim commandResult
  Dim processBufferResult
  Main = ""
  ' Call the function to send the command to the device. In this case we are
using input string as the command.
  commandResult = localSendCommandSingle(inputString)
  ' If the command was processed OK then take action.
  If commandResult Then

```

```

        processBufferResult = ProcessBuffer()
        If processBufferResult Then
            ' Take action based on the result of processing the buffer
        End If

    End If
End Function

Function localSendCommandSingle(CmdToSend)
' This is a simplified version of the send command single activity used in
the main CatTools client.
' The idea is that it can be used to handle specific output not catered for
by the main script.
    Dim retval
    Dim Mult(20)
    Dim Choices
    Dim StoredBuffer
    localSendCommandSingle = False
    StoredBuffer = ""
    ' Set up Mults - The text returned from the device is checked for any of the
mults and
    ' if a match is found different actions are taken depending on the mult.
    Mult(1) = "--More--"
    Mult(2) = "Invalid Command"
    ' Custom mults can be inserted here
    ' Mult(3) = "Device specific mult"
    Mult(18) = cl.DeviceVTYPrompt
    Mult(19) = cl.DeviceEnablePrompt
    Mult(20) = cl.DeviceConfigPrompt
    Choices = 20 'the number of mults

    ' Clear the buffer
    cl.flushrxbuffer

    ' Send the command to the device (but dont press enter yet)
    If cl.WaitForEcho Then
        If cl.SendAndWaitForEcho(CmdToSend) = False Then
            cl.Log 4, "Did not receive echo of " & CmdToSend
            Exit Function
        End If
    Else
        cl.sendData CmdToSend
    End If
End Function

```

```

End If

' We dont want to record the command being sent or the echo so clear the
buffer ready for a reply from device
cl.flushrxbuffer

' Send <cr> to execute the command
cl.sendData vbCr

' Process return results
cl.Log 4, "Waiting for a response to: " & CmdToSend
Do
    ' Wait for a response from the device
    If cl.WaitForTime = 0 Or cl.WaitForTime = "" Then
        ' If the user has not specified a WaitForTime
        retval = cl.WaitForMultData(Mult, Choices, cl.RecDataTimeOut)
    Else
        ' If the user has specified a WaitForTime
        retval = cl.WaitForMultData(Mult, Choices, cl.WaitForTime)
    End If

    ' Store any output from the device
    StoredBuffer = StoredBuffer & cl.RxBuffer

    ' Analyze results for the first matching mult found
    Select Case retval
    Case 0
        ' No valid data received
        If 2 > 0 Then
            cl.Log 2, "Did not receive expected response to comma"
        End If
        Exit Do
    Case 1
        ' --more-- prompt found so send a cr to get the next block of
        cl.sendData vbCr
    Case 2
        ' Invalid command - error with syntax
        If 2 > 0 Then
            cl.Log 2, "Error with syntax: " & CmdToSend
        End If
        Exit Do
    End Select
End Do
    
```



```
'Case 3
    ' Take action based on the "Device specific mult" found.
    ' This may involve sending a command or keystrokes to the dev
    ' You may also need to flush the buffer before or after the a
a cl.FlushRxBuffer command.
    ' Note: You may not want to exit the loop at this point becau
want to leave the device ready
    ' to receive another command so you should ensure you are at
prompt when you exit the loop.
    Case 18, 19, 20
        ' Prompt found so the device has finished its output so exit
        localSendCommandSingle = True
        Exit Do
    End Select
Loop
    ' Put the total output of the command in the RxBuffer so that it is
available for use elsewhere.
    cl.RxBuffer = StoredBuffer
End Function

Function ProcessBuffer()
    ' Do what ever processing of the data in cl.RXBuffer you need to here
    ProcessBuffer = True
End Function
```

How to enable or disable ports on a Catalyst CatOS switch

The Cisco Catalyst CatOS switches (4000, 5000, 6000) use `Set` type commands to modify the configuration. You can easily enable or disable specific ports on switches at certain times of the day via a scheduled [Device.CLI.Send commands](#) activity.

1. Create a scheduled activity to enable the ports in the morning.

Configure activity settings in the Edit scheduled activity window.

1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab
 - a. Select Type Device.CLI.Send commands
 - b. Enter a name and description, for example "Enable ports in morning"

4. Click the Time tab.
 - a. Select Custom Schedule and enter 07:00. Click Add.
 - b. Under Days of the week, select Monday, Tuesday, Wednesday, Thursday, Friday.
5. Click the Options tab.
 - a. Select List of commands to be entered on device.
 - b. Enter commands to enable ports 1 to 24 on slot 3 and enable ports 1 to 12 on slot 5:

```
Set Port 3/1-24 enable
Set Port 5/1-12 enable
```

- c. Select Enter commands in enable mode.
- d. Select Save device output to file. Specify the log file.
- e. Select Overwrite existing capture file.
- f. Select Answer yes to any confirmation prompts.
- g. Select Stop on error.

2. Create another scheduled activity to disable the ports in the evening.

Configure activity settings in the Edit scheduled activity window.

1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab.
 - a. Select Type Device.CLI.Send commands
 - b. Enter a name and description, for example "Disable ports in evening"
4. Click the Time tab.
 - a. Select Custom Schedule and enter 19:00. Click Add.
 - b. Under Days of the week, select Monday, Tuesday, Wednesday, Thursday, Friday.
5. Click the Options tab.
 - a. Select List of commands to be entered on device.
 - b. Enter commands to disable ports 1 to 24 on slot 3 and ports 1 to 12 on slot 5.

```
Set Port 3/1-24 disable
Set Port 5/1-12 disable
```

- c. **Select** Enter commands in enable mode.
- d. **Select** Save device output to file. Specify the log file name here.
- e. **Select** Overwrite existing capture file.
- f. **Select** Answer yes to any confirmation prompts.
- g. **Select** Stop on error.

How to backup Cisco IOS image to TFTP server

This example shows you how to use the [Device.CLI.Send commands](#) activity to back up a Cisco IOS device image to a TFTP server.

1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab
 - a. **Select Type** Device.CLI.Send commands
 - b. Enter a name and description, for example "Backup Cisco IOS image"
4. Click Options tab
 - a. **Select List of commands to be entered on device and enter the following code:**

```
copy flash: tftp:  
c2500-ik8s-1.122-5.bin  
192.168.1.200  
c2500-ik8s-1.122-5.bin
```

- c2500-ik8s-1.122-5.bin: name of image file to backup
 - 192.168.1.200: IP address of remote TFTP server
 - c2500-ik8s-1.122-5.bin: name of image file to use on TFTP server
5. **Select** Enter commands in enable mode.
 6. **Select** Save device output to file. Specify the log file name.
 7. **Select** Overwrite existing capture file.
 8. **Select** Answer yes to any confirmation prompts.
 9. **Select** Stop on error.

How to download Cisco IOS running config to TFTP server

This example demonstrates how to use the [Device.CLI.Send commands](#) activity to download the current running configuration of a Cisco IOS device to a TFTP server.

1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab.
 - a. Select **Type** `Device.CLI.Send commands`
 - b. Enter a name and description, for example "Download Cisco IOS config to TFTP".
4. Click the Options tab .
 - a. Select **List of commands to be entered on device** and enter the following code:

```
copy running tftp
192.168.1.200
%ctGroupName/%ctDeviceName-Config.txt
```

- 192.168.1.200: the IP address of remote TFTP server.
- %ctGroupName/%ctDeviceName-Config.txt: the file name to store config to on server.

i Note the use of a forward slash in the filename to specify a subfolder within the TFTP Download folder.

- %ctGroupName: Device Group Name
 - %ctDeviceName: Device Name
- b. Select **Enter commands in enable mode**.
 - c. Select **Save device output to file**. Specify the log file name.
 - d. Select **Overwrite existing capture file**.
 - e. Select **Answer yes to any confirmation prompts**.
 - f. Select **Stop on error**.

How to download Cisco IOS running config to SCP server

This example demonstrates how to use the [Device.CLI.Send commands](#) activity to download the current running configuration of a Cisco IOS device to a SCP server.

1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab.
 - a. Select Type `Device.CLI.Send commands`
 - b. Enter a name and description, for example "Download Cisco IOS config to SCP".
4. Click the Options tab.
5. Select List of commands to be entered on device and enter the following code:

```
copy running-config scp://Destination username:Destination
password@Destination HostIP/Destination filename
%ctCR
%ctCR
%ctCR
```

6. Select Enter commands in enable mode.
7. Select Save device output to file. Specify the log file name.
8. Select Overwrite existing capture file.
9. Select Answer yes to any confirmation prompts.
10. Select Stop on error.

Example

After the activity runs the example script above against your device

```
copy running-config scp://Admin:Password@10.190.1.10/%ctDeviceName-Config.txt
%ctCR
%ctCR
%ctCR
```

the device then responds with the prompts:

```
Address or name of remote host [10.190.1.10]?
Destination username [Admin]?
Destination filename [Device-Config.txt]?
```

By using the `%crCR` [Meta Variable](#), the device tells CatTools to send a CR (Carriage Return) response to each prompt. Note the use of the device name meta variable: `%ctDeviceName`

For more information, refer to [Backing up device via SFTP/SCP](#).


How to upload Cisco IOS config from TFTP to device

This example demonstrates how to upload an IOS to a device using the [Device.CLI.Send commands](#) activity.


1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab.
 - a. Select Type `Device.CLI.Send commands`
 - b. Enter a name and description, for example "Upload Cisco IOS config".
4. Click the Options tab.
5. Select List of commands to be entered on device and enter the following code:

```
copy tftp running
192.168.1.200
%ctGroupName%\%ctDeviceName-Config.txt (Filename to store config to on
server)
%ctCR (Required to start device TFTP)
```

- `%ctGroupName%\%ctDeviceName-Config.txt`: the filename to store config to on server.

 Note the use of a **backward slash** in the filename to specify a subfolder within the TFTP Upload folder.

- `%ctGroupName`: device group name
 - `%ctDeviceName`: device name
- `%ctCR`: carriage return. Required to start the device TFTP. The `%ctCR` variable to send a `<CR>` to the device to activate the TFTP session.
6. Select `Enter commands in enable mode`.
 7. Select `Save device output to file`. Specify the log file name.
 8. Select `Overwrite existing capture file`.
 9. Select `Answer yes to any confirmation prompts`.
 10. Select `Stop on error`.

 The [Device.TFTP.Upload Config](#) activity could also be used to upload an IOS to a device.

How to upgrade Cisco IOS via TFTP

This example demonstrates how to use the [Device.CLI.Send commands](#) activity to upgrade a Cisco IOS device using TFTP.

1. Open Kiwi CatTools.
2. In the Activities panel, click Add.
3. Click the Activity tab.
 - a. Select Type `Device.CLI.Send commands`
 - b. Enter a name and description, for example "Update Cisco IOS".
4. Click the Options tab.
5. [X] List of commands to be entered on device:

```
erase flash: (erase the existing flash files and make space for the new files)
copy tftp: flash:
192.168.1.200 (IP address of remote TFTP server)
c2500-ik8s-1.122-5.bin (source filename)
c2500-ik8s-1.122-5.bin (destination filename)
```

- `erase flash:` erase the existing flash files and make space for the new files
 - `192.168.1.200:` IP address of remote TFTP server
 - `c2500-ik8s-1.122-5.bin:` source filename
 - `c2500-ik8s-1.122-5.bin:` destination filename
6. Select `Enter commands in enable mode`.
 7. Select `Save device output to file`. Specify the log file name.
 8. Select `Overwrite existing capture file`.
 9. Select `Answer yes to any confirmation prompts`.
 10. Select `Stop on error`.

How to enter different commands onto each device for Device.CLI.Send

Configuring each of your selected devices in a different way can be achieved in two ways:

1. Create an individual [Device.CLI.Send commands](#) activity for each device and set the commands to be entered for each in the options tab. This method involves having a separate scheduled activity for each device you want to modify.

2. Create a single [Device.CLI.Send commands](#) activity and select all the devices you want to modify.
 - a. Use the `Or, read commands from file` option and specify the `%BaseFile%` value as part of the file name.
 - b. Create a separate text file for each device that contains the commands you want to have entered into the config.

Example

1. Specify a filename of `C:\CatTools\%BaseFile%.txt`. A list of devices might be:

Device name	Base filename	Config commands filename
CiscoRouter1	CiscoRouter1	C:\CatTools\CiscoRouter1.txt
CiscoRouter2	CiscoRouter2	C:\CatTools\CiscoRouter1.txt
Sales Switch	Sales_Switch	C:\CatTools\Sales_Switch.txt

i Any spaces and invalid characters are replaced by and underscore (`_`) in the base filename.

2. In each of the text files, enter a list of commands that you want to enter. For example, in the file `C:\CatTools\CiscoRouter1.txt`, enter the commands:

```
Interface s 0/0
Description New WAN link to head office
No shutdown
```

Enter different commands in each text file and then run the activity. CatTools automatically matches the correct text file with the device based on the `%BaseFile%` value.

i Any spaces and invalid characters are replaced by and underscore (`_`) in the base filename.

3. In each of the text files, enter a list of commands that you want to enter. For example, in the file `C:\CatTools\CiscoRouter1.txt`, enter the commands:

```
Interface s 0/0
Description New WAN link to head office
No shutdown
```

4. Enter different commands in each text file and then run the activity. CatTools automatically matches the correct text file with the device based on the `%BaseFile%` value.

Save outputs from multiple devices into a single file

Normally, the [Device.CLI.Send commands](#) activity saves the data from each device into a separate file. The default capture filename is: `\Captured`

`Data\%GroupName%\%DeviceName%.DeviceOutput.%DateISO%.txt`

To save the output from multiple devices into a single file, set the capture filename to `\Captured Data\MyActivityData.txt`

- On the options tab, ensure that you uncheck `Overwrite existing capture file`. This allows the data from each device to be appended to the common capture file.
- On the Activity tab, set the Client threads to use a single thread only. This ensures that the output file is written to by only one device at a time.

To have start and finish tags placed between the output from each device, place comments into the commands you send to the device. For example, if the command you enter is "Show ARP", use the following creates start and finish tags:

```
! %ctDeviceName - Start
Show ARP
! %ctDeviceName - Finish
```

When the command is sent to the device, the `%ctDeviceName` is replaced with the current device name. Ccommands starting with `!` are ignored by Cisco routers.

Catalyst switches

Catalyst switches (CatOS) use the `#` character instead. The captured data file could look like:

```
! Cisco2950Switch1 - Start
Protocol Address Age (min) Hardware Addr Type Interface
Internet 192.168.1.1 3 0008.02b9.1111 ARPA Vlan1
Internet 192.168.1.2 1 000d.9d89.2222 ARPA Vlan1
! Cisco2950Switch1 - Finish
! Cisco2950Switch2 - Start
Protocol Address Age (min) Hardware Addr Type Interface
Internet 192.168.1.3 6 0008.a118.3333 ARPA Vlan1
Internet 192.168.1.4 0 0008.02cf.4444 ARPA Vlan1
! Cisco2950Switch2 - Finish
```

The data captured from each device is wrapped with a Start and Finish tag.

Device.ConnectivityTest.Login

The **Device.ConnectivityTest.Login** activity allows you to log in to a selection of devices. This activity is similar to [Device.ConnectivityTest.Ping](#) in that it does not change anything on the device but simply confirms connectivity plus the ability to log in.

CatTools logs in to each selected device and logs any failures to the Report file `\Reports\Device.ConnectivityTest.Login.txt`. The report file provides you with an historical record of the status of tested devices.


In general, the reasons for performing log in tests are similar to those for performing Ping tests, with some differences.

The log in test goes further than a simple Ping. The test confirms interactive terminal access and network operation with the device. It also confirms that device passwords have not changed. This allows you to verify that log in passwords for the device have not been altered without your knowledge or authorization, and that staff with the correct password can log in without issue.

Setting up the activity for Device.ConnectivityTest.Login

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Enter enable mode if possible	If the device has Enable mode, and the appropriate passwords are configured in the device file, CatTools attempts to enter enable mode when it logs in to each Device.
Only record login failures	Record of login failures. The report file is contained to a minimum size, conserving hard-drive space.

Device.ConnectivityTest.Ping


The **Device.ConnectivityTest.Ping** activity allows you to ping a selection of devices. CatTools pings each selected device in turn, and logs any failures to the Report file `\Reports\Device.ConnectivityTest.Ping.txt`

The ping test is a simple Internet Control Message Protocol (ICMP) Echo request, sent from the machine running CatTools to the selected device(s). It tests the operation of your network between CatTools and the selected device, but only at the IP level. This activity is a useful test of your network, or your users' ability to use the network, depending on your specific network design. The activity report file provides a historical record of the status of the tested devices.

Setting up the activity for Device.ConnectivityTest.Ping

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity option

 Configure the options specific to this activity in the Options tab.

Response timeout (seconds)	Time, in seconds, that the activity waits until it assumes that the ping request has failed. Default value is 2 seconds.
Ping packet send count	Number of ping requests the activity sends to each device as the activity runs. Default value is 5 packets.
Only record ping failures	Record of ping failures. The report file is contained to a minimum size, conserving hard-drive space.
Log Error if 100% not reported	Raises an error if all pings are not 100% successful.

Device.InterDevice.Ping

The **Device.InterDevice.Ping** activity allows you to ping a list of hosts from the selected devices. CatTools logs into each device in turn and performs a ping test to each device listed. Alternatively, it will ping all other devices that have been selected. There are two options available for which devices are pinged:

1. Enter a list of host names and IP Addresses in the "Ping a list of Hosts" field, under the Options tab. CatTools logs in to each selected device and pings each listed host in turn.
2. Check `Or, ping all other selected devices` under the Options tab. CatTools starts with the first selected device and pings all other devices in turn. The activity then moves to the second device and pings all others in turn, and so on until all selected Devices have sent ping requests to all other selected devices. This test ensures that you have full connectivity from each device to all other devices.

CatTools creates a report in the form of a table, listing each device and the results of each ping originated by that device. Under the Options tab, you can also check the option to have CatTools report only on failed ping requests. This keeps the report as small as possible and draws attention only to fault conditions.

By default the Activity appends reporting information to the file `\Reports\Device.InterDevice.Ping.txt`. This report provides forensic data in the event of a service provider outage or other event that affects your network.

Hostname and IP address

The hostname and IP address information can be entered in two formats:

1. If you do not know the hostname, enter the IP address instead. Each IP address must be entered on a new line, marked by a carriage return or enter.
2. If you know the hostname, enter both hostname and IP address. The hostname and IP address must use a comma and space to separate them, with each listed on a new line.

The hostname is used in the report, in a dedicated column. For example:

Main Server, 192.168.1.1


Sales router, 192.168.2.1

ISP DNS, 202.30.45.21

Setting up the activity for Device.InterDevice.Ping

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Ping a list of hosts (Display name, IP address) A list of host names and IP addresses.

Or, ping all other selected devices Disable the "Ping a list of hosts" field.
This option and "Ping a list of hosts" are mutually exclusive. If you select this option, the Ping a list of hosts field is deactivated, and all selected devices under the activity are pinged.

Only record ping failures Record of ping failures. The report file is contained to a minimum size, conserving hard-drive space.

When to use the Device.InterDevice.Ping

The **Device.InterDevice.Ping** activity is a useful network management tool because it allows you to perform and report on multi-directional testing across your whole network. In many router networks, pinging the routers is not a sufficient test to provide a high level of confidence in the proper working of the network, because the router is closer to you than the devices that use the network.

It is reasonably common, for example, for pings to remote routers to use a WAN interface IP address. So as long as the WAN and router are functional the ping request may succeed, but this type of test does not tell you that the LAN on the far side of the router is operating as intended. It only tests your ability to ping the router from a particular point in your network. Pinging a router does not test from other points, nor does it test the ability of devices on the far side to successfully use your network.

If you have a Linux device on each remote LAN segment, or at least "behind" each remote router, you can access those devices from CatTools and use an activity to ping other significant parts of your network. This provides better evidence that the network is operating properly than pinging the routers WAN interface from a single central point.

Checking availability of a central device

The `Device.InterDevice.Ping` activity test allows you to test, for example, branch office connectivity back to the central office server. Select all the branch office routers (or a Linux machine sitting in the branch office) and have them ping the server back in central office. This verifies that your customers can access the central server.

Device.TFTP.Upload Config


The **Device.TFTP.Upload Config** activity enables you to upload text config files to the specified device.

To see a full list of the currently supported device types for the `Device.TFTP.Upload Config` activity, please see the [Device Matrix](#).

Setting up the activity for Device.TFTP.Upload Config

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

TFTP Server	Set the address of the TFTP server.
TFTP FileName	Set the name of the file containing the config data. You can, optionally, use the default
Save device output to file	Save the device output from the activity to a file.
Overwrite existing capture file	Overwrite or append the capture file.
Save to NVRAM	Save the new config to NVRAM for non-CatOS devices.

Device.Update.Banner

The **Device.Update.Banner** applies a banner to your devices. CatTools passes the banner command to each selected Device in turn. The activity's success or failure is logged to the report file `\Reports\Device.Update.Banner.txt`. This activity supports the use of [Command Variables](#).


The text provided either through the on screen text box or through an input file should be plain text. No delimiters or syntax command words are needed.

To see a full list of the currently supported device types for the Device.Update.Banner activity, please see the [Device Matrix](#).

Setting up the activity for Device.Update.Banner

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Banner text	Indicates if the banner text is to be read from the text box or from a file. Selecting this option allows for free text to be entered. Unselecting this option allows you to select a text file containing the banner text to be applied.
Or, read banner text from file	Reads banner text from the specified text file entered. This option and "Banner text" are mutually exclusive. If you select this option, the Banner text field is deactivated, and the text is only read from the text file.
Which banner is to be supported	Select which banner to apply. Not all banners are supported on all devices.
Save device output to file	Save the Banner command that was used to the file specified.
Overwrite existing capture file	Overwrite or append to the capture file.
Save running config to start-up config when complete	Saves the banner to NVRAM start-up configuration. Some devices do not support this option.

Device.Update.Password

The **Device.Update.Password** activity enables you to change some of the passwords on devices.


To see a full list of the currently supported device types for the Device.Update.Password activity, please see the [Device Matrix](#).

A report gives you the status of each device selected for the activity after the activity is completed. The report shows both the new and old passwords. These passwords are shown by default in plain text on the report. You may elect to hide the reported passwords by selecting the [Hide password change report passwords](#) option in the Misc tab in the CatTools Setup dialogue.

Setting up the activity for Device.Update.Password

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

New enable	<p>Configure a new enable mode password to use for devices.</p> <p>You can enter a blank password in the field. This enables you to reset the password to nothing (no password required to log in).</p>
New enable secret	<p>Configure a new enable secret mode password to use for devices.</p> <p>You can enter a blank password in the field. This enables you to reset the password to nothing (no password required to log in).</p>
Choose password for CatTools	<p>Select which of the enable mode passwords you want CatTools to remember and use for the devices.</p>
New VTY	<p>Configure a new VTY password to use for devices.</p> <p>If selected, you must enter a password in the field. Field cannot be blank.</p>
Console password	<p>Configure the console password to use for devices.</p> <p>If selected, you must enter a password in the field. Field cannot be blank.</p>
Save running config to start-up config when complete	<p>Copy the running config on the device to non-volatile memory. The modified config then becomes the start-up config.</p>

Report.ARP table

The **Report.ARP table** activity builds a report using the ARP table from selected devices as source data. On a Cisco IOS device it uses a simple "Show IP ARP" command and builds a table of results over time.


The default location of the text file is `\Reports\Master ARP table.txt`. This activity automatically indexes MAC addresses against IP addresses and device interfaces, and then resolves their host names via DNS, if required. By default, the table is updated after each run of the activity, providing a historical record of the devices attached to your network over time. Each entry is time stamped, and "First Seen" and "Last Seen" columns included in the report.

By importing the file into a spreadsheet program like Excel, you can search for specific MAC addresses or host names, or maybe list all devices on a particular port. The Options tab allows you to use an alternate command, to set the expiry date of MAC entries in the report, and to select whether to resolve IP addresses.

Setting up the activity for Report.ARP table

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Use alternate command	<p>Set an alternate command if the default command issued by CatTools does not produce the correct results.</p> <p>This may be useful if you have some devices not yet supported in the CatTools default device list.</p>
Days until entries expire	<p>Activates associated numeric field. Enter the number of days you want entries to remain in the table.</p> <p>Each time the activity runs, the existing MAC address table entries that are older than the expiration period are removed. This keeps the table to a workable size.</p>
Resolve IP addresses to host names	<p>Activates resolution of host names to IP addresses for report.</p> <p>Each time the activity runs, the program scans the table for entries that have a blank host name field. A list of IP addresses is created and then resolved in bulk. The resolver attempts up to 100 DNS lookups at once. If a hostname is returned for the IP address, the table is then updated with the hostname.</p>

Report.CDP Neighbors table


The **Report.CDP.Neighbors table** activity produces a report of all networked devices that can be seen by the specified device using the CDP neighbor command.

To see a full list of the currently supported device types for the Report.CDP Neighbors table activity, please see the [Device Matrix](#).

Setting up activity for Report.CDP Neighbors table

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Use alternate command	Enter alternate commands if the default command issued by CatTools does not produce the correct results.
	This is useful if you have devices not yet supported in the CatTools default device list.
Hide duplicate IP address entries	Hide duplicate IP address entries.

Report.Compare.Running Startup

The **Report.Compare.Running Startup activity** compares the running and the startup configs of your device(s) and reports on the differences found.

To see a full list of the currently supported device types for the activity, please see the [Device Matrix](#).

How the activity works

The activity works by first connecting to your selected devices, then issuing the commands which show the running and startup configs. These configs are saved into the `ClientTemp` folder while the activity is running. When the configs have been retrieved from the devices, a compare is run against the two captures and a report is generated, similar to the [Report.Compare.Two files](#) activity.

Report generation

The reporting of any differences found can be any combination of the following three reports:

1. As with other activities the report file on the main activity tab records a history of the jobs that have been run and their result.


2. An HTML report can be created which will allow you to see the differences between the two files next to each other in column format and is color coded.
3. A .txt file report can be generated which shows differences in top down format.

The two file paths are capable of receiving [Filename Variable](#) parameters.

Setting up the activity for Report.Compare.Running Startup

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options


 Configure the options specific to this activity in the Options tab.

Alternate Startup command	Enter a command which, when issued to your device, returns the startup config.
Alternate Running command	Enter a command which, when issued to your device, returns the running config.
Ignore Text	<p>Should the new file contain any lines of text that are likely to be different from the original file every time the comparison is run (such as a timestamp) and you would like to ignore these lines, they can be added here. More information is available on the Ignore Text field.</p> <p>To ignore certificate self-signed changes on Cisco routers, see the block text ignore example.</p>
HTML compare report	<p>Specifies the folder and file name of where to store the HTML diff report.</p> <p>This diff report file contains a side by side comparison of the Running and Startup configs. Changes are highlighted in different colors.</p>
Text summary diff report	Specifies the folder and file name of where to store the text based diff report. This diff report file contains a list of the changes found in the Running and Startup configs. This report is a smaller and simpler way to view the changes. Unlike the HTML report, where both the Running and Startup configs are shown in full. The text report only shows the lines that are different.
Only notify by e-mail if differences found	Check this option to receive emails reporting any difference found between the startup and running configs.

Attach reports to email Allows you to choose what level of diff reports you are sent via e-mail:

- HTML compare report only (default)
- Text diff report only
- Both reports

If you want full comparison reports with highlighting, use the HTML compare report. If you just want the changes, use the text report.

 If you are sending the reports via an insecure medium, such as a corporate network or internet, be aware that the HTML reports contain your full device configuration. SolarWinds recommends that you send password protected HTML reports over an insecure medium.

Zip Attachments To reduce network traffic and allow for password protection, you can choose to zip the attachments. This is the recommended and default option. It keeps the e-mail smaller and places all the files in a single attachment, which can be beneficial when you have, for example, 400 devices.

Password protect Zip file Protect your reports by adding a password to your zip file. A password protects your files over insecure mediums while in transit to your e-mail box. SolarWinds recommends using current password strength standards when creating a password for your files.

Report.Compare.Two files


The **Report.Compare.Two files** activity runs a compare against two files which you define and reports on the differences it finds.

Generating comparison reports

The reporting of any differences found can be any combination of the following three reports:

1. The Report File on the main Activity tab records a history of the jobs that have been run and their result.
2. An HTML report which allows you to see the differences between the two files next to each other in a color-coded and columned format.
3. A .txt file report which shows differences in top down format.


The HTML and .txt file paths are capable of receiving [Filename Variable](#) parameters.

 This activity does not require the selection of any devices. Filename variables relating to devices, such as %GroupName%, %BaseFile% , %DeviceName%, do not work.


Setting up the activity for Report.Compare.Two files

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Source File	The first file for the comparison to use. This can be thought of as the new file.
File to compare against	The second file for the comparison to use. This can be thought of as the original file.
Ignore Text	Should the new file contain any lines of text that are likely to different from the original file every time the comparison is run (such as a timestamp) and you would like to ignore these lines, they can be added here. For more information, see Ignore Text field .
HTML compare report	Results are shown in HTML format. Indicate where the report should be saved.
Text summary diff report	Results are shown in Text format. Indicate where the report should be saved.
Only notify by e-mail if differences found	Receive emails reporting differences found between the new and old files.
Attach reports to email	<p>Allows you to choose what level of diff reports you are sent via e-mail:</p> <ul style="list-style-type: none"> • HTML compare report only (default) • Text diff report only • Both reports <p>If you want full comparison reports with highlighting, use the HTML compare report. If you just want the changes, use the text report.</p>

 If you are sending the reports via an insecure medium, such as a corporate network or internet, be aware that the HTML reports contain your full device configuration. SolarWinds recommends that you send password protected HTML reports over an insecure medium.

Zip Attachments	To reduce network traffic and allow for password protection, you can choose to zip the attachments. This is the recommended and default option. It keeps the e-mail smaller and places all the files in a single attachment, which can be beneficial when you have, for example, 400 devices.
Password protect Zip file	Protect your reports by adding a password to your zip file. A password protects your files over insecure mediums while in transit to your e-mail box. SolarWinds recommends using current password strength standards when creating a password for your files.

Report.Error info table

The **Report.Error info table** activity collects and reports error counter information for all interfaces of each selected device. The default location for the report is `\Reports\Master Error info table.txt`.

To see a full list of the currently supported device types for the activity, please see the [Device Matrix](#).


This activity contains four options, detailed under Activity options:

1. Specify the order of the report headers, decide which columns you wish to display, and what names you want the columns headers to be if you do not use defaults.
2. Calculate column totals, providing a summary line for each of the devices in the activity.
3. Clear the interface counters after collecting the data.
4. Collect the error counters from your VLANs. This option is only supported by Cisco IOS devices.

Setting up activity for Report.Error info table

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Specify Report Headers	Specify the order of the report headers, decide which columns you wish to display and what names you want the column headers to be. The first row of the matrix displays all the information that may be reported on. You can select to report on each column or not as you wish. You may also select and drag each column to the position you want for formatting the report layout.
Calculate column totals	Calculate column totals. This provides you with a summary line for each of the errors columns. Each device in the activity has a separate totals summary line.

Clear the interface counters	Clear the devices error counters after collecting all the current data.
------------------------------	---

Collect VLAN information	Collect the error counters from your VLANs. This option is only supported by Cisco IOS devices.
--------------------------	--

Report.MAC address table

The **Report.MAC address table** activity gathers MAC address entries (CAM table entries) from switches and bridging routers and can builds a table over time. The default location of this file is `\Reports\Master MAC address table.txt`.


By default, this activity may issue more than one command depending on the device type. In these cases, not every command issued is valid for every device selected. This does not cause problems or errors. CatTools gathers information from every command that succeeds and ignores any command that does not produce a valid result.

Optionally, you to use an alternate command and to set the expiry date of MAC entries in the report. Entering an alternate command to use overrides all default commands.

Setting up the activity for Report.MAC address table

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Use alternate command	Set an alternate command if the default command issued by CatTools does not produce the correct results. This may be useful if you have some devices not yet supported in the CatTools default device list.
-----------------------	--

Days until entries expire	Activates associated numeric field. Enter the number of days you want entries to remain in the table. Each time the activity runs, the existing MAC address table entries that are older than the expiration period are removed. This keeps the table to a workable size.
---------------------------	--

Report.Port info table


The **Report.Port info table** activity creates a snapshot of the interface configuration and state of the selected devices and stores the data in a tab delimited report file. The data is stored in a text file whose default location is `\Reports\Master Port info table.txt`. The information gathered includes name, VLAN, port type (Ethernet, Fast Ethernet, Full Duplex, etc), status, and speed.

Optionally, you can enter an alternate command, should the default command not produce the correct output.

Setting up the activity for Report.Port info table

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Use alternate command	Set an alternate command if the default command issued by CatTools does not produce the correct results.
-----------------------	--

This may be useful if you have some devices not yet supported in the CatTools default device list.

Report.SNMP.System summary


The **Report.SNMP.System summary** activity produces a summary report of data available from any device that supports standard system SNMP. All columns are optional and may be renamed. You can add up to 5 optional columns to the report and enter your own OID to produce the column data.

To see a full list of the currently supported device types for the activity, please see the [Device Matrix](#).


Setting up the activity for Report.Compare.Running Startup

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Alternate Startup command	Enter a command which, when issued to your device, returns the startup config.
---------------------------	--

Alternate Running command	Enter a command which, when issued to your device, returns the running config.
Ignore Text	<p>Should the new file contain any lines of text that are likely to be different from the original file every time the comparison is run (such as a timestamp) and you would like to ignore these lines, they can be added here. More information is available on the Ignore Text field.</p> <p>To ignore certificate self-signed changes on Cisco routers, see the block text ignore example.</p>
HTML compare report	<p>Specifies the folder and file name of where to store the HTML diff report.</p> <p>This diff report file contains a side by side comparison of the Running and Startup configs. Changes are highlighted in different colors.</p>
Text summary diff report	Specifies the folder and file name of where to store the text based diff report. This diff report file contains a list of the changes found in the Running and Startup configs. This report is a smaller and simpler way to view the changes. Unlike the HTML report, where both the Running and Startup configs are shown in full. The text report only shows the lines that are different.
Only notify by e-mail if differences found	Check this option to receive emails reporting any difference found between the startup and running configs.
Attach reports to email	<p>Allows you to choose what level of diff reports you are sent via e-mail:</p> <ul style="list-style-type: none"> • HTML compare report only (default) • Text diff report only • Both reports <p>If you want full comparison reports with highlighting, use the HTML compare report. If you just want the changes, use the text report.</p> <div style="border: 1px solid orange; padding: 10px; margin-top: 10px;"> <p> If you are sending the reports via an insecure medium, such as a corporate network or internet, be aware that the HTML reports contain your full device configuration. SolarWinds recommends that you send password protected HTML reports over an insecure medium.</p> </div>
Zip Attachments	To reduce network traffic and allow for password protection, you can choose to zip the attachments. This is the recommended and default option. It keeps the e-mail smaller and places all the files in a single attachment, which can be beneficial when you have, for example, 400 devices.

Password protect Zip file	Protect your reports by adding a password to your zip file. A password protects your files over insecure mediums while in transit to your e-mail box. SolarWinds recommends using current password strength standards when creating a password for your files.
---------------------------	--

Report.Version table

The **Report.Version table** activity creates a tab delimited report of all the software and hardware version information from the selected devices. The default location of the data file is `\Reports\Master Version table.txt`.

The Version table contains the following information columns:

- Group
- Device Name
- Serial Number
- Processor
- IOS (version number)
- ROM (revision)
- Boot
- Uptime
- Flash
- NVRAM (size)
- Memory
- Image (filename)

Setting up the activity for Report.Version table

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Report.X-Ref.Port MAC ARP

The **Report.X-Ref.Port MAC ARP** activity creates a cross reference report of MAC and IP addresses against ports on a network. If IP addresses have been resolved to host names on the ARP report, the host names are reported as well.

This report uses the [Device.Port info table](#), [Report.MAC address table](#) and [Report.ARP table](#) reports as its information base. It collects all port information from the Port report and matches MAC and ARP information against the ports. These report activities **must** be run before the cross reference report can be run.


The report does not require any devices to be selected. It uses the devices selected for the port report as its starting point.

Optionally, you can select the source reports, limit the number of MAC addresses displayed, and ignore specified interfaces.

Setting up the activity for Report.X-Ref.Port MAC ARP

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Current Port info table	The path of the port info table to use for the x-ref.
Current MAC address table	The path of the MAC address table to use for the x-ref.
Current ARP table	The path of the ARP table to use for the x-ref.
Max MAC entries per port	<p>Limits the number of MAC entries to be displayed in the report on a port. This is useful for trunks and similar ports.</p> <p>If you set a maximum of 5 entries, but a port contains more than 5 MAC entries, only the 5 most recent MAC entries are displayed.</p>
Interfaces to exclude	<p>Excludes an interface. You can enter an exact port name, or you can enter a regular expression that evaluates a number of similarly named ports to exclude. An example of this might be "Giga Serial" to remove any ports with the text "Giga" or "Serial" in their name.</p> <p>For example, these may be trunk ports that link devices and that may have a large number of MAC entries linked to them.</p>

System.File.Delete

The **System.File.Delete** activity allows you to automate the deletion of files. It is primarily designed to allow the management of the dated config files, although it can be run against any folder.


The activity does not run against your physical network devices. Instead, from the specified folder and optionally sub folders, it:


- Deletes all files except a specified number of device specific files.
- Deletes files older than a certain number of days.
- Combination of the above two options.

Setting up the activity for System.File.Delete

For information on setting up activities, see [Add / Edit scheduled activity details](#).

Activity options

 Configure the options specific to this activity in the Options tab.

Select Folder	The folder containing the file you want to run the activity against.
Include sub folders	Select this if you want the activity to run against all sub folders of the selected folder.
Number of device specific dated files to retain	<p>This option allows you to cleanup outdated configs, but also ensure that you always keep a specified number of dated configuration files. Enter the number of files of each device specific type you want to keep.</p> <p>This works by using the CatTools device filename, for devices specified in the devices tab above, and comparing this to the file names in the folder to identify files that relate to the same device. It then keeps the specified number of files for each device type.</p>
<p> See Troubleshooting if your System.File.Delete activity is not retaining an \times number of files.</p>	
Delete files older than how many days	Delete files older than the specified number of days. This option is of lower precedence than the above option. If both are specified, you still keep \times number of files, even if they are older than the specified number of days.

CatTools naming convention

Troubleshoot 'Number of device specific dated files to retain' option not working:

If you have set `Number of device specific dated files to retain` but the activity is deleting all files, it is likely due to using a custom file naming convention, rather than using the default CatTools file naming convention.

CatTools default naming convention uses, in this backup activity example, the naming convention:

