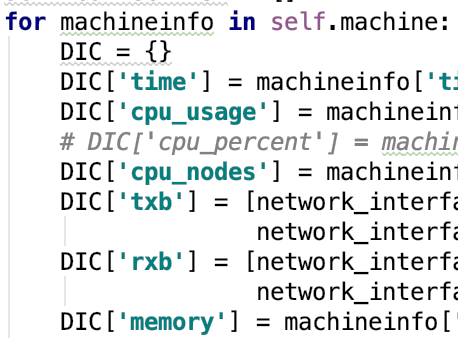
本文对Cadvisor Analysis工具的score计算公式进行详细介绍，其可分为Container与Machine两种score模式，而二者均采用相同计算公式，故借此机会与各位分享一下公示推演过程。

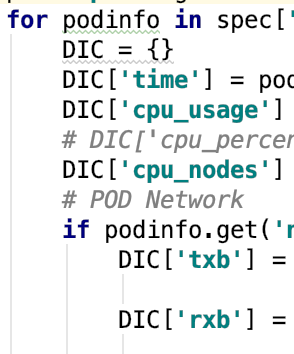
本文需强调说明：

CPU的占用比仅为单位时间下CPU空闲比，因而监控系统（包括windows资源管理器）中的占用比仅代表其在出厂规定的频率下满负荷工作，其并非实际的MIPS占用率，因此物理主机的MIPS无法完全实现规量化。

在Cadvisor Analysis与Cadvisor建立请求连接并采集到API返回的Machine CPU/Memory/txb(发送字节数)/rxb(接收字节数)后，该工具会将JSON结果进行Model抽象并以数组的形式存储至如下变量中：



MachinestatModel

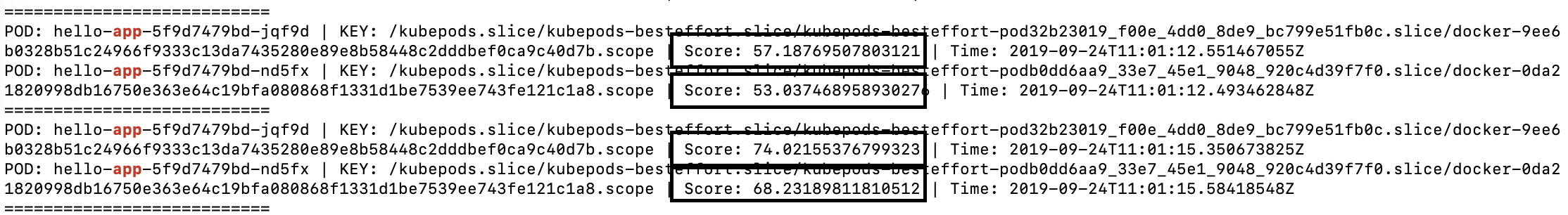


ContainerModel

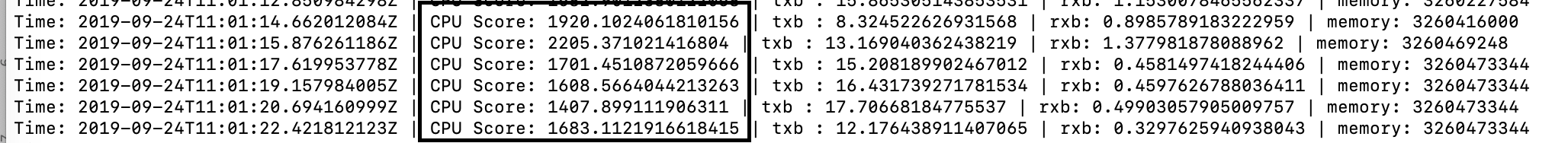
而score计算方法在ContainerScoreCollector.py与MachineScoreCollector.py中均有实现。不妨以ContainerScoreCollector为例，最简单的方法为通过获取两次相邻状态（stats）的cpu\_usage差值与时间差值time并对其做比即可获得该stats组当前时刻下的平均状态，在代码实现中本人采用的是数组中最近两次的stats做为计算锚点，其公式如下：

如需要保证score精度，可采用将逐个相邻状态做差取平均的方式实现，而考虑到Cadvisor本身监控周期与该容器宿主机性能相关，因此需在程序中将其进行过滤后取值，公式如下：

鉴于API返回时间精度为微秒级，而在大部分日常监控下的时间单位均为秒/分钟级别，故在函数处理过程中本人将时间计算单位的精度转换为秒。以上即为Cadvisor Analysis的score计算公式，接下来为实际运行状态截图：



Container运行状态打分



Machine运行状态打分