Bytecode

Bytecode ist eine Zwischensprache, die bei der Ausführung eines Programmes für den jeweiligen Prozessor durch eine Laufzeitumgebung Maschinencode generiert, der vom Prozessor ausgeführt wird.

Ein klarer Nachteil des Bytecodes ist die schlechtere Performance als bei purem Maschinencode. Aber die Laufzeitumgebungen sind recht gut optimiert, so dass der Unterschied der Laufzeitgeschwindigkeit nur gering ausfällt. Außerdem kann das Programm nicht direkt auf die Hardware zugreifen, sondern der Bytecode muss erst Ahead-of-Time, also vor der Ausführung, in Maschinencode übersetzt werden.

Ein klarer Vorteil gegenüber des Maschinencodes ist die gute Portierbarkeit von Programmen, da der Assembly Code nicht für jeden Prozessor einzeln vor der Ausführung übersetzt werden muss.

Dadurch, dass es für jeden Prozessor einen eigenen Assemblydialekt gibt, muss gegebenenfalls der Source-Code angepasst werden. Was bei größeren Projekten sehr viel Aufwand bedeuten würde. Also ist die Entwicklung von Cross-Platform Programmen weniger aufwendig.

Eine Garbage-Collection kümmert sich bei den meisten Laufzeitumgebungen, um nicht mehr benötigten Speicher. Somit muss der Entwickler selbst keine Speicherbereinigung mehr durchführen und Speicherfehler können somit vermieden werden.

Ein weiterer Vorteil sind Laufzeit-Checks, die auf bestimmte Eigenschaften des Codes prüfen, wie z.B. die Indexierung von Arrays(1). Dadurch werden Buffer-Overflow(2) Angriffe umgangen.

(1) Ein Array ist eine aufeinanderfolgende Liste von Elementen, die mit einem Index ansprechbar ist, um ein Element zu lesen oder zu verändern.

(2) Bei einem Buffer-Overflow Angriff wird versucht außerhalbt des Arrays zu schreiben, um somit Schadhaften Code auszuführen.