Basic concepts and Taxonomy of Dependable and Secure Computing

IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, 2004

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

Presented by Srdjan Pudar

Graduate student, Iowa State University

(for CprE 545: Fault-Tolerant Systems class)

Motivation

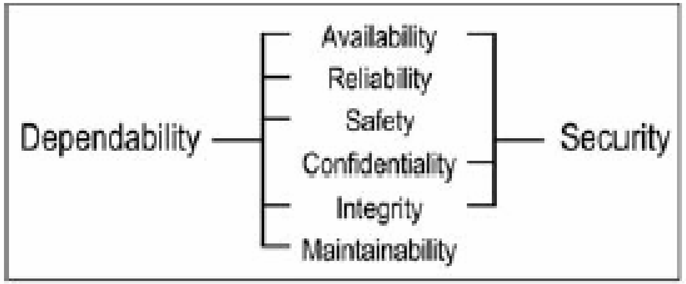
- Bring together dependability and security
- Dependability and security have followed different path that come together
- In recent years, steady increase of research efforts in cyber security due to
  - Growing concerns for cyber attacks, and
  - Increased trend in cyber attack incidents.
- Dependability
  - Restriction to non-malicious faults was only a part of the problem
- Security
  - Shift from confidentiality…
  - To concerns in regard of integrity and availability

Paper overview

- 1. Threats
  - Faults
  - Failures
  - Errors
- 2. Attributes of Dependability and Security
- 3. Means to attain Dependability and Security
  - Fault prevention
  - Fault tolerance
  - Fault removal
  - Fault forecasting

Introduction

- Attributes of dependability
  - Reliability
  - Availability
  - Safety
  - Integrity
  - Maintainability, etc.



- Security attributes
  - Confidentiality- the absence of unauthorized disclosure of information.
  - Availability- readiness for correct service.
  - Integrity- absence of improper system alterations.

Basic Concepts and Terms

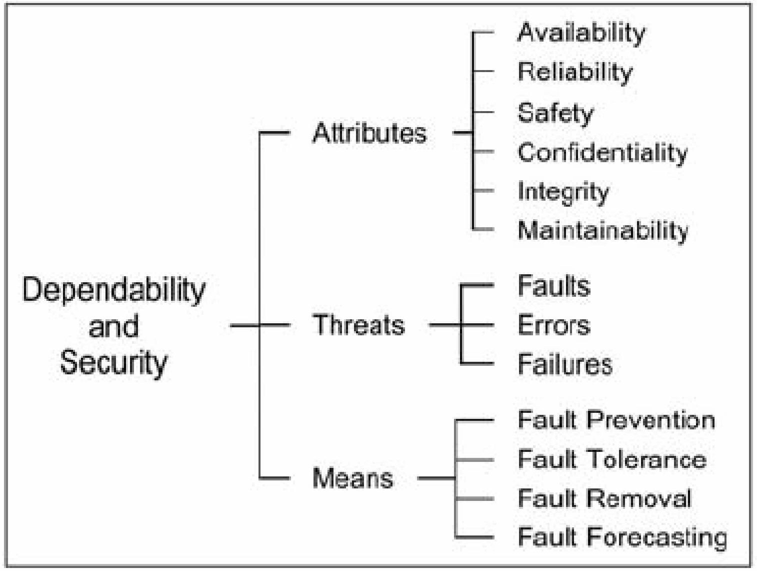
- These concepts are relevant to computer based systems.
- Correct Service
  - Implements the system function
- Service failure
  - Deviation from correct service (error)
  - Transition from correct to incorrect service
- Service outage
  - Period of delivery of incorrect service

Basic Concepts and Terms

- Failure modes and failure severities
  - Ranks of deviations in terms of severity of impact
- Fault
  - Adjudged or hypothesized cause of the error
  - Active (causes error) and dormant
- Error
  - Part of the total state of the system that may lead to its subsequent service failure
- Vulnerability
  - Presence of the internal fault that enables an external fault to harm the system
- Degraded mode
  - Limited service, partial fault

Dependability, Security, and their Attributes

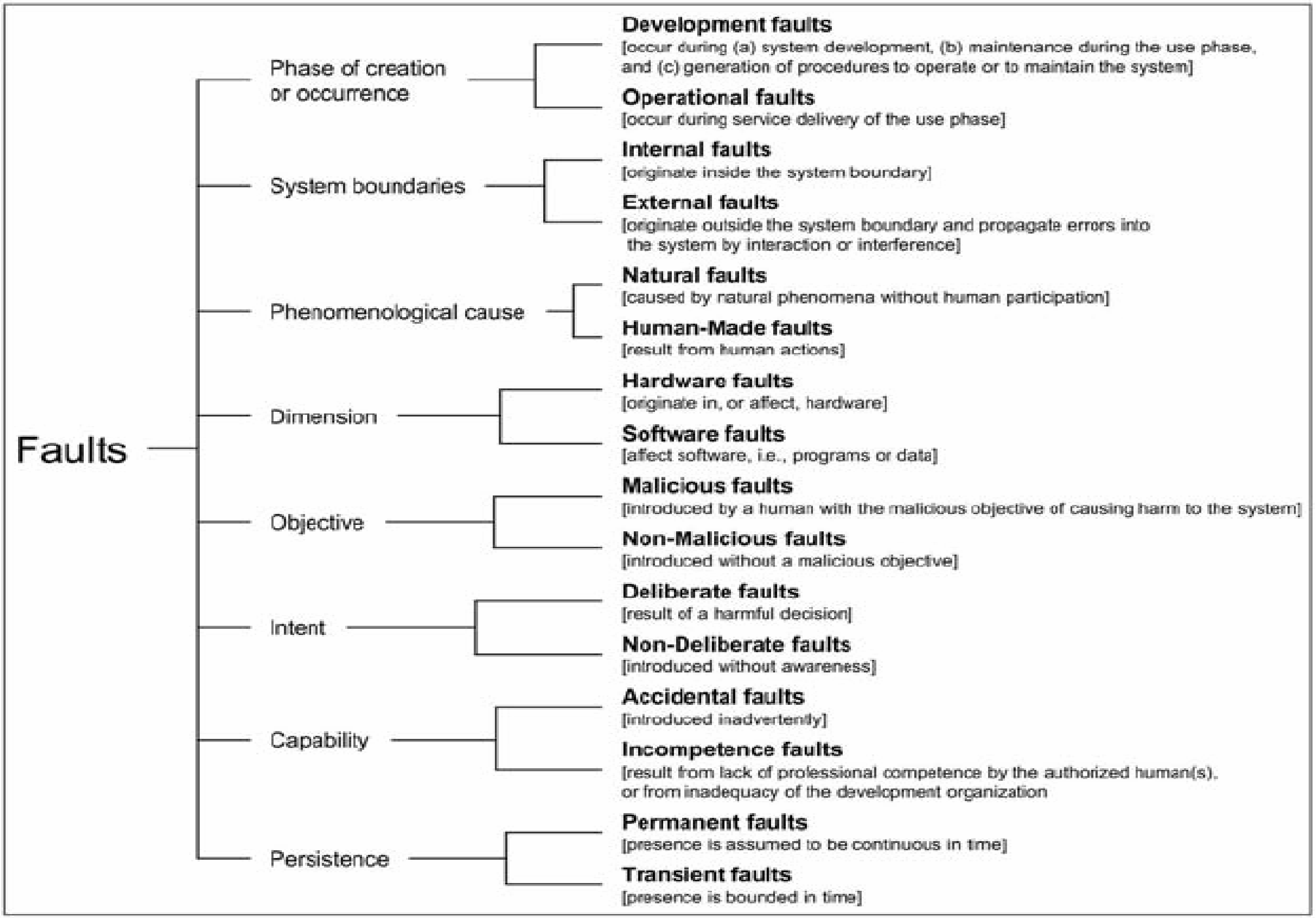
- Availability, Reliability, Safety, integrity, maintability: Dependability concept covers those
- Security concept attributes: Confidentiality, integrity, and availability
  - Confidentiality: absence of unauthorized disclosure of the information



