# Known Issues

This page contains a list of workarounds for known issues in TeamCity.

## Agent running as Windows Service Limitations

When a TeamCity build agent is installed as a Windows service, there may appear various "Permission denied" or "Access denied" errors during the build process, see details below.

### Security-related issues

The user account used by the service is required to have sufficient permissions to perform the build and manage the service. If you run the TeamCity agent service under the SYSTEM account, do the following:

1. Change SYSTEM for a usual user account with necessary permissions granted.
2. Restart the service.

### Windows service limitations

As a Windows service, the TeamCity agent and the build processes are not able to access network shares and mapped drives.

To overcome these restrictions, run TeamCity agent via console.

#### Issues with automated GUI and browser testing

These problems include errors running tests headless, issues with the interaction of the TeamCity agent with the Windows desktop, etc.

To resolve/ avoid these:

1. Run TeamCity agent via console.
2. Configure the build agent machine not to launch a screensaver locking the desktop.

> Note that there is a Windows limitation to accessing a remote computer via mstsc: the desktop of the remote machine will be locked on RDP disconnect, which will cause issues running tests. The VNC protocol allows you to remote control another machine without locking it.
> To run GUI tests and be able to use RDP, see the workaround below.

3. Configure the TeamCity agent to start automatically (e.g. configure an automatic user logon on Windows start and then configure the TeamCity agent start (via agent.bat start) on the user logon).
   For graphical tests the build agent cannot be started as a service and it is recommended to configure the build agent launch with a 1 minute delay after the user auto-logon, e.g. using the "`bin\agent.bat start`" command in the task scheduler and configuring the delay there.

### Running automated GUI tests and using RDP

RDP uses its own video driver overriding the one from the machine's video card for the session. Redirecting the session to console will unload the Windows graphical drivers. This can be done by adding the following step to your build configuration prior to your tests (the example below is for PowerShell, but other languages (DOS, Python) can be used too):

```
$sessionInfo=((quser $env:USERNAME | select -Skip 1) -split '\s+')
if ($sessionInfo[1] -like "rdp-tcp*") { tscon $sessionInfo[2] /dest:console }
```

where "`quser [current username]`" lists all the connections to that machine for the user, either console or graphical. The one listed as `rdp-tcp#*` is the remote desktop connection which can be redirected to the console using "`tscon [connection id] /dest:console`".

> An unsupervised computer with a running desktop permanently logged into a user session might be considered a network security threat, as access to it can be difficult to trace. Therefore, it is recommended to run automated GUI and browser tests on a virtual machine isolated from sensitive corporate network resources, e.g. on a machine not included in a Windows domain.

### Issues with . Net Selenium

When a TeamCity agent is started as a Windows service and automated tests for .Net applications use Selenium WebDriver, the tests may fail due to browser drivers limitations. As a solution, consider starting the agent manually.

## Early start of the service before other resources are initialized

To handle this, consider using the Automatic (Delayed Start) option of the service settings or configure service dependencies.

For more investigation steps, see the Common Problems page.

Back to top

# java.lang.OutOfMemoryError: unable to create new native thread error

If your TeamCity server is running on SUSE® Linux Enterprise (or using systemd Daemon), the java.lang.OutOfMemoryError: unable to create new native thread error may be caused by the cgroup process number controller feature limiting the number of processes and the amount of threads in a cgroup to 512 by default.

Increasing the limit (e.g. to 4096) on the TeamCity server should solve the issue.

See also this external posting.

Back to top

# Clearing Browser Caches

There is a web UI-related issue which some our users have encountered (and it cannot be reproduced on other computers) which is tied to the cached versions of content. If you have come across such problem, make sure your browser does not use cached versions of content by clearing browser caches.

Back to top

## Logging with Log4j in Your Tests

If you use Log4j logging in your tests, in some cases you may miss the Log4j output from your logs. In such cases please do the following:

- Use Log4j 1.2.12
- For Log4j 1.2.13+, add the "`Follow=true`" parameter for the console appender used in Log4j configuration:

```
<appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
  <param name="Follow" value="true"/>
</appender>
```

Back to top

## Agent Service Can Exit on User Logout under Windows x64

The used version of Java Service Wrapper does not fully support Windows 64 and this causes agent launcher process to be killed on user logout. The agent itself will be function until the next restart (server upgrade or agent properties change).