```
Config.Current.Running.Cisco_Router_2.txt
```

However, if you are using a convention which does not place a period mark (.) at the end of the device name, such as `Cisco_Router_2.`, CatTools may not group the files relating to the device correctly. As demonstrated below, if you are using a file naming convention such as:

```
Cisco_Router_2-backup.txt
```

CatTools is unable to group the `Cisco_Router_2` files together correctly. Therefore all files are deleted.

The solution is to change the dash (-) in your custom naming convention to a period (.). For example:

```
Cisco_Router_2-backup.txt
```

becomes

```
Cisco_Router_2.backup.txt
```

Kiwi CatTools internal functions

This section contains CatTools internal functions and methods that can be used to help you customize your activity.

- [Ignore Text](#)
- [Filename Variables](#)
- [Meta Data](#)
- [Meta Commands](#)
- [Meta Variables](#)

Ignore Text

When comparing files using either the [Report.Compare.Two files](#) or [Report.Compare.Running Startup](#) activities, you are provided with the option to supply ignore text.

Lines that match the ignore text instruction are classed as ignored lines. Ignored lines are still highlighted in the diff report, but they are not flagged as changes/differences. No trigger alert emails are sent.

The Ignore text syntax is made up of two parts:

1. The first part is an instruction which defines the context of the ignore text.
 - `^`: ignore lines with the sample text occurring anywhere in them.
 - `<`: ignore lines that begin with the sample text.
 - `>`: ignore lines that end with the sample text.
2. The second part is a block of sample text which is searched for when the line comparison is being made.

The following is an example of how this feature might be used:

- `^Time`: ignores any lines with the text `Time` in them.
- `<Time`: ignores any lines which begin with the word `Time`.
- `>Time`: ignores any lines which end with the word `Time`.

Examples

- `^! Last configuration change at`
Ignore only lines that contain the text `Last configuration change at`.
- `<Current configuration`
Ignore only lines that begin with `Current configuration`.
- `^! Last configuration change at|<Current configuration`
Ignore lines that contain the text `Last configuration change at` as well as lines that begin with `Current configuration`.

Multiple ignore text instructions may be concatenated by using the pipe, `|`, symbol: `^some text|^some other text|<some more text`

Ignoring changes that span multiple lines:

To have a block of changes ignored, use the following ignore syntax:

```
{start of block text match}-{end of block text match}
```

In the example below, the running-config contains a code block that is missing from the startup config.

Cisco Running-Config:

```
!
crypto ca certificate chain TP-self-signed-12345678
    certificate self-signed 01
```

```

3082022B 30820194 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D31 37363538
528BD5A8 E7E26C51 10BAB609 5B60228F C8DE0299 7BE85C2D 9769FF05 C295706F
3082022B 30820194 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D31 37363538
528BD5A8 E7E26C51 10BAB609 5B60228F C8DE0299 7BE85C2D 9769FF05 C295706F
3082022B 30820194 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D31 37363538
528BD5A8 E7E26C51 10BAB609 5B60228F C8DE0299 7BE85C2D 9769FF05 C295706F
    
```

quit

Username joe password bloggs

!

Cisco Startup-Config:


```

!
crypto ca certificate chain TP-self-signed-12345678
    certificate self-signed 01 nvram:IOS-Self-Sig#4567.cer
Username joe password bloggs
!
    
```

To ignore this entire block of changes, you could use the following ignore text:

```
{certificate self-signed 01$}-{quit}}|^certificate self-signed
```

This matches "certificate self-signed 01" at the top and ignores all changes until it matches the "quit" at the end of the block. The text contained between the {}-{} uses the Regular Expression syntax. So in our example, the \$ means match the end of the line.

 Because the ignore text field uses the pipe, |, as a delimiter between multiple inputs, you can not use | in the regular expression syntax. For example: ^some text|{crypto block start}-{crypto block end}|<some more text


To Ignore a Block of Text from Comparison itself

To identify a block of text to be ignored fully from comparison itself, use the following syntax:

```
[Start of Block]-[End of Block]
```

Filename Variables

There are a number of variables that can be used within the filename fields throughout CatTools which will help you create unique names for your backups and reports. These variables are resolved to actual values at the time the activity is started. For example: `C:\CatTools V3\Captured Data\%GroupName%\%DeviceName%.DeviceOutput.%DateISO%.txt`

 Filename variables should not be confused with [Meta Data](#) variables.

Variable	Value
%GroupName%	Device group name as a folder name
%BaseFile%	Device specific file name
%HostAddress%	Device specific Host Address
%DateISO%	ISO format date
%DateYYYY%	Date as yyyy (year only)
%DateMM%	Date as mm (month only)
%DateDD%	Date as dd (day only)
%DateWW%	Day of the week (values 1 through 7)
%DateWDN%	Week Day Name (e.g. Monday)
%TimeHHMM%	Time as hhmm (e.g. time of 13:01 will display as 1301)
%TimeHH%	Time as hh (e.g. time of 13:01 will display as 13)
%TimeMM%	Time as mm (e.g. time of 13:01 will display as 01)
%DeviceName%	File system friendly device name
%AppPath%	File system path of the CatTools executable including a trailing "\"

It should be noted that %GroupName% and %DeviceName% are device specific variables. As such they cannot be resolved against an activity, such as a summary report file, since an activity is related to multiple devices.

Meta Data

Meta data is a term used to describe the method by which a user can access internal CatTools variables or functions from within an activity.

 Meta data should not be confused with [Filename Variables](#).

The meta data commands / variables are placed within the stream of commands defined by the user for the activity. They may be the only command for an activity, but are normally interspersed amongst actual device commands.

The commands are evaluated by CatTools for each device in the Activity list and tell CatTools to either

1. Perform a CatTools internal action at that point in time. This is a [Meta Command](#).
2. Replace the variable with an actual data value. This is a [Meta Variable](#).

Activities that support the use of Meta data include :

- [Device.CLI.Send commands](#)
- [Device.CLI.Modify Config](#)
- [Device.Update.Banner](#)

Meta Commands

Meta Commands are a subset of CatTools [Meta Data](#). They are used to tell CatTools to perform a piece of internal functionality at a particular point in an activity.

Available Meta Commands

Meta Command	Description
<code>%ctDB</code>	Updates the CatTools Device table which holds the properties for each device. See the Device Table section for a list of the device fields you can update.
<code>%ctUM: EchoOff</code>	Sets a flag so commands are sent without waiting for an echo back of the command.
<code>%ctUM: EchoOn</code>	Sets a flag so commands are sent and wait for an echo back of the command.
<code>%ctUM: Timeout</code>	Sets the maximum time to wait for the response to a command.
<code>%ctUM: PauseTime</code>	Sets a period to wait before sending a command.

`%ctDB` Command

The `%ctDB` meta command enables you to directly update a field in a table in the CatTools database as part of an Activity.

Syntax

```
%ctDB:tablename:fieldname:new field value
```


- `tablename` is the name of the table in the database you wish to update a field in.
- `fieldname` is the name of the field you wish to update. See the Device Table section for a list of fields.

The new field value is the content you wish to place into the database field. This value may contain a colon, `:`, in the content as the command parser only looks for the first three colons as separators.

For instance, you may wish to change the password for devices on the remote authentication server. This would be done on the server itself, and CatTools would not be aware that the password has changed.

You can set up a [Device.CLI.Send commands](#) activity with a meta command that changes the password in the CatTools database for the selected devices:

```
%ctDB:Device:AAAPassword:thenewpassword
```

In this case, you need to enter only the one meta command as the commands for the [Device.CLI.Send commands](#) activity. When the activity runs, it sets the AAAPassword field of the Device table to "thenewpassword" for all the selected devices. CatTools would then use this new password whenever it needed to login to one of the devices.

You might also use this type of command if you were setting up the local username and password properties on a set of devices. Place the meta commands to update the CatTools database fields after the device commands that set the properties on the devices. SolarWinds recommends that you select the option to Stop on error in this situation to ensure that CatTools does not have a field set if the device command does not complete successfully.

In a [Device.CLI.Modify Config](#) activity you might enter the command: `username joe password fred...` to set a local username and password. You could use the meta commands to update the CatTools database to reflect these changes:

```
%ctDB:Device:AAAUsername:joe
%ctDB:Device:AAAPassword:fred
```

CatTools now knows to use these values when logging into the device in future.

Device Table

List of the fields that can be updated in the Device table using the `%ctDB` meta command:

Field Name	Max Size - characters
HostAddress	255
Filename	255

Field Name	Max Size - characters
Model	255
Telnet	255
TelnetPort	255
Session	255
VTYPass	255
ConsolePass	255
EnablePass	255
PrivilegeLevel	255
AAAUsername	255
AAAPassword	255
SNMPRead	255
SNMPWrite	255
RequireVTYLogin	255
LoginUsesAAA	255
EnableUsesAAA	255
VTYPrompt	255
ConsolePrompt	255
EnablePrompt	255
AAAUserPrompt	255
AAAPassPrompt	255
Address1	>255
Address2	>255
Address3	>255
ContactName	>255
ContactPhone	>255
ContactEmail	>255

Field Name	Max Size - characters
ContactOther	>255
AlertEmail	>255

While all the fields are character format fields, some, such as RequireVTYLogin, LoginUsesAAA, and EnableUsesAAA, contain numbers to indicate whether they are on or off. 0 is off, 1 is on.

`%ctUM`

The `%ctUM` meta commands enable you to set various options for use in the [Device.CLI.Send commands](#) and [Device.CLI.Modify Config](#) activities.

Commands

`%ctUM: EchoOff`

Normally when commands are sent out, CatTools waits for an echo of that command before proceeding. Some commands do not produce an echo, so the activity fails at this point. Password changing activities are notorious for this.

The `%ctUM: EchoOff` command tells CatTools to send out the next command in this activity without waiting for an echo.

`%ctUM: EchoOn`

Although in some cases it is necessary to switch of the echoing of commands it is advisable to switch echoing back on as soon as possible. By waiting for an echo of commands CatTools can better synchronize data sent and received from the device minimizing the scope for errors.

The `%ctUM: EchoOn` command tells CatTools to wait for an echo of any further commands during this activity.

`%ctUM: Timeout xx`

When waiting for a response to a command CatTools waits for a default time of 30 seconds. Some commands on devices take longer than this to respond. The `%ctUM: Timeout XX` command allows you to specify the time to wait for a response for all subsequent commands for this activity.

For example, you may want to set the timeout for the show version command to 50 seconds, to do this you should perform the following:

```
%ctUM: Timeout 50
show version
%ctUM: Timeout 0
```

Note: the use of the command with a zero value - `%ctUM: Timeout 0` - sets the timeout back to the default value of 30 seconds.

`%ctUM:PauseTime xx`

This causes CatTools to pause for the specified time in seconds before issuing the next command. The maximum time that can be specified is 3600.

For example, the commands below issues a ping, waits for 50 seconds, and then issues another ping:

```
ping 127.0.0.1
%ctUM: PauseTime 50
ping 127.0.0.1
```

Meta Variables

Several Activities allow you to enter a list of commands which are issued to your selected devices. [Device.CLI.Send commands](#), [Device.CLI.Modify Config](#) and [Device.Update.Banner](#) are examples of these activities.

Within the commands you send, you can include a number of CatTools [Meta Data](#) variables that are translated when the activity is run.


For example, in the `Device.Update.Banner` activity you could set the following text and apply it to all your devices:

```
Hello, I am %ctDeviceName. Contact %ctContactName on %ctContactPhone to gain access to me.
```

Available Meta Variables

Variable	Value
<code>%ctDeviceName</code>	The device name
<code>%ctGroupName</code>	The device group name
<code>%ctHostName</code>	The device host address
<code>%ctModel</code>	The device model field
<code>%ctAddress1</code>	The device Address1 field
<code>%ctAddress2</code>	The device Address2 field
<code>%ctAddress3</code>	The device Address3 field

Variable	Value
%ctContactName	The device Contact name field
%ctContactPhone	The device Contact phone field
%ctContactEmail	The device Contact email field
%ctTimeHHMM	Current time in HHMM format
%ctDateISO	Current date in YYYYMMDD format
%ctBaseFile	Device base file name
%ctAppPath	File system path of the CatTools executable including a trailing "\"
%ctCR	Carriage return
%ctLF	Line feed
%ctTAB	Tab
%ctCTRL-Z	Control key & Z. Normally used to exit config mode.

 The variable names are case sensitive and must be entered as shown.

Creating a custom activity

CatTools provides a facility to create your own custom activities and activity script files should the CatTools built-in activities not suffice your requirements.

Prerequisites

A reasonable understanding or experience of Visual Basic Scripting is assumed in order to successfully add custom scripts to CatTools.

The help file documentation and comments within the example code template files found in the `/Templates` sub folder of the CatTools root directory, should provide a reasonable level of assistance for a technically competent novice to follow.

Overview

To add support for a custom activity in CatTools, four files are required. Three activity files and one custom device file:

Activity files

1. The activity type file ([.ini](#) file), which defines the following:
 - activity name
 - activity ID
 - activity [main script](#) filename (associated with the activity)
 - activity [client script](#) filename (associated with the activity)
 - the user interface field values and defaults which are displayed in the activity form [Options](#) tab when adding or editing an activity.
2. The activity [main script](#) file ([.txt](#) file), which contains code to read the activity options from the CatTools database, prepare folders and files to store output data, set variables, marshal the CatTools Client threads and do any post processing of results in order to create reports or send messages to the CatTools main program.
3. The activity [client script](#) file ([.txt](#) file), which contains a number of common function calls to the device scripts, i.e. the scripts that send device specific commands in order to get the device to log in, issue the commands required to perform the activity, then log out of the device again.

Device file

The device script file - [.custom](#) file - which contains device type specific code for the custom activity, such as the commands to send to the device and any parsing of the data before sending the results back to the client activity script.

The activity client and main script files also contains function calls and references to variables within the internal CatTools program code. These are prefixed with `cl.` in the client script and `ct.` in the main script. A list of these `cl.` and `ct.` functions and variables have also been made available within this chapter to help assist in the development of your custom activity scripts.

- [How to create a custom activity](#) - a simple step-by-step guide on how to create a custom activity.
- [The custom activity type file \(.ini\)](#) - information and how to create the custom activity type file.
- [The custom activity main script file \(.txt\)](#) - information and how to create the custom activity main script file.
- [The custom activity client script file \(.txt\)](#) - information and how to create the custom activity client script file.
- [The custom activity device script file \(.custom\)](#) - information and how to create the custom activity device script file.
- [cl. / ct. variables and functionsClient script - cl. variables and functions.htm](#) - information on the CatTools internal variables and functions exposed to the custom activity script files.
- [Testing your custom activity](#) - help and tips on testing your custom activity.

How to create a custom activity

This is a simple step-by-step guide on creating a custom activity in CatTools. It contains instructions on how to add a custom activity type to the CatTools GUI and how to create a the custom main script and custom client script activity files, and how to make your custom activity available to device types.

The custom activity templates files provided by CatTools and referenced in this guide, are based on creating a simple Version report for a Cisco Router device. Once you have created the script `.txt` and `.custom` files, you need to modify them for your particular activity or report.

Further information regarding the [custom activity type file \(.ini\)](#), [custom activity main script file \(.txt\)](#), [custom activity client script file \(.txt\)](#), [custom activity device script file \(.custom\)](#), [ct. code examples](#) and [cl. code examples](#) of how to use the CatTools internal functions and sub procedures in your activity scripts, can be found in the other sub pages of this chapter.

1. Create a custom activity type file (.ini)

- a. Create a copy of the `Custom.Activity.Template.ini` in the `\Templates` sub folder of the CatTools root directory and save to the `\Activities` sub folder.
- b. Rename the new file using the syntax `Custom.:`

`Type.Name`

For example, `Custom.Report.Version`

2. Edit the .ini file

Using a text file editor, such as Notepad, open the `.ini` file created in step 1. You must change the following items in the `.ini` file from the template default values:

```
[Activity]
name=Custom.Activity.Template
script_main=Main.Custom.Activity.Template.txt
script_client=Client.Custom.Activity.Template.txt
report_file=%AppPath%Reports\Master Custom Activity Template.txt
id=3999
```

- Change `name` to a unique activity name. For example: `Custom.Report.PortStatus`
Do not use spaces. Use a period (`.`) instead.

- Change `script_main` to the name of the associated Main script file.
Use the activity name and prefix with `Main.` to keep the name unique.
- Change `script_client` to the name of the associated Client script file.
Use activity name and prefix with `Client.` to keep the name unique.
- Change `report_file` to a unique report name.
- Change `id` to a unique number (i.e. one not used in any other activity .ini file). Number must be within the range of 4000 to 4999.

3. Save & restart

- a. Once you have made your changes to the `.ini` file, save it back to the `\Activities` sub folder and close.
- b. Restart CatTools to populate the Type drop-down field in the activity setup screen with your new custom activity type.

4. Create a custom activity main script file (.txt)

- a. Create a copy of `Main.Custom.Activity.Template.txt` in the `\Templates` sub folder of the CatTools root directory.
- b. Save the copy to the `\Scripts` sub folder.
- c. Give it a new file name as specified in the 'script_main' item within the activity type .ini file **[Activity]** section in step 2 above. Ensure you retain the .txt file type suffix.

5. Edit the main script .txt file

- a. Using a text file editor, open the .txt file created in step 4.
- b. Follow the instructions in the SCRIPT NOTES section of the `.txt` file to begin customizing the script for your activity.

6. Save the main script .txt file

Once you have made your initial changes to the main script .txt file, save it back to the `\Scripts` sub folder and close.

7. Create a custom activity client script file (.txt)

- a. Make a copy of the `Client.Custom.Activity.Template.txt` in the `\Templates` sub folder of the CatTools root directory.

- b. Save the copy to the `\Scripts` sub folder. Rename the new file name as specified in the `script_client` item within the activity type `.ini` file `[Activity]` section in step 2 above. Ensure you retain the `.txt` file type suffix.

8. Edit the client script `.txt` file

- a. Using a text file editor, open the `.txt` file created in step 7.
- b. Follow the instructions in the `SCRIPT NOTES` section of the `.txt` file to begin customizing the script for your activity.

9. Save the client script `.txt` file

Once you have made your initial changes to the client script `.txt` file, save it back to the `\Scripts` sub folder and close.

10. Create a custom activity device script file (`.custom`)

- a. Make a copy of the `Custom.Device.Template.txt.custom` in the `\Templates` sub folder of the CatTools root directory.
- b. Save to the `\Scripts` sub folder and rename the file.

Ensure you save the file using the same file name, including the `.txt` file type suffix, as its associated encrypted device script file (or a custom device script file). Append an additional suffix `.custom`

The following example if you create a custom activity device script file for a Cisco Router using an encrypted script file provided by CatTools: `Cisco.Router.General.txt`. The custom activity device script file name is therefore: `Cisco.Router.General.txt.custom`

11. Edit the device script `.custom` file

- a. Using a text file editor, open the `.custom` file created in step 10.
- b. Read the `SCRIPT NOTES` section of the `.custom` file and begin customizing the script for your device.

An example of the steps you need to follow to add a custom activity to the `.custom` device file can be found in [The custom activity device script file \(.custom\)](#) help file page.

12. Save the device script `.custom` file

Once you have made your changes to the device script `.custom` file, save it back to the `\Scripts` sub folder and close.

13. Testing your custom activity scripts

- a. When all the custom activity scripts and the device .custom script files are set up, test the new activity within CatTools by setting up the activity to run with the device you created the .custom file for.
- b. Use Run Now to test the custom activity manually rather than by starting the timer.
- c. Check the [Info Log pane](#) for errors and edit your .txt files and .custom file as necessary.

See [Testing your custom activity](#) for more information and tips.

Creating custom activity checklist

- a. Create activity type file (.ini)
- b. Create activity Main script file (.txt)
- c. Create activity Client script file (.txt)
- d. Create activity device script file (.custom)

i Although the .txt files can be opened and edited using a text editor, a syntax highlighting [script editor](#) may make reading and editing the .txt file much easier.

The custom activity type file (.ini)

Activity types are defined within CatTools using text files (.ini files) and are stored in the \Activities sub folder within the root CatTools directory. Each .ini file represents a separate item in the 'Type' drop-down list in the [Add/Edit scheduled activity details form](#) setup form.

When CatTools starts, it searches for all the .ini files in the \Activities sub folder and reads their contents. All the activity types found are then available for selection in the activity Type drop down when defining activities.

To add a new custom activity type, you need to create a new .ini file defining the activity and its characteristics.

The .ini file sections overview

The contents of the .ini file are divided up within [sections]. The main sections inside the INI file are as follows:

[info] section

This is required and identifies the .ini file as a CatTools file.

```
[info]
cookie=CatTools
version=3
author=SolarWinds
```

[Activity] section

This is required and defines the name used within the CatTools user interface, such as the Activity type drop-down list field, and within scripting. It also defines the unique key (id) of the activity type in the CatTools database.

```
[Activity]
name=Custom.Activity.Template
script_main=Main.Custom.Activity.Template.txt
script_client=Client.Custom.Activity.Template.txt
description=Collects information from devices and creates a custom report
report_file=%AppPath%Reports\Master Custom Activity Template.txt
report_overwrite=1
client_threads=0
id=3999
```

Although the file name of the .ini file may be similar (or the same) to the [Activity] section `name` item, it is the `name` item that defines the activity name within the user interface and not the .ini file name.

CatTools reserves IDs from 0 to 3999 for predefined activity types. You can use the number range from 4000 to 4999 for the custom activity types you create, however you **must** ensure the number is unique.

[item] sections (Activity Options)

Each item section defines the input fields used within the CatTools user interface for the Activity setup screens.

```
[item_value1]
name=Use alternate command
type=string
checkbox_show=1
checkbox_default=0
value_default=
required=0
info="Alternative command to capture data from device."
```

Each input field in the CatTools user interface for Activity options has a separate `[item_XXX]` type section.


- The `name` item sets the text (or label) to be displayed next to the input or selection field.
- The `type` item describes the type of input or selection field adjacent to the name text/label.
- The `checkbox_show` and `checkbox_default` items describe whether a checkbox should be displayed and its default value of checked or unchecked.
- The `value_default` item sets the default value to be used when adding a new activity to the database.
- The `required` item tells the user interface if the field must contain valid data or not. For example this must be an item from within a predefined list or whether the user can enter their own values.
 - 0 = not required
 - 1 = required
- The `info` item is the description associated with this field. This text is displayed in the status bar of the user interface when a field has focus.

Limiting the range of values

The above `[item_value1]` section is an example of a checkbox and standard text box input field within the user interface. If you need to limit a range of numerical values that can be entered into an option field, then use the `type` or `range`.

The range option example below, contains a checkbox and an input field that accepts numerical values between 1 and 2000.

```
[item_value2]
name=Days until entries expire
type=range
checkbox_show=1
checkbox_default=0
value_default=30
range_min=1
range_max=2000
required=1
info="Remove entries older than X days from the Master report"
```

 Check box input fields can only accept values of 0 or 1. Any value other than a 1 is considered a 0.

For example, the check box is defaulted to 1 (checked).

```
[item_value3]
name=Resolve IP addresses to hostnames
type=checkbox
checkbox_show=1
checkbox_default=1
info="Resolve all IP addresses found to hostnames via reverse DNS lookup"
```

The length of the data that can be entered into each field is determined by the design of the CatTools database. Any data entered via the CatTools user interface that is too long for the field will be truncated accordingly.

The custom activity type .ini template

CatTools provides a starting point template file to help assist in the creation of a new custom activity type. The template file is called `Custom.Activity.Template.ini` and can be found in the `\Templates` sub folder. This template file is included as part of the predefined activity types in CatTools. As for all of the CatTools predefined activity types, it may be subject to updates and modifications with each new release of the CatTools product.

For this reason, you should never modify the template file itself in order to create a new custom activity type.

If you need to create a new custom activity type, take a copy the template file and save it to the `\Activities` sub folder giving it a new file name.

Creating your custom activity type .ini file

If adding a new custom activity type to CatTools, the first step is to take a copy of the template file `Custom.Activity.Template.ini` in the `\Templates` sub folder and save it to the `\Activities` sub folder giving it a new file name.

This gives you a new starting point activity type file to work with. When naming the file, try to use the naming convention:


```
Custom.Type.Name
```

- **Custom:** the text "Custom" to distinguish custom activity types from CatTools predefined types. They are grouped together in the activity 'Type' drop-down list field.
- **Type:** the type of activity, such as Report, CLI, Connectivity, or Update.
- **Name:** the name of activity, such as ARPTable, SendCommands, PingTest, and BannerText.

Editing the .ini file

The next step is to open the .ini file and make the required changes to the values within each section. Below is an example of an .ini file with a number of items values highlighted.

- The most important item values which must be changed are **highlighted in red**. These items must be unique otherwise your activity type may not appear in the CatTools activity Type dropdown list field.
- Items values that are no required to be changed are **highlighted in green**.
- Additional comments are **highlighted in Blue**.

 Do not include any comments in your .ini file.

SolarWinds recommends that anything else that is not highlighted should be left at its default setting. You can customize some of the items, such as the Name and Info items, to make them more relevant to your activity.

- The text value within the `Name` item is the field Label that is displayed within the form.
- The text value within the `Info` item is the text that is displayed in the Status bar when you select the field in the form.

Once you have made your amendments to the .ini file, save it to the `\Activities` sub folder. Restart CatTools in order for your new custom activity type to be read into the activity 'Type' drop-down list field.

Sample .ini file

```
[info]
cookie=CatTools
version=3
author=SolarWinds enter your name or leave as the default

[Activity]
name=Custom.Activity.Template change to a unique activity name. example:
Custom.Report.PortStatus Note: do not use spaces.
Use a 'period' mark (.)
script_main=Main.Custom.Activity.Template.txt change to associated Main
script name (use activity name and prefix with
'Main.' to keep unique)
script_client=Client.Custom.Activity.Template.txt change to associated Client
script name (use activity name and prefix
'Client.' to keep unique)
description=Collects information from devices and creates a custom report
change if required
report_file=%AppPath%Reports\Master Custom Activity Template.txt change to a
unique report name
report_overwrite=1
client_threads=0
```


```
id=3999select a number within the range of 4000 to 4999. The number used here
must be unique.
```

```
# Activity options
[item_value1]
name=Use alternate command
type=string
checkbox_show=1
checkbox_default=0
edit_button=0
value_default=
required=0
info="Alternative command to capture data from device."

