Trabalho Prático 3: Serverless Computing

Cloud Computing - Lorenzo Carneiro Magalhães

Task 1: Serverless Function and Runtime

A primeira tarefa do projeto foi criar um script para um ambiente serverless similar ao AWS Lambda, já configurado com regras específicas. A principal exigência era implementar uma função handler, responsável por receber um dicionário de métricas do sistema e um contexto, processá-los e retornar um dicionário de saída. As métricas chegam ao script via Redis pub/sub, e o código deve calcular a média móvel de 60 segundos para três indicadores: utilização das CPUs, utilização de rede de saída e percentual de memória em cache.

Para manter o estado entre execuções, a fim de calcular a média móvel, o script deve usar *context.env*, um dicionário persistente fornecido pelo ambiente serverless. Após o processamento, as métricas calculadas precisam ser armazenadas em um dicionário de saída, que o ambiente exportará automaticamente para o Redis usando a chave configurada.

Além disso, foram disponibilizados os arquivos de deployment, contendo a configuração necessária para exportação do script e definição da chave Redis via *ConfigMap* no Kubernetes, simplificando o processo de testes e deploy no ambiente serverless.

Implementação

Primeiramente foi computado a média móvel das CPU's através da coleta dos dados atuais vindos do input e do histórico passado iteração por iteração via *context.env*. Caso o histórico contenha mais de 12 pontos (60 segundos dividido por medições a cada 5 segundos), eu removo o último para adicionar o atual, de maneira que a média móvel seja baseada apenas nos últimos 12 pontos.

O cálculo da porcentagem de cache é mais fácil, já que basta coletar os dados do *input* e fazer um cálculo simples: (*virtual_memory-cached + virtual_memory-buffers) / virtual_memory-total*.

O cálculo de network egress também é tranquilo, já que considera apenas dados do *input*. O cálculo feito foi: 100 * *net_io_counters_eth0-bytes_sent / (net_io_counters_eth0-bytes_sent + net_io_counters_eth0-bytes_recv)*

Esses valores eram então retornados pela função *handler*.