Faults

- Three major groups
  - Development faults
    - All faults during development
  - Physical faults
    - All faults that affect hardware
  - Interaction faults
    - All external faults

Fault Taxonomy



Faults

- Phase of creation or occurrence
  - **Development faults** [occur during (a) system development, (b) maintenance during the use phase, and (c) generation of procedures to operate or to maintain the system]
  - **Operational faults** [occur during service delivery of the use phase]
- System boundaries
  - **Internal faults** [originate inside the system boundary]
  - **External faults** [originate outside the system boundary and propagate errors into the system by interaction or interference]
- Phenomenological cause
  - **Natural faults** [caused by natural phenomena without human participation]
  - **Human-Made faults** [result from human actions]
- Dimension
  - **Hardware faults** [originate in, or affect, hardware]
  - **Software faults** [affect software, i.e., programs or data]
- Objective
  - **Malicious faults** [introduced by a human with the malicious objective of causing harm to the system]
  - **Non-Malicious faults** [introduced without a malicious objective]
- Intent
  - **Deliberate faults** [result of a harmful decision]
  - **Non-Deliberate faults** [introduced without awareness]
- Capability
  - **Accidental faults** [introduced inadvertently]
  - **Incompetence faults** [result from lack of professional competence by the authorized human(s), or from inadequacy of the development organization
- Persistence
  - **Permanent faults** [presence is assumed to be continuous in time]
  - **Transient faults** [presence is bounded in time]

