Basic Concepts and Taxonomy of Dependable and Secure Computing

Algirdas Avizienis, Jean-Claude Laprie
Brian Randell, and Carl Landwehr
Overview

➲ Motivation
➲ Concepts in Our Taxonomy
➲ Dependability and Security
➲ Threats to Dependability and Security
➲ Means to Dependability and Security
➲ Conclusion
➲ Questions
Motivation

• Communication and cooperation among communities are difficult
  • Especially when system failures

• Explicit and clear concepts are necessary

• But, there are uncertainties and complexity in systems
Concepts of Our Taxonomy

➲ System
  • A system is an entity that interacts with other entities, i.e., other systems, including hardware, software, humans, etc.

➲ Function
  • The function of a system is what the system is intended to do and is described by the functional specification in terms of functionality and performance

➲ Behavior
  • The behavior of a system is what the system does to implement its function and is described by a sequence of states

➲ Structure
  • The structure of a system is what enables it to generate the behavior.
Service

- The service delivered by a system is its behavior as it is perceived by its user(s)
- Roles
  - Provider
  - User
- Interfaces
  - Service Interface
  - Use Interface
- States
  - External State
    - Global variables
  - Internal State
    - Local variables
Overview of a system

- A service is a sequence of the system’s external states
Dependability and Security

- Definition of Dependability
- Definition of Security
- Their Attributes
- Their Relationship
Definitions of Dependability

-definition 1
  - The ability to deliver service that can justifiably be trusted
    - Stress the need for justification of trust

-definition 2
  - The ability to avoid service failures that are more frequent and more severe than is acceptable
    - Stress the avoidance of failures
Attributes of Dependability

➲ **Availability**
  • Readiness for correct service

➲ **Reliability**
  • Continuity of correct service

➲ **Safety**
  • Absence of catastrophic consequences on the user(s) and the environment

➲ **Integrity**
  • Absence of improper system alterations

➲ **Maintainability**
  • Ability to undergo modifications and repairs
Security

**Definition of Security**
- Security is a composite of the attributes of confidentiality, integrity, and availability, requiring the concurrent existence of
  - Availability for authorized action only
  - Confidentiality
  - Integrity with “improper” meaning “unauthorized”

**Confidentiality**
- The absence of unauthorized disclosure of information
- With great prominence
Relationship between Dependability and Security
Threats to Dependability and Security

 Faults
   A fault is the adjudged or hypothesized cause of an error

 Errors
   An error is the part of total state of the system that may lead to its subsequent service failure
   - Active
   - Latent
Failures (or Service Failures)

A failure is an event that occurs when the delivered service deviates from correct service.

At least one external state of the system deviates from the correct service state.
Taxonomy of Faults

⯈ Development Faults
● All fault classes occur during the development

⯈ Physical Faults
● All fault classes that affect hardware

⯈ Interaction Faults
● All external faults
  ● e.g. Interface mismatch between components
**Taxonomy of Faults**

- **Natural Faults**
  - Caused by natural phenomena without human participation
    - Physical faults
    - Production defects originating from development
    - Internal/External

- **Human-Made Faults**
  - Result from human actions
    - Omission/Commission faults
      - e.g. Absence/Wrong action
    - Malicious/Nonmalicious faults
      - Virus/Flaw
Taxonomy of Errors

Errors
- An error is the part of total state of the system that may lead to its subsequent service failure
  - Detected/Latent
  - Content/Timing
  - Consistent/Inconsistent

Does an error cause a service failure?
- It depends on the structure of the system
  - Redundancy
- It also depends on the behavior of the system
  - What if the part of the state that contains the error never be needed for service?
Taxonomy of Failures

Service Failures

- A service failure is defined as an event that occurs when the delivered service deviates from correct service.
- 4 Viewpoints

![Diagram of service failure taxonomy]

- Content failures
  - Early timing failures
  - Late timing failures
  - Halt failures
  - Erratic failures
- Signaled failures
  - Unsignaled failures
- Consistent failures
  - Inconsistent failures
- Minor failures
  - Catastrophic failures
Development Failures

- Development Failures
  - Development faults introduced into the system by its environment, especially by human, may contribute to partial or complete development failures
    - Budget failure
    - Schedule failure

- Development failures have a very negative impact on the user community
  - Complete development failure of the AAS system resulted in the waste of $1.5 billion!!!
Chain of dependability and security threats
Means to Attain Dependability and Security

- Fault Prevention
- Fault Tolerance
- Fault Removal
- Fault Forecasting
Fault Prevention

- Part of general engineering

- Prevention of development faults is mentioned
  - Software & hardware
    - e.g. C or Java
  - Improvement of development processes
    - e.g. Recording faults in product
Fault Tolerance

Failure avoidance by error detection and system recovery

- Error Detection [identifies the presence of an error]
- Error Handling [eliminates errors from the system state]
- Recovery [transforms a system state that contains one or more errors and (possibly) faults into a state without detected errors and without faults that can be activated again]
- Fault Handling [prevents faults from being activated again]

- Concurrent Detection [takes place during normal service delivery]
- Preemptive Detection [takes place while normal service delivery is suspended; checks the system for latent errors and dormant faults]
- Rollback [brings the system back to a saved state that existed prior to error occurrence; saved state: checkpoint]
- Rollforward [state without detected errors is a new state]
- Compensation [the erroneous state contains enough redundancy to enable error to be masked]
- Diagnosis [identifies and records the cause(s) of error(s), in terms of both location and type]
- Isolation [performs physical or logical exclusion of the faulty components from further participation in service delivery, i.e., makes the fault dormant]
- Reconfiguration [either switches in spare components or reassigns tasks among non-failed components]
- Reinitialization [checks, updates and records the new configuration and updates system tables and records]
Fault Removal

- System development phase
  - 3 steps: Verification, Diagnosis, Correction
  - Verification approaches
- System use phase
  - Corrective or preventive maintenance
Fault Forecasting

Fault forecasting is conducted by performing an evaluation of the system behavior with respect to fault occurrence of activation.

- Qualitative evaluation
  - Identify, classify, and rank the failure modes

- Quantitative evaluation
  - Evaluate in terms of probabilities the extent to which some of the attributes are then viewed as measure
Relationship between the 4 means

僇 Fault Prevention vs Fault Removal
• Faults may occur after prevention, we need fault removal

僇 Fault Removal vs Fault Forecasting
• Fault removal may generates faults, we need fault forecasting

僇 Fault Tolerance is required even more
• Increasing dependence on computing systems
• Fault Tolerance needs fault removal & forecasting

僇 Nothing is perfect, we need the combined utilization of all 4 means
Conclusion

➲ We need trust various computing systems
  • Airplane, nuclear plant, etc
➲ A single conceptual framework among various systems
  • Availability
  • Reliability
  • Safety
  • Confidentiality
  • Integrity
  • Maintainability
➲ We need a system with an appropriate balance of these properties
In fault tolerance, error handling includes rollforward, can you give me an example of rollforward? Is it easy to do a rollforward?
What is symbolic execution? (in section 5.3.1)
Question 3

What’s the difference between protective redundancy and unintentional redundancy?
Question 4

Are there any computing systems, each phase of which actually uses all 4 approaches presented in the paper? (Fault prevention, fault removal, fault tolerance, fault forecasting)
More Questions?