[item_value2]
... add more options items as required (to a maximum of 10 options).
```

The custom activity main script file (.txt)

Activity main scripts are defined within CatTools using .txt files and are stored in the `\Scripts` sub folder within the root CatTools directory. When an activity is run, CatTools reads the `script_main` item value in the `[Activity]` section from the activity type .ini file to determine which activity main script file it needs to use.

 SolarWinds recommends that the .ini file and .txt files have the same file name apart from the file prefix of 'Main.' for the .txt script file.

For all the predefined activity types in CatTools, the associated activity main script .txt files have been encrypted. They are encrypted for two reasons. The first is to protect our intellectual property. The other is to prevent unauthorized modification of these files which may cause the scripts to fail at run time.

The custom activity main script .txt template

CatTools provides a starting point main script template file to help assist in the creation of a new custom activity main script. The template file is called `Main.Custom.Activity.Template.txt` and is found in the `\Templates` sub folder within the root CatTools directory.

This template file is included as part of the predefined activity scripts in CatTools, but is unencrypted. As for all of the CatTools predefined activity scripts, it may be subject to updates and modifications with each new release of the CatTools product.

For this reason, you should never modify the template file itself in order to create a new activity custom main script.

If you need to create a new custom activity main script, take a copy the template file and save it to the \Scripts sub folder giving it a new file name.

Creating your custom activity main script .txt file

If adding a new custom activity main script to CatTools, the first step is to take a copy of the template file `Main.Custom.Activity.Template.txt` in the `\Templates` sub folder and save it to the `\Scripts` sub folder, giving it with a new file name. You need to ensure the file name you use is the value entered for the `script_main` item within the activity type `.ini` file `[Activity]` section.

This gives you a new starting point device script file to work with.

Editing the main script .txt file

The next step is to open the custom main script .txt file and make any necessary changes in order to get the script to work with your device(s). The .txt file is well commented (documented) to provide instructions and assistance in making your modifications, however there are a few important sections at the top of the script file which require further mention.

You may see the following line at the very top of the script:

```
Attribute VB_Name = "Custom_Activity_Template"
```

This is the Visual Basic module name for the custom activity main script file when viewing within the Visual Basic (or equivalent) development environment. This line won't appear if editing the file in the VB development environment, but may do in others. `Custom_Activity_Template` is the name given to the custom template file module.

If you have a number of custom activity main script files, you may want to consider changing this name to reflect your custom activity main script file names. By doing so, you can then open multiple custom activity main scripts (say within a project) in your VB development environment.

If you are using a text based editor to modify your scripts , then changing the module name is optional, although to avoid confusion you may prefer to change the name anyway.

SCRIPT NOTES

The SCRIPT NOTES section provides useful information and lists tasks that are required to be carried out during the initial script creation phase. You can edit this section as and when you feel it is appropriate. An example may be deleting the initial setup tasks block of text from the SCRIPT NOTES section after you have carried out the initial script setup tasks, as it no longer applies.

Any complex routines or script specific behavior which may be of assistance to any person maintaining the activity scripts, should be mentioned in the SCRIPT NOTES section. They should be either fully documented in the SCRIPT NOTES section, or a reference made to the relevant function where they are fully documented.

ACTIVITY NOTES

The ACTIVITY NOTES section is an area where you can enter any activity specific behavior or quirks which may be of assistance to any person maintaining the activity scripts. An example may be: "*The activity only works for Cisco IOS devices*".

Required functions

There is only one function that must exist within your custom activity main script in order for CatTools to run the activity: `Function Main()`

You may also define your own Functions or Sub routines in the activity main script, however they must be called from within `Function Main()`. It is good practice to comment your Functions and Sub routines, so that you or other developers can get an idea of what the function does at a later date.

Saving the script file

Before CatTools can access the script file you have built, you must save it to the \Scripts sub folder. Normally you do not have to stop and restart CatTools in order for script file code changes to be picked up, however there may be occasions during testing that you find you may have to do this if the changes are not being recognized.

Testing your custom activity scripts

Once you have all the custom activity scripts set up, you can test the new activity with CatTools by setting up the activity to run with an existing device. The device selected must have had a [.custom](#) file created and contain the necessary code relevant to run your activity, otherwise you will see a error be logged to the Info Log of:

```
"Client script error: Type Mismatch: [activity name] on line: [line number]"
```

See [Testing your custom activity](#) for more information and tips.

Send us your working scripts

Once you have successfully created support for your custom activity in CatTools, if you would like us to consider adding it as a predefined activity type to ship with the product, then please send your custom client and main script .txt files, the custom activity type .ini file and also an example of a device .custom file which references the custom activity.

Please send as an attachment using the [Technical Support](#) form on our [website](#).

There are no guarantees as to when or if the activity will be added to the predefined activity types in CatTools, as there are a number of factors to consider: e.g. the number of requests for the activity, complexity of the script, technical resources available. However all scripts that are sent in will be cataloged for future reference.

The custom activity client script file (.txt)

Activity client scripts are defined within CatTools using text files (.txt files) and are stored in the `\Scripts` sub folder within the root CatTools directory. When an activity is run, CatTools reads the `script_client` item value in the `[Activity]` section from the activity type .ini file to determine which activity client script file it needs to use.

i SolarWinds recommends that in order to save confusion, the .ini file and .txt files have the same file name apart from the file prefix of 'Client.' for the .txt script file)

For all the predefined activity types in CatTools, the associated activity client script .txt files have been encrypted. They are encrypted for two reasons. The first is to protect our intellectual property. The other is to prevent unauthorized modification of these files which may cause the scripts to fail at run time.

The custom activity client script .txt template

CatTools provides a starting point client script template file to help assist in the creation of a new custom activity client script. The template file is called `Client.Custom.Activity.Template.txt` and can be found in the `\Templates` sub folder within the root CatTools directory.

This template file is included as part of the predefined activity scripts in CatTools, but is unencrypted. As for all of the CatTools predefined activity scripts, it may (as and when required) be subject to updates and modifications with each new release of the CatTools product.

For this reason, you should never modify the template file itself in order to create a new activity custom client script. If you need to create a new custom activity client script, take a copy the template file and save it to the `\Scripts` sub folder giving it a new file name.

Creating your custom activity client script .txt file

If adding a new custom activity client script to CatTools, the first step will therefore be to take a copy of the template file `Client.Custom.Activity.Template.txt` in the `\Templates` sub folder and save it to the `\Scripts` sub folder, giving it with a new file name. You need to ensure the file name you use is the value entered for the `script_client` item within the activity type .ini file `[Activity]` section.

This gives you a new starting point device script file to work with.

Editing the client script .txt file

The next step is to open the custom client script .txt file and make any necessary changes in order to get the script to work with your device(s).

The .txt file is well commented (documented) to provide instructions and assistance in making your modifications, however there are a few important sections at the top of the script file which require further mention.

You may see the following line at the very top of the script:

```
Attribute VB_Name = "Act_Custom_Template"
```

This is the Visual Basic module name for the custom activity client script file when viewing within the Visual Basic (or equivalent) development environment. This line won't appear if editing the file in the VB development environment, but may do in others.

`Act_Custom_Template` is the name given to the custom template file module. If you have a number of custom activity client script files, you may want to consider changing this name to reflect your custom activity client script file names. By doing so, you can then open multiple custom activity client scripts (say within a project) in your VB development environment.

If you are using a text based editor to modify your scripts, then changing the module name is optional, although to avoid confusion you may prefer to change the name anyway.

SCRIPT NOTES

The SCRIPT NOTES section provides useful information and lists tasks that are required to be carried out during the initial script creation phase. You can edit this section as and when you feel it is appropriate. An example may be deleting the initial setup tasks block of text from the SCRIPT NOTES section after you have carried out the initial script setup tasks, as it no longer applies.

Any complex routines or script specific behavior which may be of assistance to any person maintaining the activity scripts, should be mentioned in the SCRIPT NOTES section. They should be either fully documented in the SCRIPT NOTES section, or a reference made to the relevant function where they are fully documented.

ACTIVITY NOTES

The ACTIVITY NOTES section is an area where you can enter any activity specific behavior or quirks which may be of assistance to any person maintaining the activity scripts. An example may be: "The activity allows 3 Options:...", and then goes on to list what each option does and its input values.

Required functions

There is only one function that must exist within your custom activity client script in order for CatTools to run the activity:

```
Function Client()
```

You may also define your own Functions or Sub routines in the activity client script, however they must be called from within `Function Client()`. It is good practice to comment your Functions and Sub routines, so that you or other developers can get an idea of what the function does at a later date.

Saving the script file

Before CatTools can access the script file you have built, you must save it to the `\Scripts` sub folder. Normally you do not have to stop and restart CatTools in order for script file code changes to be picked up, however there may be occasions during testing that you find you may have to do this if the changes are not being recognized.

Testing your custom activity scripts

Once you have all the custom activity scripts set up, you can test the new activity with CatTools by setting up the activity to run with an existing device. The device selected must have had a `.custom` file created and contain the necessary code relevant to run your activity, otherwise you will see a error be logged to the Info Log of:

```
"Client script error: Type Mismatch: [activity name] on line: [line number]"
```

See [Testing your custom activity](#) for more information and tips.

Send us your working scripts

Once you have successfully created support for your custom activity in CatTools, if you would like us to consider adding it as a predefined activity type to ship with the product, then please send your custom client and main script `.txt` files, the custom activity type `.ini` file and also an example of a device `.custom` file which references the custom activity. Please send as an attachment using the [Technical Support](#) form on our [website](#).

There are no guarantees as to when or if the activity will be added to the predefined activity types in CatTools, as there are a number of factors to consider: e.g. the number of requests for the activity, complexity of the script, technical resources available, etc; however all scripts that are sent in is cataloged for future reference.

The custom activity device script file (.custom)

Devices scripts are defined within CatTools using text files (`.txt` files) and are stored in the `\Scripts` sub folder within the root CatTools directory.

When an activity is run, CatTools reads the name item value in the [device] section from the device type .ini file to determine which device script .txt file it needs to use. Therefore each .txt file must be given the same file name as the name item in the corresponding .ini file.

For all the predefined device types in CatTools, the associated device script .txt files have been encrypted. They are encrypted for two reasons. The first is to protect our intellectual property. The other is to prevent unauthorized modification of these files which may cause the script to fail at run time.

Device .custom file extension scripts

With the introduction of custom activity scripting in CatTools, in order to facilitate the adding of custom activities to the predefined device types, CatTools now checks for the existence of an associated device .custom file in the `\Scripts` folder, after loading the encrypted device .txt script file.

If when running an activity an associated device .custom file is found for a device type, the contents of the .custom file are read and appended to the predefined device type encrypted script contents, creating a temporary extended device type script, which is then used for the activity.

The device .custom file and the device encrypted files are merged in this manner every time an activity is run.

A device .custom file is associated with its encrypted script by file name, so for example, to add a .custom file for a Cisco Router, `Cisco.Router.General.txt`, you need to create a file called `Cisco.Router.General.txt.custom` in the `\Scripts` folder.

Whenever you upgrade or reinstall CatTools on your system, the predefined device type and activity type script files are overwritten. By defining your custom activity function calls for devices in a separate .custom file, it ensures that your custom functions do not get inadvertently lost.

The device .custom template

CatTools provides a starting point template file to help assist in the creation of a new custom activity device .custom file. The template file is called `Custom.Device.Template.txt.custom` and can be found in the `\Templates` sub folder.

This template file is included as part of the predefined scripts in CatTools, but is unencrypted. As for all of the CatTools predefined scripts, it may (as and when required) be subject to updates and modifications with each new release of the CatTools product.

For this specific reason, you should never modify the template file itself in order to create a new device .custom script.

If you need to create a new device .custom script, then take a copy the template file and save it to the `\Scripts` sub folder giving it a new file name.

Creating your device .custom script file

If adding a new .custom script to CatTools, the first step will therefore be to take a copy of the template file `Custom.Device.Template.txt.custom` in the `\Templates` sub folder and save it to the `\Scripts` sub folder, giving it with a new file name. You need to ensure you save the file using the same file name, including .txt extension, as its associated encrypted device script file, and append a suffix `'.custom'`.

This gives you a new starting point device script file to work with.

Example

You have created a new custom activity called "Custom_Report_PortStatus" and want to use this activity with your Cisco IOS switch devices. Your custom activity Client script file `Client.Custom.Report.PortStatus.txt` needs to call the function `Custom_Report_GetPortStatus()` in order to send a command, capture port data and parse the results to send back to the activity Client script.

To add this custom activity to the Cisco IOS switches, device type script file `Cisco.Switch.IOS.txt`, you need to perform the following steps:

1. Create a .custom file for the Cisco IOS switch device type. Copy the .custom template file `Custom.Device.Template.txt.custom` from the `\Templates` sub folder and save it to the `\Scripts` sub folder with the file name of `Cisco.Switch.IOS.txt.custom` in order for it to be merged with the `Cisco.Switch.IOS.txt` script at run-time.
2. Open the .custom file and delete the example functions.
3. Add your custom activity function `Custom_Report_GetPortStatus()` and code. Initially, you may just want to test the function is being called correctly, so below is a simple example of the code needed to add an 'Info' message in the Info Log from within your function.

```
Function Custom_Report_GetPortStatus()  
    cl.Log 3, "Testing custom activity: Custom_Report_PortStatus"  
End Function
```

4. Save the file.

Editing the .custom file

There are only three issues to be aware of with regards to editing the .custom script file. After copying the .custom template file in the `\Templates` sub folder to the `\Scripts` sub folder:

1. Do not delete the first line in the template file: `Attribute VB_Name = "Dev_CustomDeviceTemplate_custom"`

You can modify the value between the quotes, but it must always be defined as the `Attribute VB_Name`, otherwise a run-time error is likely to occur.

2. SolarWinds recommends you do not enter any uncommented program code in the header section and most importantly do not add any above the `Attribute VB_Name` line, otherwise your code may be stripped out or alternatively you may encounter a run-time error.

The SCRIPT NOTES section within the `.custom` file header is there for your own reference. Please feel free to add your own comments in here if it is of assistance to yourself or another developer at a later date.

3. Ensure that all program code added is contained within either a Function or a Sub routine. To avoid run-time errors or the risk of redefining a variable or object which is used in the predefined encrypted device script, try keep variable and object declarations limited to the scope of a Function or Sub routine. It is good practice to comment your Functions and Sub routines, so that you or other developers can get an idea of what the function does at a later date.

Saving the .custom script file

Before CatTools can access the `.custom` script file you have built, you must save it to the `\Scripts` folder. Normally you do not have to stop and restart CatTools in order for script file code changes to be picked up, however there may be occasions during testing that you find you may have to do this if the changes are not being recognized.

Testing your custom device .custom scripts

Once you have the scripts set up you can test the custom activity with CatTools by setting up the activity and selecting the device you have created the `.custom` file for.

See [Testing your custom device](#) for more information and tips.

Send us your working scripts

Once you have successfully created support for your custom device in CatTools, if you would like us to consider adding it as a predefined device type to ship with the product, then please send your custom client script `.txt` file and custom device type `.ini` file as an attachment using the [Technical Support](#) form on our [website](#).

There are no guarantees as to when or if the device is added to the predefined device types in CatTools, as there are a number of factors to consider, such as the number of requests for the device, complexity of the script, or technical resources available. All scripts that are sent in will be cataloged for future reference.

Testing your custom device

Testing your custom device scripts is relatively straight forward. Once you have the device type `.ini` file and the device script `.txt` file set up, you can test the new device with CatTools by setting up activities using the new device type. Start testing with is the [Device.ConnectivityTest.Login](#) activity to ensure you can access the device. Nearly all of the activities that run on a device require logging in to the device, therefore if the `Device.ConnectivityTest.Login` activity fails, it is likely the other activities will also fail.

Info Log messages

Within the Info Log pane you see messages appear while an activity is running. The level of messages that appear can be filtered by the drop-down list near the bottom of the Info Log window. By selecting level "4) Show Debug events and above" you get to see all the [cl.log](#) messages being displayed during the running of an activity.

You should be aware that all the Info Log messages are logged to a file called `InfoLog.txt` in the root CatTools folder. This file can get very large very quickly, so can be purged by selecting the [File | Delete | Delete InfoLog.txt](#) file menu item from the CatTools File menu.

The client script function [cl.Log 4](#), "message" is found throughout device scripts to display level 4 messages which help assist in troubleshooting device specific issues.

Debug log

Under the File menu of CatTools there is an entry [Enable Capture Mode](#). This turns on the logging of the conversation between CatTools and the end device and creates a disk file in the `\Debug` sub folder of the communications. This can be extremely useful when the script does not appear to be communicating correctly with the device after entering a command. The scripting mechanism waits for known prompts when issuing a command to the device, but eventually times out if an expected prompt does not appear. The debug log capture mode file shows what the device actually sent to CatTools, and you can adjust your code accordingly.

Custom activity scripts `cl.` and `ct.` variables and functions

The custom activity client script and main script files contain references to the internal CatTools variables, functions and sub routines. These variables, functions and sub routines can be identified by their `cl.` - client - or `ct.` - main CatTools program - prefixes.

Functions, variables and sub routines are not documented within the template scripts themselves, to cut down the size of the script files. However, each variable, function and sub routine exposed in the custom activity template files are detailed within the following sub pages of this chapter:

- [Client script file `cl.` functions and variables](#)
- [Main script file `ct.` functions and variables](#)

Client script - cl. variables and functions

Below is a list of CatTools client `cl.` variables, functions and sub routines, with some examples of how to implement them in your custom activity client script files.

This is not a full set of all the internal client variables and functions used within the CatTools application. It contains details of those from within the client script template files, plus a few additional ones which may be useful when creating your custom activity client scripts.

Variables

<code>cl.AppPath</code>	The application path (install directory) of the CatTools program.
<code>cl.ScheduleNumber</code>	The current schedule number.
<code>cl.DeviceName</code>	The name of the device currently connected to.
<code>cl.</code>	<p>The connection status for the instance of the protocol engine control <code>TelnetConnectionStatus</code> client thread.</p> <p>For example: Test current client connection status and exit function if status is not 'OK'.</p> <pre style="border: 1px solid black; padding: 5px;"> If StrComp(cl.TelnetConnectionStatus, "OK", vbTextCompare) <> 0 Then Exit Function </pre>
<code>cl.QuitNow</code>	A variable that is checked in a number of places within an activity to stop it running. For example, it is set to 1 when the 'STOP' button is pressed while an activity is running. This variable can only be referenced from within activity Client Scripts. Use the <code>ct.QuitNow</code> variable to reference within Main Scripts.

Functions and Sub routines

General and File

<code>cl.Log</code>	Sends a line of text to the <code>Infolog.txt</code> file and Info Log pane.
<code>cl.LogToFile</code>	Writes data to a file.
<code>cl.FileExists</code>	Checks for the existence of a file.
<code>cl.CreateFile</code>	Create a new file using a given filename and path.
<code>cl.DeleteFile</code>	Delete a file using a given filename and path.

<code>cl.ReadFromFile</code>	Read in contents of a given file.
<code>cl.ReplaceClientFilenameVariables</code>	Replace filename variables in client report filenames or meta variables with real text values.
<code>cl.SaveResults</code>	Save results to a temporary .txt file within the \ClientTemp folder.

Activity

<code>cl.DBCheckScheduleOption</code>	Determine whether a particular option has been selected within a given activity.
<code>cl.DBScheduleGetField</code>	Get activity field value from the CatTools database

Device

<code>cl.DisconnectHost</code>	Disconnect instance of the (protocol engine control) client thread from the end device.
--------------------------------	---

Functions and Sub routines - Details and examples

The functions and sub routines listed in the above table are further detailed below with their input parameters and examples of their usage are provided.

`cl.Log (iPriority, sMessage)`

This sub routine sends a line of text to the Infolog.txt file and [Info Log pane](#). It has two input parameters:

<code>iPriority</code>	Integer range 1 to 4 representing the logging level for the line being sent. <ul style="list-style-type: none"> • 1=Error • 2=Warning • 3=Information • 4=Debug
<code>sMessage</code>	Message text of the line being sent.

Example

Log a level 3 (info) message if the 'Use alternate command' field in the activity [Options tab](#) is empty.

```

Dim sAltCommand
sAltCommand = cl.DBScheduleGetField(cl.ScheduleNumber, "OptionsString1")
If Len(Trim(sAltCommand)) = 0 Then
    cl.Log 3, "No alternate command specified"
End if
    
```

cl.LogToFile (sFileName, sData, bAppend)

This sub routine writes data to a given file. It has three input parameters:

sFileName	String of the filename and path of the file to write data to.
sData	String of data to write.
bAppend	Boolean value. If bAppend is not 0 - not False - then the file is appended to.

Example

Log a level 2 (warning) message to the Info Log and write a line to a text file in the Reports folder if the 'Use alternate command' field in the activity Options tab is empty.

```

Dim sLogResultsFile
Dim sAltCommand

sLogResultsFile = cl.AppPath & "Reports\" & cl.UniqueFileID & ".txt"
sAltCommand = cl.DBScheduleGetField(cl.ScheduleNumber, "OptionsString1")

If Len(Trim(sAltCommand)) = 0 Then
    cl.Log 2, "No alternative command specified"
    Call cl.LogToFile(sLogResultsFile, "No alternative command specified in
activity Options tab", 1)
End if
    
```

cl.FileExists(sFileName) As Boolean

This function checks for the existence of a given file. The function returns `True` if the file was found, `False` if not. It has one input parameter:

sFileName	String of the filename and path of the file that existence is being tested for.
-----------	---

cl.CreateFile(sFileName) As Boolean

This function is used to create a new file using the given filename and path. The function returns `True` if the file already exists, or the file creation was successful, else it returns `False`. It has one input parameter:

<code>sFileName</code>	String of the filename and path of the file to create.
------------------------	--

cl.DeleteFile(sFileName) As Boolean

This function is used to delete a file using the given filename and path. The function returns `True` if the file has been deleted successfully, else it returns `False`. It has one input parameter:

<code>sFileName</code>	String of the filename and path of the file to delete.
------------------------	--

cl.ReadFromFile(sFileName) As String

This function is used to read in the contents of a given file. The function returns a string of the file data. It performs the same operation as the `CatTools` function `ct.ReadFromFile`, but can only be called from within an activity Client Script. It has one input parameter:

<code>sFileName</code>	String of the filename and path of the file to read in the contents of.
------------------------	---

Example

Read in the contents of a given file, in this case a list of CLI commands.

```
Dim sFileName
Dim sCommandList

If cl.DBCheckScheduleOption(cl.ScheduleNumber, 1) = 0 Then
    ' Option selected to load CLI commands to send to device from a file
    sFileName = cl.DBScheduleGetField(cl.ScheduleNumber, "OptionsString2")
    sCommandList = cl.ReadFromFile(sFileName)
End If
```

cl.ReplaceClientFilenameVariables(sData) As String

This function is used to replace any [filename variables](#) in the client report filenames or [meta variables](#) in the commands to be sent to the device, with the actual text values. It returns the modified string with the variables replaced. It has one input parameter:

<code>sData</code>	String of the filename and path of the file; or command list being checked.
--------------------	---

Example

Replace any filename variables of a given file, then read in its contents (in this case a list of CLI commands). Next, replace and meta variables within the command list.

```

Dim sFileName
Dim sCommandList

If cl.DBCheckScheduleOption(cl.ScheduleNumber, 1) = 0 Then
    ' Option selected to load CLI commands to send to device from a file
    sFileName = cl.DBScheduleGetField(cl.ScheduleNumber, "OptionsString2")

    ' Replace any filename variables
    sFileName = cl.ReplaceClientFilenameVariables(sFileName)

    ' Read in the file contents
    sCommandList = cl.ReadFromFile(sFileName)

    ' Replace meta (command) variables with values
    sCommandList = cl.ReplaceClientFilenameVariables(sCommandList)
End If
    
```

cl.SaveResults(sData, bAppend)

This sub routine saves the results to a temporary .txt file within the \ClientTemp folder. It has two input parameters:

sData	String of data to write.
bAppend	Boolean value. If bAppend is 0 (False), then the file will be overwritten, otherwise it is appended to.

Example

Save current device response `sResults` to a temporary .txt file overwriting any existing data.

```
Call cl.SaveResults(sResults, 0)
```

cl.DBCheckScheduleOption(lScheduleNumber, lOptionNumber) As Long

This function is used to determine whether particular options have been selected within a given activity. It has two input parameters:

lScheduleNumber	The current schedule number as a Long.
-----------------	--

<code>lOptionNumber</code>	The option item within the activity we are checking as a Long. <code>lOptionNumber</code> must be between 1 and 10 (inclusive).
----------------------------	--

Example

Check an activity, in this instance a [Device.CLI.Send commands](#) activity, to see if the output of the commands should be emailed.

```
lRetVal = cl.DBCheckScheduleOption(cl.ScheduleNumber, 8)
If lRetVal = 1 then
    ' code to email commands goes here...
End if
```

`cl.DBScheduleGetField(IScheduleNumber, sFieldName)`

This function is used to get a field value from the CatTools database for a given activity. It performs the same operation as the CatTools function `ct.DBScheduleGetField`, but can only be called from within an activity Client Script. It has two input parameters:

<code>IScheduleNumber</code>	The current schedule number as a Long.
<code>sFieldName</code>	The field name in the database we want to get the value of as a String.

Example

Used in conjunction with the `cl.DBCheckScheduleOption` function above, check an activity to see if the 'Use alternate command' check-box is selected in the activity [Options tab](#). If it is, it gets the value from the database for the alternate command we want to send.

```
' Check the option 1 check-box to see if selected - i.e. set to 1
If cl.DBCheckScheduleOption(cl.ScheduleNumber, 1) = 1 Then
    ' Get value from database for associated string field (being field
"OptionsString1" in the
    database) to set a variable value
    sAltCommand = cl.DBScheduleGetField(cl.ScheduleNumber, "OptionsString1")
End If
```

`cl.DisconnectHost`

This sub routine is used to disconnect the instance of the (protocol engine control) client thread from the end device. It should be called after the device has been logged out of to disconnect the client thread gracefully.

Main script - ct. variables and functions

Below is a list of CatTools application 'ct.' variables, functions and sub routines, with some examples of how to implement them in your custom activity main script files. This is not a full set of all the internal ct. variables and functions used within the CatTools application, however it contains details of those from within the main script template files, plus a few additional ones which may be useful when creating your custom activity main scripts.

Variables

<code>ct.ScheduleNumber</code>	The current schedule number.
<code>ct.Devices</code>	A pipe () delimited string of devices to connect to (i.e. the devices associated with the current schedule).
<code>ct.ScriptFileClient</code>	A string variable to store the file name of the client script file that is used by the activity.
<code>ct.QuitNow</code>	A variable that is checked in a number of places within an activity to stop it running. For example, it is set to 1 when the 'STOP' button is pressed while an activity is running. This variable can only be referenced from within an activity Main Script. Use the <code>cl.QuitNow</code> client variable is referencing within the Client Scripts.

Functions and Sub routines - Summary list

General and File

<code>ct.Log</code>	Sends a line of text to the Infolog.txt file and Info Log pane
<code>ct.ReadFromFile</code>	Read in contents of a given file
<code>ct.ReplaceFilenameVariables</code>	Replace filename variables in report filenames to real text values.
<code>ct.CreateMasterTable</code>	Creates a master table file.
<code>ct.CreateAndSendReport</code>	Creates report files (.txt .html, etc.) and sends by email.
<code>ct.RemoveClientResults</code>	Removes existing client temporary .txt files within the \ClientTemp folder.

Activity

<code>ct.MarshalThreads</code>	Manages the client threads used within the activity schedule.
<code>ct.DBScheduleGetField</code>	Get activity field value from the CatTools database.

Device

<code>ct.RemoveHeader</code>	Removes the first line of data from within a data block.
------------------------------	--

Functions and Sub routines- Details and examples

The functions and sub routines listed in the above table are further detailed below with their input parameters and examples of their usage are provided.

`ct.Log(sDevice, iPriority, sMessage)`

This sub routine sends a line of text to the Infolog.txt file and [Info Log pane](#). It has three input parameters:

<code>sDevice</code>	The name of the device as a string. If an empty string, or <code>VBNullString</code> , then the string value is "CatTools" referring to an internal application related log message rather than a device specific message.
----------------------	--

<code>iPriority</code>	Integer range 1 to 4 representing the logging 'level' for the line being sent.
------------------------	--

- 1 = Error
- 2 = Warning
- 3 = Information
- 4 = Debug

<code>sMessage</code>	Message text of the line being sent.
-----------------------	--------------------------------------

Example

Log a level 1 (error) message for CatTools, rather than any specific device, such as `sDevice= ""`, with the text "No devices have been selected".

```
ct.Log "", 1, "No devices have been selected"
```

`ct.ReadFromFile(sFileName) As String`

This function is used to read in the contents of a given file. The function returns a string of the file data. It performs the same operation as the client function `cl.ReadFromFile`, but can only be called from within an activity Main Script. It has one input parameter:

<code>sFileName</code>	String of the filename and path of the file to read in the contents of.
------------------------	---

Example

Read in the contents of a file called "ReportData", removing the first row header row.


```
Dim sReportFile
Dim sReportData

' Get the file name and path from the activity settings and replace any
filename variables.
sReportFile = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportFile")
sReportFile = ct.ReplaceFilenameVariables("", ReportFile)

' Read the contents of the report file, remove the first line of data (header
row) and save the results to sReportData variable.
sReportData = ct.RemoveHeader(ct.ReadFromFile(sReportFile))
```

ct.ReplaceFilenameVariables(sDeviceName, sFileName) As String

This function is used to replace any filename variables in the report filename to the actual text values. It returns the modified string with the filename variables replaced. It has two input parameters:

sDeviceName	String value for the device name. Set sDeviceName to be an empty string ("") when you are translating an activity Report file name, as these should not contain device specific filename variables.
sFileName	String of the filename and path of the file being checked.

Example

Get the report filename for an activity and perform any [filename variable](#) replacements, then store the value in a string variable.

```
sReportFile = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportFile")
sReportFile = ct.ReplaceFilenameVariables("", sReportFile)
```

ct.CreateMasterTable(sReportFile, sHeader, bOverwrite, sDeviceList, sClientFilePrefix, sTableFormat) As Boolean

This function is used to create the master table file. If successful, it returns a value of `True`, else if not then returns `False`. It has six input parameters:

sReportFile	String value of the report/master table file name and path to create
sHeader	Tab delimited string for the report header as defined in the activity Main Script.
bOverwrite	Boolean value. If 0 (False) the file will be appended to, otherwise it is overwritten.

sDeviceList	Pipe () delimited string of devices associated with the current schedule (use the "ct.Devices" variable)
sClientFilePrefix	File name prefix string for the temporary client (device) files that the activity creates in \ClientTemp in order to consolidate for the Master table file.
sTableFormat	Pipe () delimited string used for those activities where you can modify the table definition output (change column names; include/exclude columns from the output; and change the table column order: For example, the Options tab of the Report.SNMP.System Summary activity).

Example

Create the master table for a version report. Get the Report file name and Overwrite values from the activity settings in the database; and define the report header. Use a prefix of "CustomReportTemplate" for each temporary client (device) file.

```
Dim sReportFile
Dim bOverwrite
Dim sHeader

sReportFile = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportFile")
sReportFile = ct.ReplaceFilenameVariables("", ReportFile)

bOverwrite = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportOverwrite")
sHeader = "Group" & vbTab & "Device Name" & vbTab & "IP Address" & vbTab &
"Serial #" & vbTab & "_Processor" &
vbTab & "IOS" & vbTab & "Uptime"

' Create the master table
bRetVal = ct.CreateMasterTable(sReportFile, sHeader, bOverwrite, ct.Devices,
"CustomReportTemplate", "")
```

```
ct.CreateAndSendReport(sScheduleName, sHeader, sData, sReportFile)
```

This sub routine is used to create the report files (.txt .html, etc.) and if the relevant options are set, it will also send the reports by email. It has four input parameters:

sScheduleName	The schedule name used for the report title, and email message subject title.
sHeader	Tab delimited string for the report header as defined in the activity Main Script.
sData	String of the report data.

sReportFile	String value of the report file name and path to create.
-------------	--

Example

Create a simple version report. Get the Activity name and Report file name from the activity settings in the database; and define the report header.

```
Dim sClientFile
Dim sScheduleName
Dim bOverwrite
Dim sReportFile
Dim sHeader

' Prefix for temporary client file filenames
sClientFile = "CustomReportTemplate"

' Get the schedule name from the activity settings
sScheduleName = ct.DBScheduleGetField(ct.ScheduleNumber, "Name")

' Get overwrite option value
bOverwrite = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportOverwrite")

' Get report file name
sReportFile = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportFile")
sReportFile = ct.ReplaceFilenameVariables("", ReportFile)

' Define the report header
sHeader = "Group" & vbTab & "Device Name" & vbTab & "IP Address" & vbTab &
"Serial #" & vbTab & _"Processor" &
vbTab & "IOS" & vbTab & "Uptime"

' Create the master table
Call ct.CreateMasterTable(sReportFile, sHeader, bOverwrite, ct.Devices,
sClientFile, "")

' Call CatTools function to create and send the report
Call ct.CreateAndSendReport(sScheduleName & " (Custom Template Report)",
sHeader, ct.RemoveHeader
(ct.ReadFromFile(sReportFile)), sReportFile)

' Finally, clean up by deleting any temp client files
Call ct.RemoveClientResults(sClientFile)
```

ct.RemoveClientResults(sClientFile)

This sub routine removes any existing client temporary .txt files within the \ClientTemp folder, with a filename beginning with the string value of sClientFile. It has one input parameter:

sClientFile	String value of the file names to find.
-------------	---

ct.MarshalThreads(ByVal sDeviceList As String, ByVal sScriptFile As String, ByVals FunctionName As String, ByVal IMaxProcessCount As Long, ByVal sUniqueFileID AsString)

This sub routine manages the client threads used within the activity schedule. It has five input parameters:

sDeviceList	Pipe () delimited string of devices associated with the current schedule (use the "ct.Devices" variable for this parameter).
-------------	---

sScriptFile	String of the filename of client script file to use (specify the "ct.ScriptFileClient" variable for this parameter).
-------------	--

sFunctionName	String of the Function name within the client script file to use (normally this should be "Client").
---------------	--

IMaxProcessCount	Maximum number of client threads to use for the activity (this will be automatically restricted by your Edition of CatTools).
------------------	---

sUniqueFileID	File name prefix string for the temporary client (device) files that the activity creates in \ClientTemp within the activity.
---------------	---

Example

Start the client threads for a simple version report. Get the maximum client threads from the activity settings in the database.

```
Dim lMaxClientThreads

' Get the number of client threads from the activity settings
lMaxClientThreads = ct.DBScheduleGetField(ct.ScheduleNumber, "ClientThreads")

' Start the threads
Call ct.MarshalThreads(ct.Devices, ct.ScriptFileClient, "Client",
lMaxClientThreads, "CustomReportTemplate")
```

`ct.DBScheduleGetField(iScheduleNumber, sFieldName)`

This function is used to get a field value from the CatTools database for a given activity. It performs the same operation as the client function `cl.DBScheduleGetField`, but can only be called from within an activity Main Script. It has two input parameters:

<code>iScheduleNumber</code>	The current schedule number as a Long.
<code>sFieldName</code>	The field name in the database we want to get the value of as a String.

Example

Get the report filename for an activity and store the value in the string variable `sReportFile`.