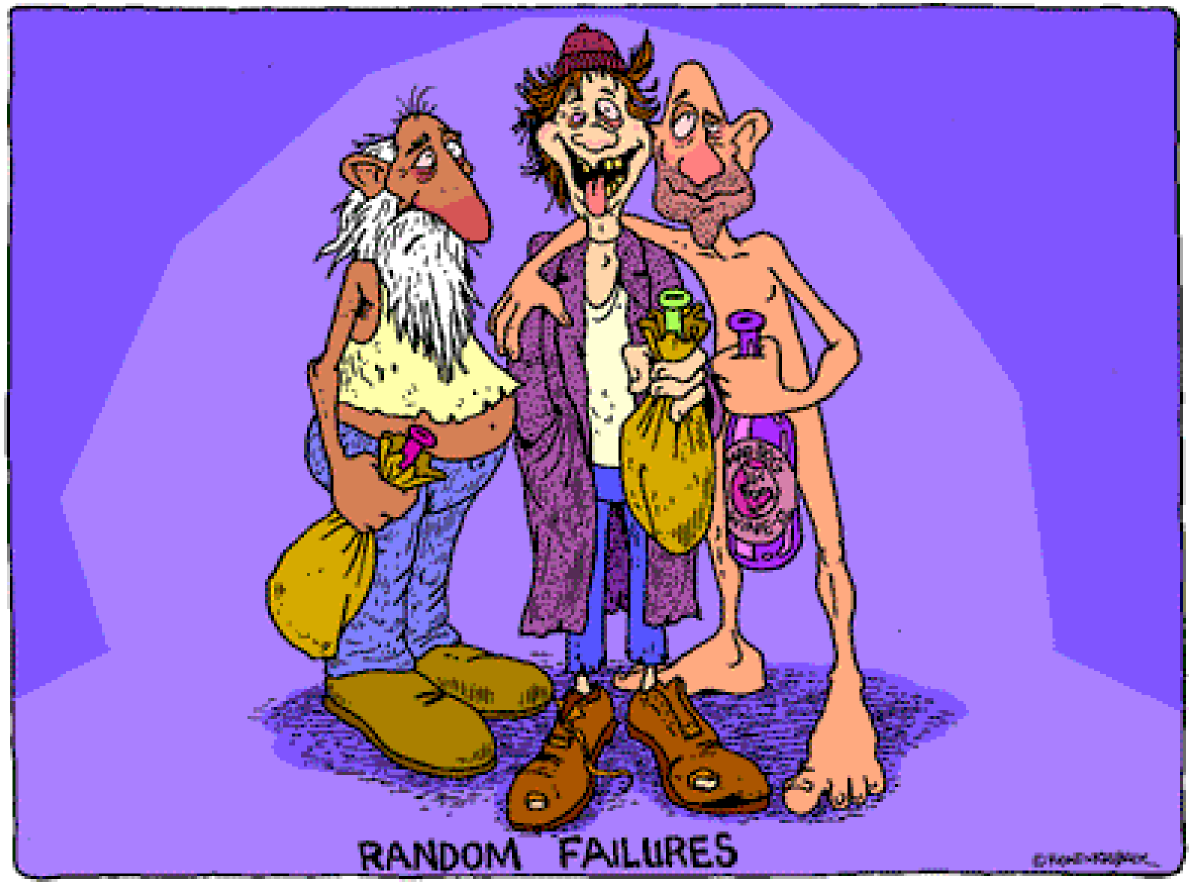
Faults Taxonomy

- Natural faults
  - Physical faults without human participation
- Human-made faults
  - Omissions(absence of actions), commissions faults (performing wrong actions)
  - By objective
    - Malicious faults
    - Non-malicious faults
- Interaction faults
  - Occur during use phase, are all *operational* faults
  - Occur in interaction with human, thus are *human-made* faults
  - May be result of the vulnerability. I.e. need vulnerability to manifest themselves

Malicious and Non-malicious Faults

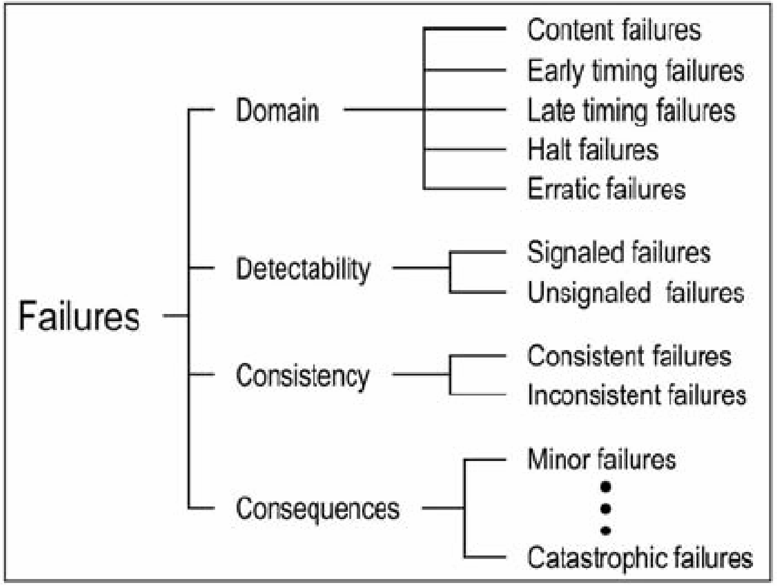
- Non-malicious faults example: *errata* in software/ hardware systems, documented in specification
- Non-malicious faults can be
  - Accidental faults
  - Incompetence faults
- Malicious faults classification
  - Malicious logic faults
    - Use development fault: Trojan horses
    - Viruses, worms
  - Intrusion attempts that are operational external faults (outages for exampple)

Failures



Service Failures

- Event that occurs when delivered service varies from correct service
- Classification



Service Failures Taxonomy

- Failure domain
  - Timing failures: timing of delivered information deviates from expected
  - Content failures: service content deviates from expected
  - Content and timing failures: ex. Erratic behavior
- Detectability viewpoint
  - Signaled failures, detecting mechanisms check validity of service
  - Un-signaled failures, otherwise

Service Failures Taxonomy

- Consistency viewpoint
  - Consistent
  - Inconsistent (Byzantine) failures: some or all users experience differently incorrect service
- Consequences
  - From minor to catastrophic consequences

Development Failures

- Introduced to system by development environment, such as human developers, development tools and facilities
- *Complete* development failure terminates development process
  - Budget and schedule failures
- *Partial* development failure
  - Budget and schedule overruns
  - Downgrading
    - Less functionality, lower performance or reduced dependability

Dependability and Security Failures

- A dependability or security failure occurs when the given system suffers *service failures* more frequently or more severely than acceptable.
- *Dependability and Security Specification* tries to address this issue.
  - Agreed upon parties

Errors

- Definition
  - Part of system's total state that may lead to a failure
- Relation between failure, error, and fault
  - Failure occurs when the error causes the delivered service to deviate from correct service. The cause of the error has been called a fault.
- Whether produces a failure depends of two things
  - Redundancy
  - Masking

Dependence and Trust

- Trust: accepted dependence
- The dependence of system A on system B represents the extent to which System A’s dependability is affected by that of System B.
- Dependence varies from *total* to *complete independence*
- *Accepted dependence* : judged that level of dependence is acceptable
- The extent to which A fails to provide means of tolerating B’s failures is a measure of A’s trust in B.

Attributes of Dependability and Security

- Secondary attributes for Security
  - Accountability
    - Availability and integrity of information of responsible person
  - Authenticity
    - Integrity of message origin, content, time, etc.
  - Nonrepudiability
    - Availability and integrity of the sender of message
- Security policy concept
  - Security-motivated constraints to be adhered by
  - Enforced via management, technical, or organizational controls
  - Security failure: lack of adherence to policy

Means to Attain Dependability and Security

- Fault Prevention
  - Design strategies, both software and hardware
- Fault Tolerance
  - Error detection and system recovery
- Fault Removal
  - Fault removal during development
    - Verification, validation, diagnosis, correction
  - Fault removal during use
    - Corrective and preventive maintenance
- Fault Forecasting
  - Evaluation of the system behavior

Conclusions

- Sophisticated computing systems are developed whose services we need to place great trust
- Simultaneous consideration of dependability and security provides a very convenient means of subsuming these various concerns within a single conceptual framework
- Strength in integration
  - Dependability attributes are not enough and often conflicting with each other
- Error, Fault, Failure model

Model-Based Evaluation: From Dependability to Security

Review of Paper by David M. Nicol, William H. Sanders, and Kishor S. Trivedi (IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, 2004) with presenter additions

Presented by Srdjan Pudar

Motivation

- In recent years, steady increase of research efforts in cyber security due to
  - Growing concerns for cyber attacks, and
  - Increased trend in cyber attack incidents.
- Can existing dependability research be used for security?
- Important to be able to quantitatively validate the efficacy of systems intended to be secure.
- How to characterize attacker behavior?
- Paper divided into tree parts
  - Defining Security parameters
  - Explain models (reliability related)
  - Application

