code intelligence

Published 19th May 2021

CI Fuzz Hardware and Software Technical Prerequisites

1. Introduction

This document details the typical and required prerequisites for running CI Fuzz as part of a secure testing and development process.

CI Fuzz is modular and flexible and can be deployed in a number of different topologies and environments to best suit the software development process and lifecycle. This document details the most common deployment approaches.

3. Deployment Scenarios

- **Scenario 1 – All-in-one Developer Machine**

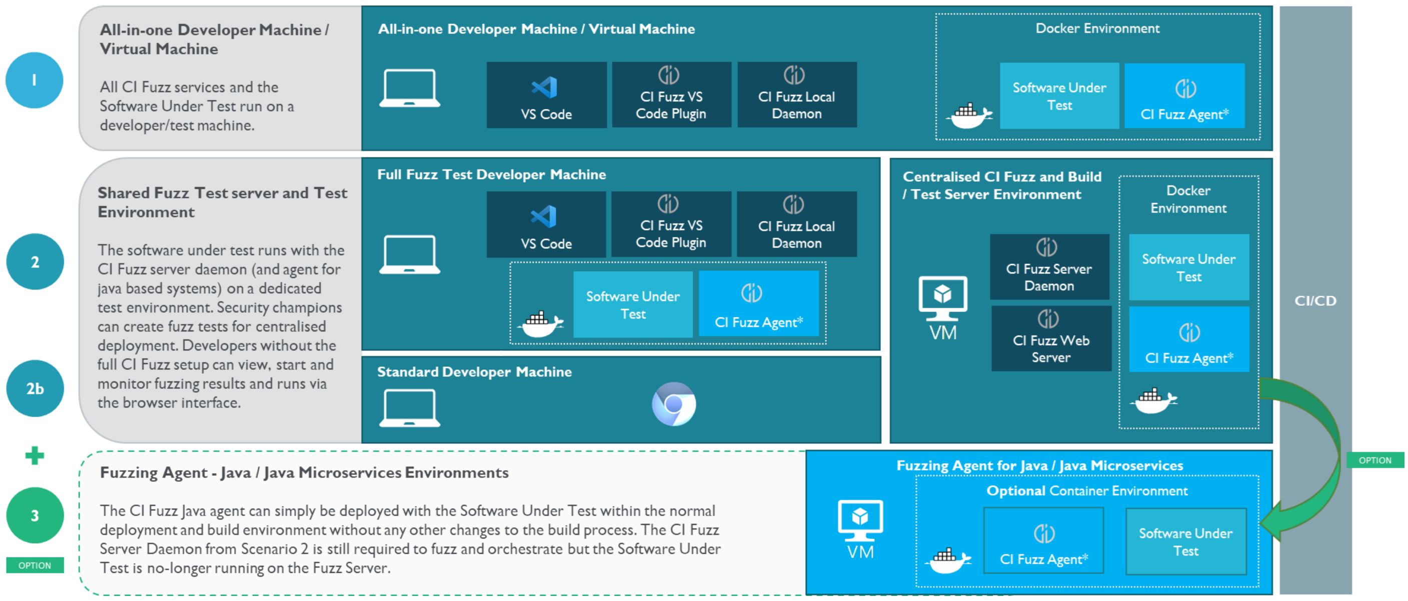
  All CI Fuzz services and the Software Under Test run on a developer/test machine.

- **Scenario 2 and 2b – Shared/Centralized Fuzz Test Server**

  The software under test runs with the CI Fuzz server daemon (and agent for java-based systems) on a centralized/dedicated test environment. A Full Fuzz Test Developer Machine is used to create fuzz tests and use full debugging, or a Standard Developer Machine is used to view and start tests via the web client.

- **Scenario 3 (Optional) - Fuzzing Agent for Java**

  For java-based Software Under Test it is possible to externalize the java agent from the shared / centralised test server to be deployed with the java application in-situ. This allows for very simple fuzz testing deployment as the original build process is not affected by the fuzz testing and also allows fuzzing of microservice architectures.