Back to top

## Failed Build Can be Reported as a Successful One With Maven 2.0.7

This is a known bug in this version of Maven. Consider using any later version.
In case it's not possible you can patch mvn.bat yourself by replacing the fragment at line 148 of `mvn.bat`:

```
:error
set ERROR_CODE=1
```

with the following one:

```
:error
if "%OS%"=="Windows_NT" @endlocal
set ERROR_CODE=1
```

Back to top

## Conflicting Software

Most common indicators of conflicting software are errors like "Access is denied", "Permission denied" or java.io.FileNotFoundException mentioning the file that is present and is writable by the user the agent/build runs under.
Also, certain software running in background (like antiviruses) can significantly slow down build agent operations like sources checkout, artifact publishing or even build running.

Certain antivirus software like Kaspersky Internet Security can result in Java process crashes or other misbehavior like inability to access files. e.g. see the issue.
ESET antivirus can also slow down Ant/IntelliJ IDEA project builds a great deal (slowing down TCP connections to localhost on agent).

If you run antivirus on the TeamCity server or agent machines and get disk access errors or experience degraded performance, please disable or better completely uninstall the antivirus software before investigating the issue and reporting the issue to

JetBrains.

It is recommended to exclude entire TeamCity server home and TeamCity Data Directory from the background checks and perform periodical checks there in the well-known maintenance window so that those do not affect server performance much. On TeamCity agent, it is recommended to exclude TeamCity agent home from the background checks.

Please disable various indexing services. e.g. there might be problems with Windows Indexing Service. See issue for more details. Windows System Restore Feature might also need disabling.

Please also do not install software with background indexing like WinCVS, TortoiseCVS, TortoiseSVN and other Tortoise* products. This applies to server and also to agents if you use agent-side checkout.

Skype software is known to:

- use port 80 on the system so you might not be able to use TeamCity server using default 80 port.
- corrupt layout of pages displayed in Internet Explorer. Internet Explorer Skype plugin is to blame. (TW-13052).

Back to top

## Subversion issues

### svn: E175002: Received fatal alert: bad_record_mac

Please add system property -Dsvnkit.http.sslProtocols=SSLv3,TLS on the build server (see Configuring TeamCity Server Startup Properties).
If you use checkout on agent, add this property on build agent as well.

### Subversion-related JVM Crashes

If JVM crashes while executing SVN-related code (e.g. under org.tmatesoft.svn package), you can try to disable it by either:

- Passing `-Dsvnkit.useJNA=false` JVM option to the crashing process (server or agent), or
- Making NTLM support less prioritative by passing `-Dsvnkit.http.methods=Basic,Digest,NTLM` JVM option.

Anyway, upgrading the JVM used to the latest available version is recommended.

Back to top

## NUnit 2.4.6 Performance

Due to an issue in NUnit 2.4.6, its performance may be slower than NUnit 2.4.1. For additional information, please refer to the corresponding issue in our issue tracker: TW-4709

Back to top

## StarTeam Performance

Using StarTeam SDK 9.0 instead of StarTeam SDK 9.3 on the TeamCity server can significantly improve VCS performance when there is a slow connection between TeamCity and StarTeam servers.

Back to top

## Perforce 2009.2 Performance on Windows

If you run Perforce 2009.2 on Windows you may experience significant slow down. This is an issue with P4 server running on Windows. Please refer to corresponding section in Perforce documentation.

## Wrong times for build scheduled triggering (Timezone issues)

Please make sure you use the latest update for Java 8 installation available for your platform (e.g. OpenJDK 8 by AdoptOpenJD K).

Back to top

## Upgrading IntelliJ IDEA May Affect Active Pre-Tested Commits

Before you upgrade to IntelliJ IDEA X (or other IntelliJ X platform products) please make sure you do not have active pre-tested commits, otherwise they will not be able to be committed after upgrade.
This is only relevant if you use directory-based IDEA project (project files are stored under `.idea` directory).

## Other Java Applications Running on the Same Server

If other web applications are available via the same hostname, a session cookie conflict can occur. This usually is visible via random user logouts or losing session-level data. (e.g. TW-12654). To resolve this, you can use different host names when accessing the applications.

Back to top

## The Server Does Not Start Claiming the Database is in Use

Only a single TeamCity server can work with one database, which is checked on the TeamCity server start.
"The Database is in Use" error on the start-up is reported in either of the following cases:

- An attempt to start more than one TeamCity server connected to the same database
- A second TeamCity instance detected
- The internal HSQL database is being used by another application

The error is most probably caused by the fact that there is another running TeamCity installation which is connected to the same database. Check that the database properties are correct and there is no other TeamCity server using the same database.

In TeamCity 8.0 and earlier, if all the settings are correct, the error can occur when the TeamCity server or the database server has been shut down incorrectly.
The resolution depends on the database type:

- MySQL: restart the MySQL server and then start TeamCity again.
- PostgreSQL, Oracle, MS SQL: kill the connections from the incorrectly shut down TeamCity, and then start TeamCity again.
- Internal database (HSQL): remove the `buildserver.lck` file from the `TeamCity Data Directory\system` directory, and then start TeamCity again.