Measures of Dependability and Security

- Classical Dependability terms
  - Reliability
    - Probability that system performs specified service (during period of time)
  - Availability
    - Quantification of alternation between proper and improper service
  - Performability
    - Quantification of system performance in the presence of failures
- Security terms
  - Security
    - Resilience of system/network to malicious attacks
  - Safety
    - Probability that a system doesn't fail in a matter that causes catastrophic damage

System Security Attributes

- Data confidentiality
  - System does not allow protected data to be read in an unauthorized fashion
- Data integrity
  - System does not allow protected data to be modified in an unauthorized fashion
- Nonrepudiation
  - System property that prevents future false denial of involvement
- Authentication
  - The claimed identity of a party to a transaction can be independently verified.
- Survivability
  - Ability of system to persevere its mission in presence of attacks and faults

Models

- Models must explain following:
  - How and why security breaches occur
  - System vulnerabilities and attackers' exploitation
  - Cost of recovery
  - Means to characterize attacker behavior
- Three types of models reviewed in paper
  - Combinatorial Methods
  - Model Checking
    - Reachability analysis using system states
  - State-Based Stochastic Methods
    - Mathematical models that specify probabilistic assumptions about time durations and transition behavior
- They all focus on different level of abstractions and system characteristics

Combinatorial Methods

- Simple approach
- Unable to capture random dependence and imperfect fault coverage
- Widely present in reliability modeling software
- Three main types
  - Reliability Block Diagrams (RBD)
  - Fault Trees
  - Attack Trees

RBDs

- Diagram consisting of nodes and links
- Nodes represent system components and some connections between components (dummy nodes)
- Dummy nodes and links model the dependency of systems to its components.
- If there is connection between two dummy nodes representing start and end, system is operational
- Can be solved in linear time
- Mostly used for reliability and availability modeling

RBDs

- Example from literature

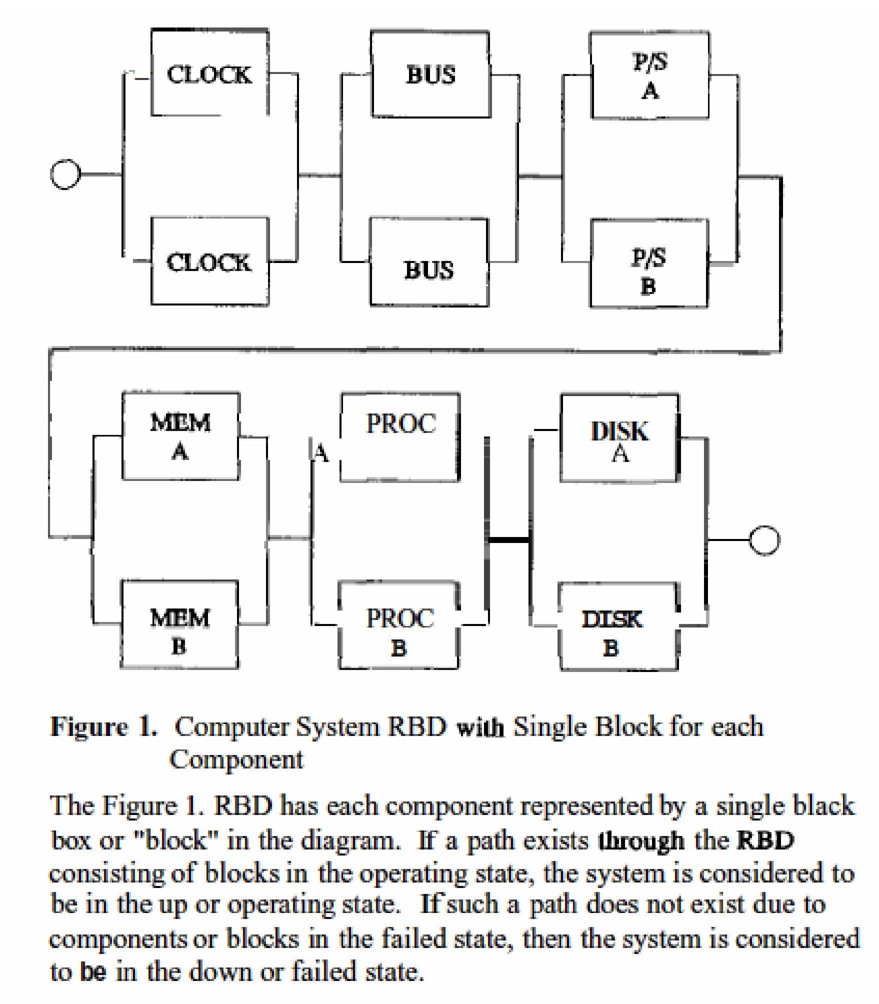
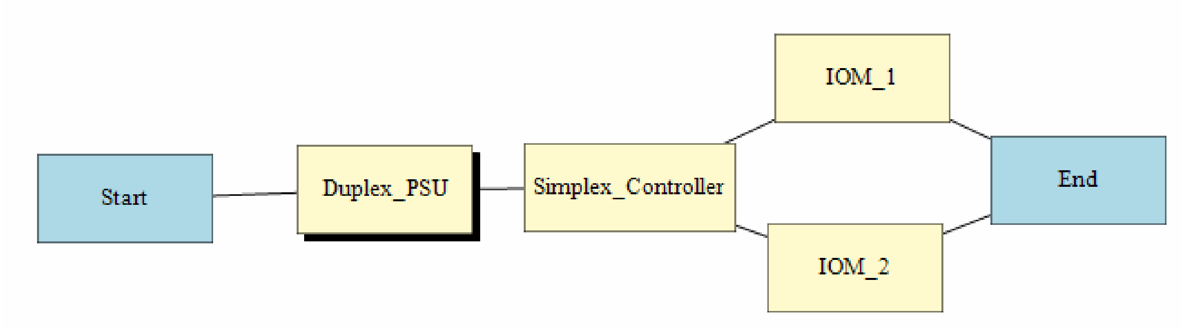