1
All-in-one Developer Machine / Virtual Machine
All-in-one Developer Machine / Virtual Machine
VS Code
CI Fuzz VS Code Plugin
CI Fuzz Local Daemon
Docker Environment
Software Under Test
CI Fuzz Agent*
2
2b
Shared Fuzz Test server and Test Environment
Full Fuzz Test Developer Machine
VS Code
CI Fuzz VS Code Plugin
CI Fuzz Local Daemon
Software Under Test
CI Fuzz Agent*
Standard Developer Machine
Centralised CI Fuzz and Build / Test Server Environment
VM
CI Fuzz Server Daemon
CI Fuzz Web Server
Docker Environment
Software Under Test
CI Fuzz Agent*
CI/CD
OPTION
3
OPTION
Fuzzing Agent - Java / Java Microservices Environments
Fuzzing Agent for Java / Java Microservices
Optional Container Environment
VM
CI Fuzz Agent*
Software Under Test

All CI Fuzz services and the Software Under Test run on a developer/test machine.

The software under test runs with the CI Fuzz server daemon (and agent for java based systems) on a dedicated test environment. Security champions can create fuzz tests for centralised deployment. Developers without the full CI Fuzz setup can view, start and monitor fuzzing results and runs via the browser interface.

The CI Fuzz Java agent can simply be deployed with the Software Under Test within the normal deployment and build environment without any other changes to the build process. The CI Fuzz Server Daemon from Scenario 2 is still required to fuzz and orchestrate but the Software Under Test is no-longer running on the Fuzz Server.

2. Hardware / Virtual Machine Requirements

As a general guidance the hardware needs to be capable to performantly run and build the Software Under Test. CI Fuzz adds an additional resource level on-top of the software under test and developer tools in particular while running fuzzing tests (these typically generate thousands of requests per second to the software under test)

	All-in-One Developer Machine	**Full Fuzz Test Developer Machine**	**Web Client**	**Shared / Centralized Fuzz Test Server**	**Java Agent**
Scenario	1	2	2b	2 and 2b	2 & 2b Option
Role	Client+Local Server	Client+Local Server	Thin Client	Server	Agent
CPU Cores 64-bit architecture (x86_64 / AMD64)	8	8	2	16 or more	As per Software Under Test requirements
RAM (GB)	16	16	8	32	As per Software Under Test requirements
HDD (GB)	50 minimum 100 Recommended	50 minimum 100 Recommended	5	50 minimum 100 Recommended	As per Software Under Test requirements
Baseline	● Software Under Test runs performantly. ● Long-running fuzzing not running in parallel to development work.	● Software Under Test runs performantly. ● Developer team between 1 and 20 developers.	CI Fuzz provinces a web application interface to allow developers to view, start and monitor fuzzing results and runs. As all computational work is done in the centralized CI Fuzz and Build/Test server environment the machine only needs to be capable to run the CI Fuzz web application in a browser	● Software Under Test runs performantly. ● Developer team between 1 and 20 developers. ● Long-running fuzzing typically with nightly builds. 5-minute fuzzing runs on every push.	The CI Fuzz Java agent can simply be deployed with the Software Under Test within the normal deployment and build environment without any other changes to the build process. The hardware requirements of a normal deployment of the Software Under Test are highly software specific and the requirements are determined by the Software Under Test.
Supported Operating System	- Microsoft Windows 10 + Microsoft WSL2 (Windows Subsystem for Linux 2) v18363.1049+ - Linux (Ubuntu, Debian, Oracle Linux Server)		Various	- Linux (Ubuntu, Debian, Oracle Linux Server)	Various

3. Software Requirements and Versions

3.1. Scenario 1: All-in-one Developer Machine

The following section describes the operating system and other software components supported or required in order to correctly run CI Fuzz.

3.1.1. Supported Operating Systems

Area	Manufacturer	Product Name	Version	Comment
Operating System	Microsoft	Windows 10 + Subsystem for Linux	18363.1049+	
Operating System	Canonical	Ubuntu	20.04, 20.10, 21.04	
Operating System	Debian community	Debian	10 (buster)	
Operating System	Oracle Corporation	Oracle Linux Server 8.2	8.2	

3.1.2. Required Software

Linux support layer for windows	Microsoft	Windows Subsystem for Linux	19042 or higher	
Container engine	Docker, Inc.	Docker	18.09.1+	
Docker image with Software under test dependencies	Build by you			The docker image needs to trust the TLS certificate of your VCS server. If you run your VCS on premise you might need to import it.
Software Development Kit	Oracle Corporation	Java Development Kit (JDK)	Openjdk-8, Openjdk-11	Only needed for fuzzing Java applications
IDE	Microsoft	Visual Studio Code	1.55.2+	Instead of the VSCode extension a command line interface is available. For best usability we recommend using the extension.
VS Code Extension	Microsoft	VS Code Extenion for WSL (Extension Identifier: ms-vscode-remote .remote-wsl)	v0.56.1+	Needed only when using Windows 10 with WSL2
VS Code Extension	Huachao Mao	VS Code Extenion REST client (Extension Identifier: humao.rest-client)	v0.24.5+	Needed only for web application fuzzing
Download Tool	Tim Rühsen, Darshit Shah and Giuseppe Scrivano	wget	1.18+	
Download Tool	curl project	curl	7.52.1+	
Browser	Google	Chrome browser	90.0.4430.93+	One browser is enough
Browser	Mozilla Corporation	Firefox	78.10.1esr	One browser is enough

