Controlled Failure

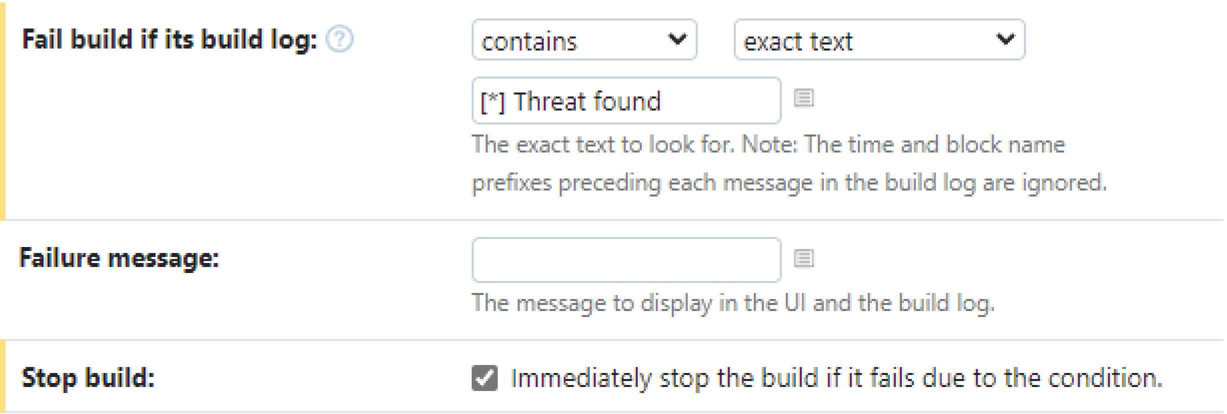
We could also integrate AV scanning into the build steps and "fail" the pipeline if the artifact gets detected. By default, a build will fail if something returns an exit code other than 0. However, most tools don't do that - they print an error to the console and return normally.

Luckily, TeamCity saves all stdout to the build log.

Within the build configuration, go to **Failure Conditions** in the left-hand menu, and under **Additional Failure Conditions** click **Add failure condition**. From the dropdown menu select **Fail build on specific text in build log**. Here, we can specify exact text to look for, or use regex.

For ThreatCheck specifically, we can look for the string `[*] Threat found`.

Also tick **Immediately stop the build if it fails due to the condition**.



Fail build if its build log: contains ▾ exact text ▾

[*] Threat found

The exact text to look for. Note: The time and block name prefixes preceding each message in the build log are ignored.

Failure message:

The message to display in the UI and the build log.

Stop build: ☑ Immediately stop the build if it fails due to the condition.

Add another build step to perform the scan (using the PowerShell or Command Line runner). Click **show advanced options** at the bottom and set the working directory to `%teamcity.build.checkoutDir%\%system.teamcity.projectName%\bin\Release`.

The actual steps to execute will then be:

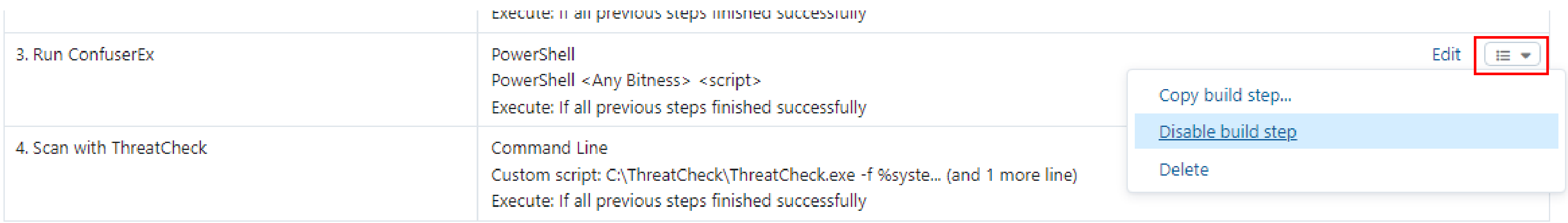
```
C:\ThreatCheck\ThreatCheck.exe -f %system.teamcity.projectName%.exe -e Defender
C:\ThreatCheck\ThreatCheck.exe -f %system.teamcity.projectName%.exe -e AMSI
```

ⓘ Remember that you will need to download a copy of ThreatCheck to the TeamCity VM.

Run the build and everything should pass. You can also inspect the build log to see ThreatCheck's output.

```
11:37:24   Step 4/4: Scan with ThreatCheck (Command Line)
11:37:24   Starting: C:\TeamCity\buildAgent\temp\agentTmp\custom_script8731328937737091980.cmd
11:37:24   in directory: C:\TeamCity\buildAgent\work\9ea9d73f7fc269a0\Rubeus\bin\Release
11:37:27   [+] No threat found!
11:37:27   [*] Run time: 2.22s
11:37:29   [+] No threat found!
11:37:29   [*] Run time: 1.86s
11:37:29   Process exited with code 0
```