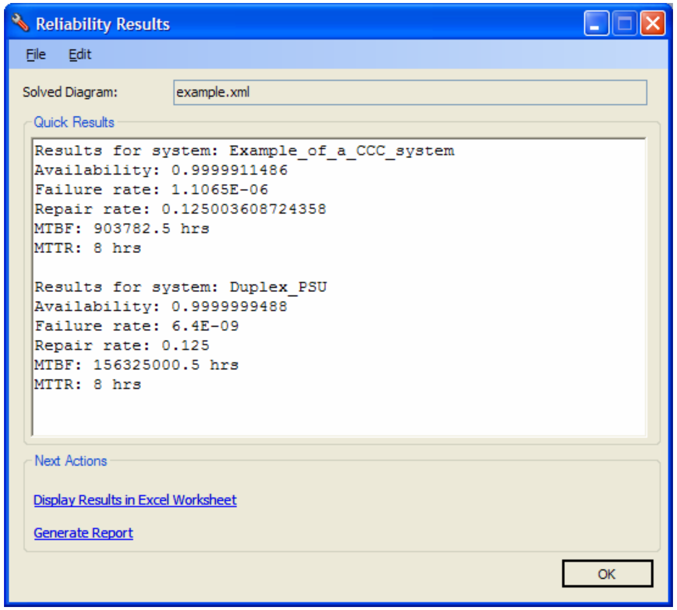
Figure 1. Computer System RBD **with** Single Block for each Component

The Figure 1. RBD has each component represented by a single black box or "block" in the diagram. If a path exists **through** the **RBD** consisting of blocks in the operating state, the system is considered to be in the up or operating state. If such a path does not exist due to components or blocks in the failed state, then the system is considered to **be** in the down or failed state.

RBDs

- Example from *CCC Reliability Tool*





Solved Diagram: example.xml

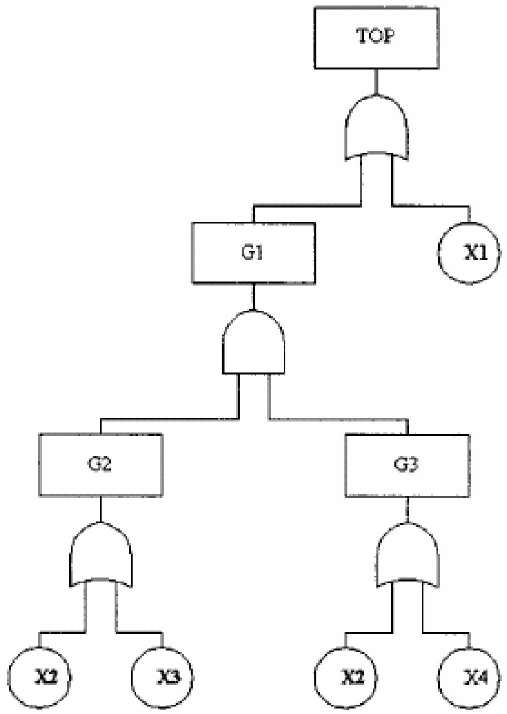
```
Results for system: Example_of_a_CCC_system
Availability: 0.9999911486
Failure rate: 1.1065E-06
Repair rate: 0.125003608724358
MTBF: 903782.5 hrs
MTTR: 8 hrs

Results for system: Duplex_PSU
Availability: 0.9999999488
Failure rate: 6.4E-09
Repair rate: 0.125
MTBF: 156325000.5 hrs
MTTR: 8 hrs
```

Fault Trees

- Acyclic graph with two types of nodes
  - Internal that are basic and advanced logic gates
  - External (leaves) that represent system information
- Similar (and in some cases equivalent) to the RBDs
- Model flow of failure towards top and system dependency of components
- Application in reliability, availability, safety modeling and modeling of software fault tolerance
- Dynamic Fault Trees incorporate some stochastic constructs
  - Priority AND gates, dependency sequence, inhibitor arcs
- Iowa State *HIMAP* software solves Fault Trees

Fault Trees

- Simple Fault tree
- X1, X2, X3, X4 – basic events

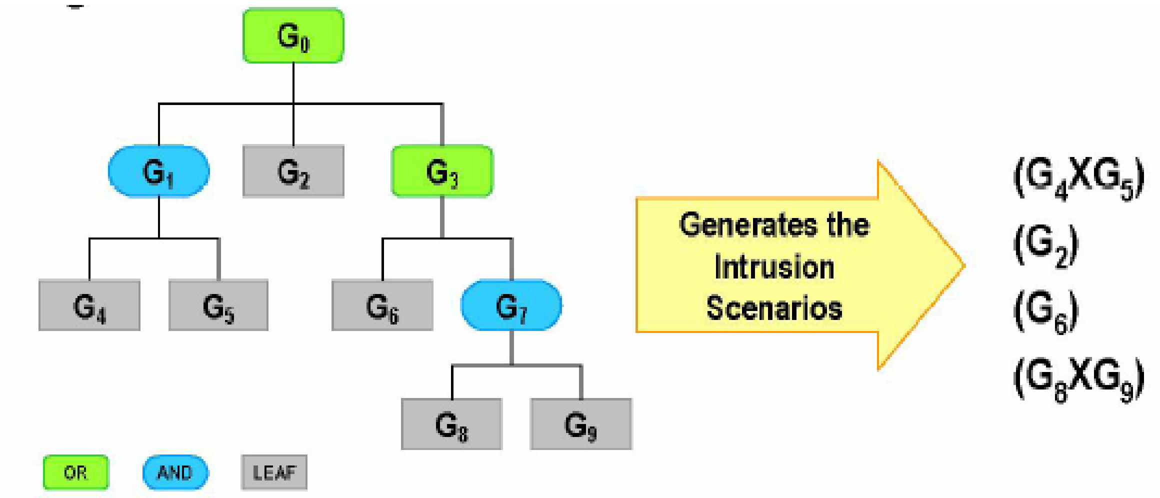


Attack Trees

- Model all possible attacks against the system
  - Attacker behavior, system vulnerabilities
- Describe sets of events that can lead to system failure in a combinatorial way
- Attacks are represented in a tree structure
- Goal is at root node
- Different ways to achieve that goal start at different leaf nodes
- Each non-leaf node represent attack goal (subgoal)
  - AND or OR node