Back to top

## Slow download from TeamCity server

If you experience slow speed when downloading artifacts from TeamCity, try checking the speed on the server machine, downloading from localhost.
If the speed is OK for the localhost, the issue can be in the network configuration or OS/hardware settings when combined with TeamCity(Tomcat) settings.

Please also make sure <TeamCity Home>\conf\server.xml file corresponds to the file included in TeamCity distribution (can be checked in .tar.gz distribution).
If you have the following "Connector" node (ports numbers can be different):

```
<Connector port="8111" protocol="HTTP/1.1"
           connectionTimeout="60000"
           redirectPort="8543"
           useBodyEncodingForURI="true"
  />
```

change it to:

```
<Connector port="8111" protocol="org.apache.coyote.http11.Http11NioProtocol"
           connectionTimeout="60000"
           redirectPort="8543"
           useBodyEncodingForURI="true"
           socket.txBufSize="64000"
           socket.rxBufSize="64000"
           tcpNoDelay="1"
                /> 
```

and restart TeamCity server.

If this does not help with the download speed, to investigate the case you might need to find an administrator with appropriate network-related issues investigation skills to look into the case.

## Failure to publish artifacts to server behind IIS reverse proxy

This problem is only relevant to configurations that involve IIS reverse proxy between build server and agents.
Sometimes a build agent can be found in infinite loop trying to publish build artifacts to server. Build log looks like this:

```
[11:15:05]Publishing artifacts
[11:15:05][Publishing artifacts] Collecting files to publish: [toZip/** => artifact.zip]
[11:15:06][Publishing artifacts] Creating archive artifact.zip (9s)
[11:15:06][Creating archive artifact.zip] Creating
C:\BuildAgent\temp\buildTmp\ZipPreprocessor2847146024236637664\artifact.zip
[11:15:15][Creating archive artifact.zip] Archive was created, file size 32142324 bytes
[11:15:15][Publishing artifacts] Sending toZip/**
[11:15:25][Publishing artifacts] Sending toZip/**
[11:15:39][Publishing artifacts] Sending toZip/**
[11:15:48][Publishing artifacts] Sending toZip/**
[11:16:01][Publishing artifacts] Sending toZip/**
[11:16:16][Publishing artifacts] Sending toZip/**
```

meanwhile teamcity-agent.log is filled with 404 responses from IIS:

```
[2012-08-01 12:04:55,514]   WARN -    jetbrains.buildServer.AGENT - <!DOCTYPE html PUBLIC
"-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
<title>404 - File or directory not found.</title>
<style type="text/css">
<!--
body{margin:0;font-size:.7em;font-family:Verdana, Arial, Helvetica,
sans-serif;background:#EEEEEE;}
fieldset{padding:0 15px 10px 15px;}
```

The most common cause for this is maxAllowedContentLength setting (in IIS) either

- is set to too small value
- is left unconfigured and so defaults to 30000000 bytes (<30 Mb)

So any artifact larger than maxAllowedContentLength is discarded by IIS
Check the settings value and try to rerun your build

## Prerelease packages are not visible in the TeamCity NuGet feed

Problem. Prerelease packages are not visible in the TeamCity NuGet feed.

Cause. NuGet clients prior to version 3 fail to list prerelease packages if the package version violates the required format.

Solution: Delete build artifacts whose versions violate the required format.

## Packages indexing is slow in TeamCity NuGet feed

Problem. After TeamCity server host machine move or upgrade to the TeamCity 2017.1 build metadata can be reset.

Cause. The TeamCity NuGet feed relies on build metadata, and packages re-indexing can take a lot of time depending on the number of packages and the idle time of the TeamCity server.

Solution. To speed up build metadata re-indexing, specify following internal properties:

```
teamcity.buildIndexer.indexPackSize=1000
teamcity.buildIndexer.packSleepDurationMs=10
```

To check the metadata indexing progress, look for lines similar to the ones below in the teamcity-server.log file:

```
INFO - .index.BuildIndexer (metadata) - Enqueued next 100 builds for indexing, builds left: 7064, last
build id: 8142
```

After re-indexing is complete, remove these internal properties.

## SSL problems when connecting to HTTPS from TeamCity (handshake alert: unrecognized_name)

This problem may happen when changing JVM from 1.6 to 1.7 and connecting some incorrectly configured HTTPS servers. The problem and workaround for it are described in this issue: http://youtrack.jetbrains.com/issue/TW-30210

## SSL problems connecting MS SQL Server and TeamCity

Since MS SQL Server always establishes an SSL connection between the jdbc client (the TeamCity application) and the server, connection problems may occur (SQL exception: Connection reset).

