High Performance Lambdas with Rust

Write (or rewrite) your critical Lambda functions in Rust to boost performance and reduce cost

What is Rust?

Rust is a modern compiled language that excels in creating high-performance, memory-safe applications. Rust is a great fit for performance critical Serverless use cases with Lambda offering **native C/C++ level performance with safety.**

Why use Rust?

- **Performance:** Rust's design prioritizes efficiency, giving execution times and a memory footprint compatible to native C/C++. Rust performs significantly better than other application language stacks (e.g. Python, Node.js, C# and Java)
- **Safety:** Rust's focus on memory safety mitigates common bugs like null pointer dereferencing and buffer overflows prevalent in C and C++.
- **Correctness:** Rust's type system highlights fallible calls and data absence, aiding developers in identifying and handling edge cases explicitly.
- **Productivity:** With a robust type system, refined compiler, and extensive developer tool ecosystem, Rust empowers teams to achieve more.

Lambda & Rust

Rust is a great choice for (re)implementing performance critical Lambda functions. Given that the lambda pricing model is:

Cost = Allocated Memory x Execution Time

Growth of Rust

The global Rust developer community surged to approximately 2.8 million in 2023. Major tech players like Google, Microsoft and AWS are actively adopting Rust for their critical infrastructure.

fourTheorem were early adopters of Rust and are active in the Rust developer community.

Low-risk adoption

Lambda functions offer an ideal starting point for Rust adoption. Small and self-contained, they often have minimal dependencies and consist of only a few hundred lines of code. Migrating a single Lambda function to Rust is a low-risk investment, allowing you to test the waters without committing to a massive rewrite.

Key Benefits of Rust on Lambda

- **Improved performance -** Rust can deliver a significant boost for performance critical Lambda functions slashing execution time.
- **Lower Cost -** Rust's efficient execution characteristics and low memory footprint can provide a significant cost reduction for highly used Lambda functions.
- **Reduced Cold Start Impact -** Thanks to its performance characteristics, Rust can significantly reduce the cost of cold-starts. Typically, Rust cold starts are in the order of 10-50ms, as opposed to 500-2000ms generally observed in Python, JavaScript and Java.
- **Reduced Carbon Footprint -** By adopting Rust for your performance critical Lambdas, you're not just optimizing for performance - you're actively contributing to more sustainable cloud deployments.

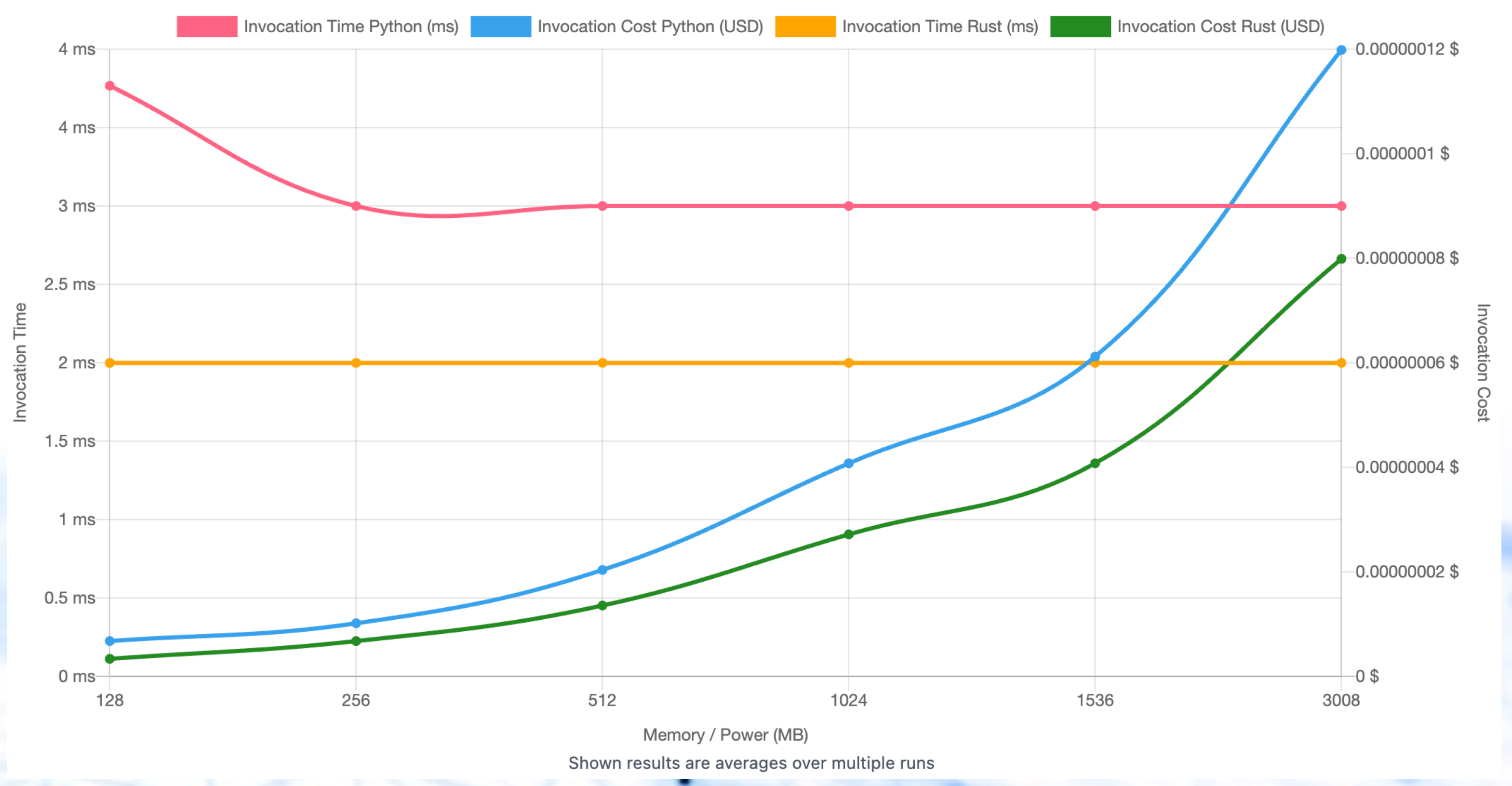
Benchmark Rust vs. Python

🚀 16x faster cold starts

⚡ 3.5x less memory

💰 3x cheaper

Full details: oidc-authorizer-benchmark



aws PARTNER Advanced Tier Services

- Advanced Tier Services
- AWS Lambda Delivery
- AWS ECS Delivery
- Migration Services
- Well Architected

Have a project you'd like to discuss or want to know more? +353 21 206 3101 hello@fourtheorem.com www.fourtheorem.com