Attack Trees

- Example of an attack tree
- G0- Goal
- G4xG5, G2, G6 and G8xG9- different intrusion scenarios



Attack Trees: Another Example

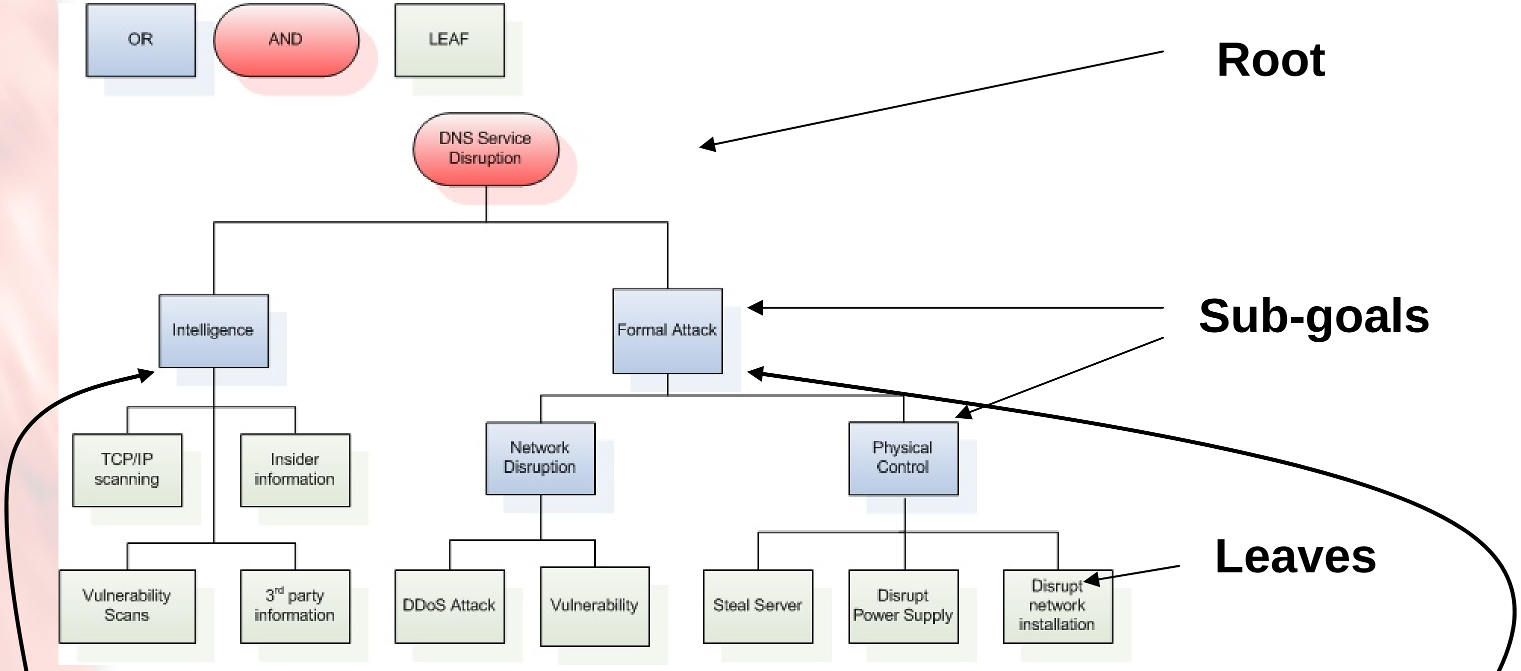


Figure 2: Attack tree model of generic DNS service disruption

- Attack model of DNS disruption
- *Obtain intelligence* and *perform formal attack*

Attack Trees Analysis Methods

- Obtain value at root node
  - Probability
  - Cost
  - Etc.
- Perform *attack scenario analysis*
  - Intrusion scenarios are various ways to reach root goal from initial attacks on leaf nodes
  - Most likely attack scenario is a intrusion scenario with lowest cost/difficulty for attacker

Model Checking

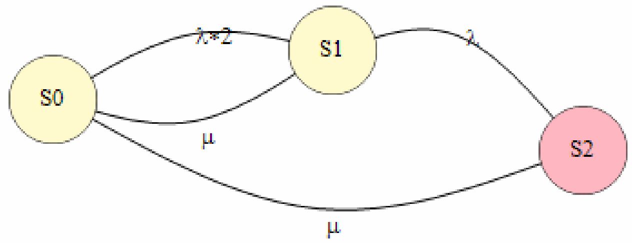
- Model Checking is actually reachability analysis of model state space
- Idea is to:
  - Analyze system states
  - Find states of interest
  - State transitions that reach failures
- Usage
  - Checking software for flaws
  - Modeling public key protocols
  - Modeling attacks on the systems
    - Network state: hosts and their capabilities
    - Attacker state: capabilities and accesses gained
- Difficulty: state size

State-based Stochastic methods

- Cover more cases that previous methods, capable of taking in account complex relationships
- Use state-space mathematic models expressed with probabilistic assumptions about time durations and transition behaviors
- Two main models based on
  - Markov chains (CTMC)
  - Stochastic Petri nets
- Strong modeling power
- For example, ability to model *sequencing*

Markov Rewards Model

- Simple example from *CCC Reliaiblity Tool*
- Represents duplex physical system modeled by failure *λ* and repair *μ* rate of components
- S2 – failed state
- Reward for every non-failure state



Markov Rewards Model

- Various usage in dependability domain
- Largeness Problem: number of system states large for systems with large number of components
- To reduce complexity researches focus on reducing number of states
  - Largeness avoidance techniques
  - Largeness tolerance techniques