```
sReportFile = ct.DBScheduleGetField(ct.ScheduleNumber, "ReportFile")
```

`ct.RemoveHeader(sData) As String`

This function is used to remove the first line of data in a given block of data. It does so by locating the first instance of a `vbCrLf`, and removes everything up to and including it. The function returns a string with the first data line removed. It has one input parameter:


<code>sData</code>	String of the data to manipulate.
--------------------	-----------------------------------

Testing your custom activity

Once you have the activity type (.ini) file, the activity [main script](#) and [client script](#) (.txt) files setup and have also created the device .custom file containing the device specific code for your custom activity; you simply create a new activity in CatTools, selecting your custom activity from the activity 'Type' drop-down list field, then in the 'Devices' tab you select the device which you created the .custom file for.

Info Log messages

Within the Info Log pane you see messages appear while an activity is running. The level of messages that appear can be filtered by the drop-down list near the bottom of the Info Log window. By selecting level "4. Show Debug events and above" you get to see all the [cl.log](#) messages being displayed during the running of an activity.

 All the Info Log messages are logged to a file called `InfoLog.txt` in the root CatTools folder. This file can get very large very quickly. Purge by selecting the [File | Delete | Delete InfoLog.txt file](#) menu item from the CatTools File menu.

The client script function `cl.Log 4, "message"` is found throughout device scripts to display level 4 messages which help assist in troubleshooting device specific issues.

The most likely `Level 1 - Error Info Log` message encountered when testing your new custom activity script is:

```
Client script error: Type Mismatch: [activity name] on line: [line number]
```

This error occurs if:


- the device you have selected to run your custom activity against does not have a `.custom` file defined for it
- the [.custom file](#) for the device does not contain a function which has been defined in your custom activity client script
- a function defined in the `.custom` file contains an error within the code.

If CatTools finds an associated `.custom` file for the device, you should see a `Level 4 - Debug Info Log` message:

```
Loading custom activities for device
```

If you do not see this message for your device, then CatTools was unable to find a `.custom` file because:

- The file does not exist.
- The file cannot be opened.
Verify the file permissions.
- A typo error has been made when naming the `.custom` file.

 You must have set your Info Log pane to "4. Show Debug events and above"

If you see the debug message above, then check the contents of the `.custom` file to ensure:

- The custom activity function exists.
- The custom activity has been spelled correctly.
- The code does not contain errors.

Add additional `cl.log 4, "..."` Info Log debug messages at strategic points in your code to determine where the script is failing.

Device does not support the custom activity

If the selected device does not support the custom activity, either deselect the device in the activity setup Devices tab, or add the following code to the custom activity Function within the .custom file for the device:

```
Function [YourCustomActivityFunctionName] ()
    Call cl.CatToolsNoSupport
End Function
```

Now, when you run this activity, the function is called, but the [cl.CatToolsNoSupport](#) internal CatTools function returns a Level 2 - Warning Info Log message:

```
Device type: [SCRIPT_NAME constant] has not had this functionality added yet.
Skipping this device
```

Other Info Log message errors

Other errors you may encounter are:

Variable is undefined

```
Client script error: Variable is undefined: '[variable name]' on line: [line
number]
```

This error occurs if:

- Your code contains a variable which has not been declared. Ensure you Dim or Redim.
- You variables correctly (check for typo errors in your variable declarations).
- Your code contains a call or reference to an internal CatTools variable or function (cl. or ct.), or to a class or type you have defined, but you mistyped 'cl.' or 'ct.' or the class name.

Object doesn't support this property or method

```
Client script error: Object doesn't support this property or method: '[
property or method]' on line: [line number]
```

This error occurs if:

- You are making a Function call or Sub routine call within the [.custom](#) file, but the Function or Sub routine does not exist. (check for typo errors in your code)

Wrong number of arguments or invalid property assignment

```
Client script error: Wrong number of arguments or invalid property
assignment: '[Function or Sub name]' on line: [line number]
```

This error occurs if:

- You are making a Function call or Sub routine call within the [.custom](#) file, but have supplied an incorrect number of arguments to the Function or Sub routine.

Debug log

Under the File menu of CatTools there is an entry [Enable Capture Mode](#). This turns on the logging of the conversation between CatTools and the end device and creates a disk file in the `\Debug` sub folder of the communications.

This can be extremely useful when the script does not appear to be communicating correctly with the device after making its initial connection. The scripting mechanism waits for known prompts and issues commands to the device at these points. It eventually times out if an expected prompt fails to appear.

The debug log capture file shows what the device is actually sending to CatTools, so you can adjust your code accordingly.

Unsupported activities for a device

Device types are updated all the time. For an up to date list of the devices types currently available in CatTools and the activities supported for each of the device types, please see the [device matrix](#).

If there is a listing to support your device, but the activity you require is not currently supported by the device type then please contact us using the [Technical Support](#) form, providing us with as much detail as you can and if possible a PuTTY capture of the relevant command(s).

Menus

The following menu items are available from the CatTools main menu:


- [File](#)
- [View](#)
- [Options](#)
- [Interface](#)
- [Help](#)

Kiwi CatTools File menu

The following menu items are available from the CatTools File menu:

Menu option	Description
Database	<ul style="list-style-type: none"> • Export: You can export activity and device details to flat files: <ul style="list-style-type: none"> ◦ Export devices to tab delimited file ◦ Export activities to tab delimited file • Import: You can import activity and device details from flat files: <ul style="list-style-type: none"> ◦ Import devices from tab delimited file ◦ Import schedules from tab delimited file • Change encryption password • Backup current database • Restore database from backup • Squeeze current database • Open other database • Create new database
Delete	Delete the info log file to recover disk space.
Enable capture mode	Capture mode can be used to help diagnose device connection or device activity problems. See Enable capture mode .
Debug	Create debugging information to send to Technical Support for debugging problems. See Create diagnostics for Technical Support .


Menu option	Description
Exit	Closes the CatTools program.

 If you are running the Service edition, the CatTools service continues run as a service in the background. The CatTools service can be stopped using the Windows 'Services' administration tool from the Control Panel.

Export devices to tab delimited file

Export your device information to a tab-delimited file, suitable for reporting using tools like MS-Excel or similar.

1. Go to File > CatTools database, and then click Export.
2. Export devices to tab delimited file.
3. In the file selection box, browse to select the folder to create the export file in.
4. Enter the name of the device export.
5. A dialog will prompt you to generate an encryption password for the database backup. Create a password and click OK.

 It is critical to remember the password you create. Without the password, your data will be lost and cannot be retrieved.

Export activities to tab delimited file

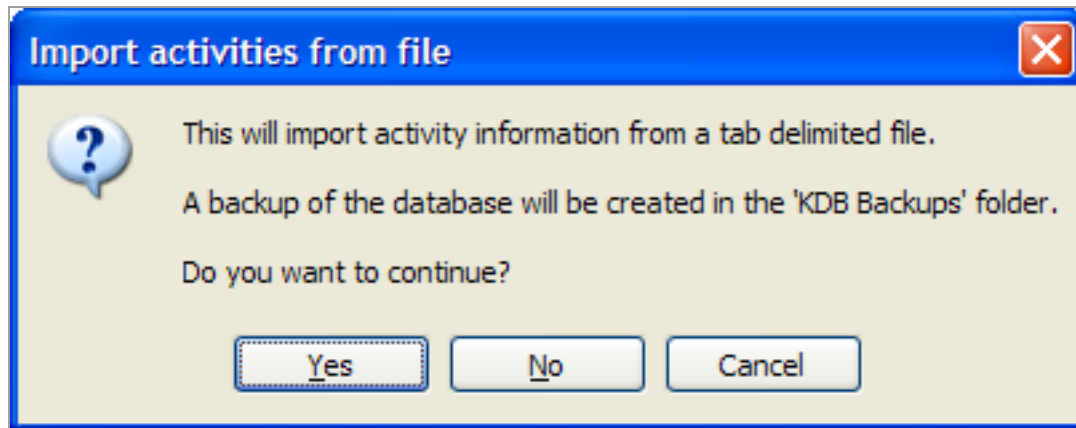
Export your activity data to a tab-delimited file, suitable for reporting using tools like MS-Excel or similar:

1. Go to File > CatTools database, and then click Export.
2. Export activities to tab delimited file.
3. In the file selection box, browse to select the folder to create the exported file in.
4. Enter the name of the activities export.


Import devices from tab delimited file

Import devices from a tab delimited file. The current database is backed up before the import operation occurs.

1. Go to File > CatTools database, and then click Import.
2. Select Import devices from a tab delimited file.
3. Confirm the import operation.



4. A dialog will prompt you to enter the encryption password for the database backup. Enter the password and click OK.

 To remove the password requirement, see [Change encryption password](#).

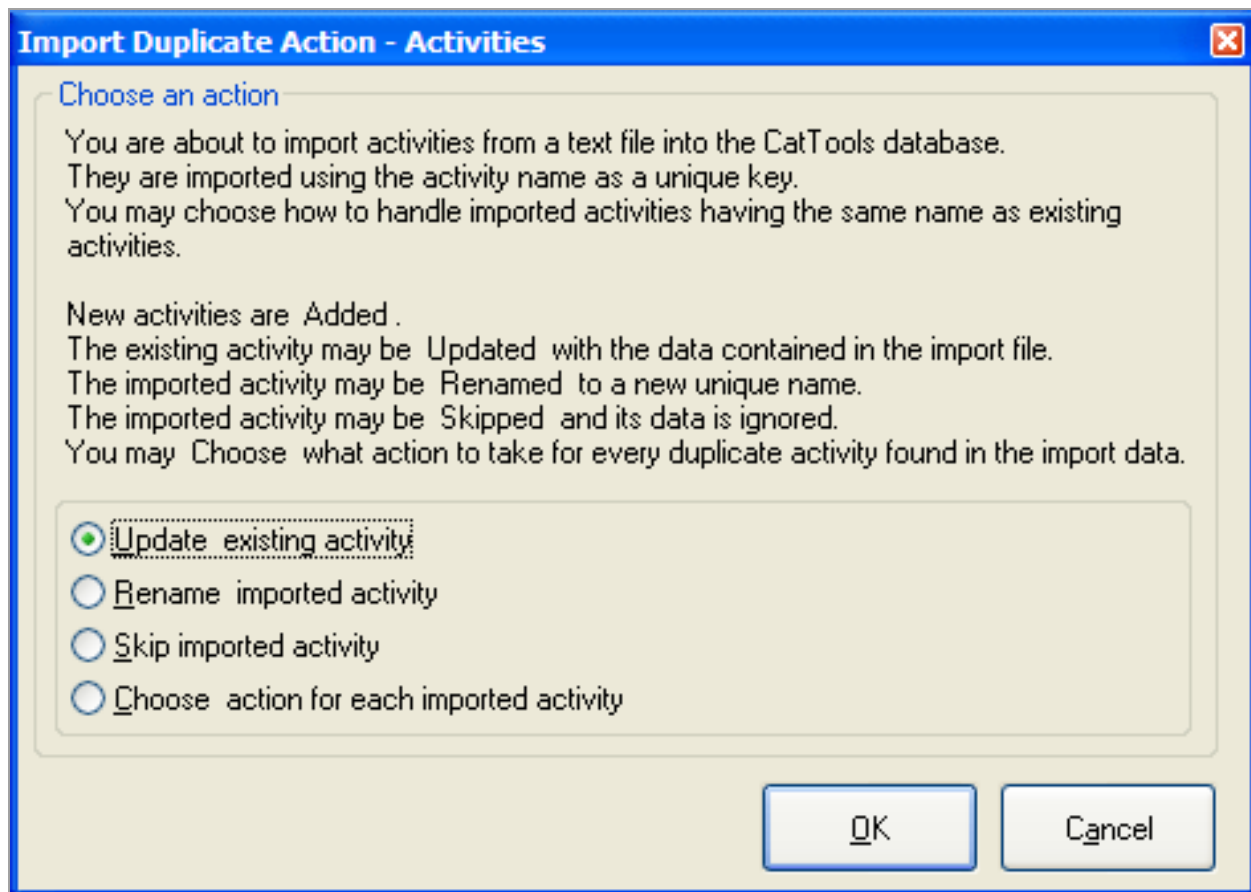
5. Select the file to import from and click Open.

By default, the file mask is set to "Export_Device.txt".

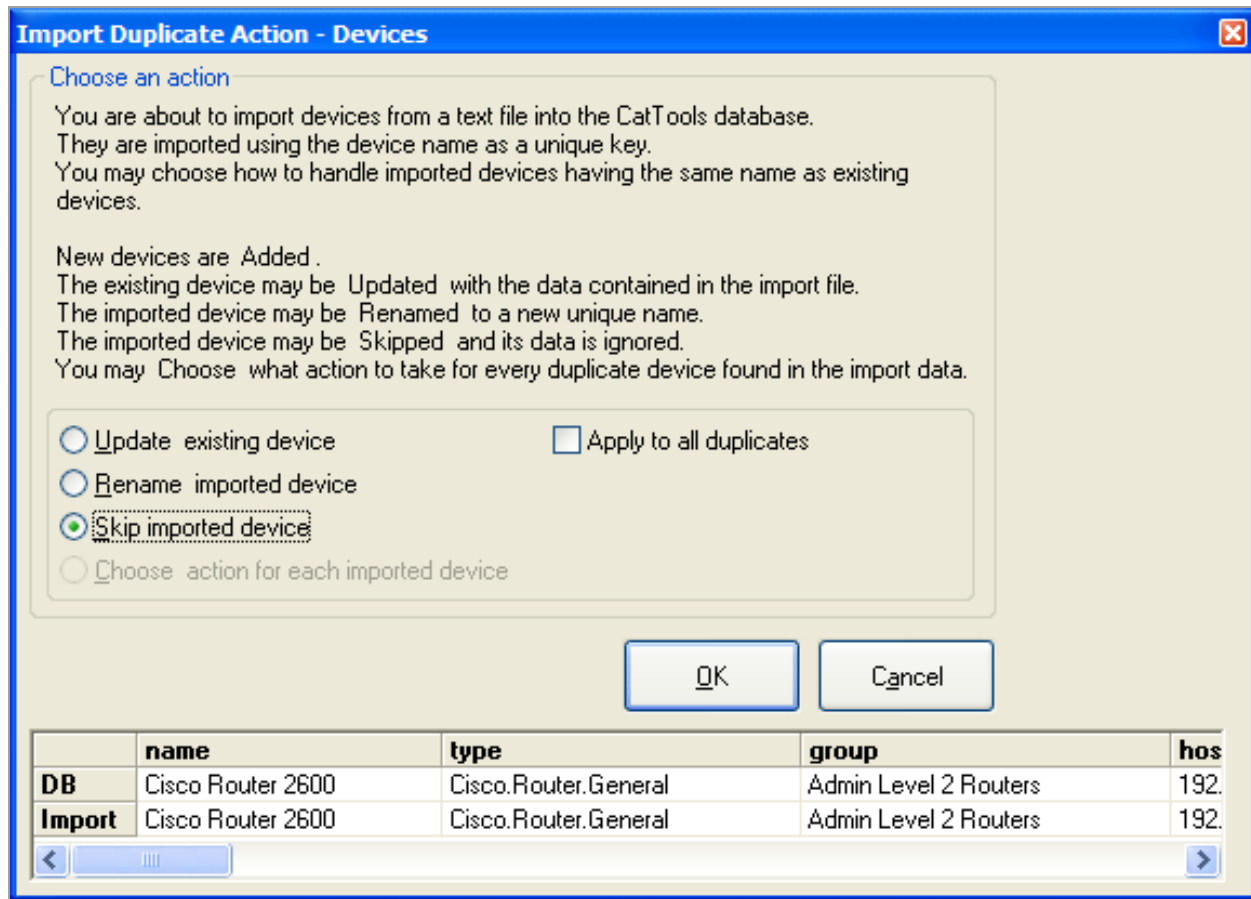
6. Determine how to handle imported devices when a duplicate is found:

 The import process always [adds a new device](#).

- a. **Update existing device** with the imported device's data. This overwrites the current device data with any fields that are found in the import data. Any fields that are not found in the import data are not changed in any way.
- b. **Rename imported device.** A new device is created with `_imported_0` appended to the current device name. The last character is a sequential digit that is incremented when more than 1 devices with the same name are encountered.
- c. **Skip imported device.** The imported device data is ignored.

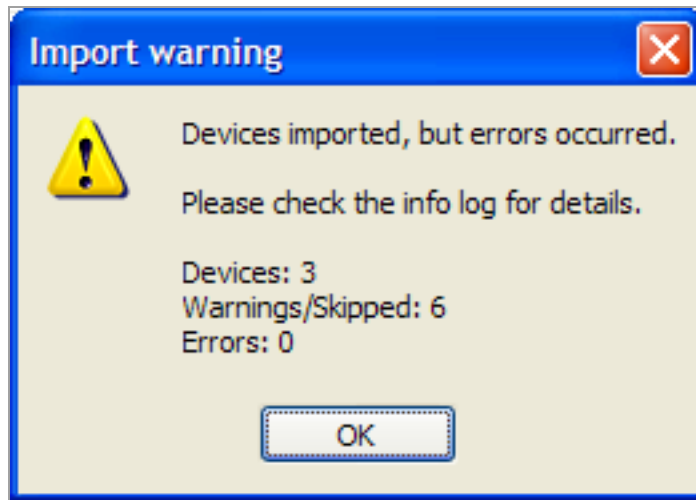



- Choose action for each imported device. When a duplicate name is found, the following dialogue is displayed:




You can view the differences between the two devices and choose what action to take for that particular device. You can also select `Apply to all duplicates`.

- When the import operation is complete, a confirmation message verifies many devices were imported from the file.



 Check the Info Log for more details on any warnings or errors received after importing.

- A dialog will prompt you to enter the encryption password for the database backup. Enter the password and click OK.

 If you want to remove the password requirement, see [Change encryption password](#).

Device import file format

The file must start with a header row that defines the fields. The rows following the header are treated as data.

Name	Type	Group	HostAddress	Filename	Model	ConnectVia	Telnet	TelnetPort
Cisco Router 2600	Cisco.Router.General	Test Group	127.0.0.1	Cisco_Router_2600	Cisco 2600	Direct connect	Telnet	23

The fields can be in any order, but must follow the same naming convention as the CatTools Export devices to tab delimited file function. At a minimum, the file must contain the following 3 fields:

- **Type**

The CatTools device type.

- **Name**


A unique name for the device.

- **HostAddress**

The IP address or hostname for the device.

There are additional device fields available to define:

Type	Group	Name	HostAddress
Filename Unique filename for device, no path or file extension required. If not specified, CatTools derives from the <code>Name</code> field.	Model	ConnectVia	Telnet
Session	VTYPass	ConsolePass	EnablePass
PrivilegeLevel	AAAUsername	AAAPassword	SSH Username
SSH Password	SNMPRead	SNMPWrite	RequireVTYLogin
LoginUsesAAA	EnableUsesAAA	VTYPrompt	ConsolePrompt
EnablePrompt	AAAUserPrompt	AAAPassPrompt	Address1
Address2	Address3	ContactName	ContactPhone
ContactEmail	ContactOther	AlertEmail	SerialNumber
AssetTag	Identification	SerialOther	ActivitySpecific1
ActivitySpecific2			

 To create a template which includes all the required fields you need to use, add a few example devices to CatTools first, and then [export to a tab delimited file](#).

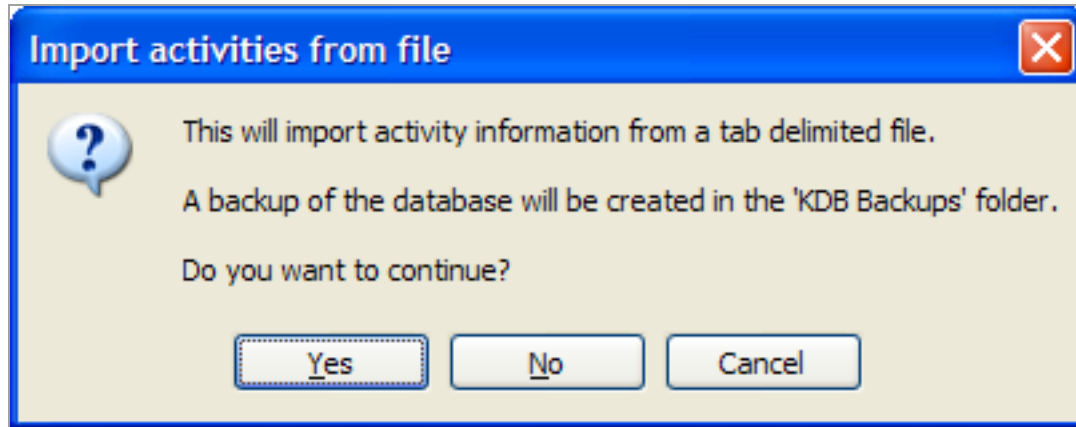
Plain text values found in fields that are normally encrypted within the database, such as HostAddress, VTYPass, EnablePass SNMPRead, are automatically be encrypted as part of the import process.

Import schedules from tab delimited file

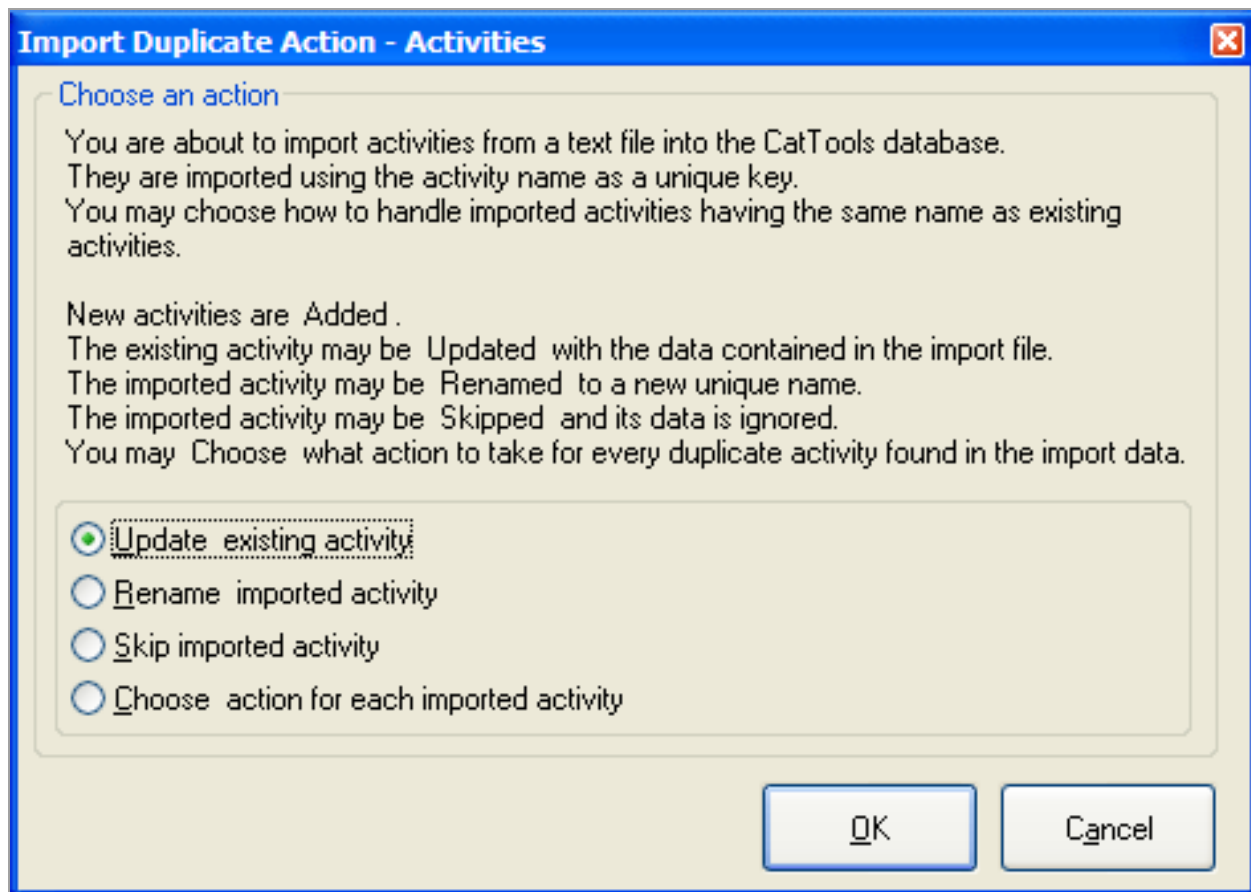
Import activities that have previously been exported from CatTools in tab delimited format.

1. Go to File > CatTools database, and then click Import.
2. Select Import devices from a tab delimited file.


3. Confirm the import operation.



4. Select the file to import from and click Open.
5. Determine how to handle imported activities, if a duplicate is found.
 - a. **Update existing activity** with the imported activity's data. This overwrites the current activity data with any fields that are found in the import data. Any fields that are not found in the import data are not changed in any way.
 - b. **Rename imported activity.** A new activity is created with `_imported_0` appended to the current device name. The last character is a sequential digit that is incremented when more than 1 activity with the same name is encountered.
 - c. **Skip imported activity.** The imported device data is ignored.




- When the import operation is complete, a confirmation message verifies many activities were imported from the file.

 Check the Info Log for more details on any warnings or errors received after importing.

- Choose action for each imported activity. When a duplicate name is found, you can view the differences between the two activities and choose what action to take for that particular activity. You can also select `Apply to all duplicates`.

Change encryption password

CatTools employs industry standard AES encryption to ensure that the contents of your device fields are protected from prying eyes. As an additional security measure, you can require a password when CatTools is run, further limiting unauthorized access to the program.

 The password is used to encrypt the device fields. If you forget the password, you can not use the program and cannot access the device data.

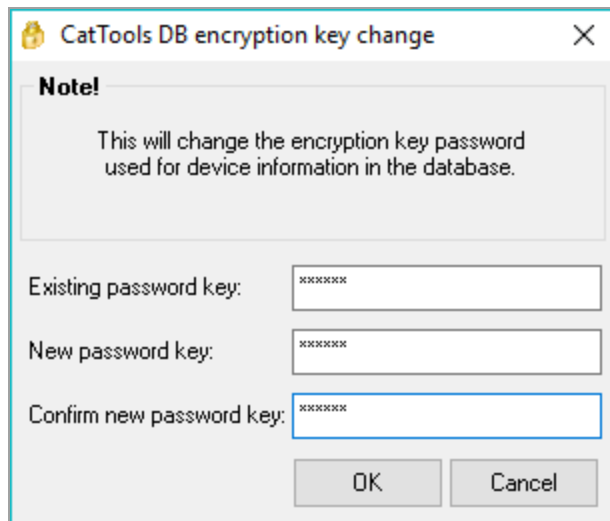
When encrypted, the Application and Manager prompts for the password on startup. The Service then opens the database and runs, regardless of whether a password is set or not.

- Resetting the password back to a blank value stops the program from prompting you for a password.
- Even with no password set, the device database is still securely encrypted.

Changing the password

To change your encryption password:

1. Go to File > CatTools database, and select Change encryption password.
2. To change the password, you must enter your existing password, if one has been set.

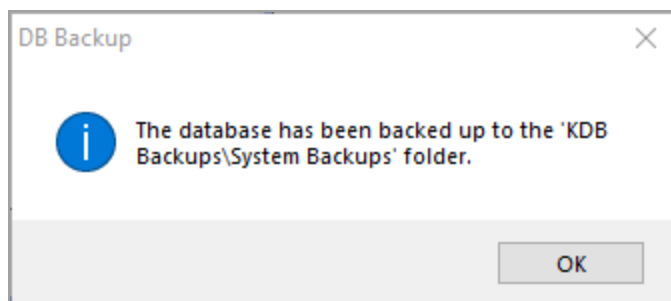


3. To reset the password to null, use a blank value for the New password key and Confirm new password key fields.


Backup current database

\KDB Backups is a standard folder created under the "CatTools" folder at installation. To backup your database files to the \KDB Backups sub folder:

1. Go to File > CatTools database.
2. Select Backup current database.



3. A dialog will prompt you to generate an encryption password for the database backup. Create a password and click OK.

 It is critical to remember the password you create. Without the password, your data will be lost and cannot be retrieved.

The `.kdb` files are not over-written or appended. Each new save creates a new file, and the file name is incremented using a numeric progression for example "Backup-1", "Backup-2", "Backup-3", and so on.

Up to nine backup events are created. Each time a backup is done, Backup 1 moves to Backup 2, and Backup 2 moves to Backup 3, and the series continues. The oldest backup event - Backup 9 - is dropped and deleted.

Restore database from backup

To create a copy of a backup database to the main CatTools folder:

1. Go to File > CatTools database.
2. Select Restore database from backup.
3. A dialog will prompt you to enter the encryption password for the database backup. Enter the password and click OK.

 To remove the password requirement, see [Change encryption password](#).

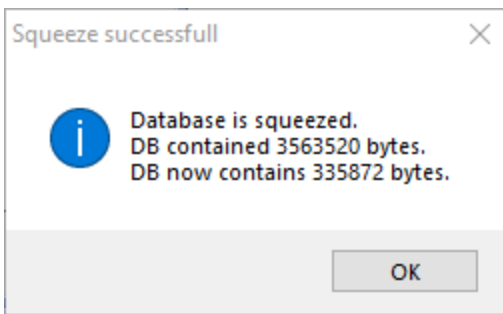
This makes the backup database file the current database file. The database is copied into the main CatTools folder and named `Restored1_KiwiDB-CatTools.kdb` where `Restored1` contains a sequence number for uniquely naming restored databases.

Squeeze current database

The CatTools database can contain unused space from normal device and activity maintenance traffic. The Squeeze database facility enables you to reclaim any unused space in the database file at regular intervals. It also refreshes all indexes to the data.

To run this facility:

1. Go to File > CatTools database.
2. Click Squeeze current database.



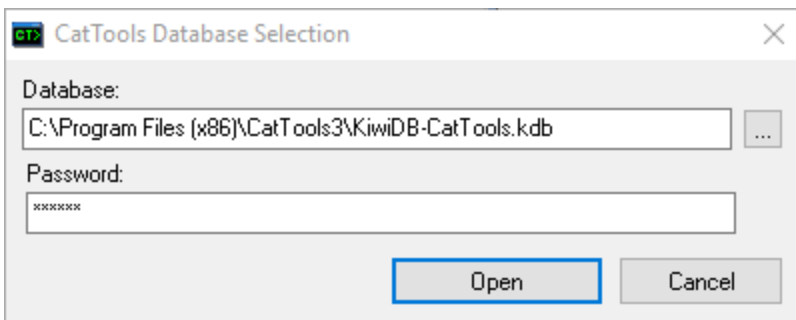
i When "Squeeze current database" is run, there cannot be any other CatTools users running against the database.

Open other database

Kiwi CatTools opens any available compatible CatTools database in any folder it has permissions to fully access. To open another database:

i SolarWinds recommends that you [squeeze the active database](#) before opening a different database.

1. Go to Files > Database and select Open/New.
2. Browse to and select the database you want to open.
3. Enter the password for the selected database, and click Open.



Use caution when opening databases from previous versions of CatTools. Incompatible database files are not opened. Compatible databases that are not at the current version may be upgraded, if required, to the version of the CatTools application that is opening it.


Create new database

The KCT 3.12.2 application and database includes new security improvements. The KCT database now uses a unique machine ID to encode and decode data. It is no longer possible to copy and paste data between standard machines with randomly generated passwords. To do so, [backup the database](#) or [manually set a database password](#).

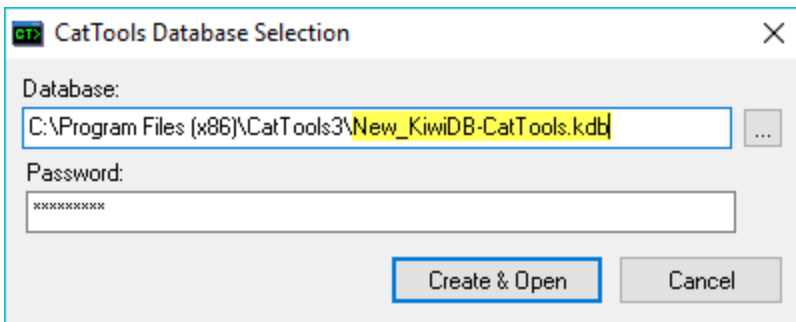
For standard machines that use randomly generated passwords, it is strongly recommended to regularly backup your database. This ensures that a database backup secured by a custom password is always available.

To create a new, empty CatTools database with no data:

1. Go to File > Database.
2. Select Open/New.
3. Browse to or enter the name and password for the new database.

 The database name must be unique. If the database name already exists, you can only open that database. A new database is not created if it shares the same name of an existing database. The name can be changed in the Database field.

4. Click Create & Open.




The new .kdb database file is named, in this example `New_KiwiDB-CatTools.kdb`.

Delete log file

The information log file is named "InfoLog.txt" and can be found in the main CatTools installation folder. By default, this file is located in `C:\Program files\CatToolsn`, where **n** is the installed version of CatTools.

To delete the log file:

1. Go to File > Delete.
2. Select Delete Infolog.txt file.

 You can set a maximum size for this file in the Options | Setup menu item: [Logging tab](#).

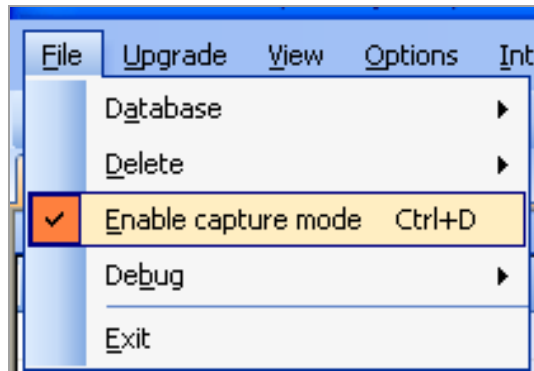
Enable capture mode

Capture mode can be used to help diagnose device connection or device activity problems.

If an activity is failing for an unknown reason, technical support may ask you to `Enable capture mode` before attempting to run the activity. This captures all the interaction between CatTools and the device to a log file. This file can then be sent to technical support to assist in troubleshooting the problem.

`Enable capture mode` functions like a tick box and by default is unselected. To enable this option:

1. Go to File and click Enable capture mode.



2. Capture mode is immediately enabled, as indicated by the check mark.

Clicking this option again disables capture mode.

3. Captured data is stored in the `\Debug` sub-folder. The file name is dependent on the name of the captured device.

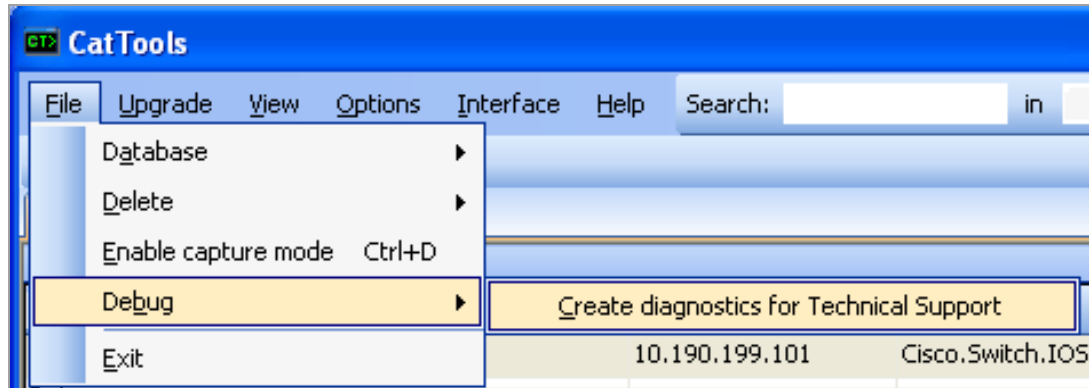
For example, a device called `2950` running on client thread `1` creates the file: `DebugLog-1-2950.txt`

Device captures can be included when [creating the diagnostics file for Technical Support](#).

Create diagnostics for Technical Support

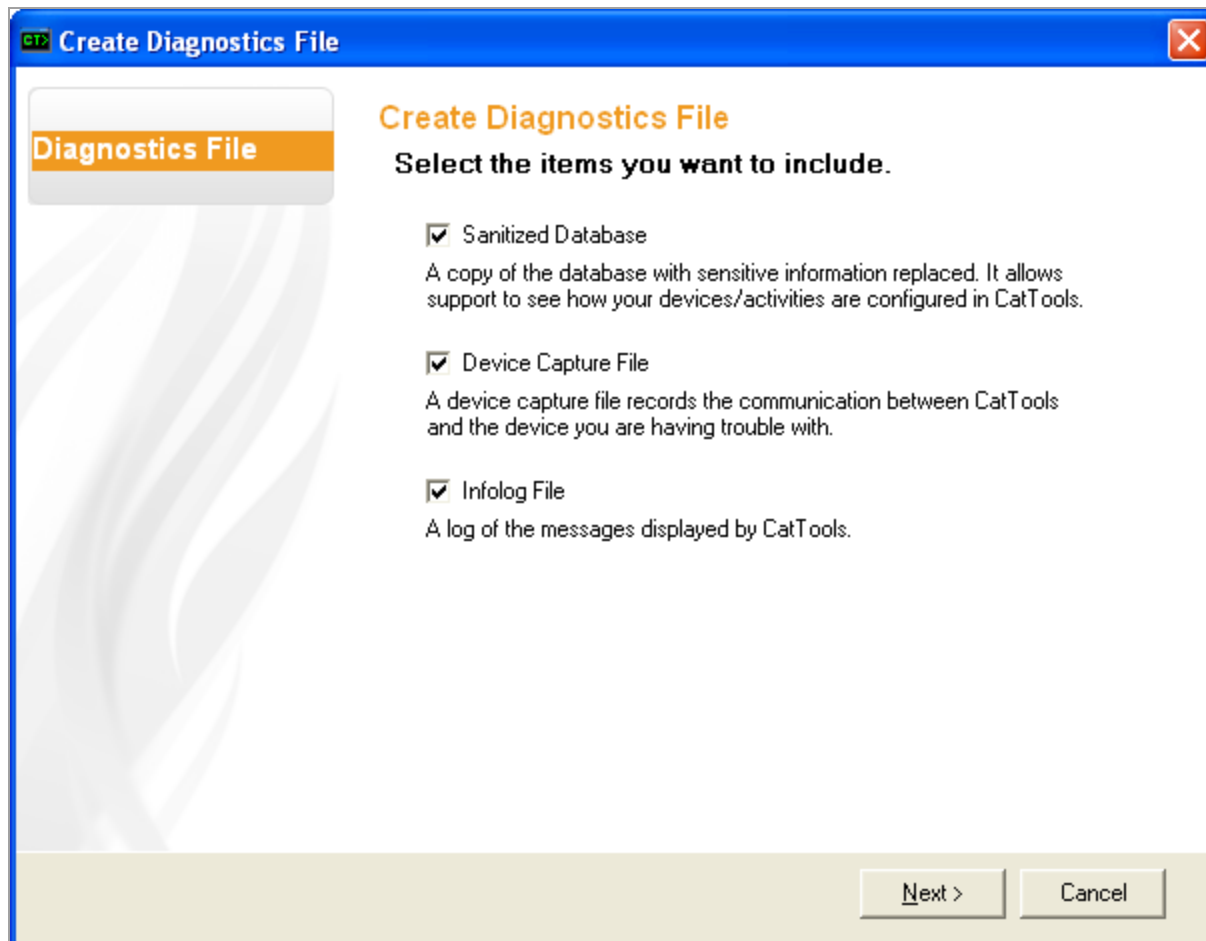
When troubleshooting issues in CatTools, Technical Support may ask you to create a diagnostic `.zip` file. To open the 'Create Diagnostics File' helper:


1. Go to File > Debug, and click Create diagnostics for Technical Support.

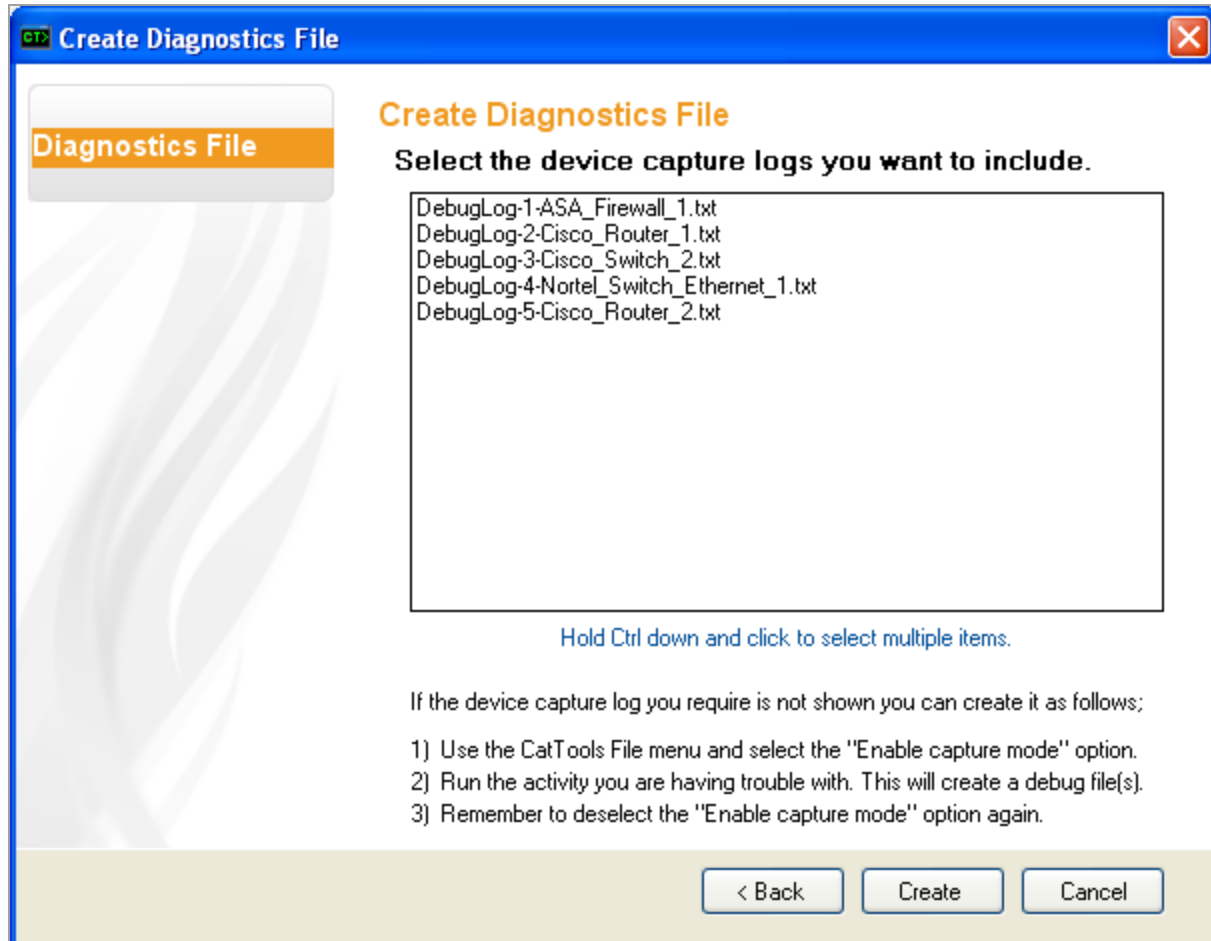


2. Choose which log or settings files you want to include in the diagnostics zip file.

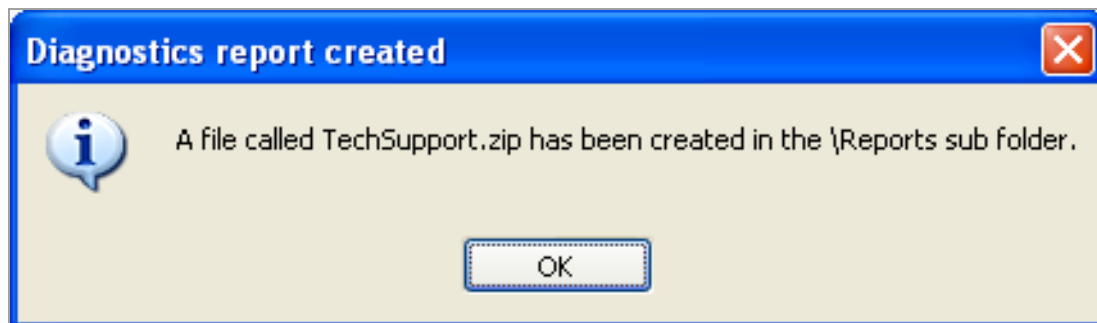
By default all three options are ticked as this helps Technical Support troubleshoot the in general should be the norm.



 If you are sending a Device Capture File to troubleshoot a device issue, before creating the diagnostics file enable capture mode and run the activity to generate the device capture files. The DebugLog selection screen allows you to choose which device capture files to include in the diagnostics zip file.



3. Click Create. You then receive a confirmation message that the diagnostics zip file has been created.



Kiwi CatTools View menu

The View menu allows you to review CatTools data files using the notepad editor.

Menu option	Description
Reports folder	<p>View the Reports folder using Windows Explorer.</p> <p>The reports from scheduled activities are stored in the <code>\Reports</code> sub folder. Clicking on this menu opens Windows Explorer to the reports sub folder so you can browse the contents.</p> <p>You can view the reports with MS-Excel, Log Viewer or use the internal viewer in CatTools from the Reports tab.</p>
Captured Data folder	<p>View captured data using Windows Explorer.</p> <p>The data captured from scheduled activities is stored in the <code>\Captured Data</code> sub folder. Clicking on this menu will open Windows explorer to the Reports sub folder so you can browse the contents.</p> <p>You can view the reports with MS-Excel, Log Viewer or use the internal viewer in CatTools from the Reports tab.</p>
Configs folder	<p>View the configs folder using Windows Explorer.</p> <p>The device configurations from the Device.Backup.Running Config configuration activity are stored in the <code>\Configs</code> sub folder. Clicking on this menu opens Windows Explorer to the Configs sub folder so you can browse the contents.</p> <p>You can view the configs with Notepad or Wordpad etc.</p>
Email log	View CatTools email activity log file using Notepad.
Activity log	View CatTools activity summary log file using Notepad.
Info log	View CatTools info log file using Notepad.

Kiwi CatTools Options menu

The Options menu allows you to change the setup options in Kiwi CatTools.

Menu option	Description
Setup	<p>This menu item displays a form that enables you to set the general properties of CatTools.</p> <ul style="list-style-type: none"> • The Email tab allows you to set all the emailing properties. • The Logging tab enables you to set CatTools to send syslog messages to a syslog server and provides control over information and activity logging functions. • The Misc tab allows you to set miscellaneous settings. • The TFTP Server tab contains the settings for controlling the CatTools TFTP server. • The DNS Resolver tab contains the settings for resolving IP Addresses in ARP reports.
Device Wizard	<p>The Device Wizard helps guide you through the addition of new devices to the CatTools database.</p> <p>Alternatively you can add new devices from the Devices pane.</p>
Activities Wizard	<p>The Activities Wizard helps guide you through the addition of new activities to the CatTools database.</p> <p>Alternatively you can add new activities from the Activities pane.</p>

Email

This section details how to set emailing properties in Kiwi CatTools for receiving notifications.

To access the email notification setup:

1. Go to Options > Setup.
2. Select the E-mail tab.

See the following topics for details:

- [Email General Options](#)
- [Primary Mail Server Properties](#)
- [Secondary Mail Server Properties](#)

General email notification options

To configure general notification email options:

1. Go to Options > Setup.
2. Select the E-mail tab.
3. Select Email General Options.

Send To

The e-mail address you want CatTools to send error notifications.

More than one e-mail address can be specified by using a comma (,) or a semi colon (;) to delimit the addresses. For example: `joe@company.com,mary@company.com,fred@company.com`

Errors

All the errors that occur during a scheduled activity are collected and sent as a single error alert. The error notification tells you what error occurred and on which devices.

You can optionally restrict sending error notifications to when you are running in timer mode.

Reports/Stats

The e-mail address you want CatTools to send reports and statistics. Most scheduled activities produce a report or statistics table of some kind.

You can optionally restrict sending error notifications to when you are running in timer mode.

From Address

Enter the "From Address" you want to appear in e-mail notifications from CatTools. This makes it easy for you to manage incoming e-mail notifications at your e-mail client, particularly SPAM filters.

SolarWinds recommends that you use a fully qualified real e-mail address. Some SMTP servers only send e-mail on behalf of addresses in their domain. For example, if your mail domain is `mycompany.com`, use a from e-mail address of `"CatTools@mycompany.com"` rather than just `"CatTools"`.

Logging Options

Select whether to keep a log of email activity through CatTools. The log normally contains informational and error level messages. You can also view the email log, and to delete the email log.

Optionally, you can set the logging message level to verbose. This is useful when there is a problem with email that requires you to see the complete email message traffic with your email server. It is not recommended that you enable this option, except as needed, due to the volume of data recorded.

Email intervals

Send Interval

You can set the email sending interval. This controls the seconds the mailer waits before sending a batch of email messages. A value of 20 means the mailer wakes up every 20 seconds to see if there is any messages to send.

Messages / Interval

This option sets the number of email messages the mailer sends to the mail server during one sending session. This allows you to control the amount of email traffic to your email server if you require it.

Send Retries

This option set the number of times the mailer will attempt to send an email that may have errors before it flags the message as not deliverable.


Test Email

Test the current e-mail settings against both mail servers. Email messages to the Errors and Reports/Stats addresses previously entered are sent.

Primary Mail Server Properties in Kiwi CatTools

To set primary mail server email options in CatTools:

1. Go to Options > Setup.
2. Select the Email tab.
3. Select the Primary Mail Server Properties tab.

 You can set an alternate mail server to be used should the primary server fail to connect. For more information, see [Secondary Mail Server Properties](#).

Use this server

This check option allows you to enable or disable the use of this server when CatTools sends email.

Server Name / Address

Enter the name or IP address of the target SMTP server where you want CatTools to send e-mail notifications.

SMTP Port

Enter the port used by the target SMTP server. The default value is 25.

Timeout

This is the timeout value for CatTools to wait when trying to connect to the target SMTP server, in seconds. The default value is 30. If you have a bad connection to your mail server, SolarWinds recommend that you use a local SMTP relay application.

Authentication

Some SMTP servers require you to authenticate before sending your e-mail, especially remote ISPs or relay servers. Most SMTP servers do not require authentication and so you can leave these values blank. Remember not to confuse SMTP sending with POP3 receiving, which always requires username/password authentication.

- **Type**

CatTools can be set to use no authentication, SMTP AUTH, or POP before send.

- **Username**

The username for authentication.

- **Password**

The password for authentication.

- **POP Server Name**

The name or address of the POP server to use for authentication.

- **POP Port**

The port number of the POP server, normally 110.

- **Test Email**

Click the button to test the current e-mail settings against the primary mail server. Email messages to the Errors and Reports/Stats addresses are sent.

Secondary Mail Server Properties

To set secondary mail server email options:

1. Go to Options > Setup.
2. Select the Email tab.
3. Select the Secondary Mail Server Properties tab.

Alternative mail servers

CatTools can have an alternate mail server set up that is used should the primary server fail to connect. If the primary server is not selected for use, the mailer always uses the secondary server.

- If the mailer is unable to connect to the primary server when it attempts to send messages, it immediately tries to connect to the secondary server.
- If a connection is lost while the mailer is sending mail to the primary server, the mailer does not immediately retry the connection. Instead it waits for the designated send interval and tries to connect to the primary server again.

Use this server

Allows you to enable or disable the use of this server when CatTools sends email.

Server Name / Address

The name or IP address of the target SMTP server where you want CatTools to send e-mail notifications.

SMTP Port

The port used by the target SMTP server. The default port value is 25.

Timeout

The timeout value for CatTools when trying to connect to the target SMTP server, in seconds. The default timeout value is 30. If you have a bad connection to your mail server, SolarWinds recommends that you use a local SMTP relay application.

Authentication

Some SMTP servers require you to authenticate before sending your e-mail, especially remote ISPs or relay servers. Most SMTP servers do not require authentication and so you can leave these values blank. Remember not to confuse SMTP sending with POP3 receiving, which always requires username/password authentication.

- **Type**

CatTools can be set to use no authentication, SMTP AUTH, or POP before send.

- **Username**

The username for authentication.

- **Password**

The password for authentication.

- **POP Server Name**

The name or address of the POP server to use for authentication.

- **POP Port**

The port number of the POP server, normally 110.

Test Email

Click the button to test the current e-mail settings against the secondary mail server. Email messages to the Errors and Reports/Stats addresses are sent.

Kiwi CatTools Logging tab

The Logging tab allows you to configure which types of logs to send and where to send those logs: to a syslog server or to a local log file. You optionally can send logs to both locations for redundancy.

Syslog

Send logs to Syslog Server	Enable sending of log messages to syslog server.
Syslog Server	Name or IP Address of the target syslog server that you want your logs sent to.
Protocol	Specify the syslog protocol, UDP or TCP. The default protocol is <code>UDP</code> .
Port	Valid port number syslog messages are coming through. The default port for UDP is <code>514</code> .
Facility	Select the facility setting for outgoing syslog messages. The facility determines the program logging the message. Depending on the facility selected, the messages are handled differently. The default facility setting is <code>Local0</code> .
Logging Level	Select what level of log messages are forwarded to the target syslog server. The default logging level is <code>"Info and above"</code> .
Test Syslog	Send a test message to the target syslog server, using the options selected above. This option verifies your configuration and network connection.

Info log


Log info log messages to local log file	<p>Enable info log messages to be logged to a local log file. The default is On.</p> <p>The log file created is named "InfoLog.txt" and can be found in the CatTools install folder.</p>
Logging level	<p>Select what level of messages are to be logged to the local file. The default level is "Debug and above".</p>
Limit File size	<p>The log file can become rather large if the logging of debug messages is enabled. By checking this option you can limit the size of the info log to the size specified in the size field.</p> <p>When the log file size reaches the limited size, the file is closed, renamed to <code>infolog_prev.txt</code>, and a new log file is created.</p> <p>The previous log file is kept in case there is a requirement to go back in the log further than what is contained in the current log. This previous log is deleted before a current log that has reached maximum size is renamed.</p> <p>The effect of having this option turned on is that there are two log files, a current and a previous.</p> <p>If the option is set off, the log file grows indefinitely.</p>

Activity log

Log Activity summary to local log file	<p>Enable the CatTools activity summary to be logged to a local log file. The default is On.</p> <p>The log file created is named "Activitylog.txt" and can be found in the CatTools install folder.</p>
Limit File size	<p>Limit the size of the info log to the size specified in the size field.</p> <p>When the log file gets to this size, it is closed, renamed to <code>activitylog_prev.txt</code>, and a new log file is created.</p> <p>The previous log file is kept in case there is a requirement to go back in the log further than what is contained in the current log. This previous log is deleted before a current log that has reached maximum size is renamed.</p> <p>The effect of having this option turned on is that there are two log files, a current and a previous.</p> <p>If the option is set off, the log file grows indefinitely.</p>

Misc tab

The Misc tab allows to you configure additional setup options for Kiwi CatTools.

Telnet client	Enter the path and file name of the local telnet client. The default file name is "Telnet.exe". When using SecureCRT as your external telnet client, you need to add the /TELNET switch to the command line. For example: C:\Program Files\SecureCRT\SecureCRT.EXE /TELNET						
SSH client	The path and file-name of the preferred local SSH client. By default, there is no SSH client specified. <div style="border: 1px solid #ccc; padding: 10px; background-color: #fff9c4;"> <p> If using PuTTY.exe you can specify PuTTY for both the Telnet and the SSH client. When CatTools detects PuTTY has been specified, the following command line switches are used to connect:</p> <p>[protocol Telnet or SSH. Derived from value set in 'Method' field of device setup form. If any SSH value is set, then the protocol used will be 'SSH']</p> <p>P [port specified in 'Port' field of device setup form]</p> <p>Example: C:\Putty.exe -SSH -P 22 HostAddress</p> </div>						
Clear log window before starting each activity	Clear the Info log window before each activity starts. <ul style="list-style-type: none"> The default is On. 						
Clear error count before starting each activity	Clear the error count displayed on the Status Line of the main window before each activity starts: <table border="1" data-bbox="349 1417 1339 1470"> <tr> <td>Manager Mode</td> <td>Schedule: Stopped</td> <td style="background-color: #ffff00;">Errors: 9628</td> <td>Device: 0 of 0</td> <td>Thread: 0 of 0</td> <td>IDLE</td> </tr> </table> <ul style="list-style-type: none"> The default is On. 	Manager Mode	Schedule: Stopped	Errors: 9628	Device: 0 of 0	Thread: 0 of 0	IDLE
Manager Mode	Schedule: Stopped	Errors: 9628	Device: 0 of 0	Thread: 0 of 0	IDLE		
Automatically enable timer mode	The timer automatically sets on the specified number of seconds after the service starts.						
Hide password change report passwords	Hide the passwords reported by the Device.Update.Password activity report. Show current database name above grids. Shows the full path of the current CatTools database.						

Terminate orphaned activity running longer than n minutes

The number of minutes an activity is allowed to run, after which it is forcibly terminated.

- The default value is 0, meaning no limit is set and the activity can run indefinitely.
- The maximum value is 2880.

This option should only be used in a situation where you have a problem with an activity never terminating. This issue may be caused by factors such as a logic loop or communications that are stalled. The timing only starts when say all bar one device sessions have completed, and the last session appears hung up.

When this is activated and the time limit is detected, CatTools attempts to terminate the activity by normal means. If it is unable to terminate the activity by normal means, the `CatTools_Client.exe` application, which runs the activities, is forcibly terminated. CatTools then waits for about 30 seconds to enable the system to clean up from the termination, and restarts the scheduler.

HTML report maximum row limit

Set the number of output rows for HTML reports.

- The default value is 5000.
- The minimum value is 1.
- The maximum value is 50000.

The maximum rows is limited to 50000 as it is unlikely you would want to scroll through an HTML file of more than 50000 rows. If all the report data is required, you should use the corresponding text format file (*.txt).


When exiting the Manager should

 This option is only available when CatTools is installed as a Service.

Since most changes need to be done when the timer is offline this feature provides the user with a reminder to start the timer again when the Manager is closed down. In this way you can ensure that your activities run even if you have forgotten to restart the timer yourself.

- **Ignore the timer state:** No action is taken.
- **Set the timer to ON:** The timer automatically sets to ON when you exit the Manager.
- **Set the timer to OFF:** The timer automatically sets to OFF when you exit the Manager.
- **PROMPT if the timer is off:** A message box asks you what you want to do when you close down the Manager if the timer is not running.

TFTP Server

 SolarWinds does not recommend usage of Trivial File Transfer Protocol (TFTP) in KCT due to potential security vulnerabilities. If TFTP must be used, it is highly recommended to take precautions and limit its exposure


To set TFTP Server properties:

1. Go to Options > Setup.
2. Select the TFTP Server tab.

For more information, see the following topics:

- [TFTP Server General Options](#)
- [Security Options](#)
- [Scripting Options](#)

TFTP server general options

 SolarWinds does not recommend usage of Trivial File Transfer Protocol (TFTP) in KCT due to potential security vulnerabilities. If TFTP must be used, it is highly recommended to take precautions and limit its exposure

To set general server options:

1. Go to Options > Setup.
2. Select the TFTP Server tab.
3. Select the General Options tab.

Automatically start TFTP server on program startup	Starts the TFTP server when CatTools starts.
--	--

TFTP Port	The port number the server listens on. This is normally port 69.
-----------	--

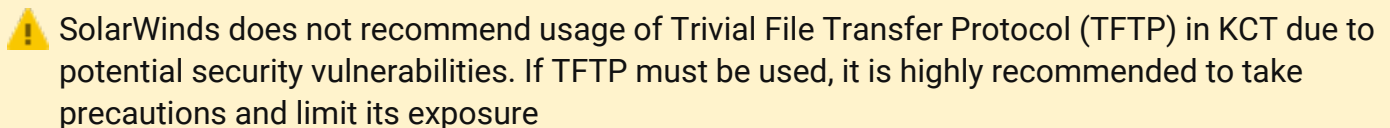
TFTP Bind to	The IP address of a network interface card to bind to. Leave the field blank to use the default card.
--------------	--

Upload folder	The folder that contains files to be uploaded to devices from CatTools.
---------------	---

Download folder	The folder that contains files downloaded from devices by CatTools.
-----------------	---

Allow overwrite of existing files	Allow files currently in the folder to be overwritten by files of the same name.
Allow automatic creation of sub folders	Allow sub-folders to be created automatically during a download.
Remove completed sessions from display after	Remove the messages created from a TFTP session from the display tab after the specified number of seconds.
Maximum file size that can be created	<p>Set the maximum file size in MB that the server can create.</p> <ul style="list-style-type: none"> The default value is 20MB. TFTP clients may also have a limit to file size they can send. There appears to be a known limitation in some TFTP clients to a file size of about 32MB, which corresponds to using the standard protocol block size of 512 bytes. If your network and devices are capable of handling larger block sizes, then you may be able to create much larger files. There are no options in CatTools to specify a block size to use; block sizes are negotiated between the TFTP Client and Server.

TFTP server security options

 SolarWinds does not recommend usage of Trivial File Transfer Protocol (TFTP) in KCT due to potential security vulnerabilities. If TFTP must be used, it is highly recommended to take precautions and limit its exposure

To set security options:

1. Go to Options > Setup.
2. Select the TFTP Server tab.
3. Select the Security Options tab.

Global Read	Must be selected for information to be read from the TFTP server.
Global Write	Must be selected for information to be written to the TFTP server.
Use Access Lists	List is used to ascertain the read and write permissions for the IP address in question.

 Access lists are subordinate to Global Read and Global Write.

Access List Usage

There are five columns in the access list table:

- **Incl|Ex IP Range:** Determines whether the IP range specified is to have the Read and Write options applied to it (**included**), or is to be exempt (**excluded**) from the Read and Write options which are then applied to all addresses outside of the range.
 - Exclude is useful for allowing only a limited range of IP's to be read or written to because everything outside of the excluded range will have the read/write settings applied to them.
- **Start IP:** Defines the start of the range of IP addresses to use.
- **End IP:** Defines the end of the range of IP addresses to use.
- **Read and Write:** When selected, reading or writing capability is allowed.

The information in the access list is evaluated in descending order until a match is found.

Access List Example

Incl Ex IP Range	Start IP	End IP	Read	Write
Exclude	192.168.1.1	192.168.1.100		
Include	192.168.1.60	192.168.1.60		


1. The first line in the access list is `EXCLUDE`.
2. Read and write properties are applied to all addresses outside of the range 192.168.1.1 – 192.168.1.100. In this case anything outside of the excluded range is unable to execute read or write commands on the TFTP server.
 - Attempts to read or write from address 192.168.1.105 would fail.
 - Attempts to read or write from address 192.168.1.98 would succeed.
3. The second line in the access list is `INCLUDE`.
4. Read and write settings will be applied to addresses included in this range. In this case, the range is a single IP address.
 - Attempts to read or write from 192.168.1.60 would fail.

Let us look at the steps that would be traversed if we tried to write to the TFTP server from address 192.168.1.60:

1. `Global Write` is examined. If `Global Write` is selected, the process continues.
2. `Use access lists` is examined. If `Use access lists` is selected, the access list is checked.
3. The first row in the access list is examined. `192.168.1.60` is within the excluded range. Consequently, the read or write settings do not apply to it.

4. The process continues to the second row of the access lists.
5. The address 192.168.1.60 is included in the range specified in this row. The write settings are applied, in this case, to **not** allow writing to the TFTP server.

Scripting Options

 SolarWinds does not recommend usage of Trivial File Transfer Protocol (TFTP) in KCT due to potential security vulnerabilities. If TFTP must be used, it is highly recommended to take precautions and limit its exposure

This pane allows you to specify VBscripts that are run before and/or after a TFTP session. To set scripting options:


1. Go to Options > Setup.
2. Select the TFTP Server tab.
3. Select the Scripting Options.

Use the tick boxes to determine whether a script is run and enter the full path to the script in the appropriate text box.

Timing Issues	It's important to note that you only have 1 second to get everything done in the pre-session external script and return a value otherwise the TFTP session times out, and retries.
Scripting Information	<p>Your script must start with Function Main (session) where session is the TFTP session being passed in.</p> <ul style="list-style-type: none"> If the script is successful it should return a value of True which allows the TFTP session to continue. Returning a value of False causes the TFTP session to abort. <p>For security reasons if the script can not be loaded or if the script fails then the TFTP session is aborted.</p>

Scripting information session properties

You can access the following properties of the session:

Session.CreatTime	<p>The time the session was started by the host.</p> <div style="border: 1px solid lightblue; padding: 5px; margin-top: 10px;"> <p> Represented as the number of seconds from midnight 1/1/1970 UTC.</p> </div>
Session.FileName	Name of the file to be written.

Session.FileSize	Size of the file on the server for a 'Get' operation. This is zero for a 'Put' operation.
Session.Host	IP address of the remote client represented as a long. The address is converted from an 8 digit hex value.
Session.Hosts	IP address of the remote client represented as a string. For example: 127.0.0.1
Session.Port	TFTP Port on the remote client.
Session.TransferSize	The number of bytes transferred so far.
<div style="border: 1px solid #add8e6; padding: 5px; display: inline-block;"> ⓘ In the pre-session script, this is zero. In the post-session script, this is the whole file size. </div>	
Session.Type	<ul style="list-style-type: none"> • Upload for a Put. • Download for a Get.

Example Scripts

The script below checks the host address of the remote client issuing a Put. If it matches a certain address then the filename is changed.


Rename of the file based on host address

