**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

Icon

Description automatically generated**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO**

**Seminar các vấn đề hiện đại của CNPM**

**Đề tài: PHÁT HIỆN VÀ PHỤC HỒI LỖI TRONG PHÁT TRIỂN PHẦN MỀM**

Giảng viên hướng dẫn: Đinh Nguyễn Anh Dũng

Lớp: SE400.P11

Sinh viên thực hiện:

* Nguyễn Hải Minh - 21522342
* Bùi Gia Huy - 21520909

**TP. Hồ Chí Minh, Ngày 14 Tháng 10 Năm 2024**

# Tổng quan

Hiện nay nay, trong lĩnh vực công nghệ thông tin, đối với những công ty phần mềm, việc tối đa hóa lợi nhuận chỉ với một khoản đầu tư nhỏ luôn là ưu tiên hàng đầu, điện toán phân tán (distributed computing) là một giải pháp hữu hiệu cho vấn đề trên. Điện toán đám mây (cloud computing), điện toán sương mù (fog computing) và Internet vạn vật (IoT) là những ví dụ nổi tiếng về điện toán phân tán, chúng cung cấp các dịch vụ và hiệu suất điện toán tốt. Tuy nhiên, việc cung cấp các dịch vụ đáng tin cậy trong môi trường thực tế, dễ xảy ra lỗi, vẫn là một thách thức. Để giải quyết vấn đề này, các bộ phát hiện lỗi được sử dụng trong các hệ thống phân tán, là các mô-đun trừu tượng chịu trách nhiệm phát hiện và giám sát hoạt động của các nút để xác định xem chúng có lỗi hay không. Ngoài ra, sau khi phát hiện lỗi, việc phục hồi lỗi cũng là một phần quan trọng trong việc giảm thiểu thiệt hại gây ra trong hệ thống.

# Giới thiệu

Về bản chất, các hệ thống phân tán bao gồm các thực thể tự chủ, độc lập, phân tán về mặt địa lý, bao gồm các quy trình, nút và cảm biến, được kết nối với nhau thông qua mạng. Các hệ thống này dễ bị lỗi do sự hiện diện của nhiều nút và kênh giao tiếp. Phát hiện lỗi là một khía cạnh quan trọng của các hệ thống phân tán giúp xác định và cô lập các nút và kênh bị lỗi. Đồng thời sử dụng các biện pháp phục hồi dữ liệu giúp hệ thống được luôn trong trạng thái sẵn sàng, hạn chế sự ngắt quãng do lỗi gây ra.

# Mục tiêu và lợi ích

* Mục tiêu của đề tài là khám phá ra những phương pháp giúp cho các hệ thống nói chung và các hệ thống phân tán nói riêng đảm bảo tính sẵn sàng ở mọi thời điểm, cụ thể:
  + Áp dụng các phương pháp phát hiện lỗi thông dụng trong phát triển phần mềm.
  + Áp dụng các thuật toán, tính toán thống kê và học máy để phát hiện ra các lỗi ngầm hoặc các cuộc tấn công dựa trên dữ liệu đầu vào.

# Ứng dụng của phát hiện và phục hồi lỗi trong ứng dụng phát nhạc trực tuyến.

* Áp dụng các phương pháp phát hiện lỗi cho chương trình như heartbeat, time-based algorhithm, ping/echo.
* Sử dụng giải thuật đồng thuận – consensus algorhitm (Paxos, Raft) để đảm bảo tính đồng nhất của dữ liệu trong các replica của một service.
* Phát hiện lỗi và các cuộc tấn công bằng tính toán thống kê, học máy (Z-score, Mạng neuron cơ bản)