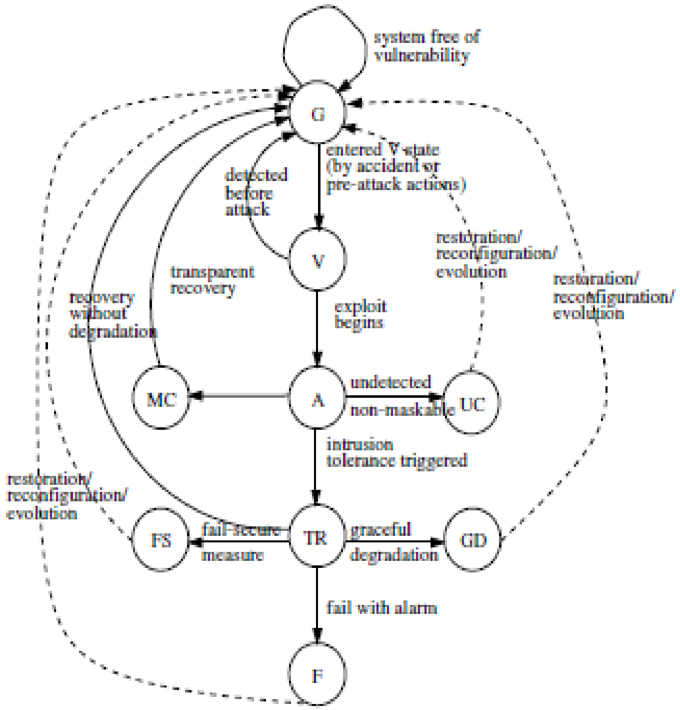
Addressing Largeness Problem

- Lumping methods: Replace multiple states with single states
  - State-level lumping
    - Finds optimal solution on CTMC level
  - Model-level lumping
    - Lump first, then create the model
  - Compositional lumping technique
    - Lump sub-systems using state-level method
- Aggregation
  - Replace multiple states with one using approximations
- Also problem with storage
  - How to store such large states

Petri Net Based Models

- Higher abstraction level compared to Markov Reward Models
- Provide modeler to more focus on the modeled system than on mathematical correctness
- Stochastic Petri nets
  - Firing time distributions are exponential
  - Reducible to a Markov chain
  - Extensions for easier use
    - Immediate transitions, inhibitor arcs
- Deterministic Petri nets
  - Firing times are fixed (constant)

Example of a Stochastic Model



system free of vulnerability
G
entered V-state (by accident or pre-attack actions)
detected before attack
V
transparent recovery
recovery without degradation
restoration/ reconfiguration/ evolution
restoration/ reconfiguration/ evolution
exploit begins
MC
A
undetected non-maskable
UC
intrusion tolerance triggered
restoration/ reconfiguration/ evolution
FS
fail-secure measure
TR
graceful degradation
GD
fail with alarm
F

G	good state	TR	triage state
V	vulnerable state	FS	fail-secure state
A	active attack state	GD	graceful degradation state
MC	masked compromised state	F	failed state
UC	undetected compromised state		

Madan et al. *"Modeling and quantification of security attributes of software systems"*

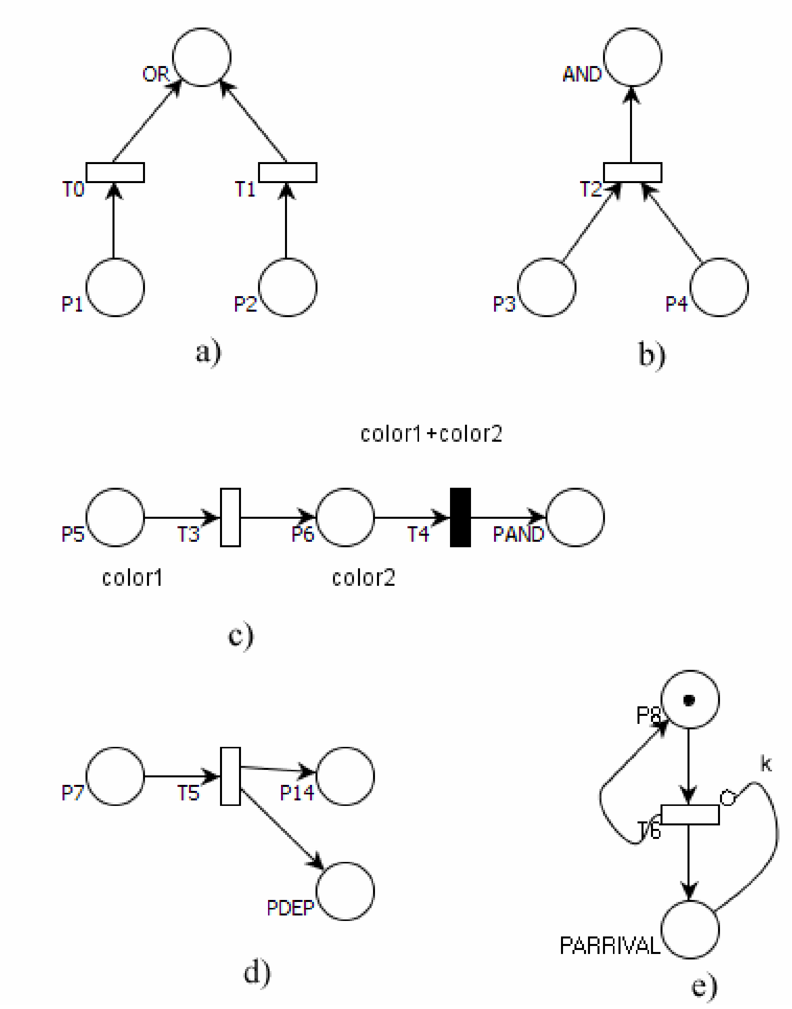
Figure 3: A state transition diagram for intrusion tolerant system

- This model is used for security evaluation
- Places
  - System states in regards of vulnerability
- Transitions
  - Events that lead to new system state

