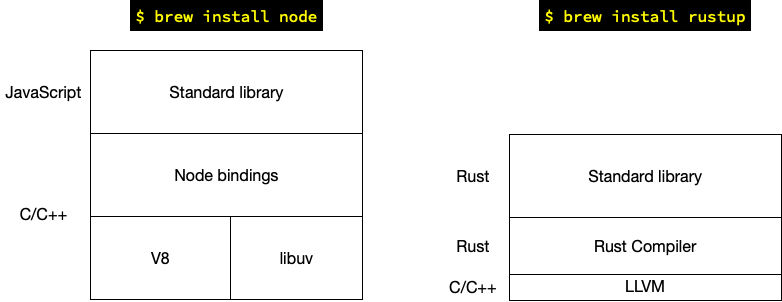
**RUST**

**1. Giới thiệu**

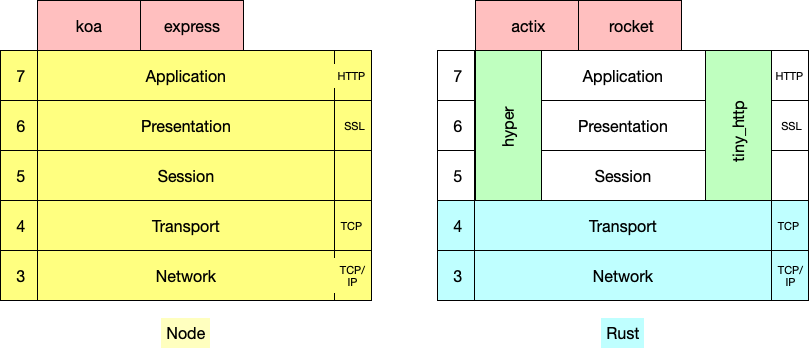
**-** Rust là ngôn ngữ lập trình bậc cao, được quảng bá là có tốc độ cực kì nhanh, ngăn chặn được phần lớn mọi bug crash (segfaults), ngăn chặn được data race, rất memory safe mà lại không cần tới garbage collector.

- Rust có độ khó khi tiếp cận cao hơn so với các ngôn ngữ như Javascript, Python, C, C++

2. Kiến trúc tổng quan



* Node cần có V8 của Google và các thư viện với javascript code
* Rust chỉ phụ thuộc vào thư viện 11vm (viết bằng C/C++)



* Node có hướng tiếp cận ngay với tầng HTTP application trong mô hình OSI 7 tầng và các framework như koa, express cũng vậy
* Với Rust, tầng TCP transport được tiếp cận trong Rust Core, còn các framework sẽ hoạt động trên tầng HTTP

**2. Ecosystem**

* Package manager

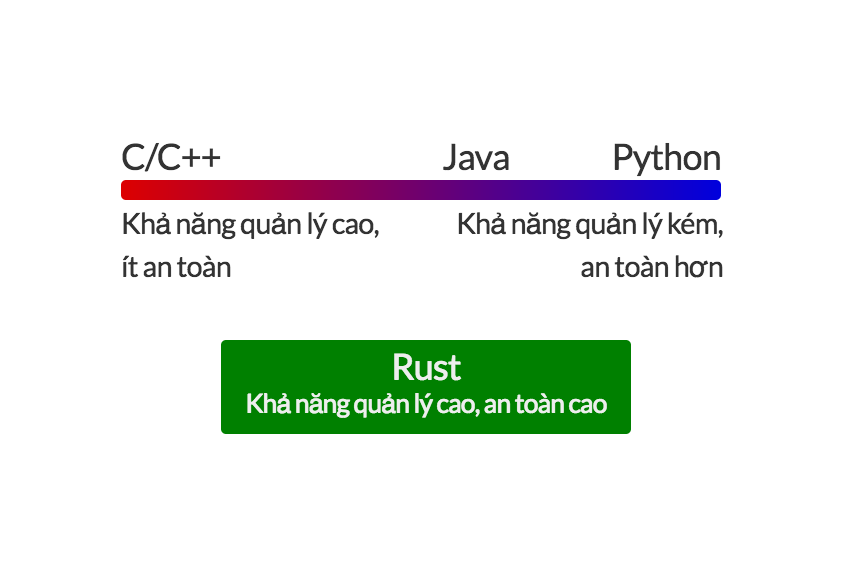
- Node đang sử dụng npm để quản lý các gói

- Với Rust, cargo đảm nhận nhiệm vụ này

Hệ sinh thái của Node là vô cùng hoàn hảo:

* Sự lựa chọn đa dạng với hàng ngàn thư viện
* Khả năng tìm sự trợ giúp khi gặp vấn đề nhanh

Rust vẫn đang trong giai đoạn phát triển và cần sự đóng góp của cộng đồng

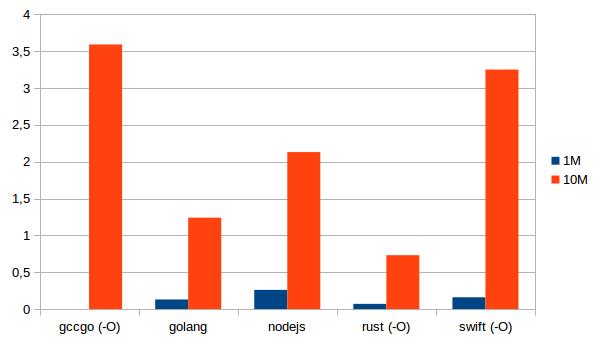
**3. So sánh hiệu năng**

Với C/C++, bạn được:

* Toàn quyền kiểm soát mọi thứ (malloc, free,...)
* Thường xuyên đau đầu với memory leak, data race, segfaults,...

JavaScript hay Ruby:

* Cơ chế cấp phát và giải phóng hoàn toàn tự động
* Nhưng trade-off của nó chính là performance không tốt trong một số trường hợp.
* Rust là sự kết hợp điểm mạnh của cả 2 mặt trên.
* Số liệu về tốc độ của Rust:



*Hình 1.2. Tốc độ khi thực hiện thuật toán Fibonaci*

**5. Rust compiler**

Một số quy chuẩn áp dụng trong rust compiler để ngăng chặn các vấn đề về bộ nhớ:

* Trong Rust, một vùng nhớ luôn có một biến sỡ hữu nó.
* Gán một biến vào một biến khác, tức là bạn chuyển quyền sở hữu vùng nhớ của biến đó cho biến mới. Biến cũ không được quyền truy xuất tới đó nữa.
* Trong một thời điểm, chỉ có một trong hai trường hợp có thể xảy ra, đó là:
* Nhiều biến khác có thể mượn vùng nhớ đó chỉ để đọc (read only).
* Chỉ có một biến có quyền mượn để ghi giá trị lên đó.
* Một biến chỉ có thể tồn tại bên trong scope của nó, ra khỏi scope đó nó sẽ bị giải phóng ngay lập tức.