Distributed Systems

0% completed

Introduction to Distributed Systems

Getting Started

Fallacies of Distributed Computing

Difficulties Designing Distributed Systems

Measures of Correctness in Distributed Systems

System Models

Types of Failures

The Tale of Exactly-Once Semantics

Failure in the World of Distributed Systems

Stateless and Stateful Systems

Quiz
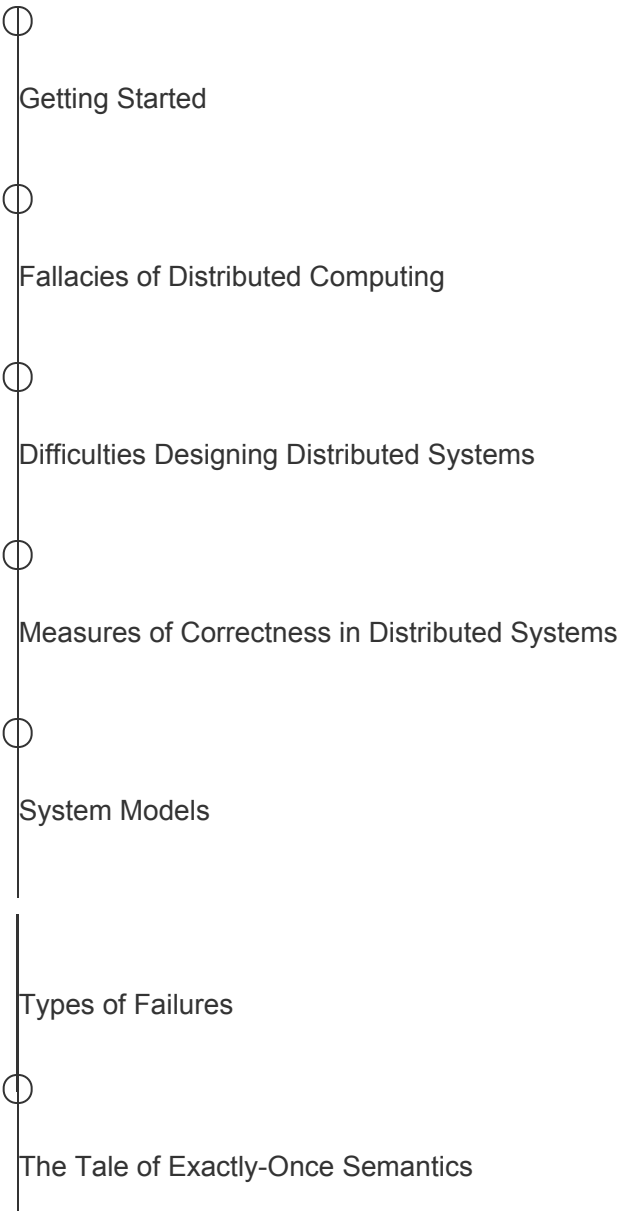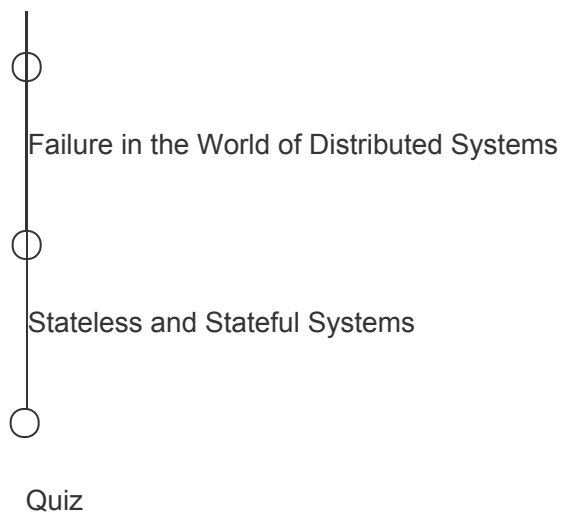
Basic Concepts and Theorems

Conclusion

Mark Module as Completed

Types of Failures

Let's see the four basic types of failures.

We'll cover the following

- Fail-stop
- Crash
- Omission
- Byzantine

There are several different types of failures. The following illustration shows the most basic categories.

Four most common type of failures that occur in distributed systems

Let's see these failures one by one.

Fail-stop#

A node halts and remains halted permanently. Other nodes can detect that the node has failed (i.e., by communicating with it).

Crash#

A node halts, but silently. So, other nodes may not be able to detect this state. They can only assume its failure by being able to communicate with it.

Omission#

A node fails to respond to incoming requests.

Byzantine#

A node exhibits arbitrary behavior: it may transmit arbitrary messages at arbitrary times, take incorrect steps, or stop.

Byzantine failures occur when a node does not behave according to its specific protocol or algorithm. This usually happens when a malicious actor or a software bug compromises the node.

To cope with these failures, we need complex solutions. However, most companies deploy distributed systems in environments that they assume to be private and secure.

Fail-stop failures are the simplest and the most convenient ones from the perspective of someone that builds distributed systems. However, they are not very realistic. This is because there are many cases in real-life systems where it's not easy for us to identify whether another node crashes or not.

> Most of the algorithms we analyze in this course work under the assumption of crash failures.

Back

System Models

Next

The Tale of Exactly-Once Semantics

Mark as Completed

Report an Issue