```
Function Main(session)
    Main= False
    With session
        If .HostS = "192.168.1.60" Then
            .FileName = "NewFileName.txt"
        End If
    End With
    Main= True
End Function
```

Reject a file based on the time of day

```
Function Main(session)
Dim CreateHour
    Main= False
    With session
        'convert the create time from the unix format to the current time
```

```
        CreateHour = DateAdd("s", session.CreateTime, DateSerial(1970, 1, 1))
        'convert to 24 hour format
        CreateHour = FormatDateTime(CreateHour, vbShortTime)
'seperate out the hour value only
        CreateHour = DatePart("h", currentTime)
        Select Case CreateHour
            'reject files created two hours either side of midnight
            Case 1,22,23,24
                'accept files created at any other time
                Main=False
            Case Else
                Main=True
        End Select
    End With
End Function
```

 There is a known bug that occurs if a script is triggered in the pre-session event that renames the filename, such as `.FileName = "NewTestFileName.txt"`. The file is put into the TFTP folder with the filename correctly changed to `NewTestFileName.txt` but the existing `Test.txt` file in the TFTP folder is deleted.

DNS Resolver

The DNS Resolver resolves the IP addresses in ARP reports based on configured server settings. To set DNS resolver options:

1. Go to Options > Setup.
2. Select the TFTP Server tab.
3. Select the DNS Resolver tab.

 The `Resolve IP addresses to Hostnames` check box must be selected for the DNS Resolver to function.

1. Select either `Resolve using known DNS server(s)` or `Resolve by auto detection of DNS server(s)`.
2. By default, internal IP address ranges are listed: `10.x.x.x` , `192.168.x.x` , `169.254.x.x` , `172.16.x.x` , `172.17.x.x` , `172.18.x.x` , `172.19.x.x` , `172.20.x.x` , `172.21.x.x` , `172.22.x.x` , `172.23.x.x` , `172.24.x.x` , `172.25.x.x` , `172.26.x.x` , `172.27.x.x` , `172.28.x.x` , `172.29.x.x` , `172.30.x.x` , `172.31.x.x`

3. To add more internal IP address ranges, click Add.

You can remove added ranges by selecting the range and clicking Remove.

4. Select the NetBIOS/DNS Server to resolve.

5. Enter the primary and alternate DNS server IP addresses for both internal and external DNS servers.

6. Specify the days to flush DNS cache.

7. Specify DNS Query time out.

8. Click Save.


Kiwi CatTools Interface menu

The Interface menu allows you to change the Panes and Themes interface options in Kiwi CatTools.

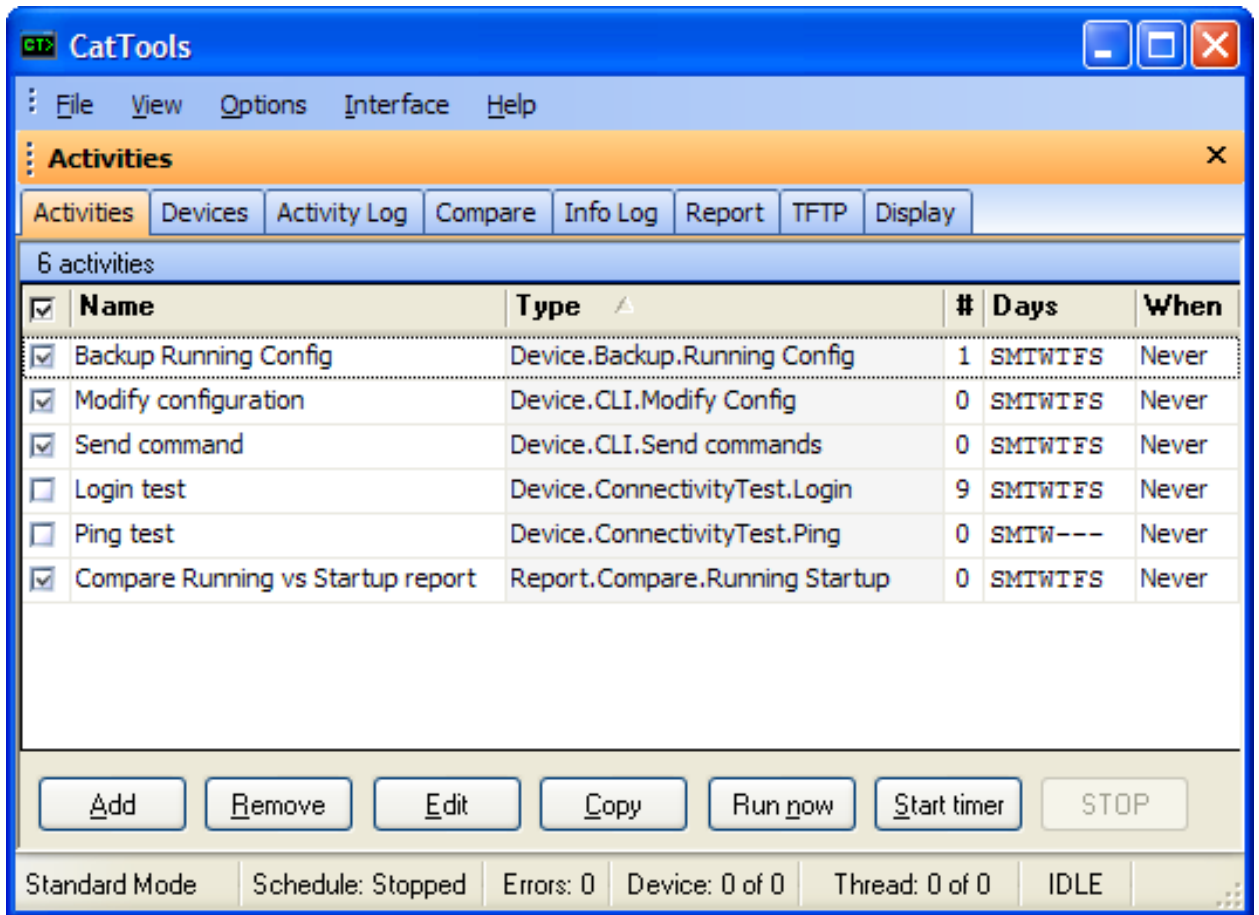
Menu option	Description
-------------	-------------

Interface Panes There are a number of options under Interface > Panes:

- Turn panes on and off by selecting or deselecting them from the panes list.
- Save your current pane layout by clicking the Save user defined layout menu item.

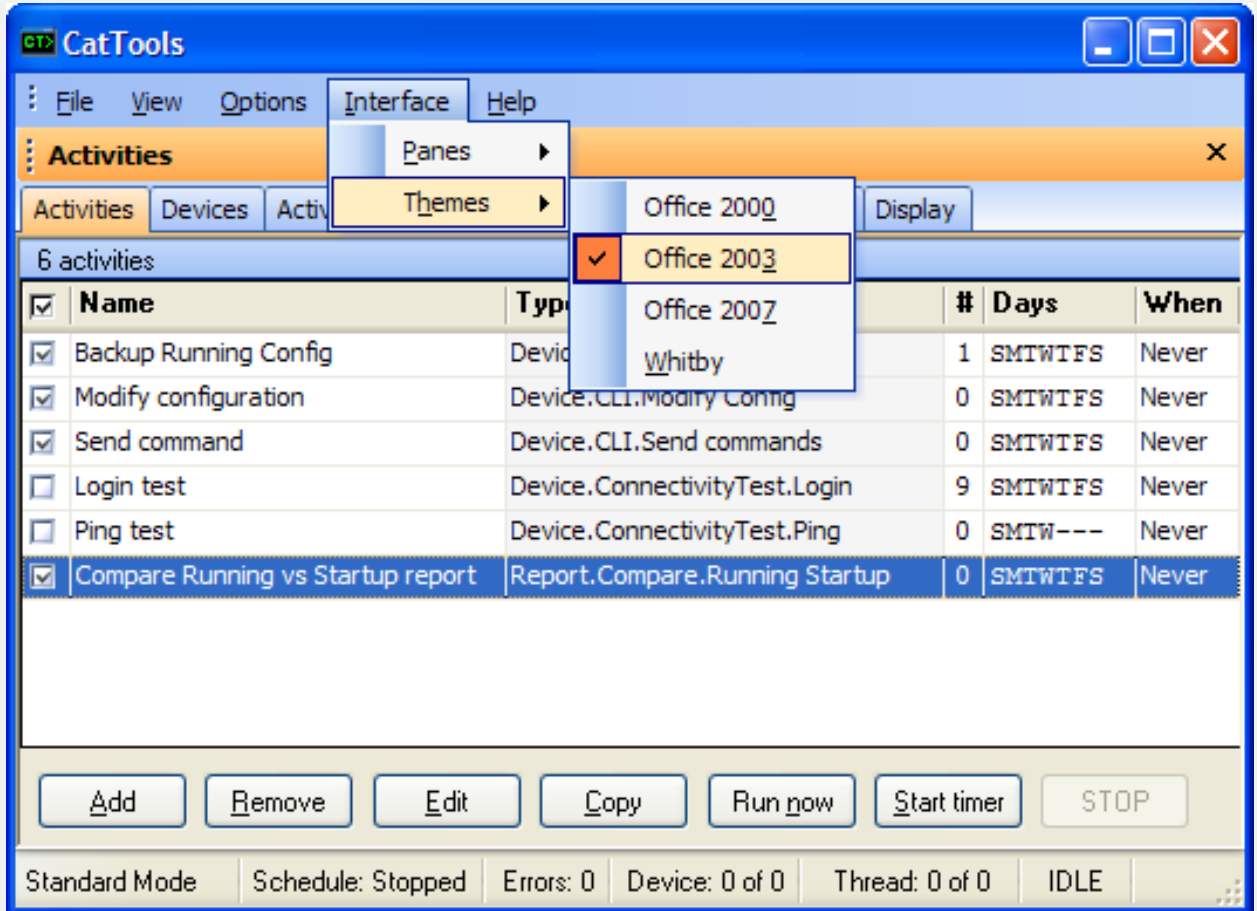
 CatTools automatically saves your current pane layout when you exit the application.

- Restore your previously saved pane layout by clicking `Restore user defined layout`.
- Restore the CatTools default pane layout by selecting `Restore to default layout`.



Menu option	Description
-------------	-------------


Themes Change the theme of CatTools by clicking the Interface|Themes menu item and selecting an option from the list of themes.



Kiwi CatTools Help menu

The Help menu enables you to view help, get or set registration details, and visit the web site.

Menu option	Description
Contents	Displays the Contents of the Help file.
Online FAQ	Opens a browser window and views the CatTools Online Frequently Asked Questions page.
Purchase CatTools	Opens a browser window to the online purchase page. Various off-line methods or payment are also available.

Menu option	Description
Enter registration details	<ol style="list-style-type: none"> 1. Enter Registration details page leads you to the Kiwi CatTools License application. 2. Activation keys can be found by logging in to the customer portal. 3. Complete the Activation wizard to upgrade your license.
About	Opens the "About" window, showing the program version number, registration details, and support information.
	<div style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px;">  When contacting technical support about a problem, please make note of your version and build number. </div>

Kiwi CatTools pane interface

Kiwi CatTools includes a docking panes interface. The panes interface is a customizable, tabbed layout allowing users to hide and show the panes that matter to them and to have two or more panes viewable at the same time.

Docking panes

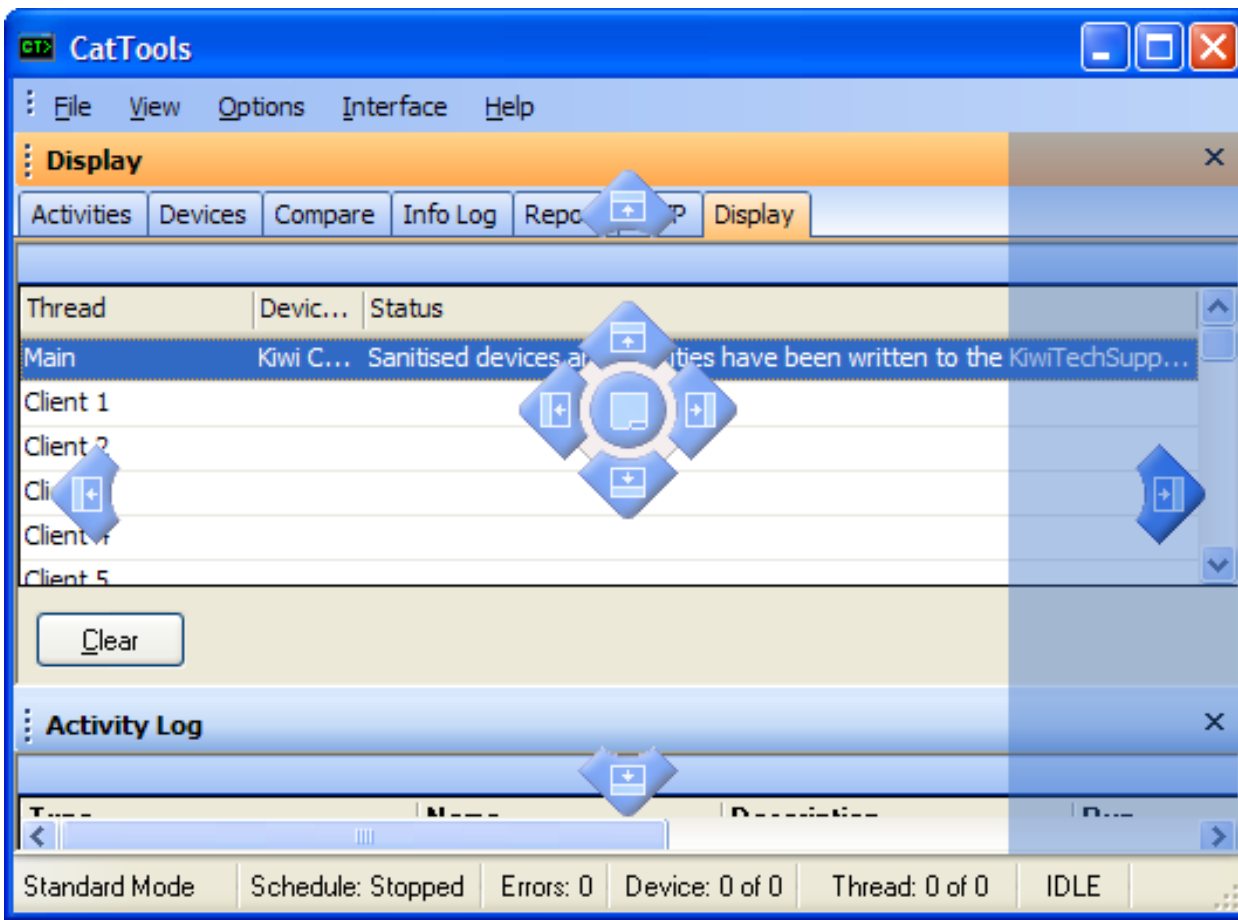
There are a number of different docking options. For example, you can dock panes at the top, bottom, left or right of the screen, or dock them to the top, bottom, left or right of other panes.

To dock a new pane, do the following:

1. Click on the pane you wish to dock and while holding the mouse button down drag it into a pane. The docking pane locator appears on screen.
2. While holding down the mouse button, move the cursor over one of the docking pane locator "stickers". The area where the pane is docked is highlighted.
3. Release the mouse to dock the pane.

The example below shows how to dock the Display pane to the right of the screen by using the right-most locator sticker.

The pane is docked to the right of **all** the existing panes. To dock the Display pane to the right of just the top-most pane, leaving the Activity Log pane to span the width of the window, drop the pane on the right-hand sticker within the center cluster.

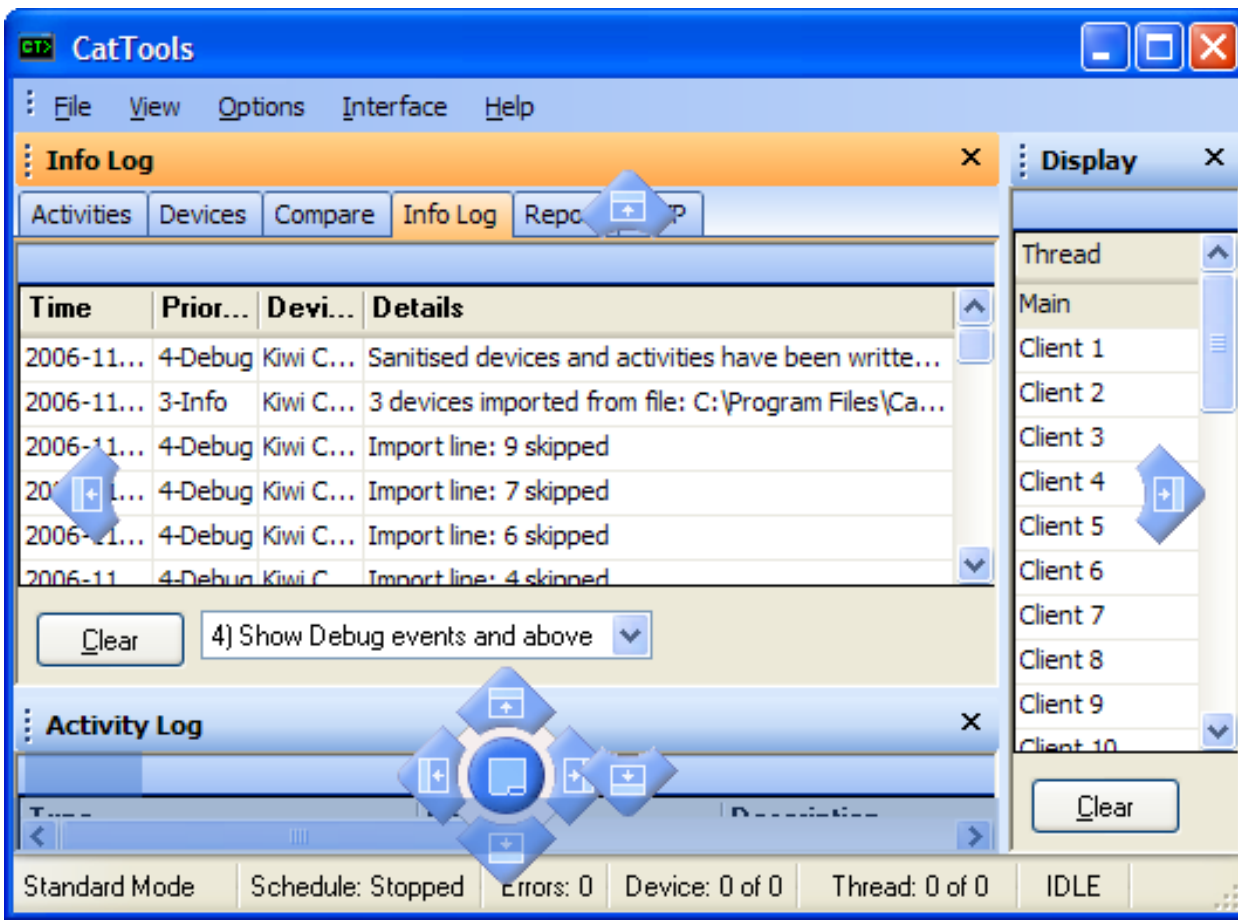


Moving and grouping panes

To move a pane from one group of panes to another, do the following:

1. Click on the pane you wish to move and while holding the mouse button down drag it into the pane you want to group it to. A docking pane locator appears on screen within the desired pane.
2. While holding down the mouse button, move the cursor over the center-most locator sticker for that pane. The whole pane is highlighted.
3. Release the mouse to drop the pane into that pane group.

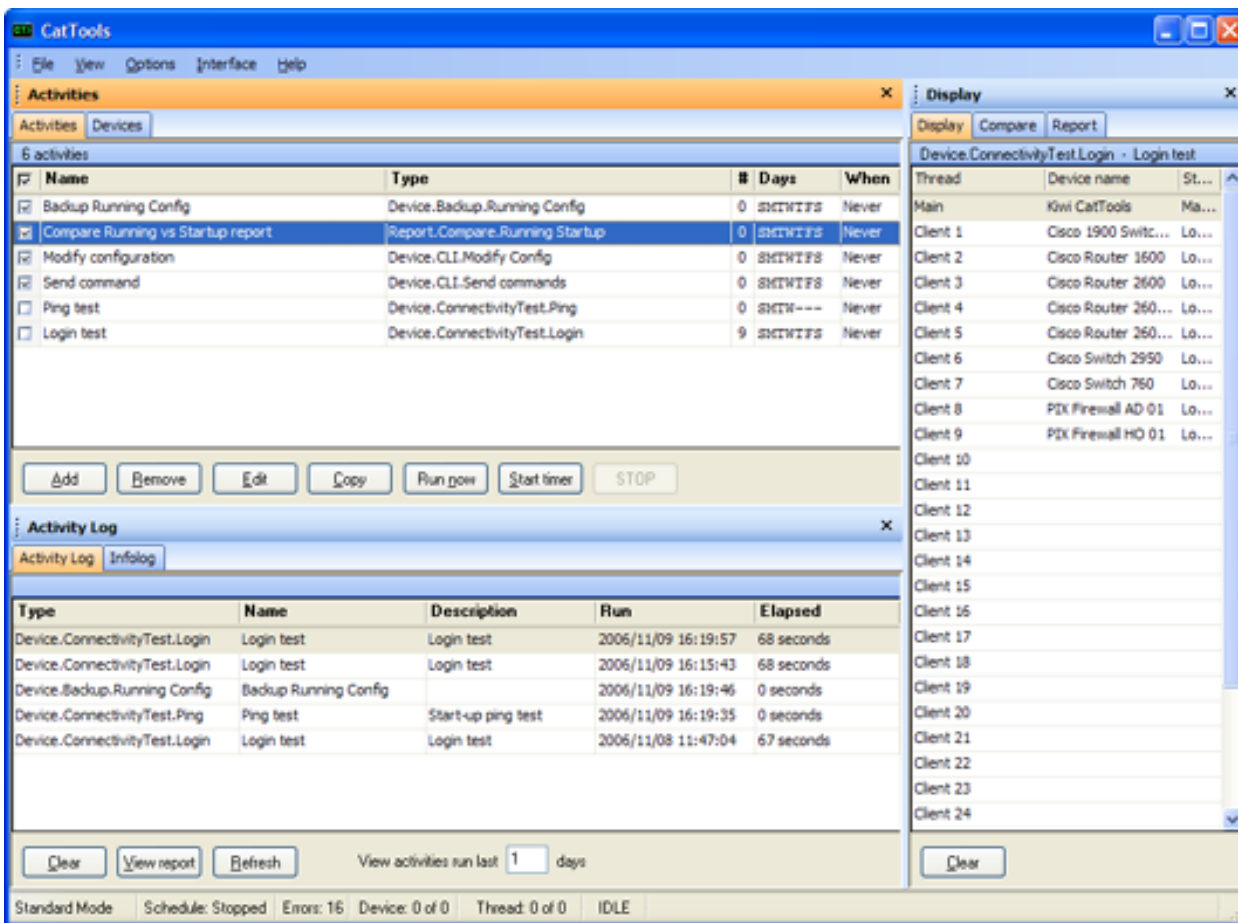
The example below shows how to move the Info Log pane from the top pane group and group it with the Activity Log in the lower pane.



Rearranging and saving your layout

You can rearrange your pane layout easily:

1. Move the mouse cursor over the edge of a pane until the "splitter" cursor appears.
2. Hold the mouse button down and move the cursor to where you want the edge of the pane to go to.
3. Release the mouse button.



Once you have decided on your layout, you can save your changes by clicking on the Interface > Panes > Save user defined layout. CatTools automatically saves your changes when you exit the application. See [Interface](#) for further details.

Reset the pane to default

To reset the panes back to the CatTools default, click on the Interface > Panes > Restore to default layout.

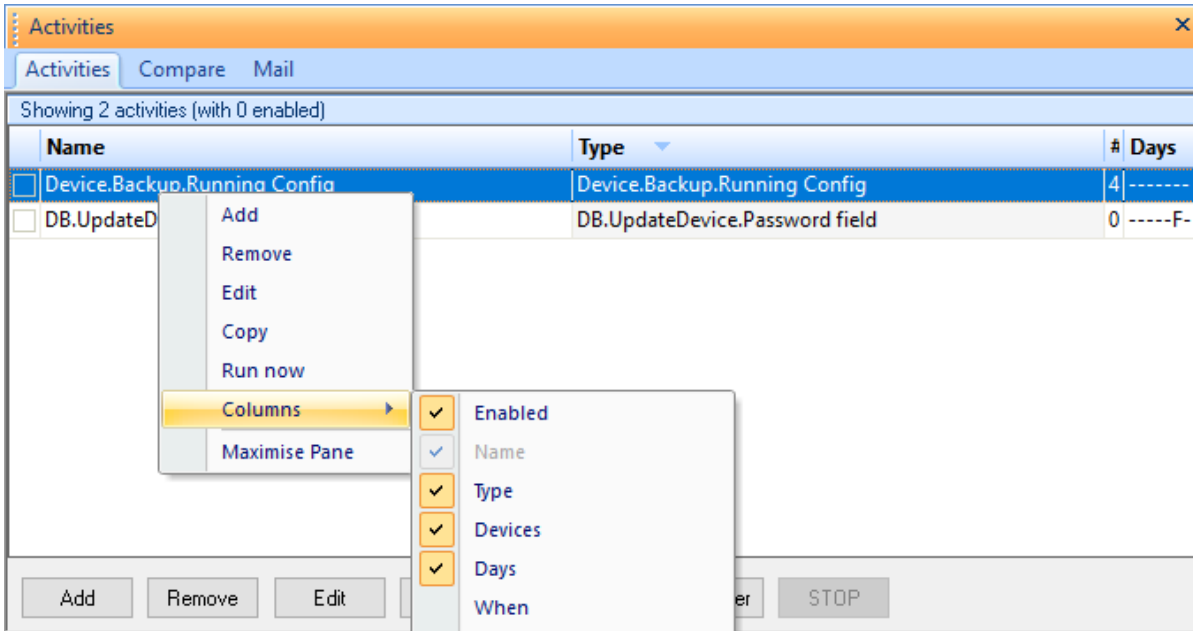
See [Interface](#) for further details.

Right-Click short menu

The functions of the buttons at the bottom of the tabs can also be accessed via the shortcut menu by selecting a row from within a pane and clicking the right mouse button.

The shortcut menu also includes other items which are specific to the pane in which you are currently in. For example, on the Activities pane, there is an item called "Columns" which contains sub items to show and hide data columns relating to activities; Show and Hide the grouping box which allows you to group your activities by data columns, and restore the pane back to its default layout.

You can maximize the current pane by right clicking and selecting the Maximize Pane menu option. To return a maximized pane to its normal size right click and select restore panes.



Chapter Topics

This chapter covers the following topics:

- [Devices pane](#)
- [Activities pane](#)
- [Activity Log pane](#)
- [Compare pane](#)
- [Info Log pane](#)
- [Report pane](#)
- [TFTP pane](#)
- [Display pane](#)
- [Mail pane](#)

Devices pane

The Devices pane displays the devices you have defined.

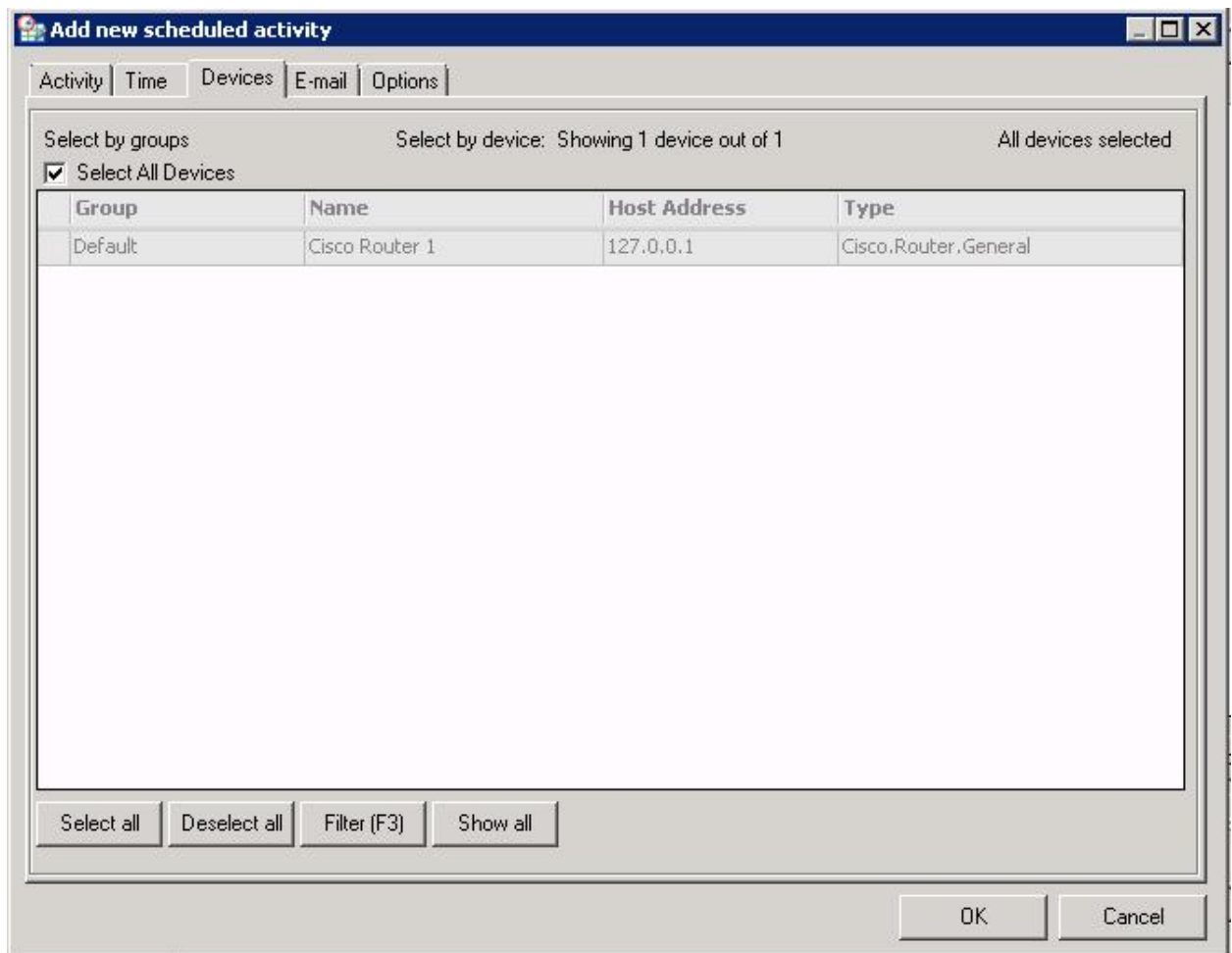
Option	Description
Add	Add a new device.
Remove	Remove a selected device.
Edit	<p>Edit a selected device. Existing configurations for the highlighted device are retained.</p> <p>Opens the "Device Information" window with the same tags and options as the "Add" device button.</p>
Copy	<p>Copy a selected device.</p> <p>Opens the "Device Information" window, with the same tags and options as with the "Add" device buttons, but with the existing configuration from the highlighted device shown.</p> <p>You must at least change the name of the device to save this configuration as a new device.</p>
Filter	Click to filter the on-screen display of devices under the Devices tab.
Show All	The Show All button removes all filtering, so that all devices in your database are displayed.

Option	Description
--------	-------------

Select All Devices	Runs an activity against all devices available without having to select them one by one. Checking this box allows you to automatically add new devices to existing activities when new devices are added.
--------------------	---

For example, you create a backup activity for your current 15 devices, but 4 weeks later you buy new equipment and add these new devices into CatTools.

If the backup activity was setup with the Select All Devices option, new devices are automatically picked up by the activity.



Add a device to Kiwi CatTools

To configure a network device in Kiwi CatTools, you must first add the device to the CatTools database. When running Kiwi CatTools for the first time, you can add a device using the [CatTools Setup Wizard](#). You can also click Add in the Devices pane to add a new device.

Device Information

Device info | Passwords | Prompts | Contact info | Extra info | Variations

Vendor: [All Vendors]
Filter by vendor or show 'All Vendors'.

Device Type: [Cisco.Firewall.ASA]

Group:* [Default]
Logically group your devices, if desired.

Name:* [ASA Firewall 2]
A unique name for this device.

Host Address:* [127.0.0.1]

File Name:* [ASA_Firewall_2]
Base file name for this device.(unique)

Model: [ASA5505]
Select model from list or type your own.

Connect via:* [Direct connect]
Name of another device to connect to first, if necessary.

Method:* [Telnet]

Port:* [23]

SolarWinds recommends using Ping device and Telnet/SSH to test and confirm the configuration and connection of any device added to Kiwi CatTools:

- **Ping Device**

Sends ICMP Ping packets to the highlighted device.

- **Telnet/SSH**

Uses the defined Telnet or SSH client program to start a manual session with the highlighted device. The Telnet & SSH client programs can be defined in the [CatTools Setup dialogue](#).


Device info

Configure the identification information for a device, such as the vendor, device type, model, and port. Once you have identified the device, ping the device to confirm a successful connection.

Vendor	The manufacturer of the device being added. Filter the 'Device Type' list box by vendor, or choose all vendors to show all Device Types.
Device Type	Select the device type. Determines how CatTools handles a session with a particular device.
Group	The name of the group the device belongs to. You can select a group from the drop down list, or create a group by entering text into the field. Use group names to organize your devices into logical or physical categories.
Name	A unique name for the device.
Host Address	The IP address of the device (in standard <code>aaa.bbb.ccc.ddd</code> format, or use a host name).
File Name	The base file name CatTools will use for the device's data and reports. Reflects the device name but converts invalid characters.
Model	The device's model number. You can select the model from the drop down menu, or enter any text you like into the free-form field. This field is only used for documentation and has no effect on the operation of CatTools.

Connect via Name of a device CatTools initially connects to. The default is "direct connection". You only need to specify another device when direct access to the desired device is not possible.

CatTools allows you to hop from one device to another using Telnet or SSH to reach your final destination. For example, if your device is behind an access list, but a Linux box has access to that device, you can connect via the Linux box first, then launch a telnet or SSH session to the destination device from there.


 When using a Cisco router as a jump point, SolarWinds recommends disabling "logging synchronous" in the Line VTY section of the config. This can cause problems when trying to telnet out from the router.

By default, many Cisco routers have been configured with 5 lines (line vty 0 4). As CatTools is multi-threaded and can support (depending on your edition) up to 30 client threads (connections). If you create an activity for more than 5 devices that connect via the router, timeout errors or connection failures can occur as a result of all the available router lines being used.

To prevent timeout or connection errors:

- A. Increase the number of VTY lines available on your main connect via router.
- B. Set the 'client threads' to use '5 threads'.
- C. Use a Linux box as a connect via device instead of a router.

Method The method used to connect to a device. The default selection is `Telnet`.

 When using SSH there may be a specific variant of the protocol that is required to connect with a device. For example, Netscreen devices supporting SSH2 require the variant SSH2-notty to connect successfully with CatTools. For more information refer to [Connecting via a session](#).

Port The port number the selected connection method uses. The default port for Telnet is 23. The SSH connections default port is 22.

Passwords

Device login/enable and other passwords.

VTY Password Enter the Virtual Terminal or initial login password here.

Enable Password Enter the enable mode or privileged password here.

Privilege Level	<p>You may set the enable mode privilege level if required. This is typically not required and so is best left blank. On a Cisco router, there are 16 privilege levels (0 to 15).</p> <p>Standard enable mode puts you in level 15.</p> <p>This is appended to the enable command when it is sent to the device, so if it is present and not required it may cause an error when the device tries to authenticate enable mode.</p>
Console Password	Enter the console password if using a direct COM port connection.
Username	Enter the Username (e.g. AAA, TACACS, RADIUS or Local username/password, etc.)
Password	Enter the Password (e.g. AAA, TACACS, RADIUS or Local username/password, etc.)
SSH Username	Enter the SSH Username if required.
SSH Password	Enter the SSH Password if required.
SNMP Read	Enter the SNMP Read community name for your device. The SNMP Read community name default is normally "public", however it is recommended you change your SNMP Read community name to some else. This field is required for running the SNMP.System.Summary activity otherwise a <code>SNMP timeout or unknown</code> error occurs.
SNMP Write	Enter the SNMP Write community name for your device. The SNMP Write community name default is normally "private", however, it is recommended you change your SNMP.
Initial login requires password	Initial login to this device requires a password (typically the VTY or initial console password).
Initial login requires username/password	Initial login to this device requires both a username and password
Enable mode requires username/password	The login to enable mode on this device requires both a username and password. The value from the Username field is used along with the Enable password.

Prompts

Describes the command line prompts used by a device if they differ from the default. Entries are only required here where a device has non-standard prompts configured.

VTY Prompt	<p>Enter the non-standard VTY login prompt for this device. The VTY prompt refers to the text that the device prompts you with when asking for the initial login password. For example:</p> <p>"Password: "</p>
Enable Prompt	<p>Enter the non-standard Enable prompt for this device. The Enable prompt refers to the text that the device prompts you with when asking for the Enable password. For example:</p> <p>"Password: " or "Super user login: "</p>