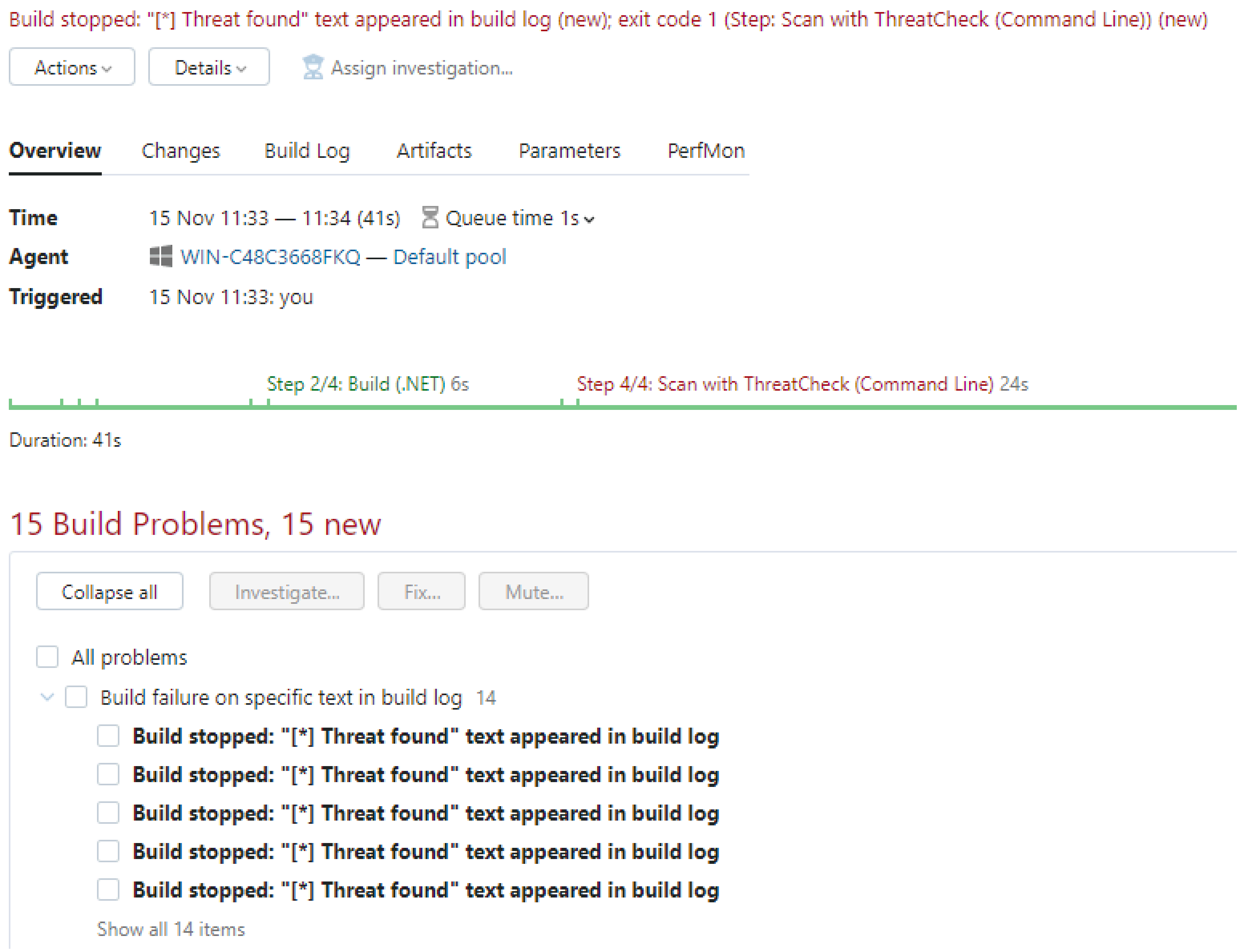
As a validation exercise, disable the ConfuserEx build step and run the build again.



Execute: If all previous steps finished successfully

3. Run ConfuserEx	PowerShell PowerShell <Any Bitness> <script> Execute: If all previous steps finished successfully
4. Scan with ThreatCheck	Command Line Custom script: C:\ThreatCheck\ThreatCheck.exe -f %syste... (and 1 more line) Execute: If all previous steps finished successfully

This time, the build should fail.



Build stopped: "[*] Threat found" text appeared in build log (new); exit code 1 (Step: Scan with ThreatCheck (Command Line)) (new)

Time 15 Nov 11:33 — 11:34 (41s) Queue time 1s

Agent WIN-C48C3668FKQ — Default pool

Triggered 15 Nov 11:33: you

Step 2/4: Build (.NET) 6s

Step 4/4: Scan with ThreatCheck (Command Line) 24s

Duration: 41s

15 Build Problems, 15 new

□ All problems

□ Build failure on specific text in build log 14

□ **Build stopped: "[*] Threat found" text appeared in build log**

□ **Build stopped: "[*] Threat found" text appeared in build log**

□ **Build stopped: "[*] Threat found" text appeared in build log**

□ **Build stopped: "[*] Threat found" text appeared in build log**

□ **Build stopped: "[*] Threat found" text appeared in build log**

Show all 14 items

This can also be expanded upon to suite your needs. For instance, we could build the artifact, calculate its checksum and then check VirusTotal for known matches. If any positive results come back, then fail the build.