Affected  MS SQL versions: Any version prior to the ones listed below:

- SQL Server 2012 Production version and later
- SQL Server 2008 Service Pack 3 Cumulative Update 4
- SQL Server 2008R2 Service Pack 1 Cumulative Update 6
- SQL Server 2008R2 Service Pack 2

Cause: The problem is caused by the Java 1.6 update addressing this security vulnerability.

Solution: Microsoft SQL Server upgrade is recommended.

Workarounds: If Microsoft SQL Server upgrade is not possible for some reason, TeamCity can be set up to use older Microsoft SQL Server versions with the database connection still SSL- or TLS-encrypted.

Any of the two workarounds listed below will make the connection between TeamCity and the database server vulnerable

- Continue using a block cipher such as `AES_128_CBC` or `3DES_EDE_CBC`, but disable CBC protection via `-Djsse.enableCBCProtection=false` Java command-line option (that can be added to `TEAMCITY_SERVER_OPTS` environment variable, as described here). The `jsse.enableCBCProtection` Java system property is also available in all OpenJDK

8 versions and IBM J9 8.0.0 SR1 and later. Secure connection between TeamCity and Microsoft SQL Server would be stable but still vulnerable to CVE-2011-3389 also known as BEAST.
- Fall back to a stream cipher (which is not susceptible to BEAST) such as `RC4_128`. This will render the connection vulnerable to CVE-2015-2808.

Please try running with antivirus software uninstalled before reporting the issue to JetBrains. e.g. see the issue.

## Windows Docker Containers

### Windows Docker containers

- Since Windows 10 version 1803 with KB4340917 it's possible to use port mapping from containers to localhost. For previous Windows versions it works for the non-localhost IP address associated with this machine and you can access a running application via the machine's hostname or determine the IP address via the `ipconfig` command. Note that the `netstat -an` command may not show that the port is open on any IP address, while in fact it can work. This is also a known problem of Docker on Windows.
- On Windows 10, the memory allocated per container is 1GB by default. To increase this value, use the following memory options:

```
docker run ... -m 2GB -e TEAMCITY_SERVER_MEM_OPTS="-Xmx2g
-XX:ReservedCodeCacheSize=350m"
```

- On Windows 10 containers work via Hyper-V and may experience problems with network and other subsystems. To diagnose these problems, execute the following PowerShell script:

```
Invoke-WebRequest https://aka.ms/Debug-ContainerHost.ps1 -UseBasicParsing |
Invoke-Expression
```

- When starting a TeamCity server from a Windows Docker image, make sure to grant `Authenticated Users` Full control over the directories used as volumes. See the related issue.
- When starting a Windows Docker container with the directory C:/BuildAgent/work mapped as a volume to the container host, Git for Windows fails with a following error: "`Invalid path '/ContainerMappedDirectories': No such file or directory`". The workaround is not to add "C:/BuildAgent/work" as a volume.

To analyze the script output, please refer to the following documents. If it shows that there are problems with the container network subsystem, try resetting it using the cleanup scripts.

More details on troubleshooting Docker for Windows are available in the Docker and Microsoft documentation.

### Information about installed Docker server OS on Windows missing on Agent

On Windows 10, the Docker server depends on Hyper-V service and its start may take a significant amount of time. To resolve the issue, configure the TCBuildAgent Windows service to depend on the Docker for Windows Service, "com.docker.service" by default.

### Linux Docker Containers under Windows

Since TeamCity 2017.2, the Docker Wrapper works on Windows when Windows-based containers are started.

If a Linux container is started on a Windows machine, TeamCity displays the error message "Starting Linux Docker containers under Windows is not supported. To avoid this problem, add the ‘teamcity.agent.jvm.os.name does not contain Windows’ agent requirement.

If you need to support a use case when the Docker wrapper runs Linux containers under Windows platform, please vote for /comment on TW-51820.

### Problems with local time in Windows containers

When using Windows Docker containers, there is an issue with incorrect time in Windows containers when system time in container goes out of sync with the time on the host machine. It could cause problems in integrations where response time is significant (e.g. OAuth tokens).

To address it, upgrade your host machine to Windows Server 2019 / Windows 10 1809 and use TeamCity docker images comp atible with Windows containers 1809.

## "Access is denied" or "Access to the path is denied" problem on container start

When docker is starting Windows containers with process isolation, it's using the SERVICE user account which lacks the write access to directory with docker volumes. To resolve it, grant the "Full control" permission to the SERVICE user for the "%PROGRAMDATA%\docker\volumes" directory.