3.1.3. Required Access and Permissions

Requirement	Purpose	Alternative
Internet Access	Download CI-Fuzz installer from aws S3	If your network access of the machine targeted by the installation is restricted you can download the installer at a different machine and transfer it using a network share or flash drive.
Access to the source code	Build, instrument and run the software under test	Not required for java web application fuzzing, but beneficial
Pull access to the VCS	Pull and fuzz the newest version	Recommended
Root privileges	Install to system folders (/opt)	Installation can be configured in a way that does not require this
Clipboard	When supporting your setup process our engineers might ask you to run commands they send you on the target machine. Being able to use copy and paste saves time.	If the network access of the target machine is restricted a network share with a network-connected machine could be helpful. Retyping commands manually would work but is impractical.

3.2. Scenario 2: Shared/Centralized Fuzz Test Server

3.2.1. Supported Operating Systems

Manufacturer	Product Name	Version	Comment
Canonical	Ubuntu	20.04, 20.10, 21.04	
Debian community	Debian	10 (buster)	
Oracle Corporation	Oracle Linux Server 8.2	8.2	

3.2.2. Required Software

Area	Manufacturer	Product Name	Version	Comment
Software Development Kit	Oracle Corporation	Java Development Kit (JDK)	Openjdk-8, Openjdk-11	Only needed for fuzzing Java applications
Container engine	Docker, Inc.	Docker	18.09.1+	
Docker image with Software under test dependencies	Build by you			The docker image needs to trust the TLS certificate of your VCS server. If you run your VCS on premise you might need to import it.
Download Tool	Tim Rühsen, Darshit Shah and Giuseppe Scrivano	wget	1.18+	
Download Tool	curl project	curl	7.52.1+	
Download tool	Google	Chrome browser	90.0.4430.93+	
Download tool	Mozilla Corporation	Firefox	78.10.1esr	

3.2.3. **Required Access and Permissions**

Requirement	**Purpose**	**Alternative**
Internet Access	Download CI-Fuzz installer from aws S3	If your network access of the machine targeted by the installation is restricted you can download the installer at a different machine and transfer it using a network share or flash drive.
Access to the source code	Build, instrument and run the software under test	Not required for java web application fuzzing, but beneficial
Pull access to the VCS	Pull and fuzz the newest version automatically	
Access to CI-Fuzz port (443) from your VCS or CI/CD platform and users' computers	Your VCS or CI/CD service needs to upload fuzz tests, start fuzzing, download results Users need access to CI Fuzz web interface to view fuzzing results and configure CICD integration	
Root privileges	Install to system folders (/opt) and bind to TLS port 443	Installation can be configured in a way that does not require this
Clipboard	When supporting your setup process our engineers might ask you to run commands they send you on the target machine. Being able to use copy and paste saves time.	If the network access of the target machine is restricted a network share with a network-connected machine could be helpful. Retyping commands manually would work but is impractical.

3.2.4. **Fuzz Test Developer Machine**

The requirements for the Fuzz Test Developer Machine are the same as described in scenario 1.

3.2.5. **Scenario 2b: Web client Access**

Developers without the full CI Fuzz setup can view, start and monitor fuzzing results and runs via the browser interface.

Area	**Manufacturer**	**Product Name**	**Version**	**Comment**
Web Browser	Google	Chrome browser	90.0.4430.93+	
Web Browser	Mozilla Corporation	Firefox	78.10.1esr	
Web Browser	Microsoft	Microsoft Edge	88+	

3.3. **Scenario 3 (Optional) - Fuzzing Agent for Java**

The CI Fuzz Java agent can simply be deployed with the Software Under Test within the normal deployment and build environment without any other changes to the build process. The deployment needs to contain all dependencies of the software under test.

In case when fuzzing input is delivered to the Java application using HTTP requests (recommended), all anti-automation measures like restrictions on the number of HTTP requests that can be sent to the application over a period of time, captcha etc. must be disabled before starting fuzzing.