Console Prompt	<p>Enter the non-standard Console prompt for this device. The Console prompt refers to the text that the device prompts you with when asking for the initial console login password. For example:</p> <p>"Password: "</p>
Username Prompt	<p>Enter the non-standard Username prompt for this device. The Username prompt refers to the text that the device prompts you with when asking for the initial login user name, normally used along with a password for AAA/TACACS/RADIUS or Local authentication. For example:</p> <p>"Username: " or "Please enter login name: "</p>
Password Prompt	<p>Enter the non-standard Password prompt for this device. The Password prompt refers to the text that the device prompts you with when asking for the initial login password, normally used along with a username for AAA/TACACS/RADIUS or Local authentication. For example:</p> <p>"Password: " or "Please enter login password: "</p>

Contact info

Describes contact information of personnel responsible for a specific device.

Extra info

Additional device specific information and fields. These fields are all free form and you can enter any text you wish.

Serial number	The serial number of this device.
Asset Tag	Asset tag information.
Identification	Identification info for this device.
Other info	Any other serial number information.

Activity Specific1 / Specific2

Information specific to a particular activity.

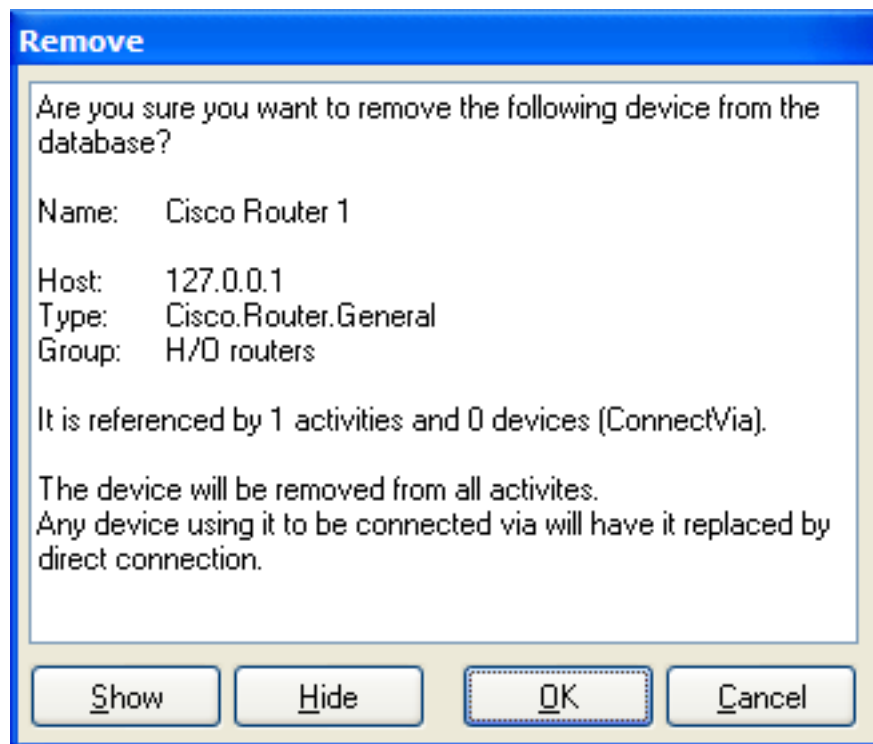
Variations

From the Variations tab you can launch the Device Variations wizard. A number of device specific variables are used by Kiwi CatTools during execution of scripts. If device characteristics do not match specific variables, the scripts do not execute correctly.

The variations wizard allows you to override any unsupported device specific variables on a device-by-device basis with user defined variables.

Remove devices from the Kiwi CatTools database

When removing a selected device from the CatTools database, the following dialogue is shown:



This dialogue box identifies whether the device is used in any activities or devices. If you choose to remove the device any relationships it has with activities or devices are also removed.



Use the Show button to see a list of entities the device is connected to.

Edit a device on the Kiwi CatTools database

To edit a network device that has been added in Kiwi CatTools, select a device on the Devices Pane and click Edit.

The Device Information window opens, with the same tags and options available as with adding a new device.

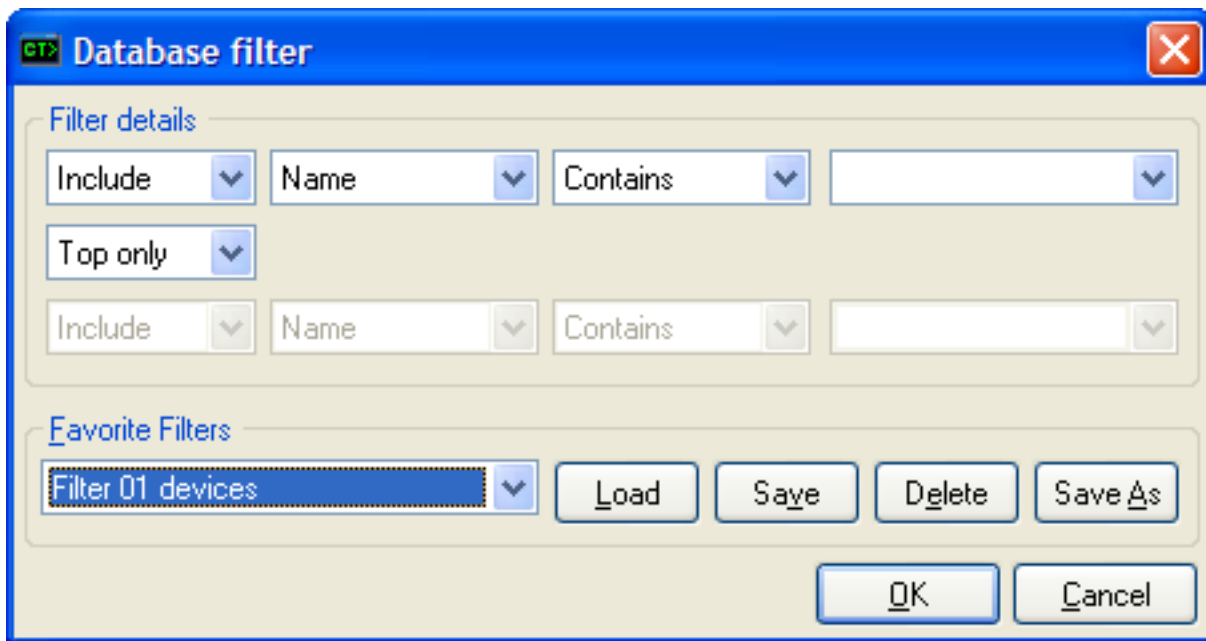
The screenshot shows the "Device Information" dialog box with the following configuration:

- Vendor: [All Vendors]
- Device Type: Cisco.Firewall.ASA
- Group: Default
- Name: ASA Firewall 2
- Host Address: 127.0.0.1
- File Name: ASA_Firewall_2
- Model: ASA5505
- Connect via: Direct connect
- Method: Telnet
- Port: 23

The existing configurations for the device are selected. Once you've made the changes to your device, test the configuration by clicking Ping Device. If the test returns successful, click OK.

Filter devices in the Kiwi CatTools database

To filter the on-screen display of devices under the Devices tab, click **Filter** (F3). This opens the "Database filter" window.



1. Use the Include and Exclude fields to either include or exclude specific device types, names, or other properties as shown in the following fields.
2. Determine the different sets of devices to be included or excluded from the filtered device display based on the device's group, name, host name, type, model or location.
3. Specify which part of the Group or Name the filter operates on:
 - **Contains**
Any part of the name may contain the specified string.
 - **Starts With**
The start of the name matches the specified string.
 - **Ends With**
The end of the name matches the specified string.
4. Enter the specific string that you want the filter to match against the device name, or group, etc.
5. (Optional) Select AND / OR Boolean operators to combine filters. The third option, Top Only, displays only the top-most device that satisfies the criteria entered in the first line of filter options.
6. Create a second filter to combine with the first using Boolean operators.
7. Use the "Favorite filters" box to save your filter for later use.

Kiwi CatTools Panes Overview

Docking panes

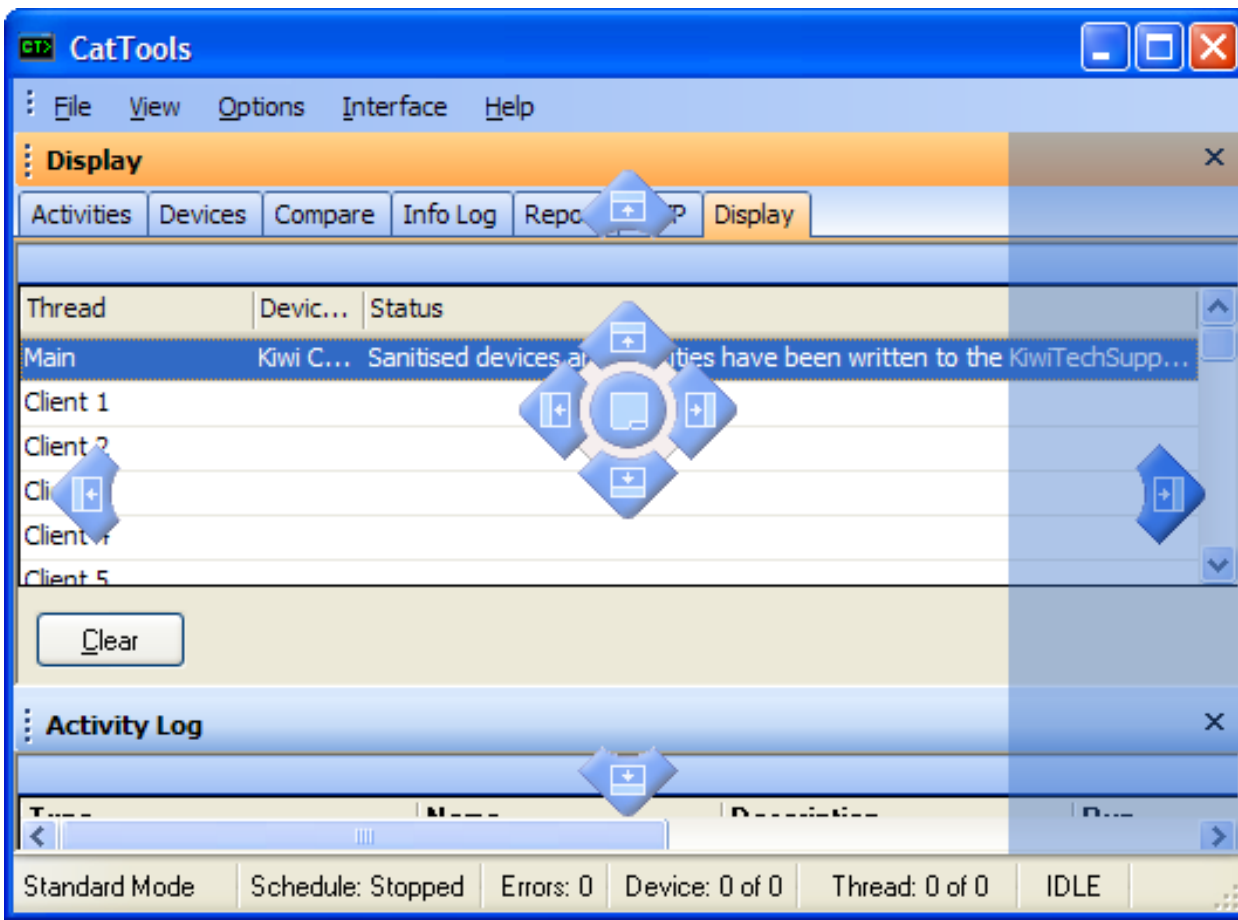
There are a number of different docking options. For example, you can dock panes at the top, bottom, left or right of the screen, or dock them to the top, bottom, left or right of other panes.

To dock a new pane, do the following:

1. Click on the pane you wish to dock and while holding the mouse button down drag it into a pane. The docking pane locator appears on screen.
2. While holding down the mouse button, move the cursor over one of the docking pane locator "stickers". The area where the pane is docked is highlighted.
3. Release the mouse to dock the pane.

The example below shows how to dock the Display pane to the right of the screen by using the right-most locator sticker.

The pane is docked to the right of **all** the existing panes. To dock the Display pane to the right of just the top-most pane, leaving the Activity Log pane to span the width of the window, drop the pane on the right-hand sticker within the center cluster.

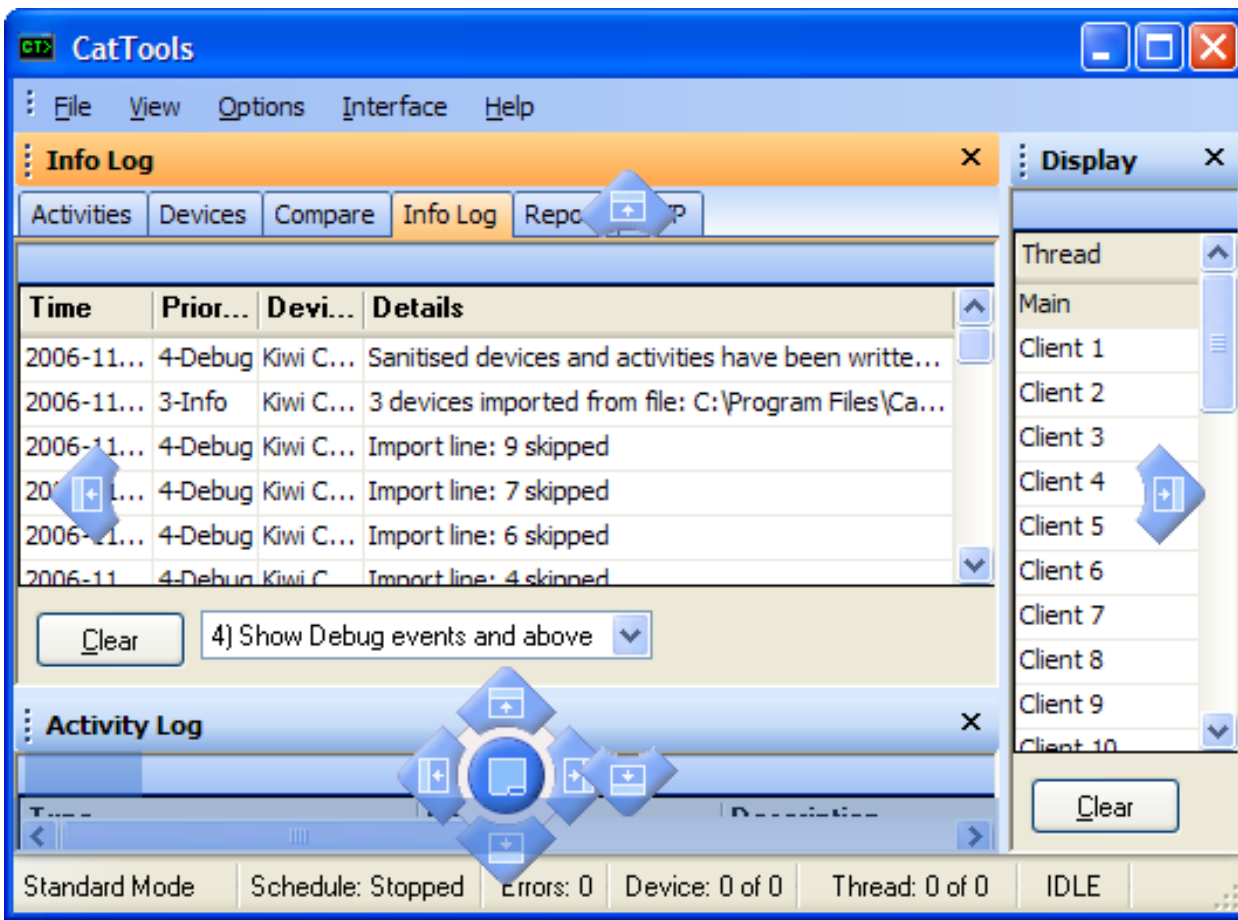


Moving and grouping panes

To move a pane from one group of panes to another, do the following:

1. Click on the pane you wish to move and while holding the mouse button down drag it into the pane you want to group it to. A docking pane locator appears on screen within the desired pane.
2. While holding down the mouse button, move the cursor over the center-most locator sticker for that pane. The whole pane is highlighted.
3. Release the mouse to drop the pane into that pane group.

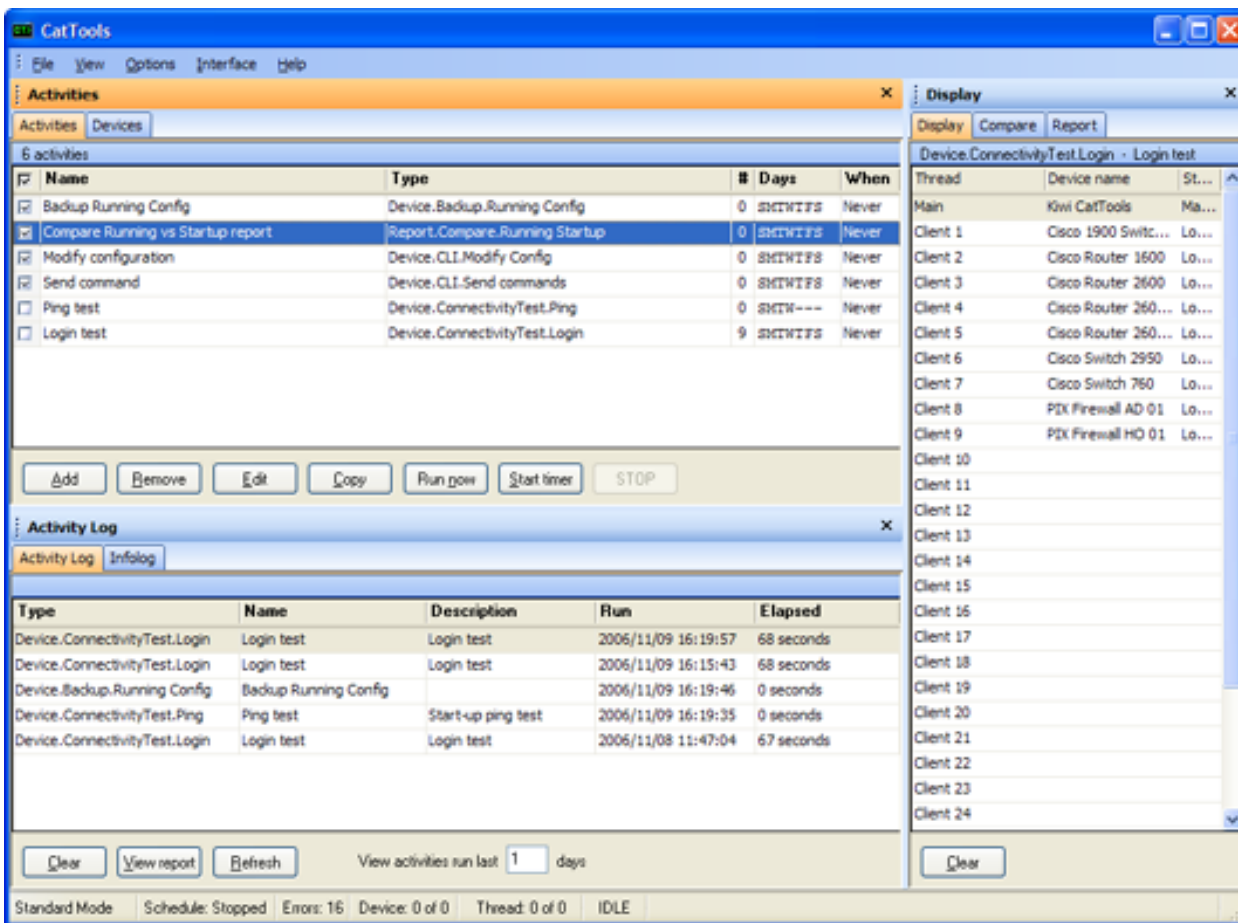
The example below shows how to move the Info Log pane from the top pane group and group it with the Activity Log in the lower pane.



Rearranging and saving your layout

You can rearrange your pane layout easily:

1. Move the mouse cursor over the edge of a pane until the "splitter" cursor appears.
2. Hold the mouse button down and move the cursor to where you want the edge of the pane to go to.
3. Release the mouse button.



Once you have decided on your layout, you can save your changes by clicking on the Interface > Panes > Save user defined layout. CatTools automatically saves your changes when you exit the application. See [Interface](#) for further details.

Resetting pane to default

To reset the panes back to the CatTools default, click on the Interface > Panes > Restore to default layout.

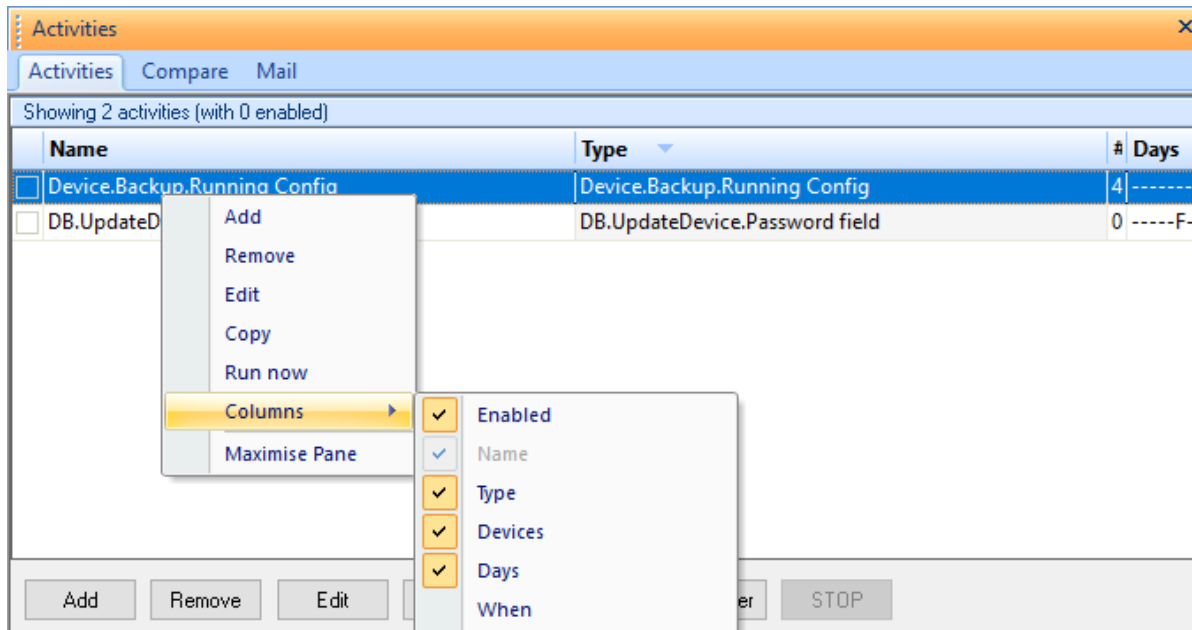
See [Interface](#) for further details.

Right-Click short menu

The functions of the buttons at the bottom of the tabs can also be accessed via the shortcut menu by selecting a row from within a pane and clicking the right mouse button.

The shortcut menu also includes other items which are specific to the pane in which you are currently in. For example, on the Activities pane, there is an item called "Columns" which contains sub items to show and hide data columns relating to activities; Show and Hide the grouping box which allows you to group your activities by data columns, and restore the pane back to its default layout.

You can maximize the current pane by right clicking and selecting the Maximize Pane menu option. To return a maximized pane to its normal size right click and select restore panes.



Kiwi CatTools Activities Pane

This pane displays all the activities you have defined. Right-clicking in the pane brings up a context menu allowing customization and access to core functions.

Option	Description
Add activities	Add a new activity using Add a new scheduled activity .
Edit activities	Edit an existing activity using Edit scheduled activity details.
Copy activities	Copy an existing activity using Edit scheduled activity details.
Run now	Immediately runs a new or edited activity that is selected from the grid.
Start timer	Starts the activity timer and runs scheduled activities. When the timer is active, any selected activities with set schedules run at the specified interval you have set for them.
Remove activities	Delete the selected activity.

Kiwi CatTools Activity log pane

This pane displays a log of all activities that have run since CatTools last started. These entries are also saved to `activitylog.txt` in the main CatTools folder.

Option	Description
Clear	<p>Clears the display of any activity information.</p> <p>This does not delete the activity log file. It only clears the display of logged events.</p>
View report	<p>View the report associated with the highlighted activity.</p> <p>This automatically switches the display to the Report pane and opens the report for viewing.</p>

Kiwi CatTools Compare pane

This pane enables you to create a comparison report of two files of your choice. The report is the same format as that created by the [Device.Backup.Running Config](#) activity when changes are detected.

Option	Description
Save activities	Saves the comparison data displayed to a disk file.
Clear activities	Clears the display of comparison report information.
Select activities	Displays a dialogue that allows you to select the two files to compare.

Kiwi CatTools Info log pane

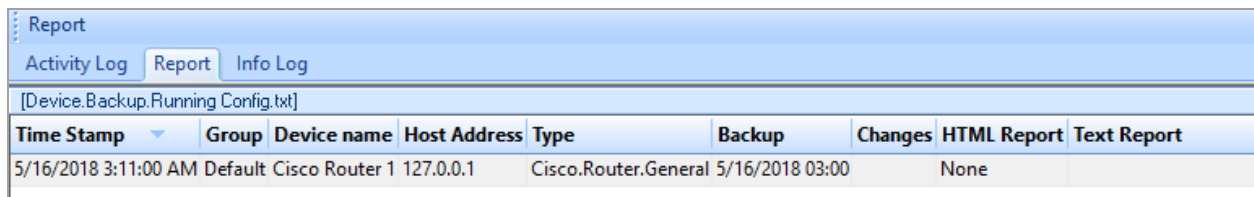
The Info log pane displays a grid of all messages displayed by CatTools while it is running, in the order that they occur. The display may be filtered by using the View filter.

The messages are also logged to `infolog.txt` in the main CatTools folder.

Option	Description
View Filter Drop-Down	<p>This allows you to change the filter options on the Info Log display which controls which level of messages are displayed.</p> <p>There are four filters available:</p> <ol style="list-style-type: none"> 1. Show Error events only. 2. Show Warning events and above. 3. Show Info events and above. This is the default setting. 4. Show Debug events and above
Clear Info log	Clears the Info Log display.

Kiwi CatTools Report pane

The Report pane enables you to view any text reports that are created by CatTools. You may also view any tab delimited text file in the grid.



Time Stamp	Group	Device name	Host Address	Type	Backup	Changes	HTML Report	Text Report
5/16/2018 3:11:00 AM	Default	Cisco Router 1	127.0.0.1	Cisco.Router.General	5/16/2018 03:00		None	

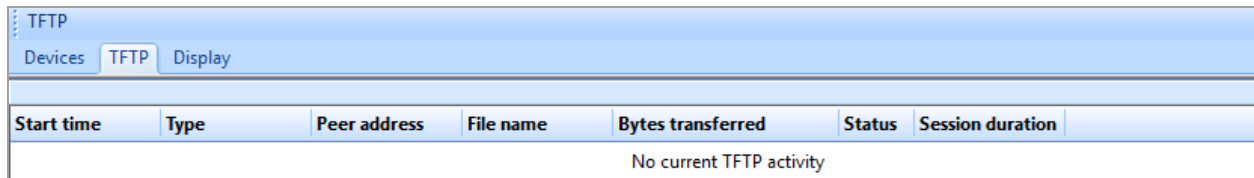
Fields in the viewed file may be modified and saved as required.

Option	Description
Open	<p>Click on this menu option to open a report file.</p> <ol style="list-style-type: none"> 1. Select the file you want to open and click Open. 2. The in-built report viewer opens a tab delimited file. 3. Review the <code>.CSV</code> file to view, sort, edit and save.
Save	Saves the currently displayed report to a file.
Clear Report	Clears the report display.
Delete Report	Delete the report file currently being viewed.

Option	Description
Refresh	Refreshes the view of the report file currently being displayed. This is useful when a report file has been updated while the view is open.

Kiwi CatTools TFTP pane

The TFTP pane enables you to view messages originating from the CatTools TFTP server. You can also stop or start the TFTP server from this tab.

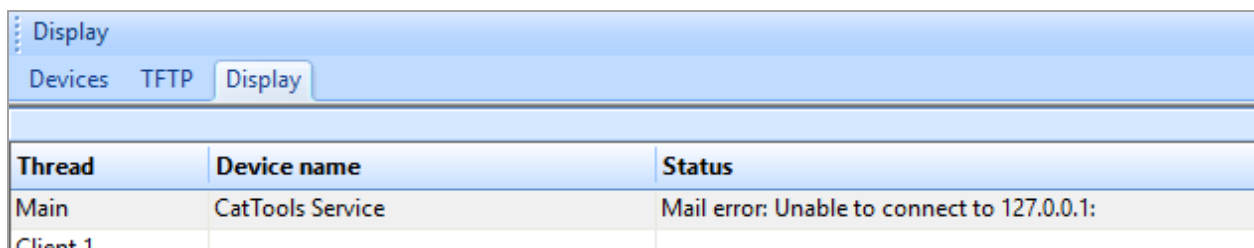


Start time	Type	Peer address	File name	Bytes transferred	Status	Session duration
No current TFTP activity						

Option	Description
Start	Starts the TFTP server, if it is not already started.
Stop	Stops the TFTP server if it is running.
Clear TFTP	Clears the display of TFTP server messages.

Kiwi CatTools Display pane

The Display pane shows the current message displayed by each of the various tasks and clients associated with CatTools.



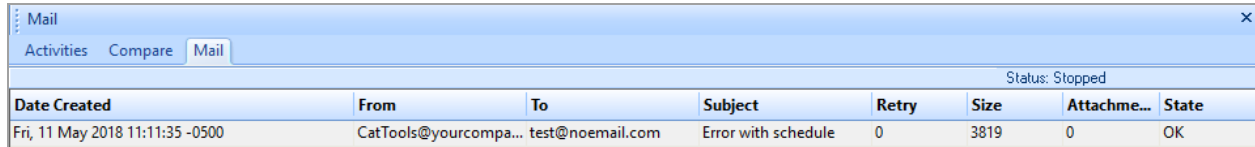
Thread	Device name	Status
Main	CatTools Service	Mail error: Unable to connect to 127.0.0.1:
Client 1		

- The Thread column identifies the originating task, which may be the main CatTools task or one of the clients.
- The Device column identifies the device currently linked to the task.
- The Status column shows the latest message from the task.

Option	Description
Clear Display	Clears the Display pane of any messages.

Kiwi CatTools Mail pane

The Mail pane shows messages currently in the mail queue waiting to be sent.



Status: Stopped							
Date Created	From	To	Subject	Retry	Size	Attachme...	State
Fri, 11 May 2018 11:11:35 -0500	CatTools@yourcompa...	test@noemail.com	Error with schedule	0	3819	0	OK

1. Summarized in the title bar at the top of the window is the total number of messages currently in the queue as well as the current status of the Mailer.
2. The Mailer can be stopped and started using the buttons at the bottom of the window or through the pop-up menu which is shown when the pane is right clicked.

Stop the mailer if you need to change an email message for some reason before it is sent, or if you want to perhaps delete a message from the queue.

3. Entries in the list can have their To, From, and Subject, fields edited in-line should the need be necessary.

The retry state may also be reset on failed messages by selecting this option from the right click pop-up menu.

4. The same right click pop-up menu also contains options for customizing the display allowing the user to turn on and off various columns.
5. Entries may be deleted by selecting them and then choosing the Delete option from the right click pop up menu.
6. If the State of a message cannot be sent due to some kind of error, you can reset the state by selecting the message and choosing the Reset State function. This makes the message available to be sent by the mailer again.

Event Email Notification

Sending an email message is the method CatTools uses to notify you of selected events. The mailer engine of CatTools provides flexible facilities for emailing reports and error notifications created by activities.

This engine normally runs asynchronously to the main CatTools task. When CatTools assembles a notification for emailing, it passes this to the mailer. Rather than sending the message immediately while CatTools waits, the mailer creates the message and puts it on a queue.

Queuing messages

CatTools continues processing the event action once the message is queued successfully. At set intervals, the mailer connects to the mail server and sends it the messages that are on the queue. Meanwhile the CatTools main task continues to function without interruption.

The queuing mechanism enables messages to remain in safety until they can be sent. If the mail server is unavailable for some reason, messages are saved on the queue until the server becomes available.

The mailer can also send messages via a secondary mail server if the primary server is unavailable.

Email server authentication

Mail server authentication is supported:

- A username and password can be used with standard SMTP AUTH type authentication.
- The mailer can also send the username and password to a linked POP server before sending mail to the SMTP server.

Further details are available on the [Mail Pane page](#).

Custom scripting

Kiwi CatTools provides a custom scripting facility to add your own custom devices and activities to the user interface. Custom scripting is beneficial should your device not be supported by one of the pre-defined device types, or the supplied activities not meet your specific requirements.

Detailed information on custom scripting, such as how to create the custom files, location of the custom scripting templates, code examples, and so on, can be found in:

- [Creating a custom device](#) within the [Devices](#) chapter for custom device scripting.
- [Creating a custom activity](#) within the [Activities](#) chapter for custom activity and reports scripting.

When adding custom devices, activities, or reports to CatTools, you need to edit the script files using a script editor.

Script variables in Kiwi CatTools

Kiwi CatTools includes several built in variables available to use in Client scripts. Use the "cl." prefix to access a variable or function from the client. For example:

