Externe und applikationsinterne Autorisierung in OAuth2 Systemen

Mit OAuth2 ist eine Sicherung von http-Schnittstellen möglich. Hierbei erhält ein Client von einem Autorisationsserver einen Token, mit dem er Zugriff auf Schnittstellen eines Resourceservers erhält. Ein valider Token ist mit einer Authentifizierung gleichzusetzen. Die Autorisierung, also die Entscheidung, ob der Token die benötige Berechtigungen besitzt, lässt sich grundsätzlich in dem Resourceserver selbst implementieren. Dies kann aber bei einer heterogenen Applikationslandschaft mit verschiedenen Programmiersprachen und sich häufig wechselnden und komplexen Zugriffsrichtlinien schnell zu einem hohen Wartungsaufwand führen. Deswegen ist es sinnvoll die Autorisierung zu entkoppeln und dies ist mit „Open Policy Agent“ (OPA) möglich. Das hat allerdings zur Folge, dass der Resourceserver jedes Mal bei eingehenden http-Anfragen den OPA-Service den u.U. umfangreichen Token zusenden muss, der wiederum dann den Token dekodieren und parsen und eine Zugriffsentscheidung zurücksenden muss, d.h. es besteht das Risiko von Performanceeinbußen. Da bei einer hohen Last auf Schnittstellen diese i.d.R. durch horizontale Skalierung entlastet werden, können Performanceeinbußen äußerst kostspielig sein.

Um den Einfluss auf die Performance durch externe Autorisierung mit OPA im Vergleich zur applikationsinternen Autorisierung zu untersuchen, wurden beide Systeme unter den Kriterien der Round-Trip-Time, Datendurchsatz, RAM-Belegung, CPU-Auslastung sowie Ausfallsicherheit getestet. Dazu wurden zwei Testsysteme implementiert und Performance-und-Lasttests mit dem Tool Apache JMeter durchgeführt und mit JMeter sowie dem Windows Ressourcemonitor die Messwerte protokolliert und ausgewertet.

Dabei ist man zu dem Ergebnis gekommen, dass …