Petri Net Based Models

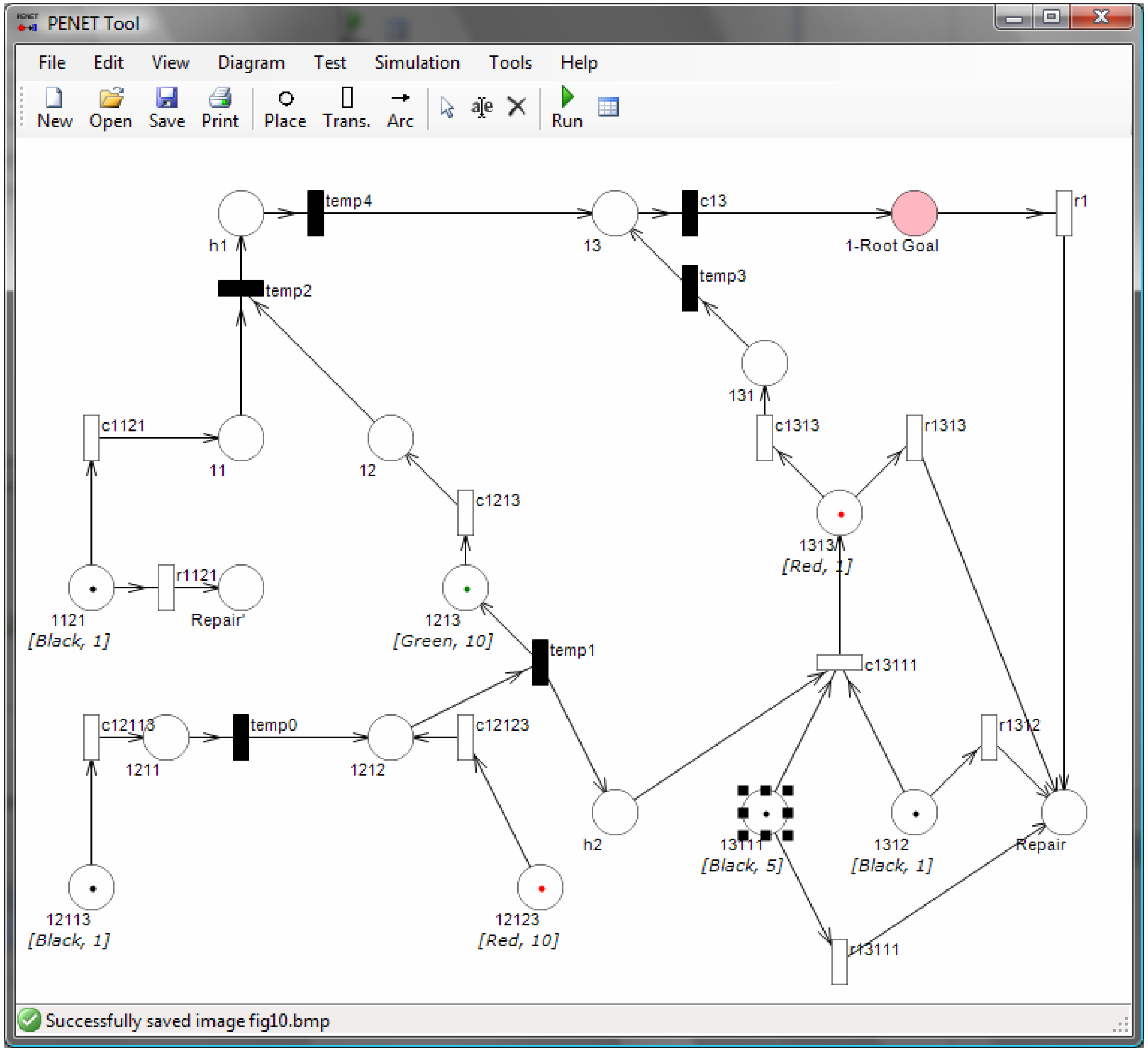
- PENET Methodology
  - Custom Petri net
    - Address existing issues in attack trees
    - Introduce concepts of reoccurring attacks, defense modeling, and dynamic constructs
    - Introduce sound analysis approach that follows attack execution in time domain
    - Provide means to evaluate system survivability and defense strategies
  - Software tool implements new approach and establishes its practical use.

PENET Constructs



- "Static" event gates
  - OR
  - AND
- Dynamic event gates
  - Priority AND
  - Dependency
  - Arrival construct

PENET Tool Example



Successfully saved image fig10.bmp

Application of Models

- Dependability calculations for various systems
- Vendor and research software such as
  - Reliability Workbench
  - MEADEP
  - HIMAP
    - Fault Tree
    - Markov model
  - CCC Reliability Tool
    - RBDs
    - Markov model
    - Mix
  - SHARPE
  - And many others

Conclusion

- Importance
  - No such thing as secure system
  - Rising number of cyber attacks
- Much of classical dependability analysis that can be transferred to security analysis.
- However some attributed of security cannot be transferred
- Promising new field of security research
- Quantifying Security
  - Modeling attacker behavior
  - Creating methodology for evaluating whether system meets requirements related to security

Conclusion

- Stochastic methods inspired by dependability models are promising to be useful for security evaluation
- Attack Trees can be used for simpler scenarios
- These two models can be combined
- Issues
  - In Reliability analysis, Reliability of elements is known exponential function
  - In Security analysis, Reliability and Security attributes of system elements are not known functions
  - Actual attack data is not being utilized in modeling efforts

References

- *Basic concepts and Taxonomy of Dependable and Secure Computing.* Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, 2004
- *Model-Based Evaluation:From Dependability to Security.* David M. Nicol, William H. Sanders, and Kishor S. Trivedi. IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, 2004
- *Choosing a Heuristic for the "Fault Tree to Binary Decision Diagram" Conversion, Using Neural Networks*. Lisa M. Bartlett and John D. Andrews IEEE TRANSACTIONS ON RELIABILITY, VOL. 51, NO. 3, SEPTEMBER 2002
- *Survivability analysis of distributed systems using attack tree methodology.* Casey Fung, Yi-Liang Chen, Xinyu Wang, Joseph Lee, Richard Tarquini, Mark Anderson, and Richard Linger
- *Analysis of reliability block diagrams with multiple blocks per component.*Forche, R.F.; Reliability and Maintainability Symposium, 1990. Proceedings., Annual
  23-25 Jan. 1990 Page(s):145 - 148
- *Modeling and quantification of security attributes of software systems,* Madan, B.B.; Gogeva-Popstojanova, K.; Vaidyanathan, K.; Trivedi, K.S., Dependable Systems and Networks,2002. Proceedings. International Conference on, vol., no.pp. 505- 514, 2002

QUESTIONS?

Thank you.