# Công nghệ và phương pháp phát triển

* Sử dụng TypeScript để cài đặt các phương pháp phát hiện lỗi và thông báo cho người quản lý sử dụng webhook của Slack.
* Sử dụng sản phẩm phần mềm cũ và áp dụng công nghệ Docker để thiết kế nó thành một kiến trúc microservice đơn giản:
  + Service quản lý người dùng.
  + Service quản lý playlist.
  + Service phát nhạc.
  + Service gợi ý nhạc.
* Cài đặt các phương pháp phát hiện lỗi đơn giản: heartbeat, time-based algorhitm và ping/echo.
* Sử dụng các thuật toán đồng thuận:
  + Raft: Cài đặt một giải thuật để cho các service (hay còn gọi là các node) bầu ra một leader node, node này sẽ đóng vai trò ghi nhận thay đổi sau đó phân phối chúng cho các node khác, đảm bảo một bản sao thống nhất của cơ sở dữ liệu.
  + Paxos: Thay vì bầu một leader node, các node cùng nhau đồng thuận về một quyết định.
* Sử dụng các giải thuật thông kê và học máy:
  + Z-score: Đầu tiên nhóm sẽ đưa ra một tập dữ liệu mẫu, sau đó tính toán trung bình (mean) và độ lệch chuẩn. Áp dụng công thức tính Z-score và nếu nó không nằm trong khoảng -3 và 3 (mức đột biến bất thường) thì sẽ báo lỗi (báo hiệu cho một cuộc tấn công hoặc lỗi kỹ thuật).
  + Mạng neuron: Sử dụng python, mẫu mạng autoencoder và train trên một tập dữ liệu mẫu bình thường (tập trung vào hành vi người dùng) để phát hiện lỗi và các cuộc tấn công.
* Quá trình cài đặt phục hồi lỗi cho các service (Failover và Load balancing).
  + Áp dụng load balancer để điều hướng request cho một trong 2 instance của một service cụ thể nào đó.
  + Nếu một bên lỗi, báo hiệu cho load balancer để đẩy hết request cho instance hoạt động.
  + Nếu instance lỗi phục hồi, báo hiệu cho load balancer.

# Thách thức và giải pháp

* **Thách thức**:
  + Khó khăn trong việc xác định quy mô của các phương pháp để sắp xếp thời gian hợp lý.
  + Các phương pháp khác được nêu trong bài báo:
    - Byzantine Fault Tolerance (BFT): Cho phép hệ thống hoạt động cho dù có một số node hỏng.
    - Thuật toán hồi quy: dự đoán dựa trên các biến đầu vào và so với giá trị thực tế để đưa cảnh báo phù hợp.
    - Giải thuật phân cụm: Biểu diễn bằng biểu đồ phân tán và phát hiện điểm ngoại lai.
    - Sao chép dữ liệu, dữ liệu dự phòng và dữ liệu dư thừa.
* **Giải pháp**: Trong quá trình cài đặt, tính toán thời gian hợp lý để thay đổi linh hoạt, ưu tiên các giải thuật thú vị với độ khó vừa phải.

# Kế hoạch

|  |  |  |  |
| --- | --- | --- | --- |
| **Giai đoạn** | **Thời gian** | **Công việc** | **Kết quả** |
| **Tìm hiểu lý thuyết** | 2 tuần | - Tìm hiểu về 5 cơ chế và thuật toán phát hiện lỗi.  - Tìm hiểu về 5 chiến lược phục hồi lỗi.  - Tìm hiểu về Javascript. | Có đủ kiến thức về việc phục hồi và phát hiện lỗi. |
| **Khảo sát dự án** | 1 tuần | - Xác định phạm vi dự án.  - Khảo sát trường hợp cần gặp lỗi và cần phục hồi lỗi. | Dự án cụ thể hoá được vấn đề cần làm. |
| **Cài đặt** | 9 tuần | - Cài đặt thuật toán phát hiện lỗi (Heartbeat, Timeout-based Algorithm, Ping/Echo, Raft, Neuron Networks).  -Cài đặt thuật toán phục hồi lỗi (Failover, Replication, Load Balancing, Data Backups, Redundancy) | Có thể phát hiện lỗi và phục hồi hiệu quả cách khác nhau qua các cuộc tấn công giả lập. |
| **Kiểm thử và đánh giá** | 1 tuần | - Kiểm thử và đánh giá độ chính xác của 10 thuật toán trên. | Dự án hoạt động ổn định. |
| **Làm hồ sơ nghiệm thu** | 1 tuần | - Chuẩn bị các tài liệu liên quan.  - Làm hồ sơ nghiệm thu. | Bản báo cáo nghiệm thu. |

# Tài liệu tham khảo

[1] “Failure Detection and Recovery in Distributed Systems”, Geeksforgeeks, 2024. [Online]. [Truy cập lần cuối: 24-9-2024].

[2] “An in-depth and insightful exploration of failure detection in distributed systems.”, Bhavana Chaurasia, A. Verma, P. Verma 2024. [Online]. [Truy cập lần cuối: 05-10-2024].

[3] “Paxos Algorithm in Distributed System”, Geeksforgeeks, 2024. [Online]. [Truy cập lần cuối: 24-9-2024].

[4] “Raft Consensus Algorithm”, Geeksforgeeks, 2024. [Online]. [Truy cập lần cuối: 24-9-2024].

[5] "Z-Score in Statistics | Definition, Formula, Calculation and Uses”, Geeksforgeeks, 2024. [Online]. [Truy cập lần cuối: 24-9-2024].