```
If cl.QuitNow = 1 then exit sub
```

Variables

QuitNow As Long

- If set to a value other than 0, indicates that the script must stop processing and exit immediately.
- When the `STOP` button is pressed on the Activities page, the value of `QuitNow` is set to 1 for each running thread.

i Ensure that this variable is checked within any long duration loops so that the client responds to the stop events in a timely manner.

DeviceName As String

- Name of the current device that has been passed from the main script via the marshal threads function.
- Use this value to perform further database lookups based on this device name.

ScheduleNumber as integer

- Contains the current schedule number. Each activity is assigned a unique identifying number.
- This number is used as the unique key for database lookups of activity information.

ScriptFileClient As String

- File name of the current script file that is currently being run

UniqueFileID As String

- Unique file ID prefix for this script (set from the Main script).
- Use this unique value when writing results files that are to be read by the main script. This avoids other threads from overwriting the same results file.

ProcessID As Long

- The ID number of this thread (1 to MaxThreads).
- This value is not unique since the thread can be reused for running a different script and connecting to a different device.
- This variable can be used when signalling the main script as a way of identifying the number of this thread.

AppPath As String

- The folder where Kiwi CatTools is installed.
- This folder is always terminated with a "\" character.
- Use this variable as the root path when saving data to results files. The ClientTemp folder for example is located under the root path.

DeviceHostnameID

- Hostname as recovered from the device after login. For example: `sales1600`
- When a connection to a device is made, the hostname is extracted from the prompt.
- A prompt of `"sales1600>"` gets extracted and converted to `"sales1600"`.

DeviceVTYPrompt

- This value holds the prompt that is expected to be returned after an action is taken in VTY mode.
- For example, after a show interface command has completed, the DeviceVTYPrompt can be expected, such as: `sales1600>`

DeviceEnablePrompt

- This value holds the prompt that is expected to be returned after an action is taken in enable mode.
- For example, after a show interface command has completed, the DeviceEnablePrompt can be expected: `sales1600#`

RecDataTimeOut as long

- Holds the number of seconds to wait for a response from the device.
- For example, time to wait for login prompt to be received after sending a command.

RxBuffer As String

- Holds the data returned from the active connection.
- For example: `SendData abcde`, then `WaitForMultData`
- After a return value has been matched or the timeout has occurred, the received data will be held in `RxBuffer`.

StripVT100ESC as Long

- If set to non-zero, vt100 type escape sequences are removed from the buffer stream at the point of reception.
- This is useful for devices with menu based telnet interfaces.
- This variable can be toggled at any time to enable or disable the removal of the escape sequences.
- For example, set this variable to 1, then call the `WaitForMultData` function.

DebugMode as long

- Determines when to debug data that is sent or received from an active connection. Debugged data is save to a file.
- The file name is made up from the formula: `AppPath + "Capture" + UniqueFileID + ".txt"`.
- 0 = normal operation.
- 1 = sent and received data from the active connection is written to a debug text file.

Device values

After calling the function `DBDevicesGetAllFields(DeviceName)`, the following fields are loaded with the information from the database.

CurDevType As String

- The current device type.
- For example: `Cisco.Router.General` or `Cisco.Firewall.PIX`
- To ensure a good match, always test for lowercase values. If `Lcase(CurDevType) = "cisco.router.general"` then...

CurDevGroup As String

- Group to which the device belongs. This is a free form field and can be any value that the user specifies.

CurDevName As String

- The current device's name. This is a free form field and can be any value that the user specifies.
- The device name must be unique.

CurDevHostAddress As String

- IP address or hostname of the device.

CurDevFilename As String

- A base filename for use when saving information to file for this device.
- The filename is made from the device name.
- Any invalid filename characters are replaced with an underscore.

CurDevModel As String

- Current device model, for example: `1601` or `3300`. This is a free form field and can be any value that the user specifies.

CurDevConnectVia As String

- The name of the device to connect to first before attempting a telnet or SSH session.
- For example, you connect to Device A , then attempt a telnet connection to the current device.

CurDevTelnet As String

- Method used for connection, for example: `Telnet`, `SSH1`, or `COMM`.
- The connection method is dependent on the particular device.

CurDevTelnetPort As String

- Port to use for connection attempt, for example: `23` for telnet, `22` for SSH, `1` for COMM1.

CurDevSession As String

- Session number to use once connected to device.
- For example: Telnet to device, then issue `session 15` command to connect to LANE card.

CurDevVTYPass As String

- VTY password for the device.

CurDevConsolePass As String

- Console password for the device.
- The console password is only required when the connection is via the console (COMM port connection).

CurDevEnablePass As String

- Enable password for the device. This is usually the enable-secret password on Cisco devices.

CurDevPrivilegeLevel As String

- The privilege level to use when entering enable mode.
- For example: `Enable 7`

CurDevAAAUsername As String

- Username to use when device is set to use AAA (TACACS/RADIUS/Username)

CurDevAAAPassword As String

- Password to use when device is set to use AAA (TACACS/RADIUS/Username)

CurDevVTYPrompt As String

- The VTY prompt to expect from the device.
- Normally set to "" when the default prompt is expected.
- For example: `Login:`

CurDevConsolePrompt As String

- The console prompt to expect from the device.
- Normally set to "" if the default prompt is expected.
- For example: `Login:`

CurDevEnablePrompt As String

- The Enable mode prompt to expect from the device.
- Normally set to "" if the default prompt is expected.
- For example: `Password:`

CurDevAAAUserPrompt As String

- The username prompt to expect from the device.
- Normally set to "" if the default prompt is expected.
- For example: `Username:`

CurDevAAAPassPrompt As String

- The password prompt to expect from the device.
- Normally set to "" if the default prompt is expected.
- Example: `Password:`

CurDevRequireVTYLogin As String

- VTY password is required to login to the device. Check box value.
- 0 = no VTY password expected
- 1 = VTY password expected

CurDevLoginUsesAAA As String

- AAA username and password required to login to the device. Check box value.
- Uses AAA user authentication.
- 0 = no username or password required
- 1 = username or password required

CurDevEnableUsesAAA As String

- AAA username or password required when entering enable mode on the device. Check box value.
- Uses AAA user authentication.
- 0 = no username or password required for enable mode
- 1 = username or password required

CurDevAddress1 / CurDevAddress2 / CurDevAddress3 As String

- Free form address information.

CurDevContactName As String

- Name of contact for device.

CurDevContactPhone As String

- Contact phone number for device contact.

CurDevContactEmail As String

- Contact e-mail address for device contact.

CurDevContactOther As String

- Other contact info for device contact.

CurDevAlertEmail As String

- E-mail to send alert info to for this device.
- If left blank, the e-mail address defined in the properties is used.

CurDevSerialNumber As String

- Device serial number. Free form field.

CurDevAssetTag As String

- Device asset tag information. Free form field.

CurDevIdentification As String

- Other device identification. Free form field.

CurDevSerialOther As String

- Other device identification. Free form field.

CurDevActivitySpecific1 As String

- Activity specific information for this device.

CurDevActivitySpecific2 As String

- Activity specific information for this device.

Temp01 to Temp10 as Variant

- General purpose global variables for use by the client scripts.
- Can be used to allow transfer of data from one script to another.
- These values are not initialized on startup and may contain data from previous script instances.

Script functions in Kiwi CatTools

Kiwi CatTools includes several built in functions available to Client scripts. Use the "cl." prefix to access a function from the client. For example:

```
cl.Log 1, "Expected response not received from command ..."
```

Functions

ConnectHost() As Boolean

- Attempts to connect to the current host specified in `CurDevHostAddress`.
 - `True` = the connection is successful.
 - A warning message is logged if a connection attempt fails.
- The connection is attempted three times before failing. After three failed attempts, an error message is logged.

ConnectHostSingleAttempt() As String

- Attempts to connect to the current host specified in `CurDevHostAddress`.
- If the connection is successful, a value of "OK" is returned. Otherwise the error description is returned.
- This function is called 3 times by [ConnectHost](#).

ConvertToFilename(Filename as variant) As String

- Converts a device name into a suitable filename.
- Replaces any invalid characters in the filename with an underscore, including quotes and spaces. The following characters are considered invalid in a filename: "&+? : ; , () = | \ / < > * ^ % @

CreateFile(Filename) As Boolean

- Attempts to create the specified file even if the path does not exist.
 - If the file already exists, no action is taken.
 - If the file does not exist, the path and a file are created. The file is 0 bytes in length.

CurrentDateStamp(Format as long) as String

- Returns the current date formatted based on the value of "Format".
- Format values:
 - 0 = ISO Format (YYYY/MM/DD)
 - 1 = US Format (MM/DD/YYYY)
 - 2 = UK Format (DD/MM/YYYY)

CurrentDateTime(Format as long) as String

- Returns the current date and time formatted based on the value of "Format".
- Format values:
 - 0 = ISO Format (YYYYMMDDHHNNSS)
 - 1 = US Format (MMDDYYYYHHNNSS)
 - 2 = UK Format (DDMMYYYYHHNNSS)

CurrentDateTimeStamp(Format as long) as String

- Returns the current date and time without separators, formatted based on the value of "Format".
- Format values:
 - 0 = ISO Format (YYYY/MM/DD HH:MM:SS)
 - 1 = US Format (MM/DD/YYYY HH:MM:SS)
 - 2 = UK Format (DD/MM/YYYY HH:MM:SS)

DBCheckScheduleOption(ScheduleNumber, Option Number) As Long

- Returns the activity specific option check box value.
 - 0 = unchecked
 - 1 = checked.
- Each activity has 10 optional data values that can be set for the activity. Pass the activity number and an option value - from 1 to 10 - and the return value of the specific option check box is returned.
- Refer to the activity .ini file for details of each specific option.

Sub DBDevicesClearCurDevFields()

- Clears all the current device variables and sets them to "".

Sub DBDevicesGetAllFields(DeviceName)

- Locates the specified device in the database and loads all the fields into the CurDevXXX variables.
- To check if the function worked: `check len(CurDevName) > 0`

Function DBDevicesGetField(DeviceName, FieldName) as Variant

- Returns the data contained in the field "FieldName" of device "DeviceName" from the database file.
- Returns blank if the data retrieval fails.

Function DBScheduleGetField(ScheduleNumber, FieldName) as String

- Returns the data contained in the field "FieldName" of activity "ScheduleNumber" from the database.
- Returns blank if the data retrieval fails.

Function DeleteFile(Filename) As Boolean

- Deletes the specified file and returns true or false based on if the file exists after the delete attempt.
 - True = File deleted.
 - False = File still exists and was not deleted successfully.

Sub DisconnectHost()

- Closes the active connection to the current device

Function FileExists(Filename) As Boolean

- Tests to see if the specified file exists.
 - True = the file is found.
 - False = the file is not found.

Sub FlushRxBuffer()

- Flushes the data found in the receive buffer and waits until there is no further data being received from the active socket.

Sub InfoUpdate(Message)

- Sends an info update message back to the main program.

Sub LogToFile(Filename, Data, Append)

- Opens a file for writing. Writes the data stored in Data to the file.
 - Append = 1, the data is appended to the end of the file.
 - Append = 0, the file is overwritten with the new data.

Function Ping(Hostname As String, Timeout As Long, PingCount As Long, ProcessID As Long) As String

- Attempts to ping the host name and returns the results in the format:

```
!!!! TAB 100% TAB 10 TAB 100 TAB 80 or ..... TAB 0% TAB TAB TAB
```

- `Hostname` is the IP address or hostname of the device to ping Timeout is in seconds. The default is 5 seconds.
 - `PingCount` is the number of Ping echo requests to send. Default is 5 pings.
 - `ProcessID` is the current processID (`cl.ProcessID`). This is used to identify the sent ping packets so that the responses can be tied back to the correct process that sent them.
- The min, max and average response times are calculated from any ping echoes received.
- The return data is tab delimited for easy parsing and reporting.

Function ReadFromFile(Filename) As String

- Reads the contents of the specified filename and returns the data as a string value.
- A "" value is returned if the file doesn't exist or contains no data.

Function ReplaceClientFilenameVariables(Filename) As String

- Replaces the `%Variable%` item with the associated value.
 - `%AppPath%` = `cl.AppPath`
 - `%GroupName%` = `cl.CurDevGroup`
 - `%DeviceName%` = `cl.CurDevName`
 - `%DateISO%` = `cl.CurrentDateStamp(0)` ("YYYY-MM-DD")

Sub SaveResults(Data, Append)

- Saves results data into separate files.
- The file name is created from the device name and `UniqueFileID` set in the Main script in the format:

```
Filename = cl.AppPath + "\ClientTemp\" + UniqueFileID + ConvertToHex(DeviceName) + ".txt"
```

- `DeviceName` is converted to hex format to remove any invalid characters present.

Sub SendData(MessageText)

- Sends the string in `MessageText` to the current connection.

Function WaitForMultData(Mult, Choices, Timeout) As Long

- Monitors the `RxBuffer` for data contained in the `Mult()` array.
- If a match is found, the array index value is returned.
 - A return value of 0 indicates that no match was found before the timeout occurred.
- The timeout is specified in seconds and the variable `cl.RecDataTimeOut` is usually passed as this value.
- Choices indicates the number of elements in the `Mult` array. For example: `Mult(1) = "Login:" Mult(2) = "Password:", Choices=2`

Function ReplaceControlChrs(Message) As String

- Replaces any control characters less than ASCII value of 32 with a text string of the numeric value.
- For example: A tab character (ASCII 9) becomes `<009>`.
- This is used when logging debug data to the display or file for later analysis.

cl.Log

Sub Log(Priority, Message)

- Sends a log event message back to the main program.
- Use one of the following priority codes to classify the message:
 - 4 = Debug
 - 3 = Info
 - 2 = Warning
 - 1 = Error

Script editors

In order to modify custom script files, you need to load them into a text file or script editor. While Notepad.exe is a capable text file editor, you may find the script file much easier to read and modify within a syntax highlighting (color-coding) script editor.


- If you have Microsoft Visual Basic or Visual Studio installed on your system, open the .txt files with either of these programs to view the files with syntax highlighting.
- If you do not have access to a syntax highlighting editor, SolarWinds recommends downloading one from the Internet.

Once installed, change the highlighting settings to open a custom script .txt or .custom file with syntax highlighting for MS VBScript language.

For instructions on configuring your text editor to apply syntax highlighting for the Microsoft VBScript language, refer to your editor's reference materials or support forum.

Kiwi CatTools API


The Kiwi CatTools API is an included tool you can use to directly read from and write to the CatTools database. The API performs the same functions as [Add](#), [Remove](#), and [Edit](#) in the CatTools UI.

 The CatTools API is only available in the Licensed edition of CatTools.

If you create an application or code that uses the CatTools API, then you can only run this on the system where CatTools has been installed. The application must read the CatTools license details in order to work.

- The API is limited to database connectivity and accessing of the device and device related data.
- There are no activity related public classes available within the API to modify activities.
- The API is accessed using various [Classes](#) which are explained in this section.
- The API includes sample code you can customize to help you get started.

API development environments

 Due to the capabilities of the CatTools API, specifically that it can directly modify device data in the CatTools database, it is assumed that:

- Users accessing the CatTools API have a reasonable knowledge of Visual Basic programming skills and are familiar with the Visual Basic development environment, such as Visual Studio.
- Alternatively, users know how to utilize the VBA programming environment within Microsoft Office applications.

The API sample code is written in the VB/VBA programming language.

Within development environments, you can add a reference to the CatTools API called "CatTools API" to expose the classes available. Once this reference has been added, copy and paste the sample code from the CatTools API help pages into your VB project to execute the code.

Text editors

If you are using a text editor, such as Notepad, you cannot add a [reference](#) to the CatTools API. For text editors, you must change the code to use the `CreateObject` function to reference the CatToolsAPI available classes. For example:

```
Dim DB
Set DB = CreateObject("CatToolsAPI.Database")
```

This would be used to replace the lines in the sample code:

```
Dim DB As CatToolsAPI.Database
Set DB = New CatToolsAPI.Database
```

Save the text file as a VBScript `.vbs` file and run it.

Register the CatTools API library

In order for the API code to work, the CatTools API library file - `Cattools.dll` - must be registered on the system correctly:

1. From the Windows Explorer, navigate to Start and open the Run dialog.
2. Open the `cattools.dll`:

```
regsvr32 "C:\Windows\system32\cattools.dll"
```

Adding a reference within a VB editor

To add a reference using the Visual Basic editor within Microsoft Office applications:

1. Go to Tools > References.
2. Select CatTools API from the available references.

To add a reference using the Visual Basic 6 IDE application:

1. Go to Project > References.
2. Select the CatTools API from the available references.

Kiwi CatTools API classes

There are several API classes are contained within the CatTools API.

Database	Used to connect to the database and verifies license requirements.
Devices	An enumerator for a collection of Device objects as well as the method by which to get and put devices into the database.
Device	Encapulates each Device stored in the CatTools database.

DeviceTypes	A collection of DeviceType objects
DeviceType	A CatTools DeviceType. For example: <code>Cisco.Router.General</code>
Groups	A collection of Group objects
Group	A CatTools device Group. For example: <code>Default</code>

Kiwi CatTools API: Database class

The Database class is used to connect to the CatTools database. In doing so, this class also verifies your license. The database that the API connects to is the same as the one stored in your `KiwiCatTools-Settings.ini` file.

Properties

Version	The version of the <code>API.dll</code> file.	Read Only
Path	The database being accessed. This is evaluated from your <code>.ini</code> file.	Read Only
EncryptionPassword	The encryption password needed to decrypt the database.	
Err	The ID of the last error encountered.	Read Only
ErrDescription	The last error encountered.	Read Only
Connected	Verifies the database is connected.	Read Only

Methods

OpenConnection	Opens a connection to the database as defined in your <code>.ini</code> file.
CloseConnection	Closes the connection to the database.
Devices	A collection of the devices contained in the database.
ErrClear	Clears any error messages.
DeviceTypes	A collection of the device types contained in the database.
Groups	A collection of the groups contained in the database.

Example: Connect to a database using CatTools API

The following sample Visual Basic code showing you how to connect to your database using the CatTools API.

```
Dim DB As CatToolsAPI.Database
Set DB = New CatToolsAPI.Database

DB.EncryptionPassword = "MyPassword"

If DB.OpenConnection Then
    MsgBox "Connected ok"
Else
    MsgBox "DB connection failed: " & DB.ErrDescription
End If

DB.CloseConnection
Set DB = Nothing
```

Kiwi CatTools API: Devices class

The Devices class is used to enumerate CatTools [Device objects](#) and to enact changes to a Device, such as add, edit and delete.

Properties

Count	Returns the number of devices in your database.
-------	---

Methods

AddNew	Returns a Device. This Device can then have its settings modified before being saved to the database.
--------	---

- Takes as a property the name of the device-type that you would like to create.

DeleteByDevice	Deletes a device from the database.
----------------	-------------------------------------

- Takes a device object as a parameter.

DeleteByName	Deletes a device from the database using the devices name as an identifier.
--------------	---

GetByName	Returns a Device object.
-----------	--------------------------

- Takes the devices name as a parameter.

Items	The collection of Device objects.
Refresh	Resynchronize with the database.
SaveDevice	Saves a Device object to the database. <ul style="list-style-type: none"> • Takes the Device to be saved as a parameter.

Example code: Device iterations

The following sample Visual Basic code showing you how to iterate devices using the CatTools API.

```

Dim DB As CatToolsAPI.Database
Dim Devices As CatToolsAPI.Devices
Dim Device As CatToolsAPI.Device

Set DB = New CatToolsAPI.Database

DB.OpenConnection
If DB.Connected Then
    Set Devices = DB.Devices
    Debug.Print Devices.Count & " devices in the database"
    For Each Device In Devices
        If Device.Group = "Default" then
            Debug.Print Device.Name & " " & Device.HostAddress
        End If
        DoEvents
    Next
Else
    Debug.Print "Connection failed: " & DB.ErrDescription
End If

DB.CloseConnection

Set Device = Nothing
Set Devices = Nothing
Set DB = Nothing
    
```

Kiwi CatTools API: Device class

The Device class is used encapsulates a Kiwi CatTools device.

Properties

AAAUsername / AAAPassword	The username and password to use if your device requires a username/password login.
AAAUserPrompt / AAAPassword Prompt	The prompt request for the username and/or password of your device.
ActivitySpecific1 / ActivitySpecific2	<i>Not used</i>
Address1 / Address2 / Address3	The location of the device.
AlertEmail	User to receive notifications by e-mail for any triggered alerts from this device.
AssetTag	Asset tag information.
ConnectionMethod	The method to use when connecting to this device. <ul style="list-style-type: none"> • For example: Telnet or SSH2.
ConnectVia	The name of the device to connect via.
ConnectViaID	The ID of the device to connect via. <ul style="list-style-type: none"> • An ID of <code>-1</code> indicates a direct connection.
ConsolePassword	The console (com port connection) password.
ConsolePrompt	What does the console password request look like from your device.
ContactName / ContactEmail / ContactPhone / ContactOther	Contact information for the person responsible for a device.
EnablePassword	The password to get into enable mode.
EnablePrompt	What does the enable password request look like from your device.
EnableUsesAAA	Does entering enable mode require username/password validation?
FileName	The base file name for the device.
Group	The name of the Group the device belongs to. For example: <code>Default</code>
GroupID	The id corresponding to the Group field. <code>-1</code> indicates a new Group is to be created.

HostAddress	The IP address or host address for this device.
ID	The id that identifies this device in the database.
Identification	Identification info for this device.
LoginUsesAAA	Does connecting to the device require username/password validation?
Model	A value describing the model of the device.
Name	A unique name that describes this device.
Port	The port to use when connecting to this device.
PrivilageLevel	Sets the enable mode privilege level.
RequireVTYLogin	Does connecting to the device require password only validation?
SerialNumber	The serial number for the device.
SerialOther	Additional serial number information.
SNMPRead	SNMP Read community name.
SNMPWrite	SNMP Write community name.
SSHPassword	The SSH password required for SSH connections to this device.
SSHUsername	The SSH username required for SSH connections to this device.
TypeID	The id of the device-type eg 10
TypeOfDevice	The name of the device-type. For example: <code>Cisco.Router.General</code>
VTYPassword	The password to use if your device verifies using a password only to log in.
VTYPrompt	What does a password request look like from your device.

Example code: Add, edit, or delete entries in a database

The following sample Visual Basic code showing you how to use the Device object of the CatTools API to add, edit or delete an entry in the database.

Add an entry to a database

```

Dim DB As CatToolsAPI.Database
Dim Device As CatToolsAPI.Device

Set DB = New CatToolsAPI.Database
If DB.OpenConnection Then
    Set Device = DB.Devices.AddNew("Cisco.Router.General")
    If Not Device Is Nothing Then
        Device.Name = "My Device 01"
        Device.HostAddress = "192.168.1.1"
        Device.Group = "My Test Group"
        Device.RequireVTYLogin = True
        Device.VTYPass = "My Password"
        Device.ConnectionMethod = "Telnet"
        If DB.Devices.SaveDevice(Device) Then
            Debug.Print Device.Name & " saved to the database OK."
        Else
            Debug.Print "Adding " & Device.Name & " failed: " & DB.ErrDes
        End If
    Else
        Debug.Print "Creation of a new device failed: " & DB.ErrDescription
    End If
Else
    Debug.Print "DB connection failed: " & DB.ErrDescription
End If

Set Device = Nothing
Set DB = Nothing
    
```

Edit an entry in a database

```

Dim DB As CatToolsAPI.Database
Dim Devices As CatToolsAPI.Devices
Dim Device As CatToolsAPI.Device

Set DB = New CatToolsAPI.Database
If DB.OpenConnection Then
    Set Devices = DB.Devices
    Set Device = Devices.GetByName("My Device 01")
    If Not Device Is Nothing Then
    
```

```

        Debug.Print Device.Name & " Loaded OK"
        Device.AAAPassword = "My New Password "
        If Devices.SaveDevice(Device) Then
            Debug.Print Device.Name & " saved OK"
        Else
            Debug.Print Device.Name & " NOT saved: " & DB.ErrDescription
        End If
    Else
        Debug.Print "Device not found"
    End If
Else
    Debug.Print "DB connection failed: " & DB.ErrDescription
End If

Set Device = Nothing
Set Devices = Nothing
Set DB = Nothing
    
```

Delete an entry from a database

```

Dim DB As CatToolsAPI.Database

Set DB = New CatToolsAPI.Database
If DB.OpenConnection Then
    If DB.Devices.DeleteByName("My Device 01") Then
        Debug.Print "Deleted OK"
    Else
        Debug.Print "Delete failed: " & DB.ErrDescription
    End If
Else
    Debug.Print "DB connection failed: " & DB.ErrDescription
End If

Set DB = Nothing
    
```

Kiwi CatTools API: DeviceTypes class

The DeviceTypes class is used to enumerate CatTools [DeviceType objects](#).

Properties

Count	Returns the number of device types in your database.
-------	--

Methods

Items	The collection of Device objects.
-------	-----------------------------------

Refresh	Resynchronise with the database.
---------	----------------------------------

Sort	Sort the collection of DeviceType objects in ascending or descending order.
------	---

DeviceTypes sample code

The following sample Visual Basic code showing you how to list the Device Types found in your database using the CatTools API.

```

Dim DB As CatToolsAPI.Database
Dim DeviceTypes As CatToolsAPI.DeviceTypes
Dim DeviceType As CatToolsAPI.DeviceType

Set DB = New CatToolsAPI.Database

DB.EncryptionPassword = ""
DB.OpenConnection
If DB.Connected Then
    Set DeviceTypes = DB.DeviceTypes
    If Not DeviceTypes Is Nothing Then
        Debug.Print DeviceTypes.Count & " device types in the database"
        For Each DeviceType In DeviceTypes
            Debug.Print "->" & DeviceType.Name
            DoEvents
        Next
    Else
        MsgBox "Error:" & DB.ErrDescription
    End If
Else
    MsgBox "Connection failed: " & DB.ErrDescription
End If

DB.CloseConnection
Set DeviceType = Nothing
    
```

```
Set DeviceTypes = Nothing
Set DB = Nothing
```

Kiwi CatTools API: DeviceType class


The DeviceType class encapsulates a CatTools device type.

Properties

ID	The ID used to identify this Device Type in the database.
IniFile	The ini file used to define this DeviceType
Name	The friendly name of the DeviceType that you see on screen.

Add, edit, devices using DeviceType class

Use the [DeviceTypes class](#) to enumerate DeviceType objects and populate them into a drop list for your user to select from.

 It is not possible to add, edit, or delete device types using the DeviceType class.

Kiwi CatTools API: Groups class


The Groups class is used to enumerate CatTools [Group objects](#).

Properties

Count	Returns the number of device types in your database.
-------	--

Methods

Items	The collection of Group objects.
Refresh	Resynchronise with the database.
Sort	Sort the collection of Group objects in ascending or descending order.

 You can only enumerate Groups you can not add edit or delete them using this object. To add a Group simply give the text value to your [Device](#) and when it is saved the Group will be created.

Groups sample code

The following sample Visual Basic code shows you how to list the Groups found in your database using the CatTools API.

```

Dim DB As CatToolsAPI.Database
Dim Groups As CatToolsAPI.Groups
Dim Group As CatToolsAPI.Group

Set DB = New CatToolsAPI.Database

DB.OpenConnection
If DB.Connected Then
    Set Groups = DB.Groups
    If Not Groups Is Nothing Then
        Debug.Print Groups.Count & " groups in the database"
        For Each Group In Groups
            Debug.Print "->" & Group.Name
            DoEvents
        Next
    Else
        MsgBox "Error:" & DB.ErrDescription
    End If
Else
    MsgBox "Connection failed: " & DB.ErrDescription
End If


DB.CloseConnection
Set Group = Nothing
Set Groups = Nothing
Set DB = Nothing
    
```

Kiwi CatTools API: Group class

The Group class encapsulates a CatTools device Group object. These values are presented in the Group drop list when you edit a device.

Properties


ID	The ID used to identify this Group in the database.
Name	The friendly name of the Group that you see on screen.

 When CatTools starts it removes from the database any unlinked entries.

Add a group using the Group class

To add a new group simply assign it to a [Devices.Group property](#).

Use the [Groups class](#) to enumerate Group objects and populate them into a drop list for your user to select from.

 It is not possible to add, edit, or delete Groups using the Groups class.

Kiwi CatTools API Limitations

There are several limitations to be aware of when working in the Kiwi CatTools API.

1. The API is only available with the licensed edition of CatTools. It is not available with the Freeware edition.
2. The API is a part of CatTools and must be installed on the same computer as CatTools. For example, you can not take the .dll and use it from another computer to access CatTools on a remote computer.
3. The API is limited to database connectivity and accessing of the device and device related data. There are currently no activity-related public classes available within the API to manage activities.
4. The API is directly modifying the device data within the CatTools database, such as the underlying device values used within an activity. The API always recommends that the CatTools scheduler timer is stopped within the CatTools UI and that no activity is currently running before the API code or script is executed. Failing to do so may result in undesirable results.
5. You cannot start or stop the CatTools scheduler timer if it has not been exposed in the API.

Automating the installation of Kiwi CatTools

Automate the installation and startup of CatTools without the need for any human interaction.

Install as a standard interactive application

1. To install and start CatTools as a standard interactive application, create a batch (.bat) file that contains the following information:

```
"*AppPath\CatTools_X.X.X.exe" /S INSTALL=APP /D=*InstallPath
"*InstallPath\CatTools.exe"
```

2. Your .bat file now looks similar to:

```
"C:\CatTools_3.5.0.setup.exe" /S INSTALL=APP /D="C:\Program
Files\CatTools3"
"C:\Program Files\CatTools3\CatTools.exe"
```

Install as a Windows service

1. To install and start CatTools as a Windows service, create a batch (.bat) file that contains the following information:

```
"*AppPath\CatTools_X.X.X.exe" /S INSTALL=SERVICE /D=*InstallPath
"*InstallPath\CatTools_Manager.exe"
```

2. Your .bat file now looks similar to:

```
"C:\CatTools_3.5.0.setup.exe" /S INSTALL=SERVICE /D="C:\Program
Files\CatTools3"
"C:\Program Files\CatTools3\CatTools_Manager.exe"
```


Troubleshooting

If you are having problems look here to see if there is an answer to some issues users may experience.

File naming conventions	Debug logs	Device specific issues	Error 70: Permission denied	Reporting problems
Remote desktop systems	Remote authentications	Anti-virus tools	Windows XP Firewall	Unscheduled service is running

- For additional support, check out the [Kiwi CatTools Knowledge Base](#).
- If you require further help, [contact our technical support team](#).

Collect and send troubleshooting logs to SolarWinds Support

If requested, you can collect and send troubleshooting logs to SolarWinds Support to aid in the research and resolution of issues such as KCT application crashes or memory leaks in Windows.

When making a support ticket, include the following information when available:

- Version of Kiwi CatTools installed and the operating system on the KCT server.
- Screenshots of the error message and/or issue experienced.
- Approximate Time and date of when the issue started and if Kiwi CatTools was working before the issue occurred.
- Any changes made to the environment prior to the issue.
- Note if the issue has been replicated or if it occurs at random.

Gather and send the following troubleshooting logs to support for further analysis:

1. If the error appears in your Event Viewer Logs, gather a copy of the Kiwi CatTools error.
2. Make a copy of the System Information file (NFO) from your computer.
3. (optional) Make a copy of the Windows DXDiag logs.
4. If available, collect memory dump files caused by Kiwi CatTools.

5. Copy the following .txt files in the C:\Program Files (x86)\CatTools3 folder:
 - InfoLog.txt
 - SendMailLog.txt
 - ActivityLog.txt
6. Generate a Kiwi CatTools Diagnostic log from the console. Go to File > Debug > Create diagnostics for Technical Support
7. Submit the collected files to support when [Submitting a Ticket](#).

Best practices

- Provide the complete wording of the error you are receiving. The error message should be present in `infolog.txt` that describes the error.
- Provide the type of device the problem is happening to. CatTools [supports a number of different devices](#) that can exhibit very different problem behaviors.
- Do not send raw screen shots of messages or setup issues, unless there is no other way of getting the information across. Mail servers often block this data. Use the CatTools facilities from the File menu to provide textual information.
- Send the complete `infolog.txt` file as a zipped attachment. Trying to copy and paste entries from the `infolog` into an email can produce poorly formatted emails.

Debug Log

The debug log gives the exact picture of communication between devices and CatTools. The debug log is used when there may be some communication errors between Kiwi CatTools and a device.

The four main notations used in debug log are:

- <R- = data read from the device.
- <W- = data written to the device.
- <C- = connected to the device.
- <D- = disconnected from the device.

Sample Debug log

```
<NEWSESSION CatTools 3.10.0 1/29/2014 3:47:29 AM>
<PROTOCOL=Telnet>
<DEVICE TYPE=Cisco.Router.General>
<ACTIVITY TYPE=Device.Backup.Running Config>
```

```

<ACTIVITY SCRIPT=C:\Program
Files\CatTools3\Scripts\Client.Device.Backup.Running Config.txt>
<USERS NAME FOR DEVICE=Cisco Router 1>
<C OK 3:47:30 AM>
<R-3:47:30 AM>[13][10][13][10]User Access Verification[13][10][13]
[10]Username:
<W-3:47:30 AM>admin[13]
<R-3:47:31 AM>admin
<R-3:47:31 AM>[13][10>Password:
<W-3:47:31 AM>password[13]
<R-3:47:31 AM>[13][10]bgp-2651-03#
<W-3:47:31 AM>[13]
<W-3:47:41 AM>logout[13]
<D 3:47:41 AM>
<SCRIPT VALUES>
<HOSTNAME="bgp-2651-03">
<PROMPT VTY="bgp-2651-03">
<PROMPT ENABLE="bgp-2651-03#">
<PROMPT CONFIG="">
  
```

Sample debug log where activity is not running

```

<NEWSESSION CatTools 3.10.0 1/29/2014 3:52:23 AM>
<PROTOCOL=Telnet>
<DEVICE TYPE=Juniper.Router>
<ACTIVITY TYPE=Device.Backup.Running Config>
<ACTIVITY SCRIPT=C:\Program
Files\CatTools3\Scripts\Client.Device.Backup.Running Config.txt>
<USERS NAME FOR DEVICE=Juniper Router 1>
<C OK 3:52:24 AM>
<R-3:52:25 AM>[13][00][13][10]stp-j2320 (ttyp0)[13][00][13][10][13][00][13]
[10]login:
=====
WFMDRetVal=1 Waiting for: "Username:"
WFMDRetVal=2 Waiting for: "Password"
WFMDRetVal=3 Waiting for: "Password required, but none set"
WFMDBuffer="[13][00][13][10]stp-j2320 (ttyp0)[13][00][13][10][13][00][13]
[10]login: "
=====
  
```

How to use debug log

The debug log allows you to check the exact communication between the device and CatTools.

If you get an output similar to the second log sample above, then CatTools is waiting for a username prompt, but the prompt given by device is `login` from buffer. So, using the variations functions we can change the device scripts and manually enter information for the prompts.

A user can add a [Variations](#) prompt to the return value from the buffer. When the activity is running the next time, the return value is matched and communication is continued.

Troubleshoot - Kiwi CatTools file name conventions

'Number of device specific dated files to retain' option is not working

You have set the `Number of device specific dated files to retain` option, but the activity is deleting all files.

Cause

This issue is likely due to the file naming convention you have used to replace the default CatTools file naming convention. CatTools default file naming convention uses, for example:

```
Config.Current.Running.Cisco_Router_2.txt
```

If you are using a convention which does not place a period at the end of the device name, such as `Cisco_Router_2` in the above example, CatTools cannot group the files relating to the device correctly. For example, if you are using a file naming convention such as:

```
Cisco_Router_2-backup.txt
```

CatTools cannot group the `Cisco_Router_2` files together correctly, so all files are therefore deleted.

Resolution

Following the device naming conventions for Kiwi CatTools. In the above example, is to change the dash '-' in your own naming convention to a period.

Specifically, `Cisco_Router_2-backup.txt` must be renamed to `Cisco_Router_2.backup.txt`


Troubleshoot: Device specific settings

There are a number of devices that require specific settings or configuration setups within CatTools in order for CatTools to handle their behavior.

Please see the [Device specific information](#) section for a list and of these devices.

Troubleshoot - Error: 70: Permission denied

Overview

 The following error message is found in older Windows versions and may not be applicable to newer versions.

You receive the error message in the InfoLog:

```
Problem starting CatTools_Client instance #1. Error:70: Permission denied
```


Cause

The user running CatTools does not have permission to launch the CatTools_Client DCOM component. The EventLog would show this as a DCOM error with Event ID: 10016.

Resolution

To troubleshoot this error:

1. Navigate to Start > Run menu, and run the program `dcomcnfg.exe`
2. Navigate to Component Services > Computers > My Computer > DCOM Config.
3. Select `CatTools_Client.clsMain`.

 Depending on your version of Kiwi CatTools, this item may also be called `CatTools_Client.KiwiSNMP`.

4. Right-click the file, and choose Properties.
5. Click the Security tab.
6. Under the Launch and Activation permissions, click Customize.
7. Click Edit.

- Adjust the permissions of the user running CatTools to allow local launch and local activation.

i Note: This service runs under the LocalSystem account.

- After making the permissions changes, reboot the system and verify that CatTools is able to start the client threads correctly.

Troubleshoot - Reporting problems

When reporting a problem with CatTools to the SolarWinds technical support team, ensure you provide as many details as possible. Doing so allows our support team to better review and troubleshoot your inquiry.

Best practices

- Provide the complete wording of the error you are receiving. The error message should be present in `infolog.txt` that describes the error.
- Provide the type of device with the problem. CatTools supports a number of different devices that can exhibit very different problem behaviors.
- Include the approximate time the problem occurred.
- Do not send raw screen shots of messages or setup issues, unless there is no other way of getting the information across. Mail servers often block this data. Use the CatTools facilities from the File menu to provide textual information.
- Send the complete `infolog.txt` file as a zipped attachment. Trying to copy and paste entries from the `infolog` into an email can produce poorly formatted emails.

Troubleshoot: Remote Desktop Systems

CatTools may experience problems starting the Client threads when running on machines that use remote desktop software.

i KiwiCatTools has been successfully tested on RealVNC 4.3.2, TightVNC 1.3.9, and PCAnywhere 12.1. With the exception of Windows Vista, no issues occurred during the testing of CatTools.

RealVNC and TightVNC

Some older versions of VNC can cause issues with starting the client threads. Versions of VNC software older than TightVNC 1.3.9 and RealVNC 4.3.2 interrupt the Windows system and cause the client to only start one thread.

Resolution

1. SolarWinds recommends upgrading your current version of VNC.
2. If you are unable to upgrade your VNC software, you can attempt to change the registry settings. Use RegEdit to set the `use_Deferral` setting to 0.



SolarWinds strongly recommends that you back up your registry before making any edits to your system registry. You should only edit the registry if you are experienced and confident in doing so. Using a registry editor incorrectly can cause serious issues with your operating system, which could require you to reinstall your operating system to correct them. SolarWinds cannot guarantee resolutions to any damage resulting from making registry edits.

```
Key: HKEY_CURRENT_USER\Software\ORL\VNCHooks\Application_Prefs\CatTools_Client.exe
Set Dword: use_Deferral to 0
Key: HKEY_CURRENT_USER\Software\ORL\VNCHooks\Application_Prefs\CatTools.exe
Set Dword: use_Deferral to 0
```

Troubleshoot: Remote Authentication

Some difficulties have been experienced when running Kiwi CatTools on a network where remote authentication is used. The symptoms are:

- Random failures of CatTools to log on to devices during the running of a scheduled activity.
- When the activity runs again, CatTools logs on to the same devices with no problems.

The problem can become more apparent on heavy loaded networks, or where the remote authentication system is under load.

You may be able to set the device to wait longer for a remote authentication server response before timing out. For instance, with some Cisco devices you can issue the following command to set the timeout period to 20 seconds:

```
set tacacs timeout 20
```



The default setting for timeout is 5 seconds.

Troubleshoot: Anti-virus Software

Some anti-virus tools can provide barriers to accessing the network, either incoming or outgoing.

If you experience CatTools failing to connect to any devices or unable to connect to the mail server to send alerts, check that you do not have an anti-virus program blocking the CatTools software.

Some anti-virus programs are set up to include program and port blocking. This can prevent CatTools from connecting to an email server for sending reports.

Troubleshoot: Windows XP Firewall

If you are running CatTools on an XP system you need to be aware that after Service Pack 2 the XP Firewall is turned on by default.

You may need to configure an exception for CatTools or make another adjustment if you continue to run with the XP Firewall on.

Troubleshoot: The service is running but nothing is scheduled

There may be occasions when Kiwi CatTools appears to be running and the scheduler is on, but no activities are running.

- A look at the `infolog.txt` file may show that an activity appears to be hung up and has not terminated. This may be caused by a logic loop or communications hang up where CatTools cannot detect a timeout.
 - There is a setting on the [Misc](#) tab in the [CatTools Setup](#) that might be useful in allowing CatTools to continue running productively after such an occurrence. This allows you to forcibly [terminate](#) an orphaned activity after waiting for a set number of minutes. CatTools attempts to recover and resume scheduling activities afterward. Leave this setting at 0, unless there is a problem, so you do not forcibly terminate an activity that takes a long time to run. This activity may still successfully run to conclusion.
- Another cause of this problem may be due to a [permissions issue on the DCOM component](#). To verify this as a cause, check your Info Log for the following message:

```
Problem starting CatTools_Client instance #1. Error: 70: Permission denied
```